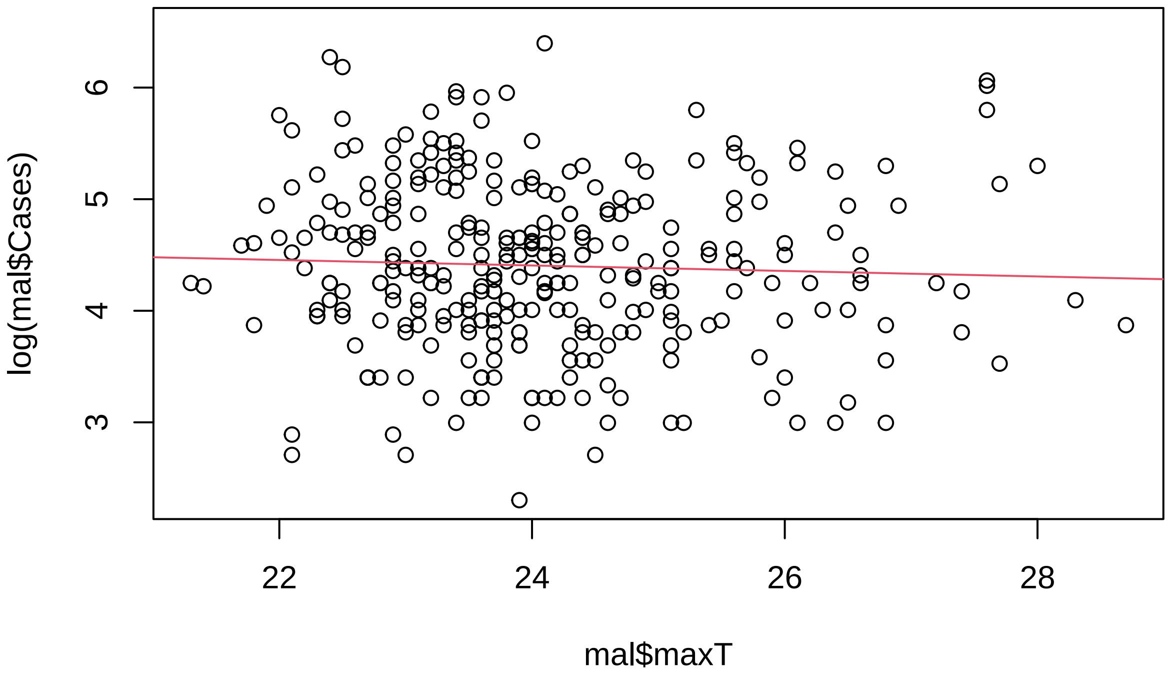
Time series in epidemiology: Lab Exercises

After loading the `Kericho.csv` data in R. Please provide an answer to the following questions.

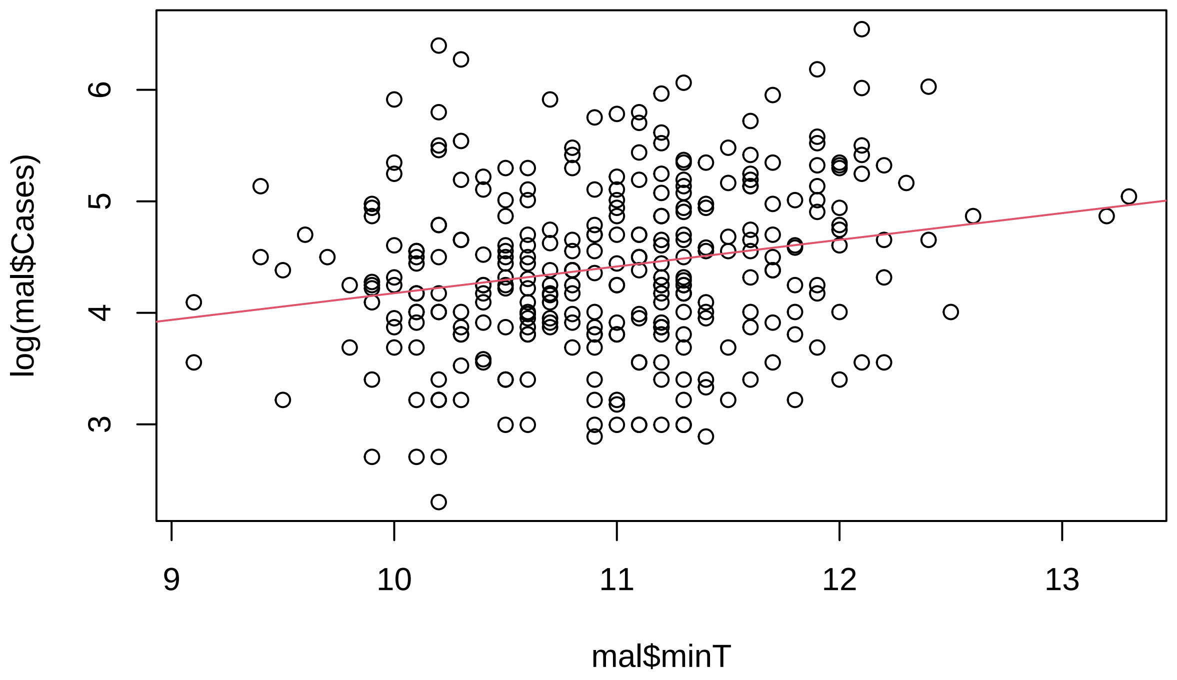
Exercise 1

1. Explore the association between maximum temperature (`maxT`), minimum temperature (`minT`) and rainfall (`Rain`) with the log-transformed number of reported malaria cases (`Cases`). What relationships do your observe and how strong are these?

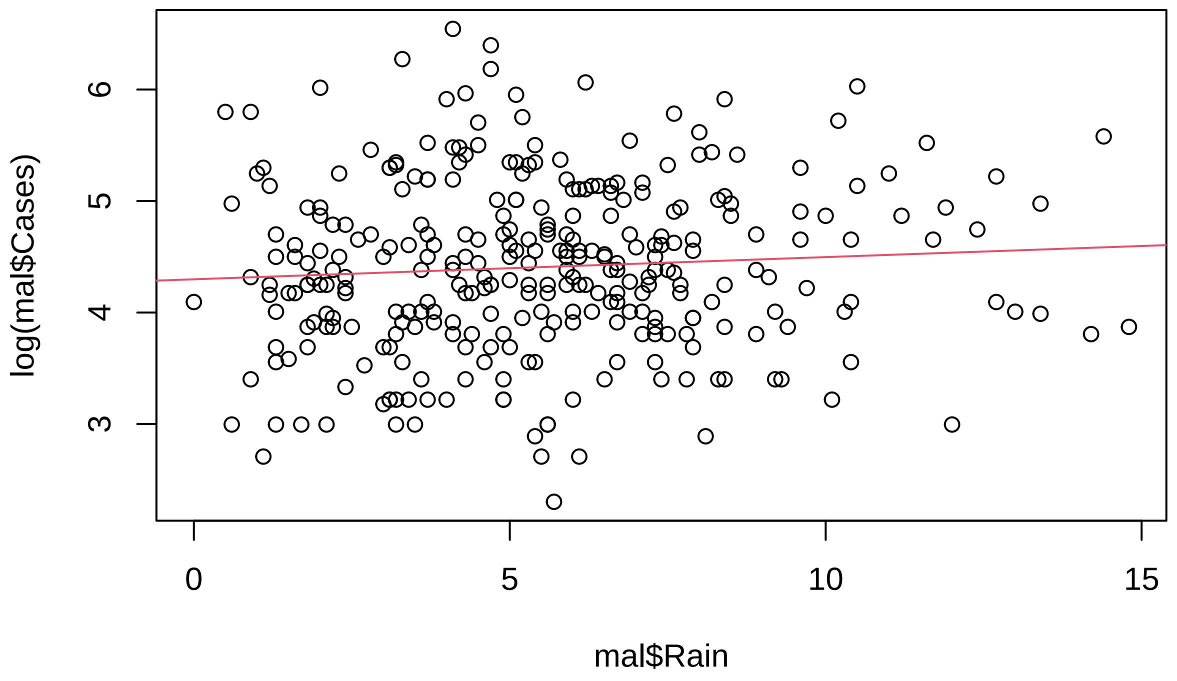
```r
mal <- read.csv("Kericho.csv")

plot(mal$maxT,log(mal$Cases))
abline(lm(log(Cases)~maxT,data=mal),col=2)
```



```r
plot(mal$minT,log(mal$Cases))
abline(lm(log(Cases)~minT,data=mal),col=2)
```



```r
plot(mal$Rain,log(mal$Cases))
abline(lm(log(Cases)~Rain,data=mal),col=2)
```



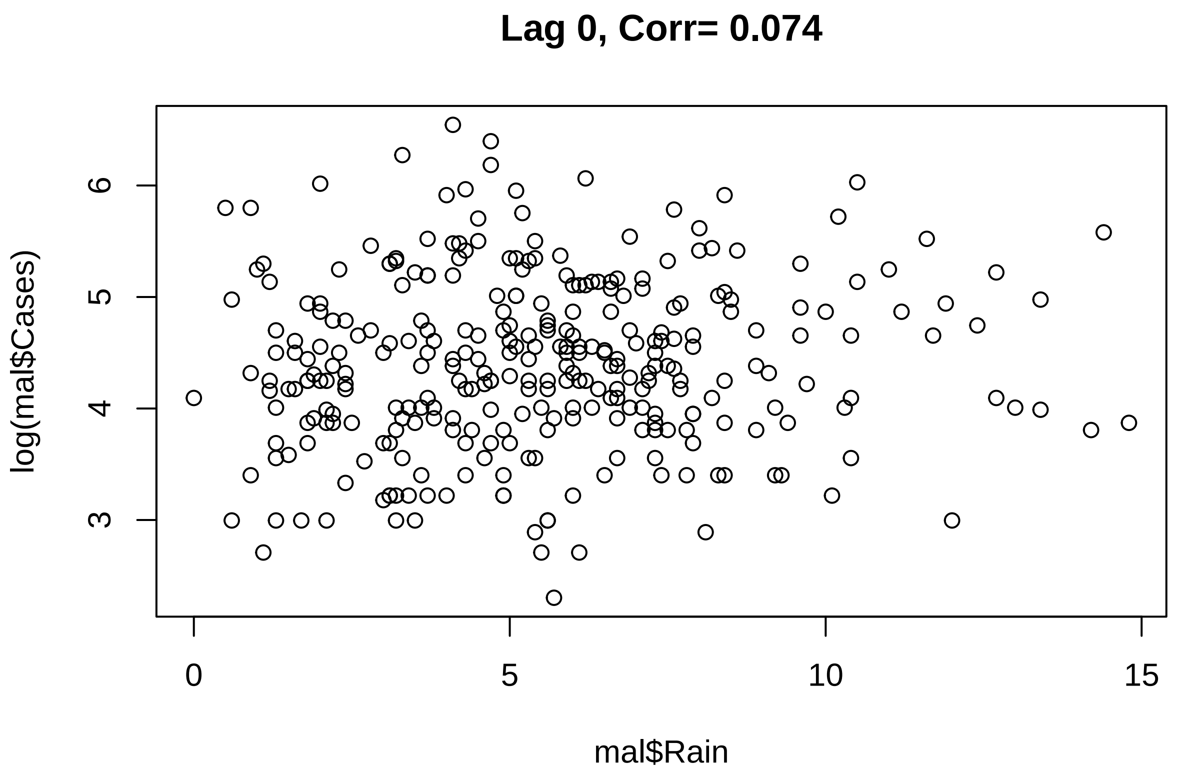
2. Consider the rainfall variable `Rain`. Create 4 time lagged variables that for a given month, give the rainfall amount of 1 month, 2 months, 3 months ago and 4 months ago. Plot the log-transformed number of cases against each of these variables. What do you observe?

```r
mal$Month <- factor(mal$Month,levels =
                 c("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep",
                   "Oct","Nov","Dec"),ordered=TRUE)
mal$t <- as.numeric(mal$Month)+12*(mal$Year-1979)
```

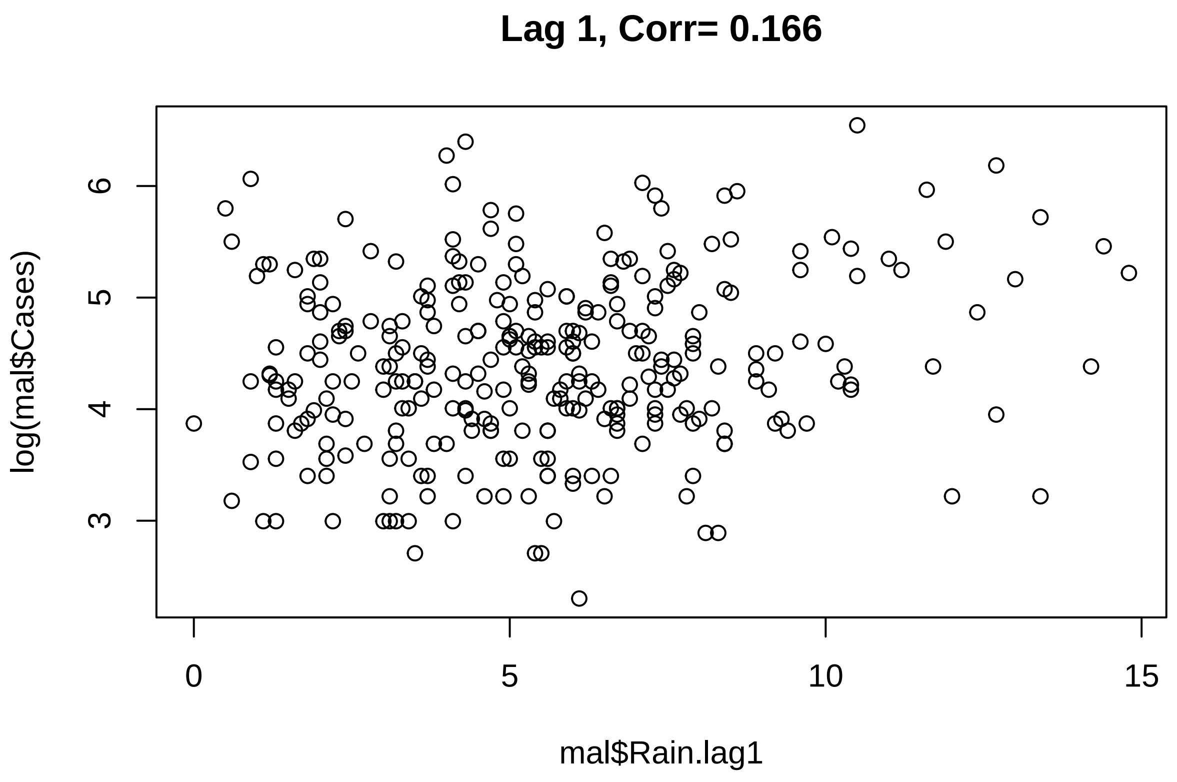
```
create.lag.var <- function(x,lag) {
  tlag <- mal$t-lag
  ind.lag <- sapply(tlag,function(x) {
                out <- which((mal$t-x)==0)
                if(length(out)==0) out <- NA
                return(out)
              })
  x[ind.lag]
}


mal$minT.lag1 <-create.lag.var(mal$minT,lag=1)

# Check on newly created variable
head(mal[sort.list(mal$t),c("minT.lag1","minT","t")])
```

```
##   minT.lag1 minT t
## 1        NA 11.8 1
## 2      11.8 11.3 2
## 3      11.3 10.9 3
## 4      10.9 12.0 4
## 5      12.0 10.9 5
## 6      10.9 11.4 6
```

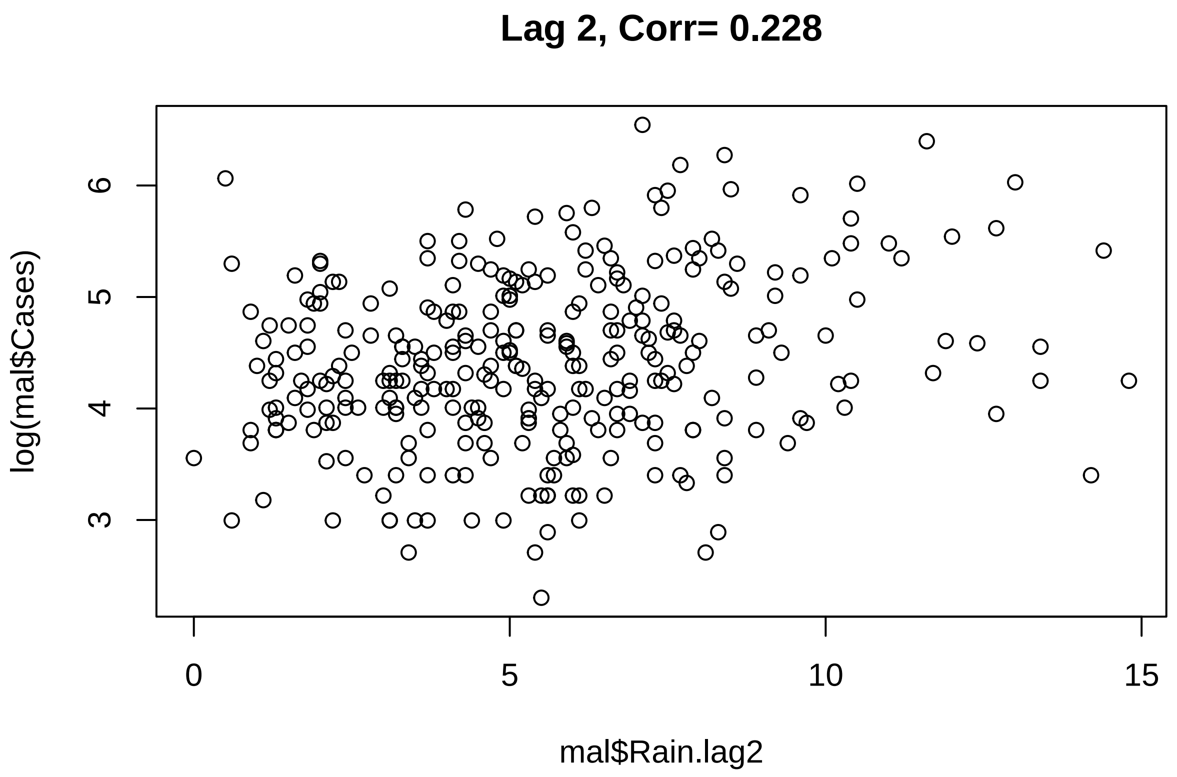
```
mal$Rain.lag1 <- create.lag.var(mal$Rain,lag=1)
mal$Rain.lag2 <- create.lag.var(mal$Rain,lag=2)
mal$Rain.lag3 <- create.lag.var(mal$Rain,lag=3)
mal$Rain.lag4 <- create.lag.var(mal$Rain,lag=4)

plot(mal$Rain,log(mal$Cases),
     main=paste("Lag 0, Corr=",round(cor(mal$Rain,log(mal$Cases)),3)))
```



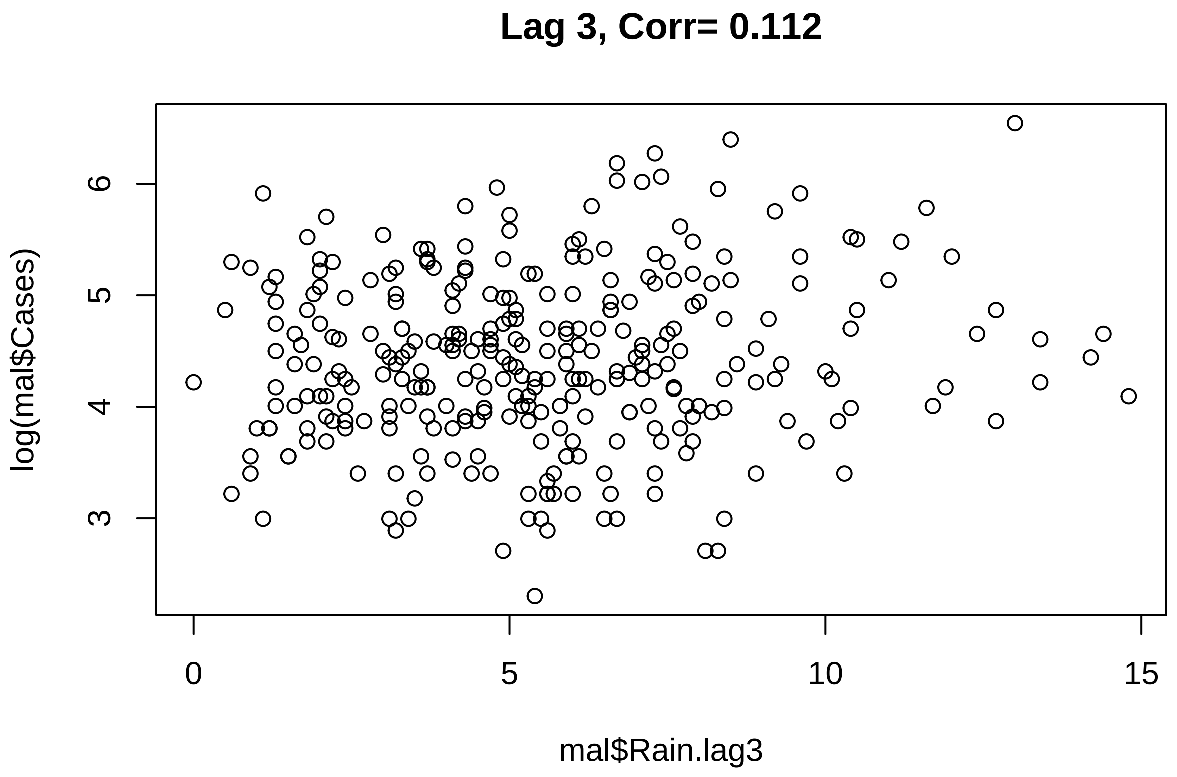
```r
plot(mal$Rain.lag1,log(mal$Cases),
     main=paste("Lag 1, Corr=",round(cor(mal$Rain.lag1,log(mal$Cases),use="complete.obs"),3)))
```



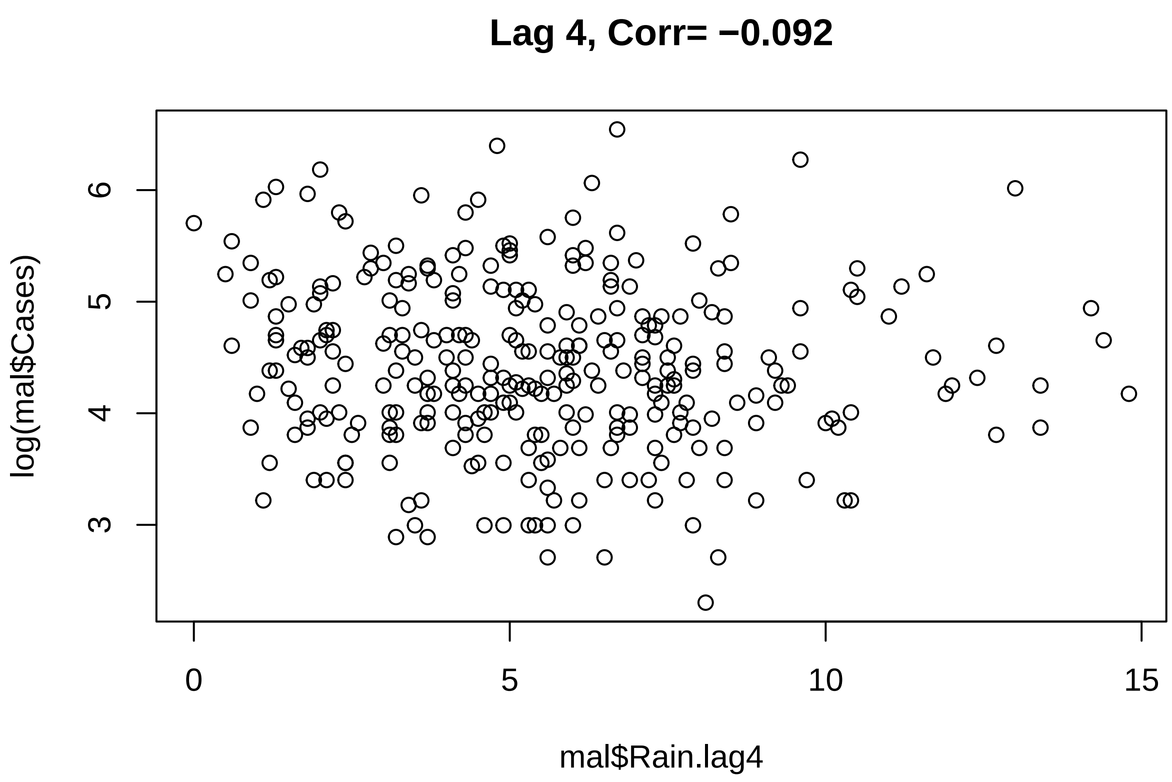
```r
plot(mal$Rain.lag2,log(mal$Cases),
     main=paste("Lag 2, Corr=",round(cor(mal$Rain.lag2,log(mal$Cases),use="complete.obs"),3)))
```



```
plot(mal$Rain.lag3,log(mal$Cases),
     main=paste("Lag 3, Corr=",round(cor(mal$Rain.lag3,log(mal$Cases),use="complete.obs"),3)))
```



```
plot(mal$Rain.lag4,log(mal$Cases),
     main=paste("Lag 4, Corr=",round(cor(mal$Rain.lag4,log(mal$Cases),use="complete.obs"),3)))
```



3. Fit a linear model for the log-transformed number of cases for each of the 4 lagged rainfall variables created in the previous point. Based on the fitted models, which lag has a stronger relationship with the reported malaria cases?

```
lm.lag0 <- lm(log(Cases) ~ Rain, data = mal)
lm.lag1 <- lm(log(Cases) ~ Rain.lag1, data = mal)
lm.lag2 <- lm(log(Cases) ~ Rain.lag2, data = mal)
lm.lag3 <- lm(log(Cases) ~ Rain.lag3, data = mal)
lm.lag4 <- lm(log(Cases) ~ Rain.lag4, data = mal)

summary(lm.lag0)
```

```
##
## Call:
## lm(formula = log(Cases) ~ Rain, data = mal)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.10892 -0.49873 -0.03262  0.52464  2.16426
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.29802    0.09424  45.607   <2e-16 ***
## Rain         0.01991    0.01522   1.308    0.192
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7678 on 308 degrees of freedom
## Multiple R-squared:  0.005526,	Adjusted R-squared:  0.002297
## F-statistic: 1.712 on 1 and 308 DF,  p-value: 0.1918
```

```r
summary(lm.lag1)
```

```
##
## Call:
## lm(formula = log(Cases) ~ Rain.lag1, data = mal)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.13518 -0.50605 -0.02626  0.53511  2.03922
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.16645    0.09336  44.626  < 2e-16 ***
## Rain.lag1    0.04448    0.01505   2.955  0.00337 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7575 on 307 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.02765,    Adjusted R-squared:  0.02448
## F-statistic:  8.73 on 1 and 307 DF,  p-value: 0.003372
```

```r
summary(lm.lag2)
```

```
##
## Call:
## lm(formula = log(Cases) ~ Rain.lag2, data = mal)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.11249 -0.48276 -0.00874  0.55499  2.03159
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.08080    0.09198  44.367  < 2e-16 ***
## Rain.lag2    0.06078    0.01483   4.098 5.34e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7462 on 306 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.05202,    Adjusted R-squared:  0.04893
## F-statistic: 16.79 on 1 and 306 DF,  p-value: 5.344e-05
```

```r
summary(lm.lag3)
```

```
##
## Call:
## lm(formula = log(Cases) ~ Rain.lag3, data = mal)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.11411 -0.48978 -0.04558  0.56805  1.90081
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.25582    0.09353  45.502   <2e-16 ***
## Rain.lag3    0.02979    0.01508   1.976    0.049 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7585 on 305 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.01264,    Adjusted R-squared:  0.009405
## F-statistic: 3.905 on 1 and 305 DF,  p-value: 0.04904
```

```
summary(lm.lag4)
```

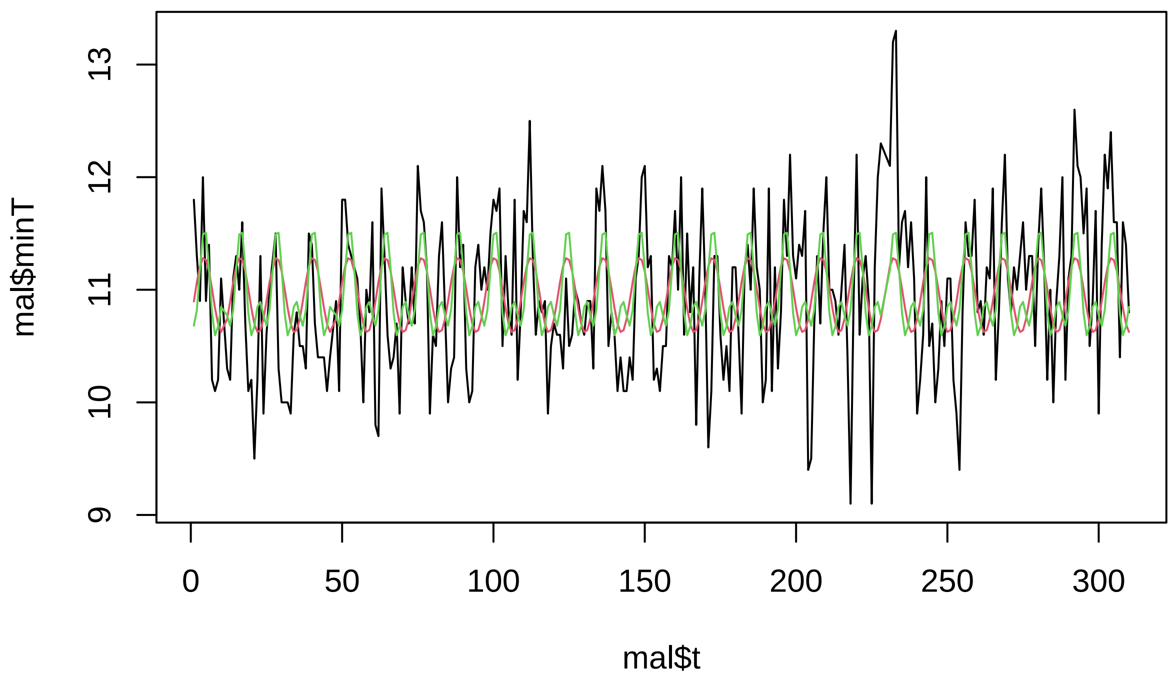
```
##
## Call:
## lm(formula = log(Cases) ~ Rain.lag4, data = mal)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.05699 -0.49388 -0.01158  0.54307  2.15010
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.55766    0.09380  48.588   <2e-16 ***
## Rain.lag4   -0.02445    0.01510  -1.619    0.106
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7591 on 304 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.008552,   Adjusted R-squared:  0.005291
## F-statistic: 2.622 on 1 and 304 DF,  p-value: 0.1064
```

Exercise 2

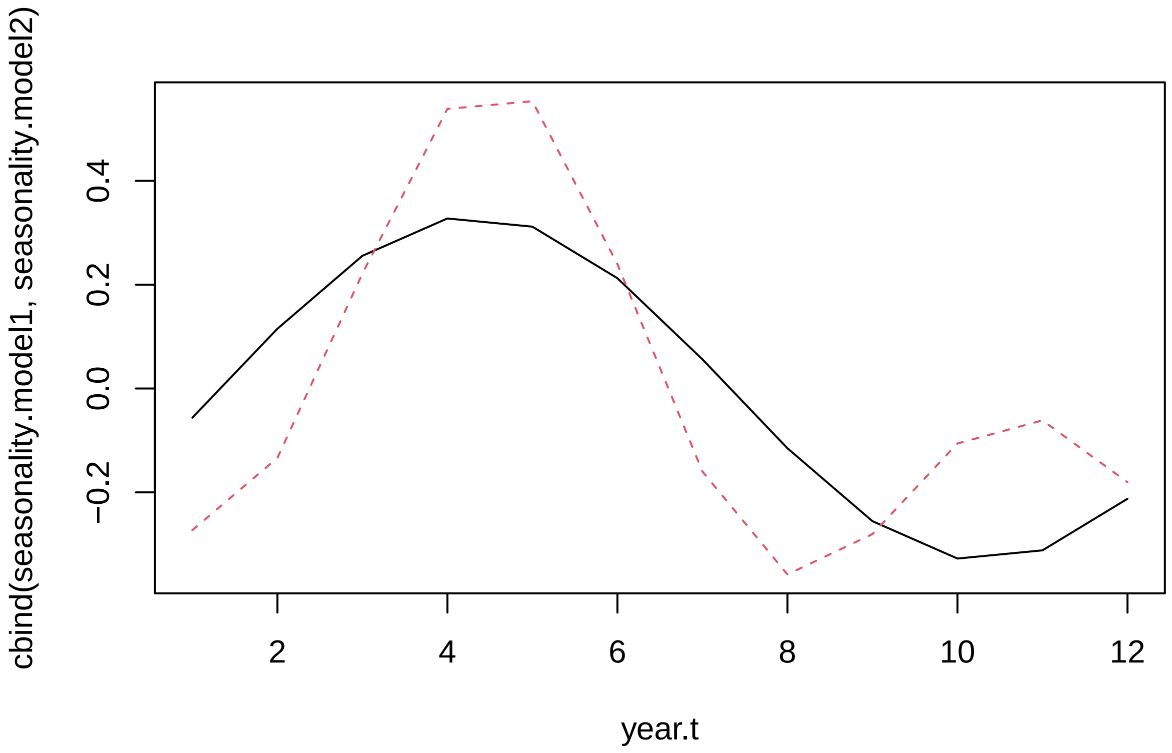
1. Plot the minimum temperature variable (`minT`) against time. What patterns do you observe?
2. Write down the equation of a linear regression model that accounts for a seasonal trend with a period of year. Fit the model defined in the previous point and add the curve of predicted values to the plot generate in the first point.
3. Add another sinusoidal function with period of 6 months to model fitted in the previous point. Generate the following plots: 1) one that displays both the seasonal trends estimated from the first model with a single sinusoidal functions with a one-year period, and the second model that add the sinusoidal curve with a six-month period; 2) the autocorrelgram based on the residuals from the two models. Based on these plots, which model do you believe to the best one?
4. Repear points 1 to 3, for `maxT` and `Rain`.

```
rm(list=ls())

mal <- read.csv("Kericho.csv")
mal <- mal[complete.cases(mal),]

mal$Month <- factor(mal$Month,levels =
                      c("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep",
                        "Oct","Nov","Dec"),ordered=TRUE)
mal$t <- as.numeric(mal$Month)+12*(mal$Year-1979)
```

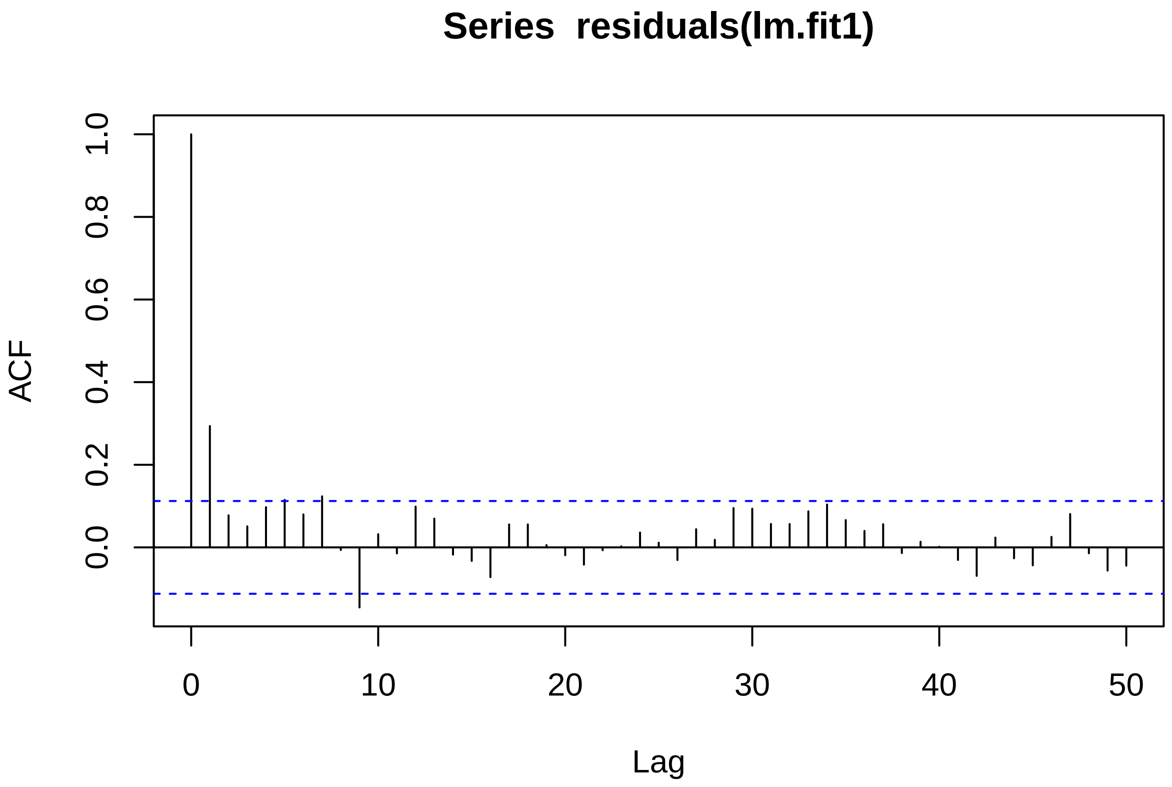
```r
plot(mal$t,mal$minT,type="l")
lm.fit1 <- lm(minT~sin(2*pi*t/12)+cos(2*pi*t/12),data=mal)

lm.fit2 <- lm(minT~sin(2*pi*t/12)+cos(2*pi*t/12)+
                  sin(2*pi*t/6)+cos(2*pi*t/6),data=mal)

lines(mal$t,predict(lm.fit1),col=2)
lines(mal$t,predict(lm.fit2),col=3)
```



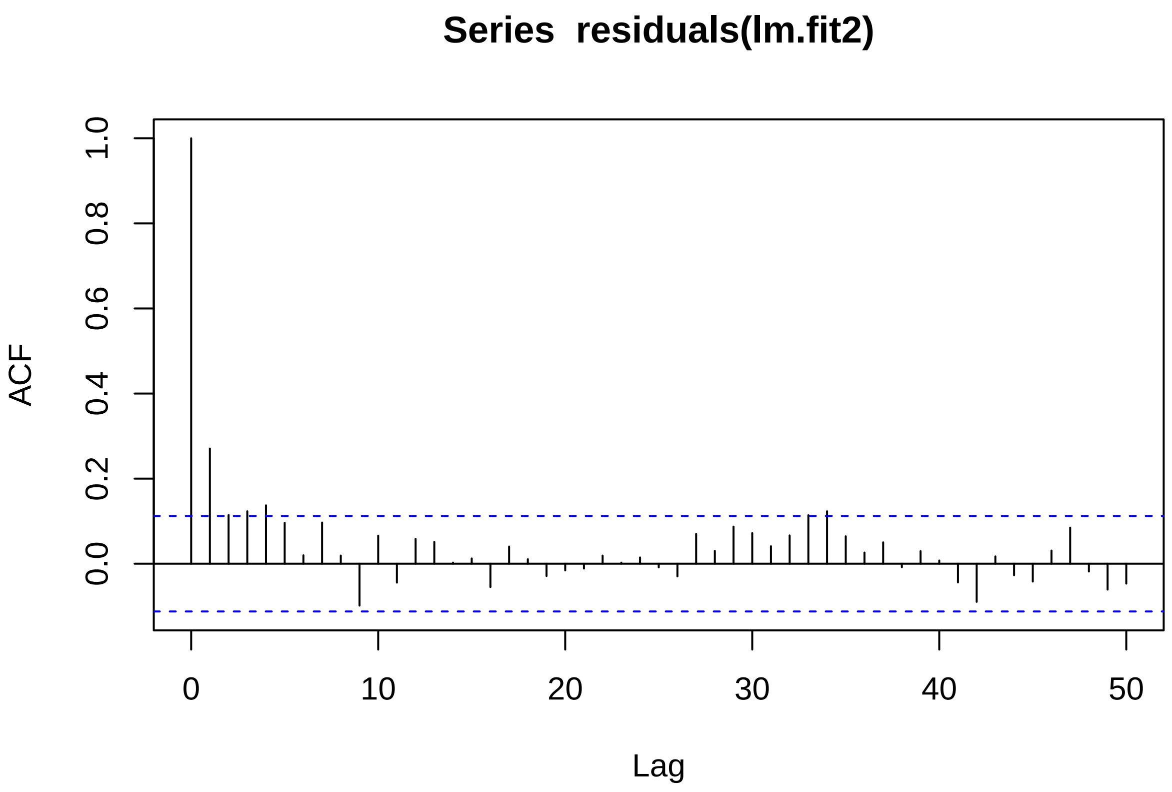
```r
beta.hat1 <- coef(lm.fit1)
beta.hat2 <- coef(lm.fit2)

year.t <- 1:12

seasonality.model1 <-  beta.hat1[2]*sin(2*pi*year.t/12)+beta.hat1[3]*cos(2*pi*year.t/12)
seasonality.model2 <- beta.hat2[2]*sin(2*pi*year.t/12)+beta.hat2[3]*cos(2*pi*year.t/12)+
               beta.hat2[4]*sin(2*pi*year.t/6)+beta.hat2[5]*cos(2*pi*year.t/6)
matplot(year.t,
     cbind(seasonality.model1,seasonality.model2),type="l")
```



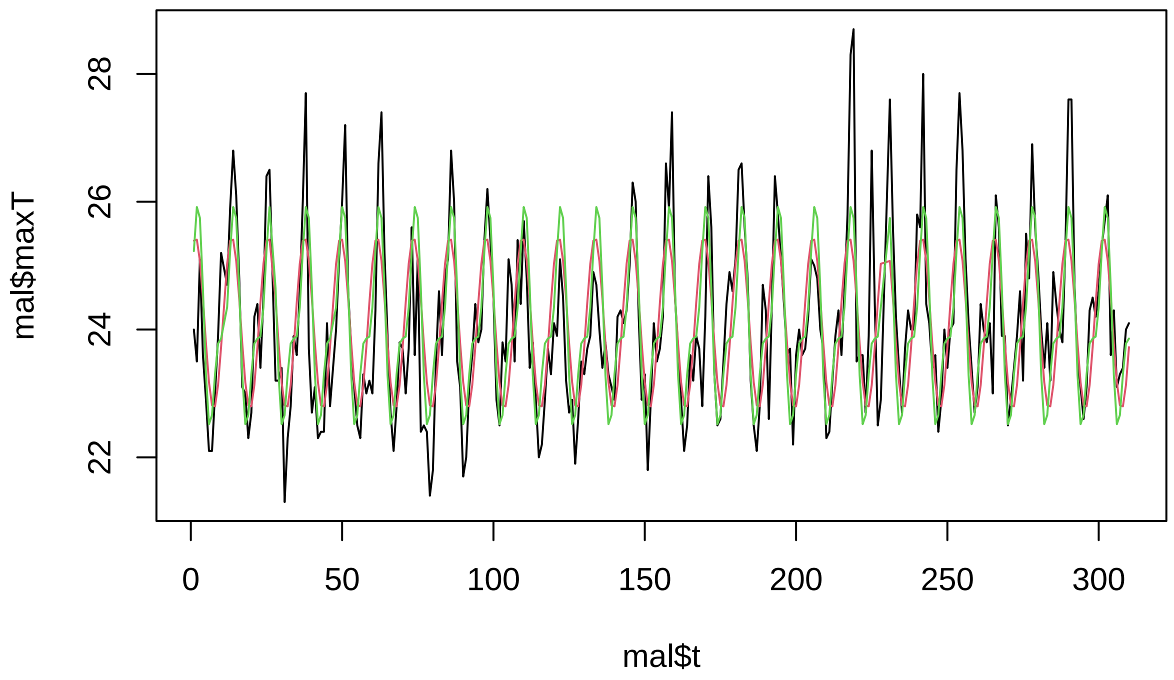
```
acf(residuals(lm.fit1),lag.max = 50)
```



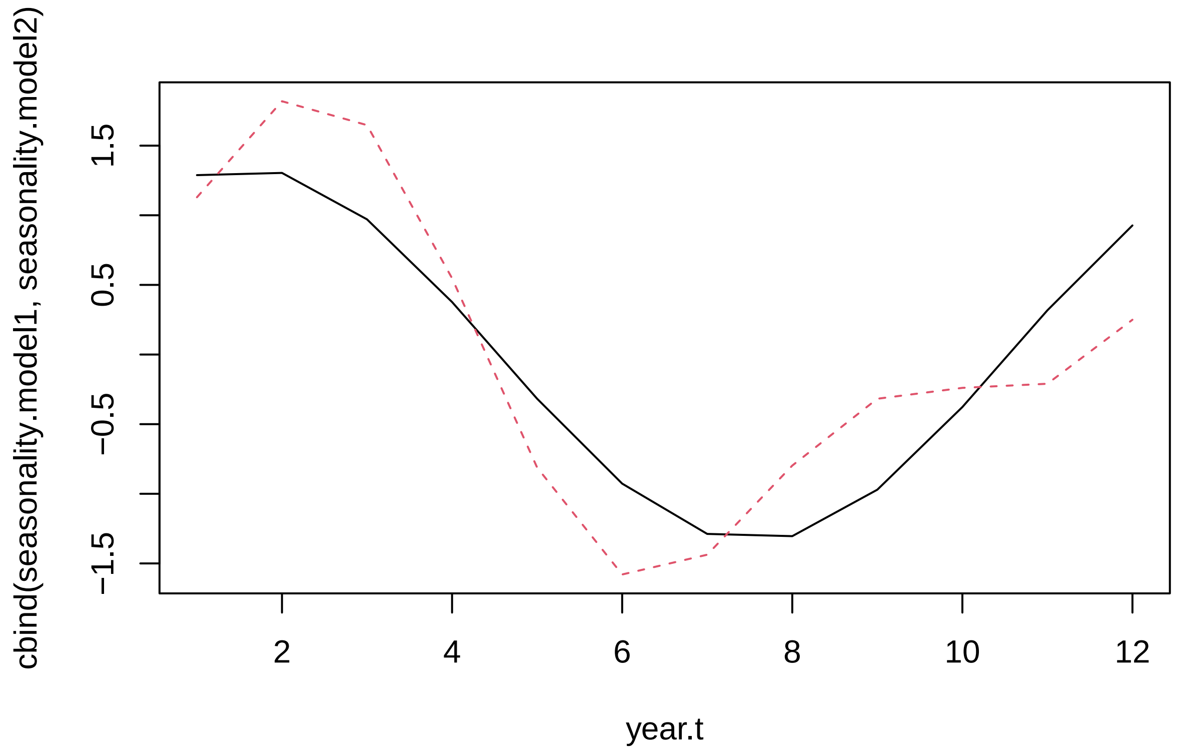
```
acf(residuals(lm.fit2),lag.max = 50)
```



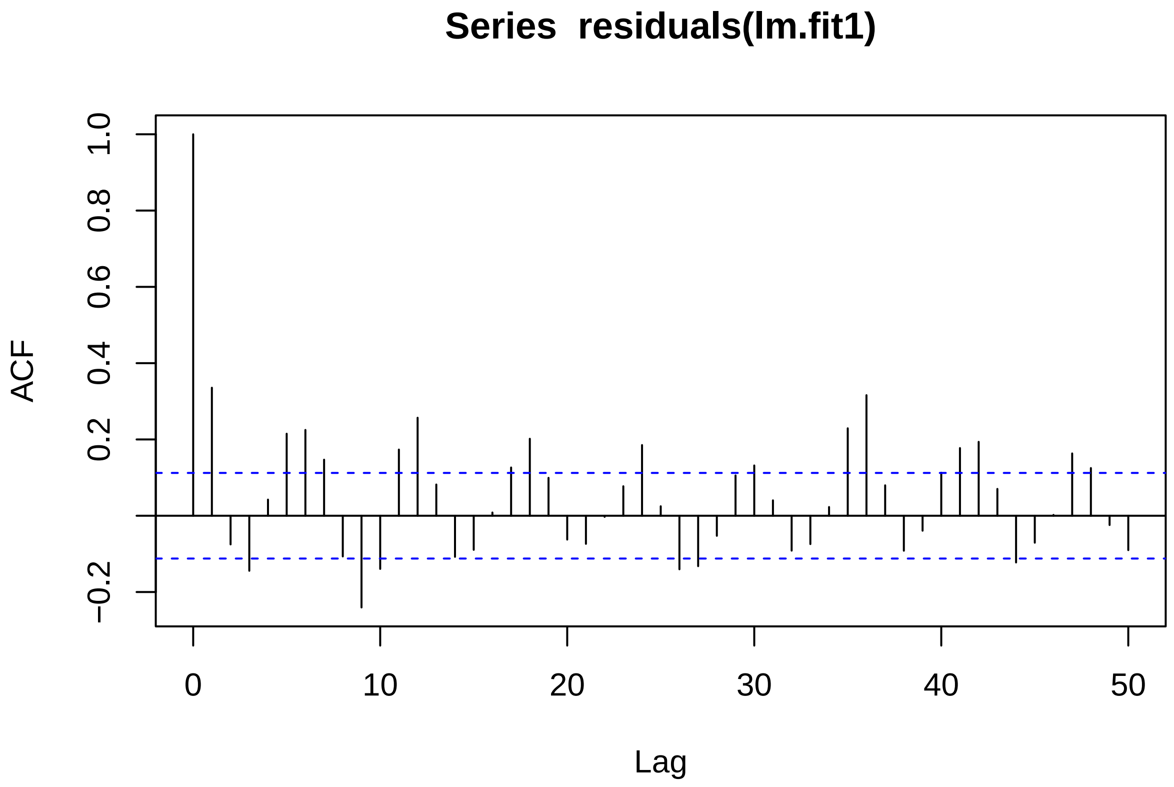
```
####
plot(mal$t,mal$maxT,type="l")
lm.fit1 <- lm(maxT~sin(2*pi*t/12)+cos(2*pi*t/12),data=mal)

lm.fit2 <- lm(maxT~sin(2*pi*t/12)+cos(2*pi*t/12)+
                sin(2*pi*t/6)+cos(2*pi*t/6),data=mal)

lines(mal$t,predict(lm.fit1),col=2)
lines(mal$t,predict(lm.fit2),col=3)
```



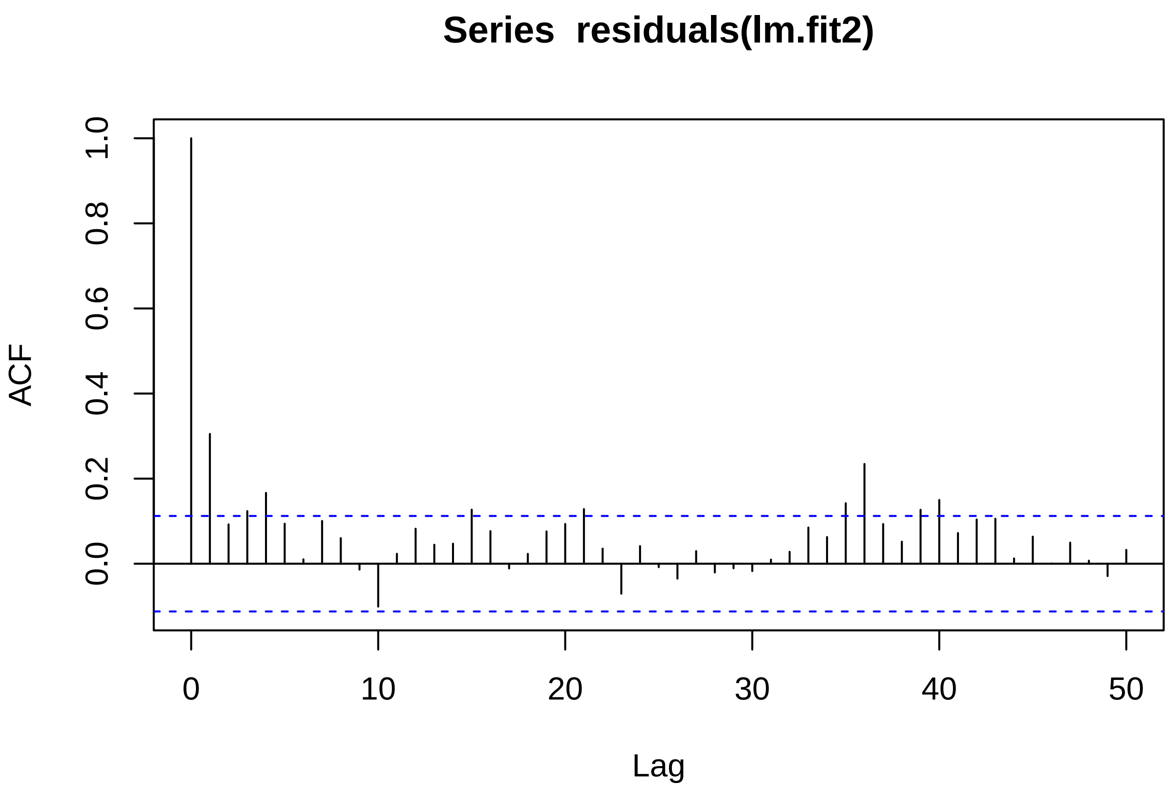
```r
beta.hat1 <- coef(lm.fit1)
beta.hat2 <- coef(lm.fit2)

year.t <- 1:12

seasonality.model1 <-  beta.hat1[2]*sin(2*pi*year.t/12)+beta.hat1[3]*cos(2*pi*year.t/12)
seasonality.model2 <- beta.hat2[2]*sin(2*pi*year.t/12)+beta.hat2[3]*cos(2*pi*year.t/12)+
  beta.hat2[4]*sin(2*pi*year.t/6)+beta.hat2[5]*cos(2*pi*year.t/6)
matplot(year.t,
        cbind(seasonality.model1,seasonality.model2),type="l")
```



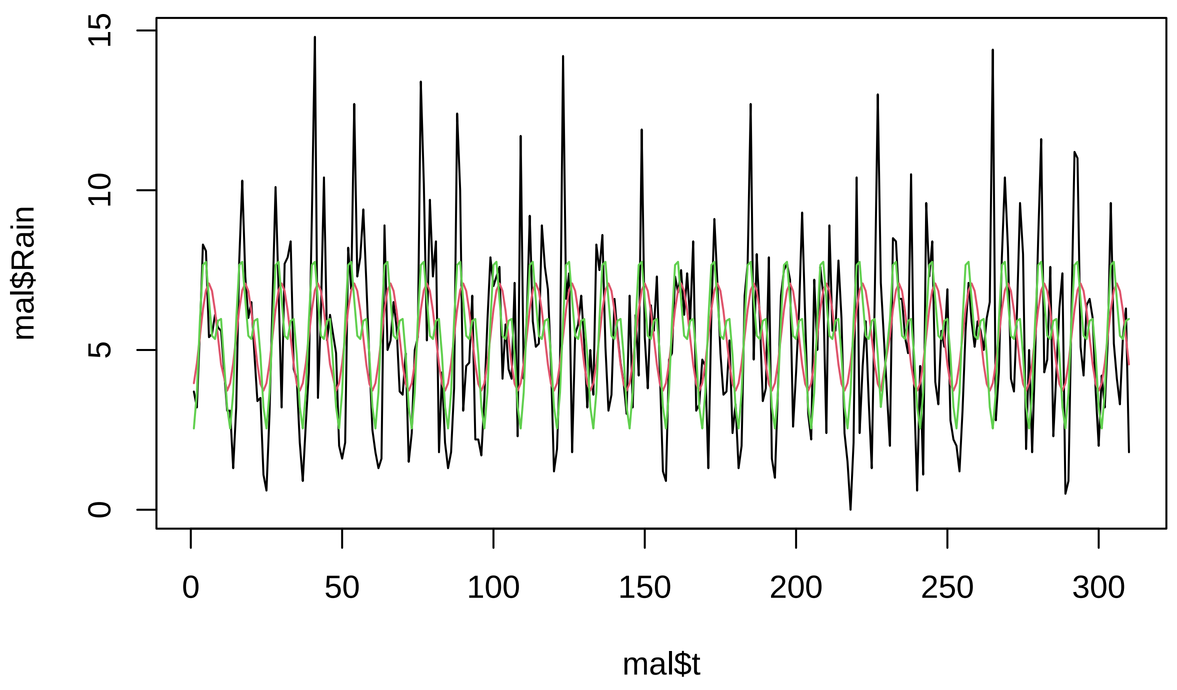
```
acf(residuals(lm.fit1),lag.max = 50)
```



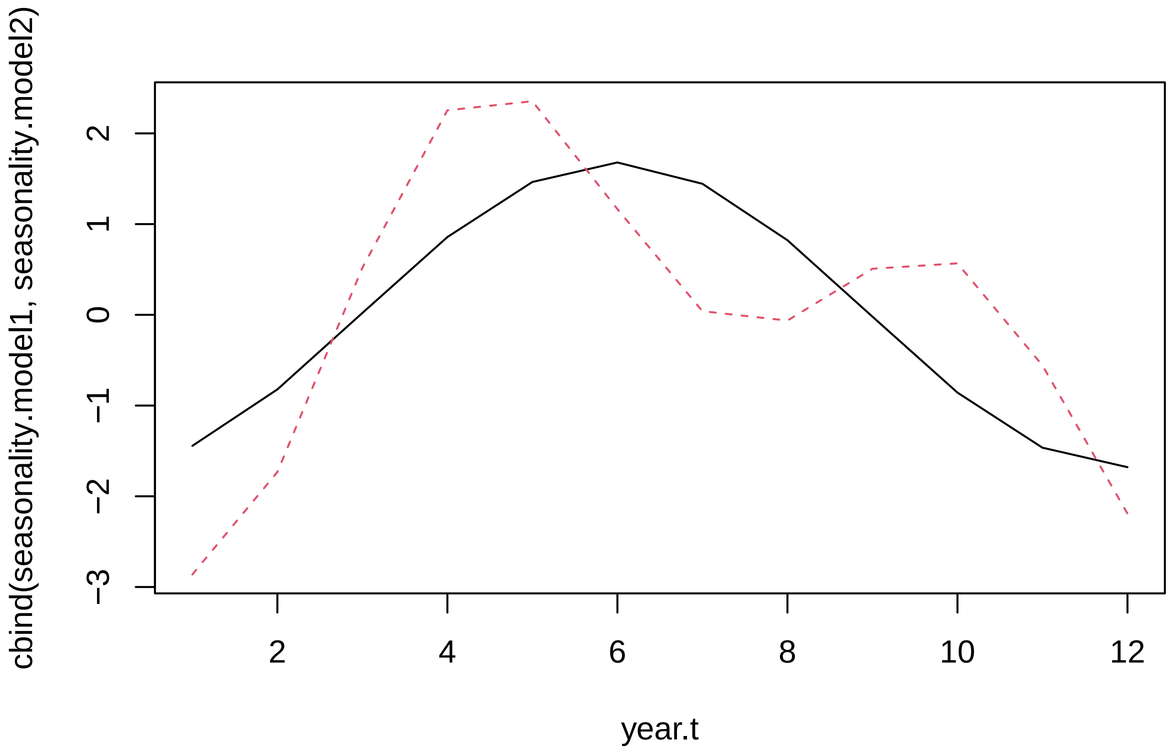
```
acf(residuals(lm.fit2),lag.max = 50)
```



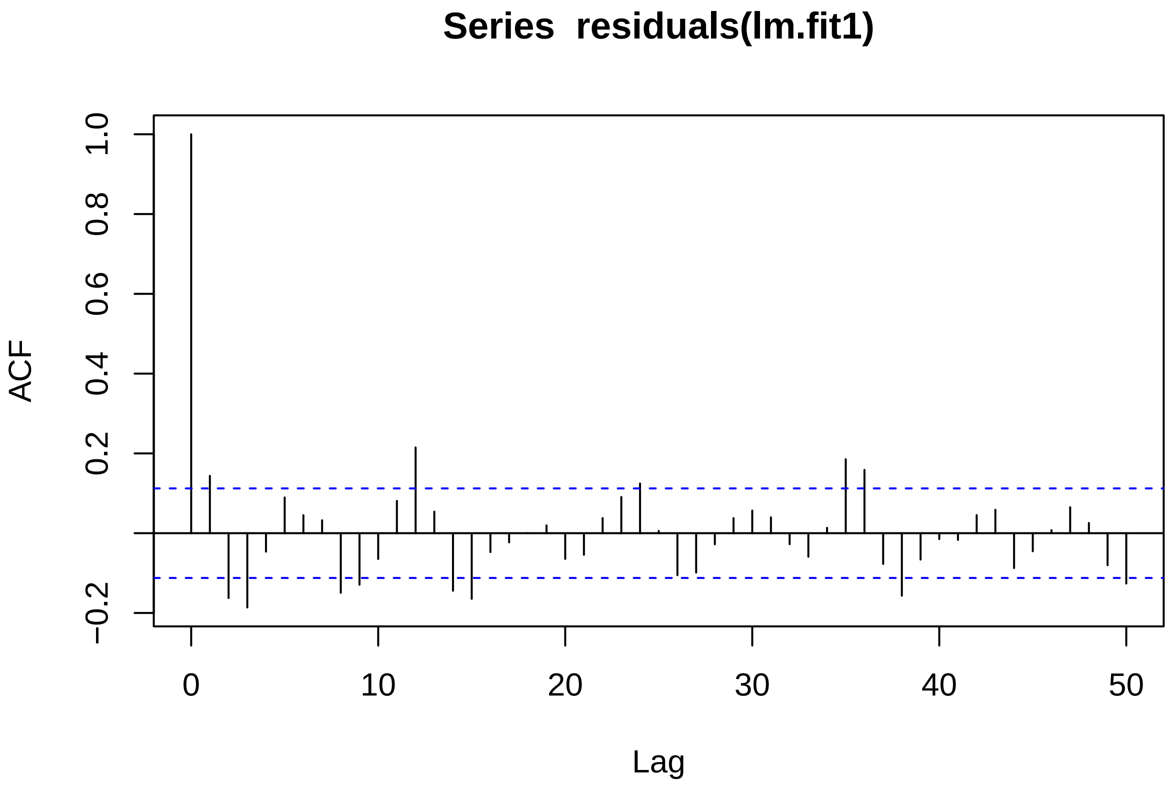
```
####
plot(mal$t,mal$Rain,type="l")
lm.fit1 <- lm(Rain~sin(2*pi*t/12)+cos(2*pi*t/12),data=mal)

lm.fit2 <- lm(Rain~sin(2*pi*t/12)+cos(2*pi*t/12)+
                sin(2*pi*t/6)+cos(2*pi*t/6),data=mal)

lines(mal$t,predict(lm.fit1),col=2)
lines(mal$t,predict(lm.fit2),col=3)
```



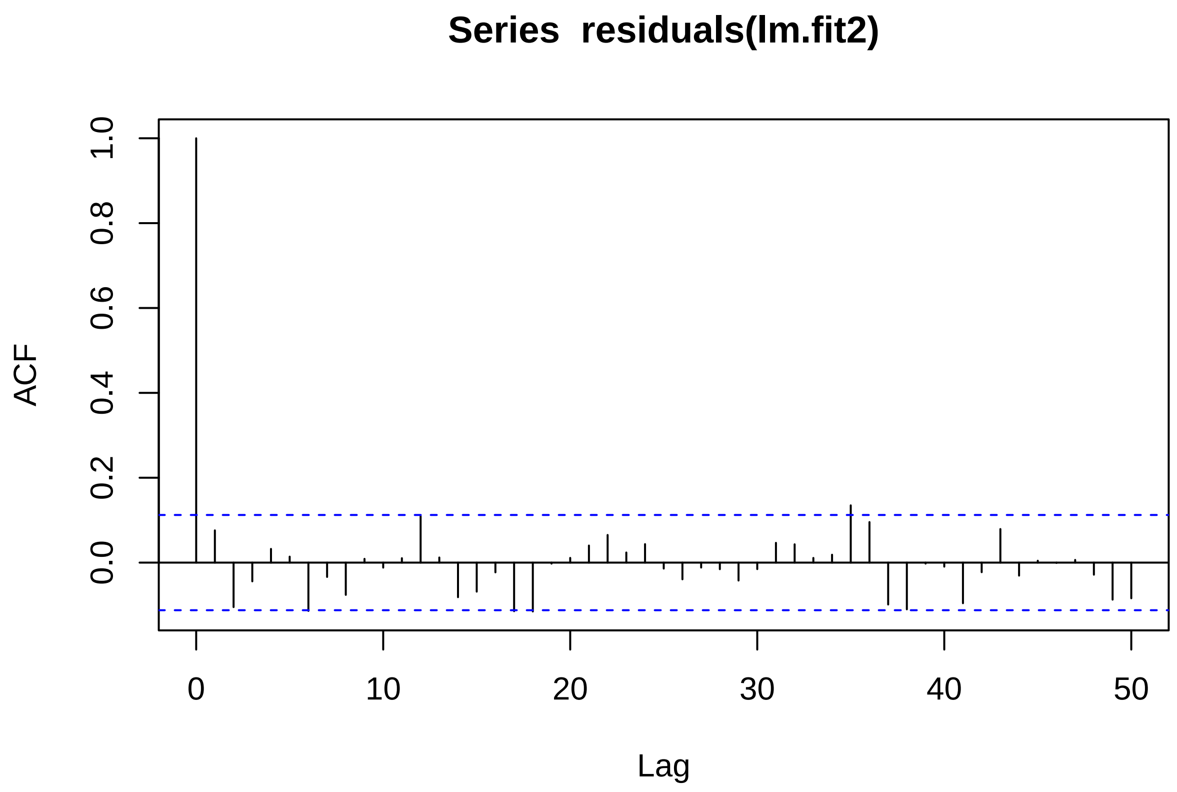
```r
beta.hat1 <- coef(lm.fit1)
beta.hat2 <- coef(lm.fit2)

year.t <- 1:12

seasonality.model1 <-  beta.hat1[2]*sin(2*pi*year.t/12)+beta.hat1[3]*cos(2*pi*year.t/12)
seasonality.model2 <- beta.hat2[2]*sin(2*pi*year.t/12)+beta.hat2[3]*cos(2*pi*year.t/12)+
  beta.hat2[4]*sin(2*pi*year.t/6)+beta.hat2[5]*cos(2*pi*year.t/6)
matplot(year.t,
        cbind(seasonality.model1,seasonality.model2),type="l")
```



```
acf(residuals(lm.fit1),lag.max = 50)
```

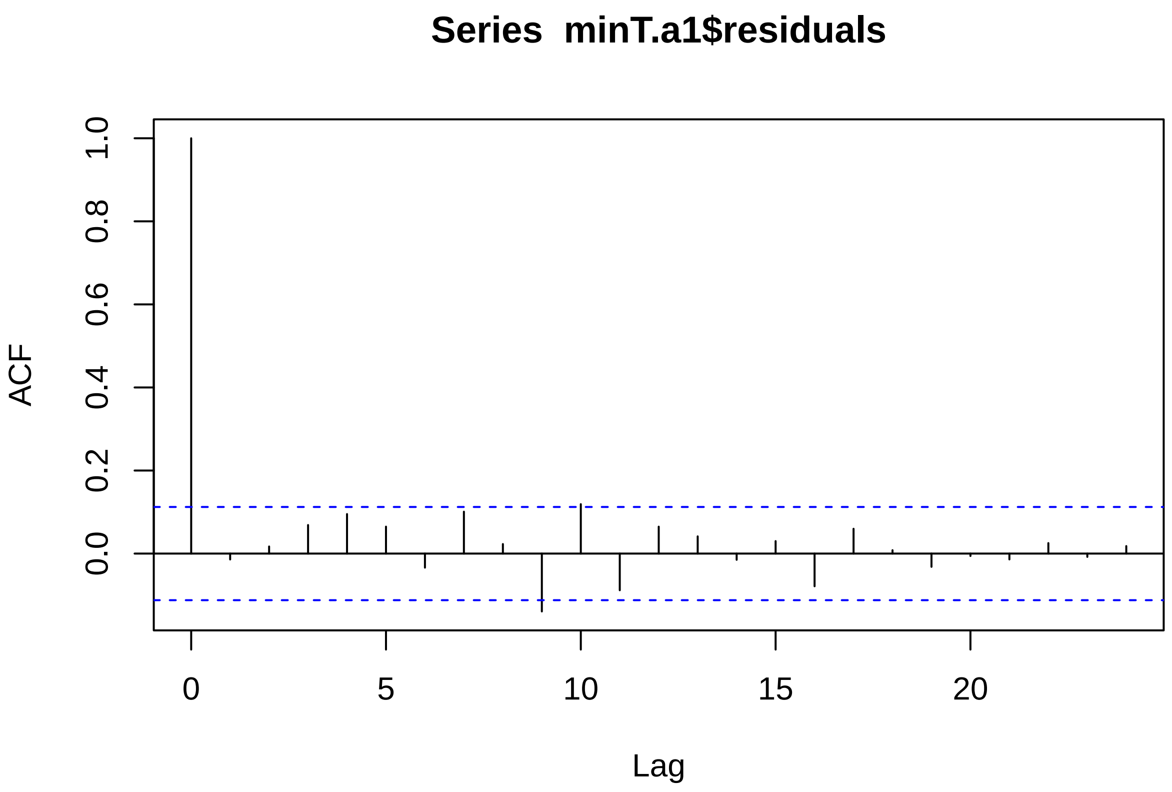


```
acf(residuals(lm.fit2),lag.max = 50)
```

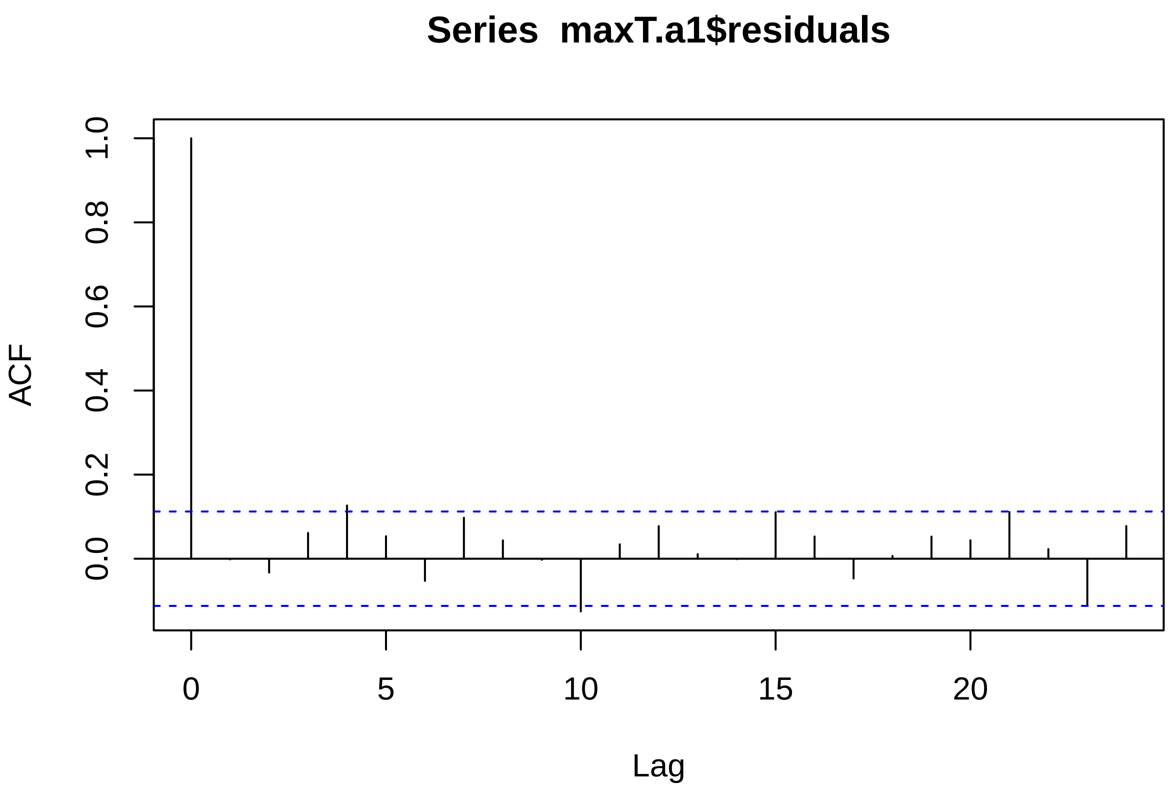


4. For all the models fitted in the previous points, fit an autoregressive process of the first order. Do the residuals from these models show any further evidence of residual temporal correlation?

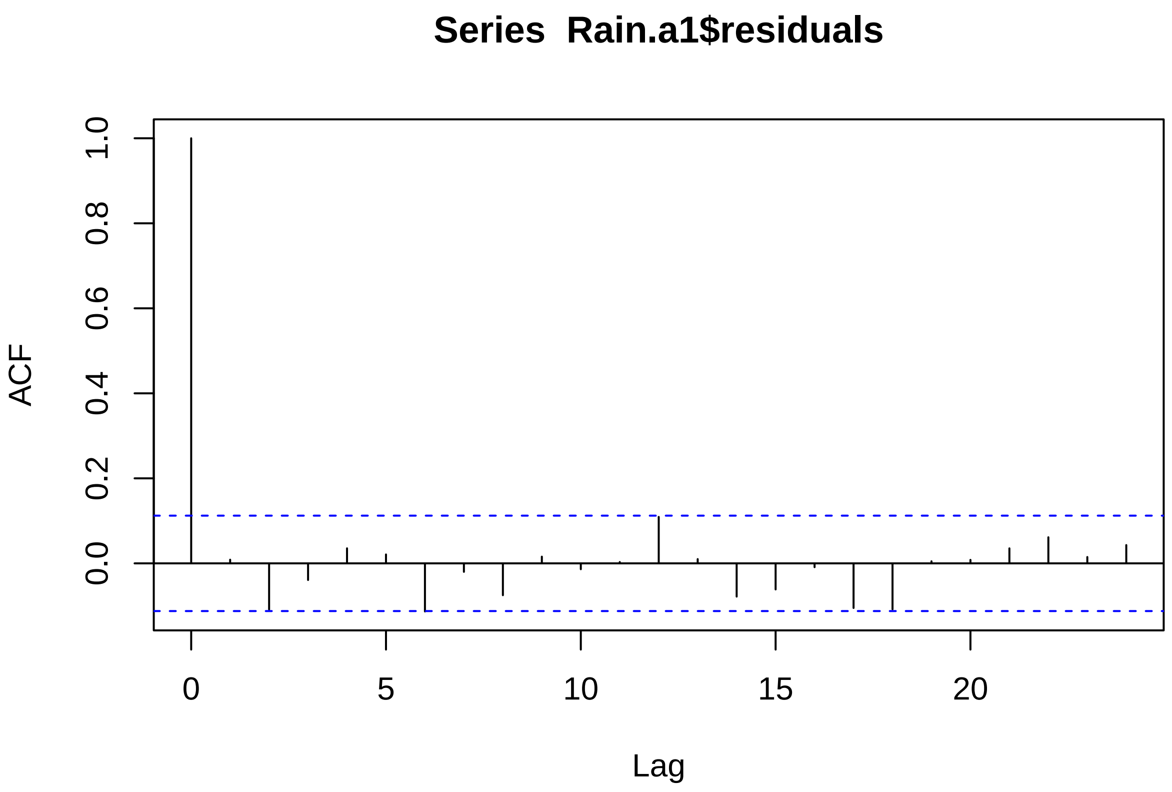
```
# minT
lm.fit.minT <- lm(minT~sin(2*pi*t/12)+cos(2*pi*t/12)+
                  sin(2*pi*t/6)+cos(2*pi*t/6),data=mal,
                  x=TRUE)
minT.a1 <- arima0(x=mal$minT,
       order=c(1,0,0),
       xreg = lm.fit.minT$x[,-1])

acf(minT.a1$residuals)
```



```
# maxT
lm.fit.maxT <- lm(maxT~sin(2*pi*t/12)+cos(2*pi*t/12)+
                  sin(2*pi*t/6)+cos(2*pi*t/6),data=mal,
                  x=TRUE)
maxT.a1 <- arima0(x=mal$maxT,
       order=c(1,0,0),
       xreg = lm.fit.minT$x[,-1])

acf(maxT.a1$residuals)
```



```
# Rain
lm.fit.Rain <- lm(Rain~sin(2*pi*t/12)+cos(2*pi*t/12)+
                  sin(2*pi*t/6)+cos(2*pi*t/6),data=mal,
                  x=TRUE)
Rain.a1 <- arima0(x=mal$Rain,
       order=c(1,0,0),
       xreg = lm.fit.Rain$x[,-1])

acf(Rain.a1$residuals)
```



5. Write your own likelihood function in R for fitting an autoregressive process of the first order. Then use your own function to obtain the parameter estimates in the previous point and compare them to those obtained using the `arima` function.

```
# y: data of the outcome to model
# D: matrix of covariates (use x=TRUE to obtain this from the linear model)
mle <- function(y,D) {
  p <- ncol(D) # number of covariates (including intercept)
  n <- nrow(D) # number of observations

  llik <- function(par) {
    beta <- par[1:p]
    phi <- 2*exp(par[p+1])/(1+exp(par[p+1]))-1
    sigma2 <- exp(par[p+2])

    stat.var <- sigma2/(1-phi^2)
    mu.t <- D%*%beta

    llik <- rep(NA,n)

    # Write the log-density function of Y1
    llik[1] <- dnorm(y[1],mean = mu.t[1],
                     sd=sqrt(stat.var),log=TRUE)

    for(i in 2:n) {

      # Write the log-density function of Yt given Yt-1
      llik[i] <- dnorm(y[i],mean = mu.t[i]+phi*(y[i-1]-mu.t[i-1]),
                     sd=sqrt(sigma2),log=TRUE)
```

```r
    }
    sum(llik)
  }
  estim <- nlminb(rep(0,p+2),
                  function(x) -llik(x))
  estim$par[p+1] <- 2*exp(estim$par[p+1])/(1+exp(estim$par[p+1]))-1
  estim$par[p+2] <- exp(estim$par[p+2])
  names(estim$par) <- c(colnames(D),"phi","sigma2")
  estim
}

estim.minT <- mle(y=mal$minT,D=lm.fit.minT$x)
estim.minT$par
```

```
##        (Intercept) sin(2 * pi * t/12) cos(2 * pi * t/12)  sin(2 * pi * t/6)
##         10.9549992          0.2505165         -0.2050617         -0.2620755
##  cos(2 * pi * t/6)                phi             sigma2
##          0.0328837          0.2729554          0.3608053
```

```r
minT.a1
```

```
##
## Call:
## arima0(x = mal$minT, order = c(1, 0, 0), xreg = lm.fit.minT$x[, -1])
##
## Coefficients:
##          ar1  intercept  sin(2 * pi * t/12)  cos(2 * pi * t/12)
##       0.2729    10.9546              0.2506             -0.2059
## s.e.  0.0553     0.0473              0.0624              0.0626
##       sin(2 * pi * t/6)  cos(2 * pi * t/6)
##                 -0.2617             0.0332
## s.e.             0.0542             0.0539
##
## sigma^2 estimated as 0.3608:  log likelihood = -277.35,  aic = 568.71
```

Exercise 3

The R function below allows to generate a time series for a set of equally spaces observation time points, between a given starting point (`start`) and end point (`end`).

```r
# start: first point in time of observation
# end: last point in time of observation
# mu: the mean of the outcome to simulate
# sigma2: the variance of the Matern process
# phi: the scale of the temporal correlation of the Matern process
# kappa: smothness parameter of the Matern process
# tau2: variance of the noise term

simulate.matern <- function(start,end,
                            mu,
                            sigma2,phi,kappa,
                            tau2) {
  t.set <- seq(start,end,by=1)
  n <- length(t.set)
  library(geoR)
  Sigma <- sigma2*matern(as.matrix(dist(t.set)),phi=phi,kappa=kappa)
  diag(Sigma) <- diag(Sigma)+tau2
  out <- as.numeric(mu+t(chol(Sigma))%*%rnorm(n))
  out
}
```

The function uses the following model to generate a time serie.

$$Y(t) = \mu(t) + S(t) + Z(t)$$

where $\mu(t)$ is the mean component of the model that includes the effects of covariates; $S(t)$ is a Matern process with variance σ^2, scale parameter ϕ and smoothness parameter κ; the random variable $Z(t)$ is Gaussian noise with variance τ^2.

1. After setting $\mu = 0$, $\sigma^2 = 1$, $\kappa = 0.5$ and $\tau^2 = 0$, choose a suitable value of ϕ to simulate three processes with a correlation between two consecutive observations of 0.5, 0.75 and 0.98. For the three simulates time series, display the correlogram and the empirical variogram. How do these two change across the three simulated time series?

```
rm(list=ls())
source("auxiliary_function.R")
simulate.matern <- function(start,end,mu,sigma2,phi,kappa,tau2) {
  t.set <- seq(start,end,by=1)
  n <- length(t.set)
  library(geoR)
  Sigma <- sigma2*matern(as.matrix(dist(t.set)),phi=phi,kappa=kappa)
  diag(Sigma) <- diag(Sigma)+tau2
  out <- as.numeric(mu+t(chol(Sigma))%*%rnorm(n))
  out
}


######## Point 1 (kappa=0.5)
# 0.5
phi <- -1/log(0.5)
exp(-1/phi)
y_0.5 <- simulate.matern(start=1,end=100,
                  mu=0,
                  sigma2=1,
                  phi=phi,
                  tau2=0,
                  kappa=0.5)
acf(y_0.5)
vari_0.5 <- vari.time(time=1:100,data=y_0.5,
                      uvec=seq(1,20,length=15))
plot(vari_0.5,type="l")

# 0.75
phi <- -1/log(0.75)
y_0.75 <-
  simulate.matern(start=1,end=192,
                  mu=0,
                  sigma2=1,
                  phi=phi,
                  tau2=0,
                  kappa=0.5)
```

```r
acf(y_0.75)
vari_0.75 <- vari.time(time=1:100,data=y_0.75,
                      uvec=seq(1,20,length=15))
plot(vari_0.75,type="l")

# 0.98
phi <- -1/log(0.98)
y_0.98<-
  simulate.matern(start=1,end=192,
                  mu=0,
                  sigma2=1,
                  phi=phi,
                  tau2=0,
                  kappa=0.5)
acf(y_0.98)
vari_0.98 <- vari.time(time=1:100,data=y_0.98,
                      uvec=seq(1,20,length=15))
plot(vari_0.98,type="l")
```

2. Repeat the previous point, for $\kappa = 1.5$ and keep all other parameters unchanged.

```r
rm(list=ls())
source("auxiliary_function.R")

simulate.matern <- function(start,end,mu,sigma2,phi,kappa,tau2) {
  t.set <- seq(start,end,by=1)
  n <- length(t.set)
  library(geoR)
  Sigma <- sigma2*matern(as.matrix(dist(t.set)),phi=phi,kappa=kappa)
  diag(Sigma) <- diag(Sigma)+tau2
  out <- as.numeric(mu+t(chol(Sigma))%*%rnorm(n))
  out
}
######## Point 2 (kappa=1.5)
# 0.5
phi <- uniroot(function(x) matern(1,x,1.5)-0.5,
               lower=0.1,upper=10)$root
matern(1,phi,1.5)

y_0.5 <- simulate.matern(start=1,end=100,
                         mu=0,
                         sigma2=1,
                         phi=phi,
                         tau2=0,
                         kappa=1.5)
plot(y_0.5,type="l")
acf(y_0.5)
vari_0.5 <- vari.time(time=1:100,data=y_0.5,
                      uvec=seq(1,20,length=15))
plot(vari_0.5,type="l")

# 0.75
phi <- uniroot(function(x) matern(1,x,1.5)-0.75,
               lower=0.1,upper=10)$root
```

```
matern(1,phi,1.5)

y_0.75 <-
  simulate.matern(start=1,end=192,
                  mu=0,
                  sigma2=1,
                  phi=phi,
                  tau2=0,
                  kappa=1.5)
acf(y_0.75)
vari_0.75 <- vari.time(time=1:100,data=y_0.75,
                       uvec=seq(1,20,length=15))
plot(vari_0.75,type="l")

# 0.98
phi <- uniroot(function(x) matern(1,x,1.5)-0.98,
               lower=0.1,upper=10)$root
matern(1,phi,1.5)

y_0.98 <-
  simulate.matern(start=1,end=100,
                  mu=0,
                  sigma2=1,
                  phi=phi,
                  tau2=0,
                  kappa=1.5)
acf(y_0.98)
vari_0.98 <- vari.time(time=1:100,data=y_0.98,
                       uvec=seq(1,30,length=15))
plot(vari_0.98,type="l")

plot(y_0.5,type="l")
plot(y_0.98,type="l")
```

3. **Challenge question.** For this point, we shall consider the CO2 data-set analyzed in the lectures (`muana_loa.csv`). Consider the following model for the data.

$$\begin{aligned} Y(t) &= \beta_0 + \beta_1 t + \beta_2 \sin(2\pi t/12) + \beta_3 \sin(2\pi t/12) + \\ &\quad \beta_4 \sin(2\pi t/6) + \beta_5 \sin(2\pi t/6) + \\ &\quad S(t) + Z(t). \end{aligned}$$

   After estimating the regression coefficients, from β_0 to β_5, using a standard regression linear model (i.e. one that completely ignores temporal correlation), use the `simulate.matern` function to simulate a CO2 time series by setting $\mu(t)$ to the mean of the process defined in the equation above. Make your own choice for the other parameters of the simulation. After simulating this time series, obtain the maximum likelihood estimates using the simulated data and compare these to the values that you have chosen.

```
rm(list=ls())
simulate.matern <- function(start,end,
                            mu,
                            sigma2,phi,kappa,
                            tau2) {
  t.set <- seq(start,end,by=1)
  n <- length(t.set)
  library(geoR)
  Sigma <- sigma2*matern(as.matrix(dist(t.set)),phi=phi,kappa=kappa)
  diag(Sigma) <- diag(Sigma)+tau2
  out <- as.numeric(mu+t(chol(Sigma))%*%rnorm(n))
  out
}

co2 <- read.csv("mauna_loa.csv")
co2$t <- co2$Month+12*(co2$Year-1965)

lm.fit <- lm(CO2 ~ t +
                sin(2*pi*t/12)+cos(2*pi*t/12)+
                sin(2*pi*t/6)+cos(2*pi*t/6),
              data=co2,x=TRUE)

co2$CO2.sim <-
simulate.matern(start=1,end=192,
                mu=predict(lm.fit),
                sigma2=1,
                phi=3,
                tau2=0.5,
                kappa=0.5)
plot(co2$t,co2$CO2.sim,type="l")

source("auxiliary_function.R")
fit.co2.sim <-
fit.matern(form = CO2.sim~t +
            sin(2*pi*t/12)+cos(2*pi*t/12)+
            sin(2*pi*t/6)+cos(2*pi*t/6),
           data=co2,
           time="t",
           kappa=0.5,
           start.cov.pars=c(1,1),
           method="nlminb")

coef(fit.co2.sim)
```

Exercise 4 \

Consider the following time series model for malaria cases reported in the Kericho data.

$$\begin{aligned} Y(t) \quad = \quad & \beta_0 + \beta_1 t + \beta_2 \max\{t-50,0\} + \beta_3 I(t>225) + \\ & \beta_4 \texttt{minT}(t-2) + \beta_5 \texttt{maxT}(t-2) + \beta_5 \texttt{Rain}(t-2) + \\ & W(t) + Z(t) \end{aligned} \tag{1}$$

where: $I(t > 225)$ is a dummy indicator variable that takes value 1 if the time identifies is larger than 225; $\texttt{minT}(t-2)$, $\texttt{maxT}(t-2)$ and $\texttt{Rain}(t-2)$ are the minimum and maximum temperature, and rain, each lagged of 2 months; $W(t)$ is a Matern process with $\kappa = 2.5$; $Z(t)$ is Guassian noise.

1. Fit the model in equation (1) and report the confidence intervals for the model parameters. For σ^2, ϕ and τ^2, obtain these on the log-scale and exponentiate them.
2. Based on the fitted model in the previous point, generate a plot that shows the point predictions and

95% prediction intervals for the log-Cases from the first to the last month of cases reported in the data-set.

3. Using a hold-out sample consisting of the last year of reported cases, use the the remaining data as a training data set and compute the mean-square prediction error and bias for the holdout data-set.
4. Repeat the previous point for a model that replaces of `minT`$(t-2)$, `maxT`$(t-2)$ and `Rain`$(t-2)$, with sinusoidal with frequencies of 1/12 and 1/6. Are there tangible differences in terms of mean square prediction error and bias between the two models?

```
rm(list=ls())

source("auxiliary_function.R")
mal <- read.csv("Kericho.csv")

mal$Month <- factor(mal$Month,
                    levels=c("Jan","Feb","Mar","Apr","May","Jun","Jul",
                             "Aug","Sep","Oct","Nov","Dec"),ordered = TRUE)
mal$t <- as.numeric(mal$Month) + 12*(mal$Year-1979)

### Point 1

create.lag.var <- function(x,lag) {
  tlag <- mal$t-lag
  ind.lag <- sapply(tlag,function(x) {
    out <- which((mal$t-x)==0)
    if(length(out)==0) out <- NA
    return(out)
  })
  x[ind.lag]
}


mal$minT.lag2 <-create.lag.var(mal$minT,lag=2)
mal$maxT.lag2 <-create.lag.var(mal$maxT,lag=2)
mal$Rain.lag2 <-create.lag.var(mal$Rain,lag=2)

mal <- mal[complete.cases(mal),]


matern.fit <-
  fit.matern(form = log(Cases)~t + I((t>50)*(t-50)) + I(t>225)+
               minT.lag2 + maxT.lag2 + Rain.lag2,
             data=mal,
             time="t",
             kappa=2.5,
             start.cov.pars=c(1,1),
             method="nlminb")

get.CI <- function(fitted.model) {
  s.model <- summary(fitted.model)
  reg.coef <-
    cbind(s.model$coefficients[,1],
          s.model$coefficients[,1]-qnorm(0.975)*s.model$coefficients[,2],
          s.model$coefficients[,1]+qnorm(0.975)*s.model$coefficients[,2])
  cov.pars <-
    exp(
      cbind(s.model$cov.pars[,1],
            s.model$cov.pars[,1]-qnorm(0.975)*s.model$cov.pars[,2],
            s.model$cov.pars[,1]+qnorm(0.975)*s.model$cov.pars[,2]))

  tab <- rbind(reg.coef,cov.pars)
  rownames(tab)[(nrow(tab)-2):nrow(tab)] <- c("sigma^2","phi","tau^2")
  colnames(tab) <- c("Estimate","Lower limit","Upper limit")
  tab
}

get.CI(matern.fit)

### Point 2

pred <-
time.predict(matern.fit,
             predictors = data.frame(t=mal$t,
                                     minT.lag2=mal$minT.lag2,
                                     maxT.lag2=mal$maxT.lag2,
                                     Rain.lag2=mal$Rain.lag2),
             time.pred = mal$t,
             scale.pred = "linear")


matplot(pred$time.pred,cbind(pred$predictions,pred$quantiles),
        lty=c("solid","dashed","dashed"),type="l",col=1)
points(mal$t,log(mal$Cases),pch=20,col=2)

### Point 3

# year.holdout: is the number of years that are part of the holdout data-set
year.holdout <- 1
time.split <- max(mal$t)-year.holdout*12


# Identify the rows in the data-set that correspond to the training data-set
train <- which(mal$t<=time.split)
# Identify the rows in the data-set that correspond to the holdout data-set
holdout <- which(mal$t>time.split)


matern.fit.train <-
  fit.matern(form = log(Cases)~t + I((t>50)*(t-50)) + I(t>225)+
               minT.lag2 + maxT.lag2 + Rain.lag2,
             data=mal[train,],
             time="t",
             kappa=2.5,
             start.cov.pars=c(1,1),
             method="nlminb")

t.pred.holdout <- mal$t[holdout]
```

```r
pred.holdout <- time.predict(fitted.model=matern.fit.train,
                             predictors = data.frame(t=t.pred.holdout,
                                                     minT.lag2=mal$minT.lag2[holdout],
                                                     maxT.lag2=mal$maxT.lag2[holdout],
                                                     Rain.lag2=mal$Rain.lag2[holdout]),
                             time.pred = t.pred.holdout,
                                scale.pred = "linear")

matplot(pred.holdout$time.pred,
        cbind(pred.holdout$predictions,pred.holdout$quantiles),
        lty=c("solid","dashed","dashed"),type="l",col=1,
        ylab="log(Cases)",xlab="Time")
points(mal[holdout,]$t,log(mal[holdout,]$Cases),pch=20)

# Bias
mean((pred.holdout$predictions-log(mal[holdout,]$Cases)))
# Mean-square error
mean((pred.holdout$predictions-log(mal[holdout,]$Cases))^2)

# Point 4

matern.fit.with.sin <- fit.matern(form=
                       log(Cases) ~ t+I((t-50)*(t>50))+I(t>229)+
                       sin(2*pi*t/12)+cos(2*pi*t/12)+
                       sin(2*pi*t/6)+cos(2*pi*t/6),
                       time="t",
                       start.cov.pars = c(1,5),
                       kappa=2.5,
                       data=mal[train,], # NOTE: selecting the training data
                       method="nlminb")


pred.holdout.with.sin <- time.predict(fitted.model=matern.fit.with.sin,
                                predictors = data.frame(t=t.pred.holdout),
                                time.pred = t.pred.holdout,
                                scale.pred = "linear")


matplot(pred.holdout.with.sin$time.pred,
        cbind(pred.holdout.with.sin$predictions,pred.holdout.with.sin$quantiles),
        lty=c("solid","dashed","dashed"),type="l",col=1,
        ylab="log(Cases)",xlab="Time")
points(mal[holdout,]$t,log(mal[holdout,]$Cases),pch=20)


# Bias
mean((pred.holdout.with.sin$predictions-log(mal[holdout,]$Cases)))

# Mean-square error
mean((pred.holdout.with.sin$predictions-log(mal[holdout,]$Cases))^2)
```