hw1code

Han-Yuan Hsu

2022-09-09

```
set.seed(111)
```

3

Download the dataset on the annual size of the Resident Population of California from https://fred.stlouisfed.org/series/CAPOP. This dataset gives the annual population of California from 1900 to 2021 (units are in thousands of persons and there is no seasonal adjustment to this data). The goal of this exercise is to fit the linear trend model (1) to this dataset.

Note the unit for CAPOP is thousands of persons.

```
df <- read.csv('CAPOP.csv')
head(df)
```

```
##         DATE CAPOP
## 1 1900-01-01  1490
## 2 1901-01-01  1550
## 3 1902-01-01  1623
## 4 1903-01-01  1702
## 5 1904-01-01  1792
## 6 1905-01-01  1893
```

```
n = length(df$DATE) # number of timepoints, 122
```

a

Provide point estimates for β_0, β_1 along with appropriate uncertainty intervals. Interpret your point estimates and explain why they make sense. (4 points)

Our model is $Y_i = \beta_0 + \beta_1 t_i + Z_i$. The point estimates of β_0 and β_1 are $\hat{\beta}_0$ and $\hat{\beta}_1$, which are obtained from the lm function.

```
t <- 1:n
lin <- lm(df$CAPOP ~ 1 + t)
summary(lin)
```

```
##
## Call:
```

```
## lm(formula = df$CAPOP ~ 1 + t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3519.8 -1456.6  -209.5  1629.5  5488.0
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4357.912    395.791  -11.01   <2e-16 ***
## t             359.874      5.585   64.44   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2172 on 120 degrees of freedom
## Multiple R-squared:  0.9719, Adjusted R-squared:  0.9717
## F-statistic:  4152 on 1 and 120 DF,  p-value: < 2.2e-16
```

From the summary, we see that $\hat{\beta}_0$ is -4357.912 and $\hat{\beta}_1$ is 359.874. The negative intercept may be frowned upon because the predicted population of the first timepoint (t=1) will be -4357.912 + 359.874, which is negative and so does not make sense. But the regression line does capture the long-term increasing trend of the population, which is shown by a positive $\hat{\beta}_1$.

b

Along with a plot of the observed dataset, plot lines corresponding to 30 samples from the posterior distribution of (β_0, β_1). Comment on the range of uncertainty revealed in this plot. (4 points)

From lecture, we know the posterior (β_0, β_1) follows

$$t_{n-2}(\hat{\beta}, \frac{S(\hat{\beta})}{n-2}(X'X)^{-1}).$$

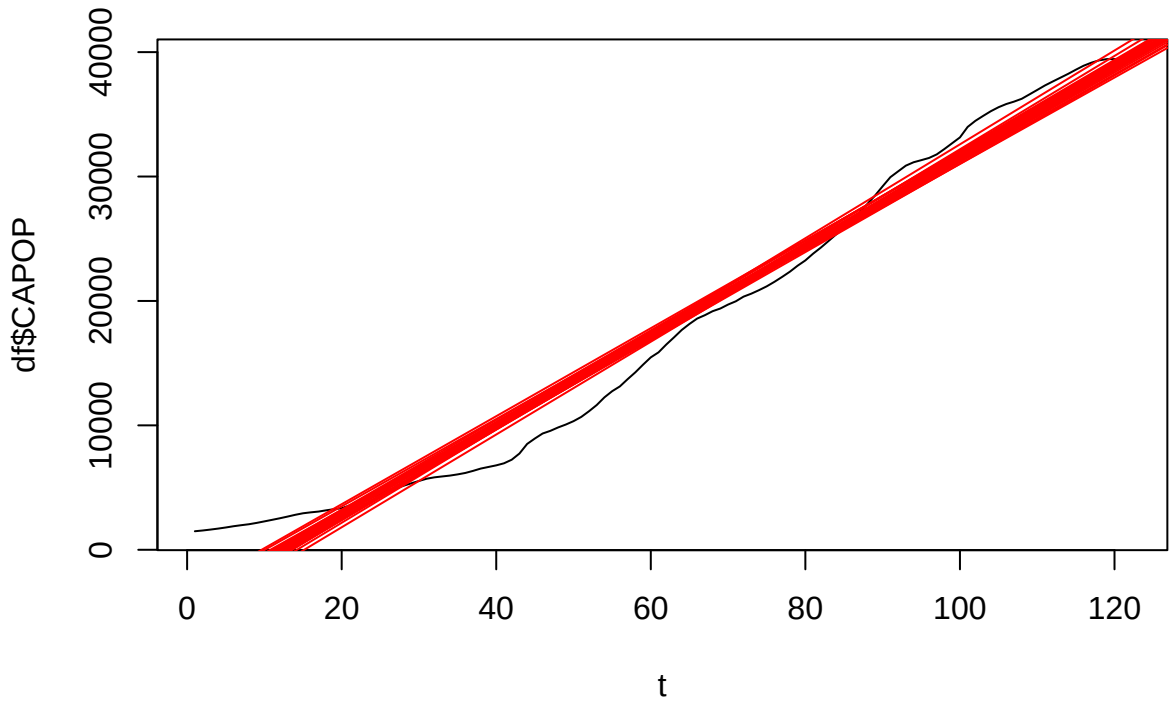
```
beta_hat = as.vector(lin$coefficients)
S = sum(lin$residuals^2) # residual sum of squares
X = as.matrix(cbind(rep(1, n), t))
Sigma = S / (n-2) * solve(t(X) %*% X)
```

The following function gives you N independent samples from the multivariate t distribution with specified mu, Sigma, and df (degrees of freedom):

```
library(mvtnorm)
get_random_t <- function(N, mu, Sigma, df) {
  # get N samples
  #d = length(mu) # dimension
  rbind(mu)[rep(1, N), ] + rmvnorm(N, sigma = Sigma)/sqrt(rchisq(N, df = df)/df)
}
```

Plot:

```
plot(t, df$CAPOP, type='l')
t30 <- get_random_t(N=30, mu=beta_hat, Sigma=Sigma, df=n-2)
for(i in 1:30) {
  abline(a=t30[i,1], b=t30[i,2], col='red')
}
```



The uncertainty of the fitted lines look pretty small. The reason is as follows: from the Sigma matrix,

```
Sigma
```

```
##                         t
##   156650.475 -1918.16909
## t  -1918.169    31.18974
```

we know that β_0 follows the t distribution $t_{n-2}(\hat{\beta}_0, 156650.475)$. Since the distribution is close to normal, it is unlikely that β_0 will lie outside of the 2-sd interval $[\hat{\beta}_0 - 2\sqrt{156650},\ \hat{\beta}_0 + 2\sqrt{156650}] = [\hat{\beta}_0 - 791.5,\ \hat{\beta}_0 + 791.5]$. But $\hat{\beta}_0$ is -4357.9, so 791.5 is small compared with $\hat{\beta}_0$. The same reasoning works for β_1, and I expect its confidence interval is even tighter because 31.18974 is way smaller.

c

Using the results of Problem 2, provide a point estimate along with appropriate uncertainty quantification of the Resident Annual Population of California for the year 2025. Comment on whether your answer makes intuitive sense. (5 points)

Year 2025 corresponds to the following timepoint:

```
time_point = (2025 - 2021) + n
```

From problem 2, we know that the population at `time_point`, conditioned on data of past population values, follows the t distribution $t_{n-2}(\text{mu}, \text{scale}^2)$, where mu and scale are as given in the code block below:

```
mu = beta_hat[1] + beta_hat[2] * time_point
sigma_hat_squared = S / (n-2)
time_point_mat = as.matrix(c(1, time_point))
scale = sqrt(sigma_hat_squared + t(time_point_mat) %*% Sigma %*% time_point_mat)
```

Thus, the predicted population is just mu.

```
mu
```

```
## [1] 40986.16
```

Let Y_{n+1} be the population at `time_point`. Then

$$\frac{Y_{n+1} - \text{mu}}{\text{scale}},$$

again conditioned on past data, follows the standard t distribution with n-2 degrees of freedom, so we can calculate the 95% confidence interval.

```
c(mu + scale*qt(p=.025, df=n-2), mu + scale*qt(p=.975, df=n-2))
```

```
## [1] 36608.87 45363.46
```

The confidence interval makes sense to me; I expect the uncertainty of a new prediction to be larger, and indeed, the scale is slightly larger than sigma_hat, as shown below:

```
as.vector(scale)
```

```
## [1] 2210.831
```

```
sqrt(sigma_hat_squared)
```

```
## [1] 2172.403
```

d

Discuss the appropriateness of the linear trend model for this dataset. Can you think of any alternative models that would perhaps be more appropriate for this dataset? (4 points).

It is not really appropriate to assume the population growth follows the linear model because in reality, the growth rate of the population is affected by the events happening to the world at various time points. Since we can see from the actual trend that the slope before the 1940s is smaller and yet the slope after that gets larger, I would fit a piecewise linear model instead. I guess that increase of slope is due to baby boom after WW2?

4

Download the google trends time series dataset for the query yahoo. This should be a monthly time series dataset that indicates the search popularity of this query from January 2004 to August 2022. The goal of this exercise is to fit the polynomial trend model

$$Y_i = \beta_0 + \beta_1 t_i + \cdots + \beta_k t_i^k + Z_i$$

with Z_i iid $\sim N(0, \sigma^2)$ to this data set for an appropriate value of k ≤ 5.

a

Visually evaluate the fit of the least squares estimate for this model to the observed data to pick an appropriate value of k ≤ 5. Explain the reason for your choice of k. (4 points).

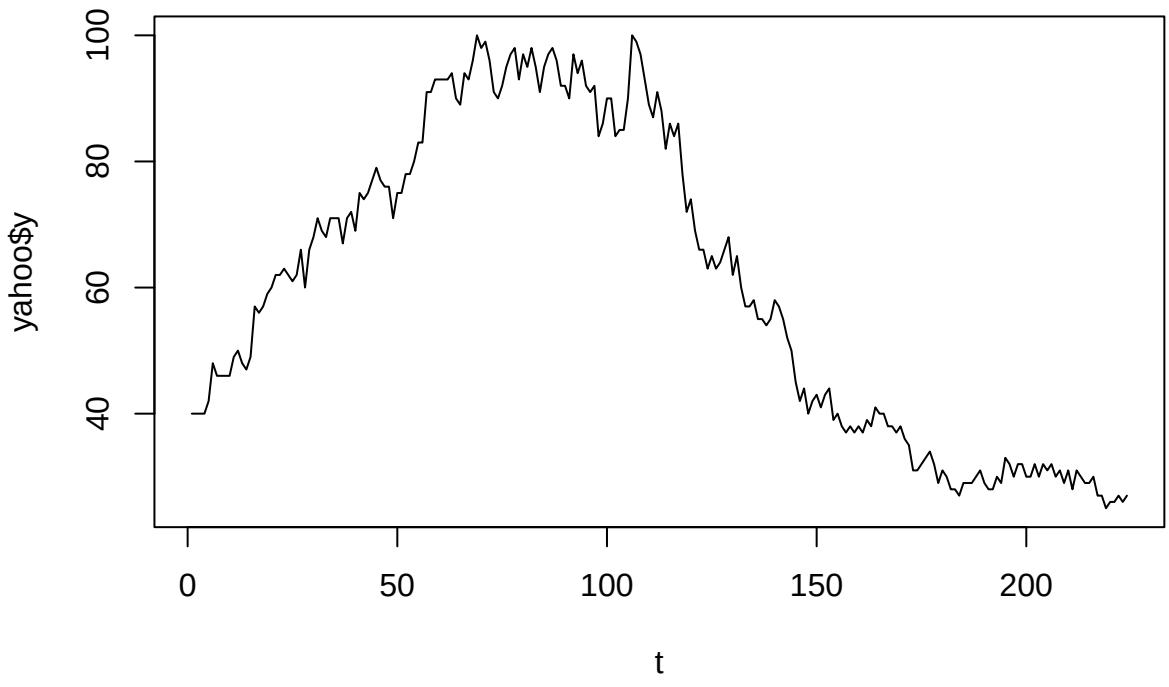
```
yahoo <- read.csv('yahoo.csv', header=T, skip=1)
colnames(yahoo) = c('Month', 'y')
yahoo <- yahoo[1:(nrow(yahoo)-1), ] # drop last row, which corresponds to Sep 2022
#yahoo.ts <- ts(yahoo$y, start = c(2004, 1), end = c(2022, 8), frequency = 12)
head(yahoo)
```

```
##     Month  y
## 1 2004-01 40
## 2 2004-02 40
## 3 2004-03 40
## 4 2004-04 40
## 5 2004-05 42
## 6 2004-06 48
```

```
n = length(yahoo$y) # number of timepoints
```

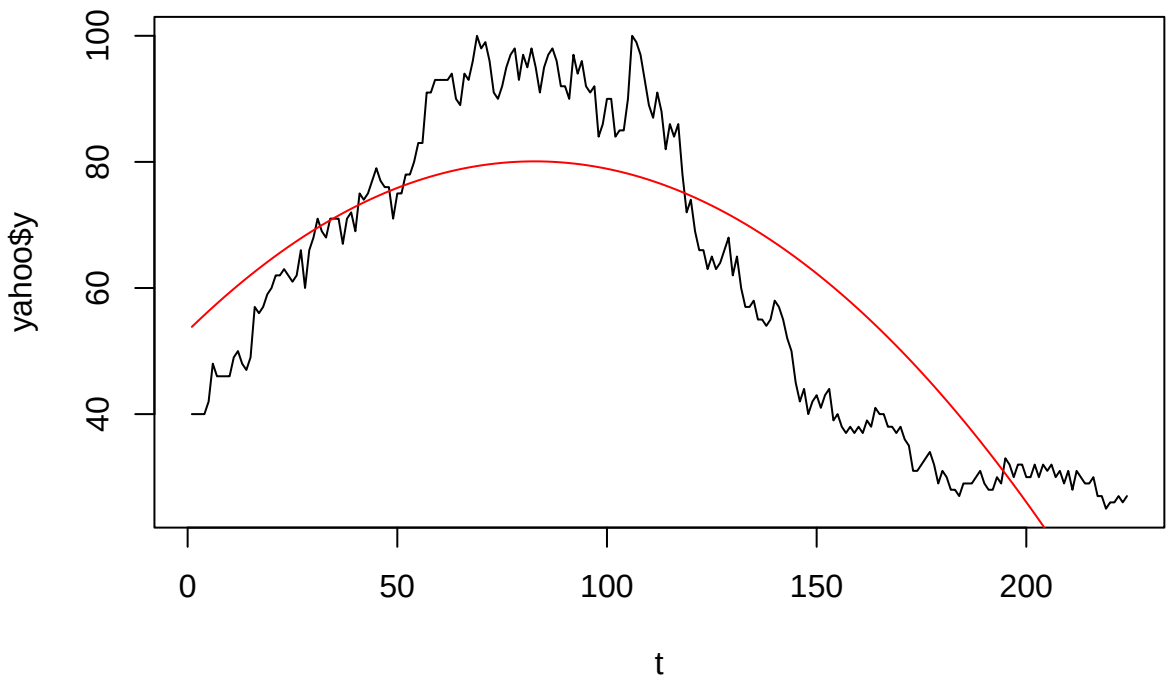
By looking at the plot of the yahoo trend data, I decided not to use a linear model because the data has a clear trend of going up and then going down.

```
t <- 1:n
plot(t, yahoo$y, type='l')
```



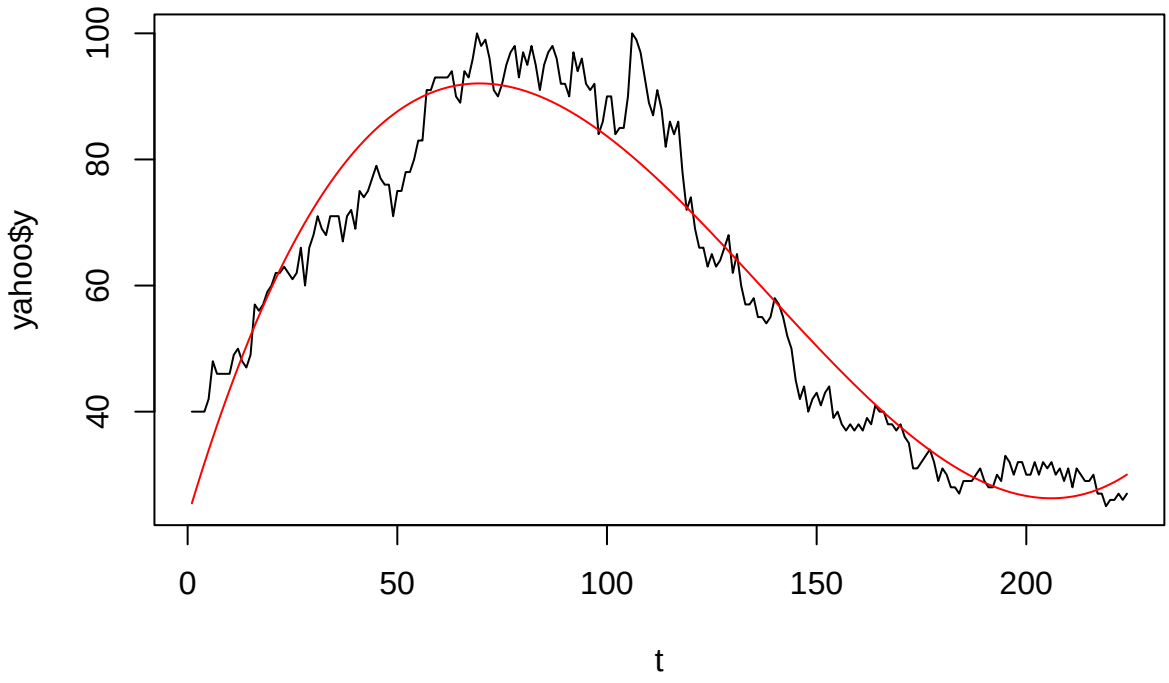
Thus, let's try k from 2. k=2:

```
plot(t, yahoo$y, type='l')
lin = lm(yahoo$y ~ 1 + t + I(t^2))
lines(t, lin$fitted.values, col='red')
```



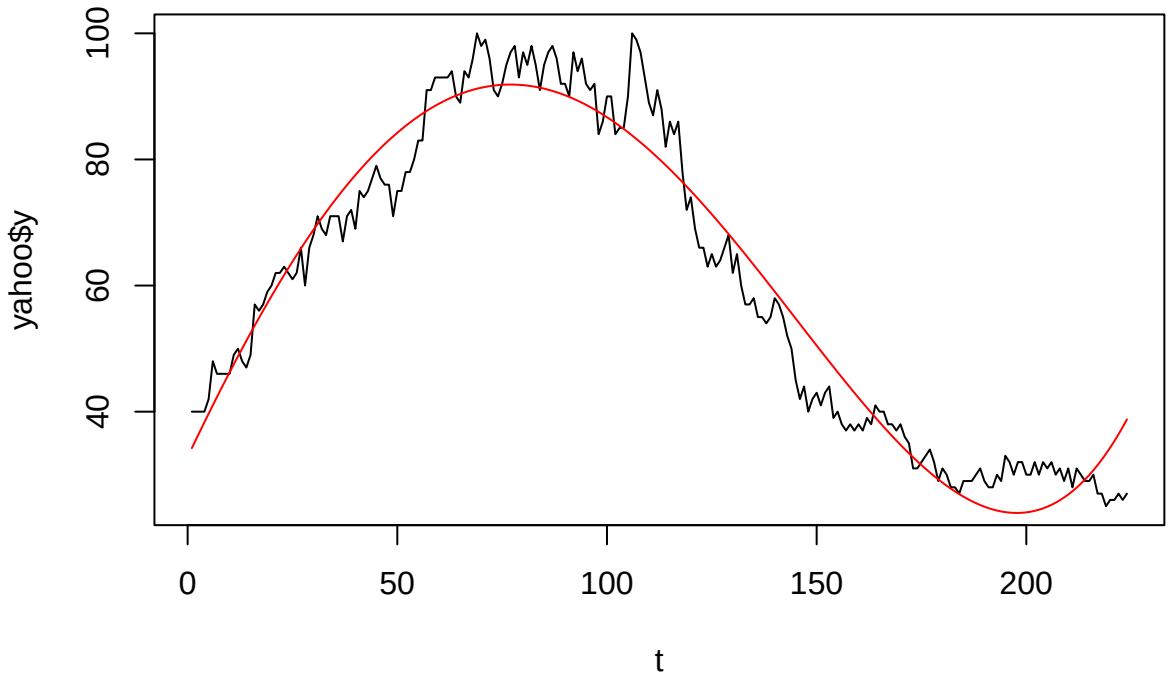
k = 3:

```
plot(t, yahoo$y, type='l')
lin = lm(yahoo$y ~ 1 + t + I(t^2) + I(t^3))
lines(t, lin$fitted.values, col='red')
```



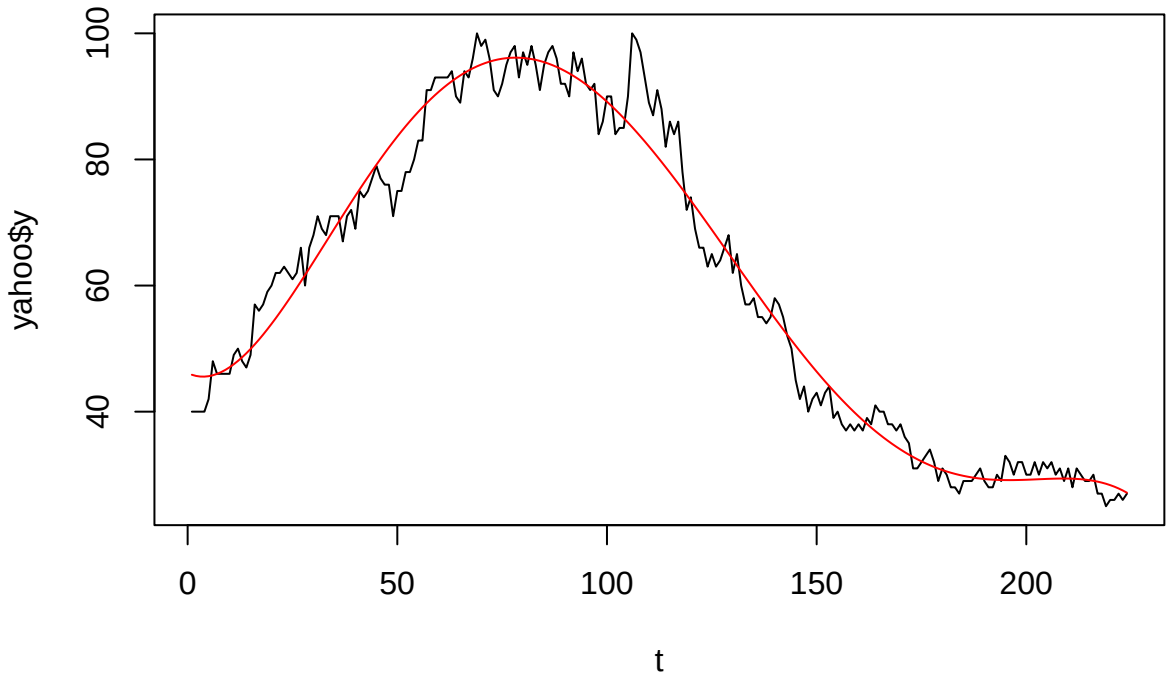
k = 4:

```
plot(t, yahoo$y, type='l')
lin = lm(yahoo$y ~ 1 + t + I(t^2) + I(t^3) + I(t^4))
lines(t, lin$fitted.values, col='red')
```



k = 5:

```
plot(t, yahoo$y, type='l')
lin = lm(yahoo$y ~ 1 + t + I(t^2) + I(t^3) + I(t^4) + I(t^5))
lines(t, lin$fitted.values, col='red')
```



I would choose k = 5 because it successfully captures the decreasing trend in the ending part of the data. k = 3 is the second best, but the polynomial goes up at the end, which the actual trend is still going down. Besides, k = 5 overall fits the data better.

Update: it turns out if I choose k = 5, R will not be able to invert the matrix $X'X$ because its entries, which involve numbers being raised to the kˆth power, are too large. After reading the GSI's response on Ed, I learned that k = 3 will be better. It is the second best fit among the plots above and it is less vulnerable to overfitting. After all, a model with a trend of a high-degree polynomial will not do well on forecasting - the model will soar to infinity very fast, while our common sense tells us that the trend should remain bounded for a long period of time. So, we will use k = 3 from now on.

b

On a plot of the observed dataset, plot the polynomial corresponding to the least squares estimate for the model with your chosen value of k. On the same figure, plot polynomials corresponding to 30 samples from the posterior distribution of the coefficients. Comment on the range of uncertainty revealed in this plot. (5 points)

The posterior $\vec{\beta} = (\beta_0, ..., \beta_3)^T$ conditioned on data $\vec{y}$ follows

$$t_{n-4}\big(\hat{\beta},\ \frac{S(\hat{\beta})}{n-4}(X'X)^{-1}\big),$$

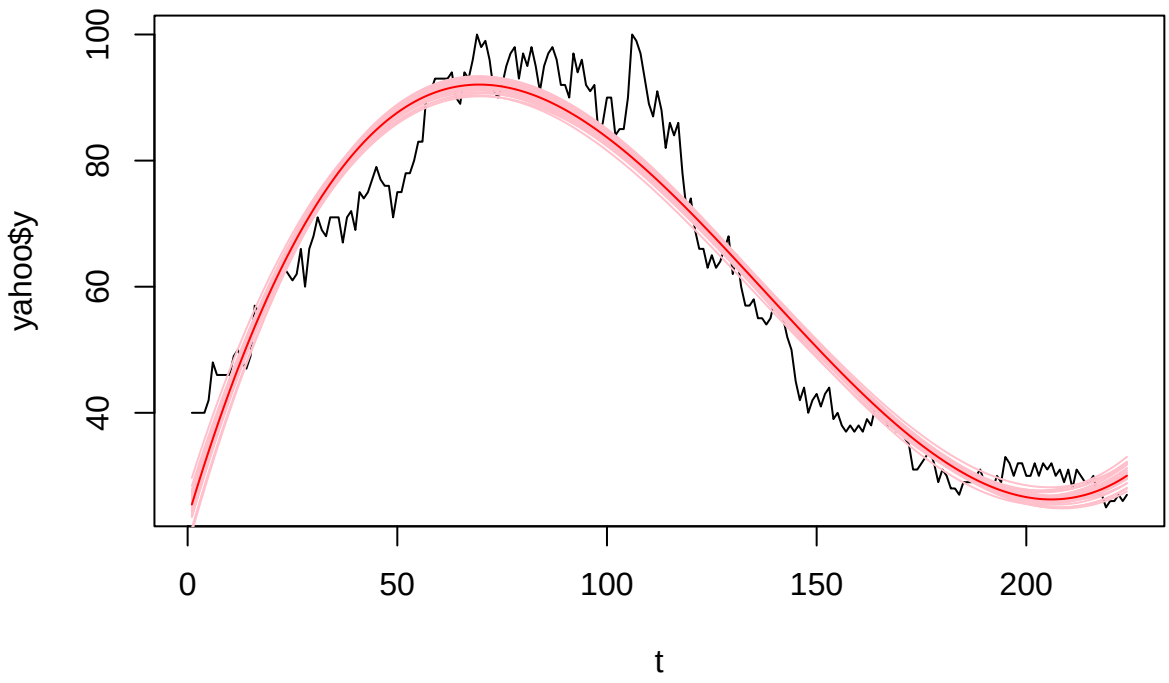
where X is now

$$\begin{bmatrix} 1 & t_1 & t_1^2 & t_1^3 \\ 1 & t_2 & t_2^2 & t_2^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_n & t_n^2 & t_n^3 \end{bmatrix}$$

```
lin = lm(yahoo$y ~ 1 + t + I(t^2) + I(t^3))

beta_hat = as.vector(lin$coefficients)
S = sum(lin$residuals^2) # residual sum of squares
X = model.matrix(lin)
Sigma = S / (n-4) * solve(t(X) %*% X)
```

In the following plot, the red polynomial curve corresponds to the least square estimate, whereas the other 30 pink curves are polynomials whose coefficients are sampled from the posterior distribution.

```
plot(t, yahoo$y, type='l')
beta_samples <- get_random_t(N=30, mu=beta_hat, Sigma=Sigma, df=n-4)
for(i in 1:30) {
  y_model = lapply(t, function(x) {
    sum(beta_samples[i, ] * c(1, x, x^2, x^3))
  })
  lines(t, y_model, col='pink')
}
lines(t, lin$fitted.values, col='red')
```



The range of uncertainty is still pretty tight. The reason is the same as in 3b. We can calculate the 95% confidence intervals for $\beta_0, ..., \beta_3$ as follows:

```
c(beta_hat[1] - 1.96*sqrt(Sigma[1,1]), beta_hat[1] + 1.96*sqrt(Sigma[1,1]))
```

```
## [1] 19.86559 26.64604
```

```
c(beta_hat[2] - 1.96*sqrt(Sigma[2,2]), beta_hat[2] + 1.96*sqrt(Sigma[2,2]))
```

```
## [1] 2.098874 2.359273
```

```
c(beta_hat[3] - 1.96*sqrt(Sigma[3,3]), beta_hat[3] + 1.96*sqrt(Sigma[3,3]))
```

```
## [1] -0.02277641 -0.02009033
```

```
c(beta_hat[4] - 1.96*sqrt(Sigma[4,4]), beta_hat[4] + 1.96*sqrt(Sigma[4,4]))
```

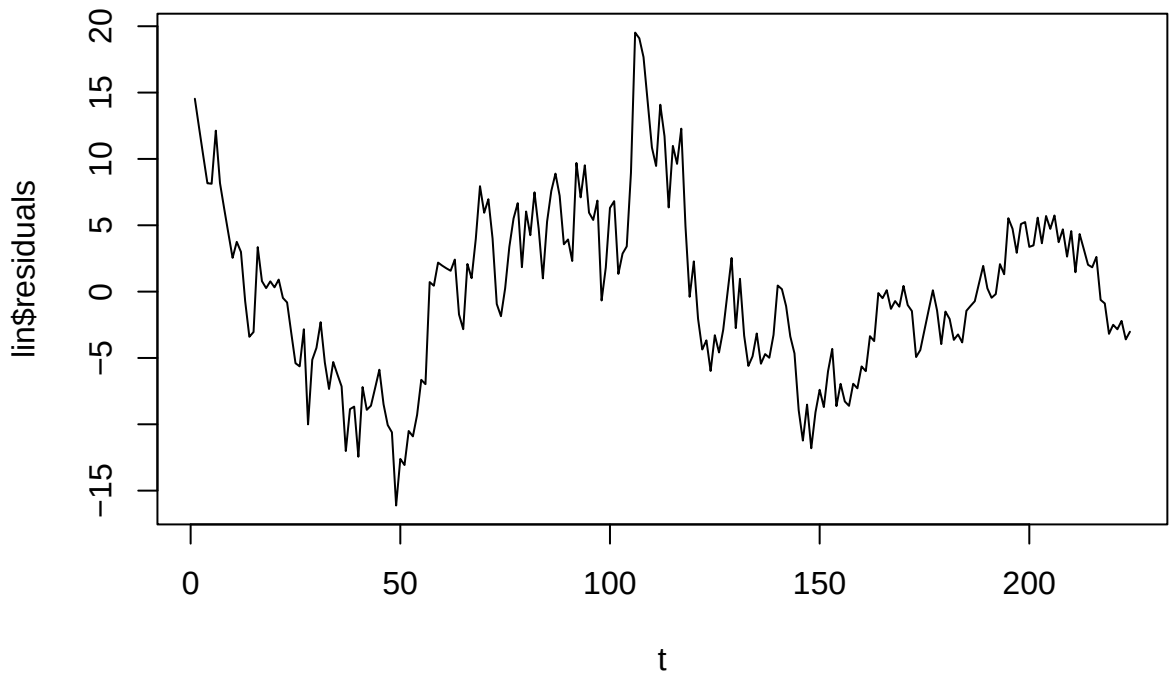
```
## [1] 4.793732e-05 5.578575e-05
```

We see that except for the CI for β_0, the other CIs are pretty tight. The intercept is allowed to change more because it only shifts the model curve vertically, but changing other betas may change the shape of the curve dramatically, so their range of uncertainty must be small.

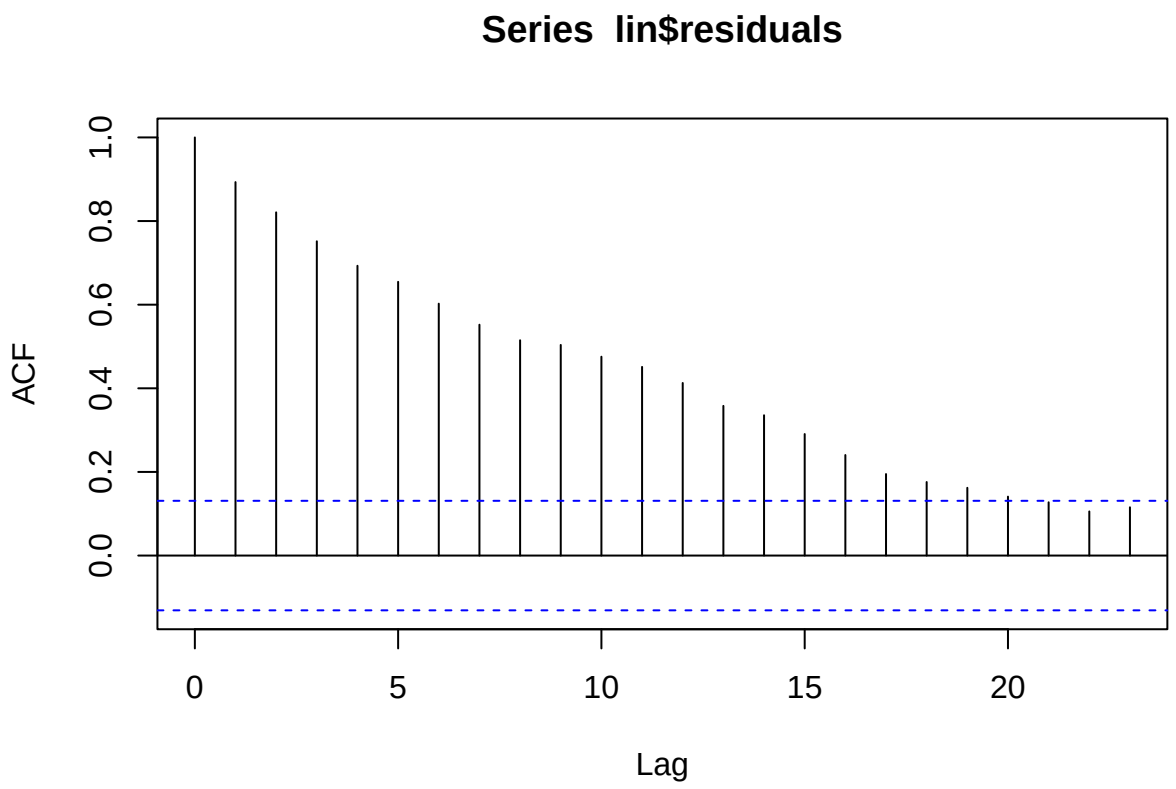
c

Plot the residuals obtained after fitting your model. Also plot the correlogram of the residuals. Is Gaussian White Noise suitable as a model for the residuals? (4 points)

```
plot(t, lin$residuals, type='l')
```



```
acf(lin$residuals)
```



Since the acf has a visible decreasing pattern and many acfs are above the blue threshold, Gaussian White Noise is not suitable as a model for the residuals. Otherwise, the autocorrelations should be approximately iid normal and so should not have a visible trend (also, I believe most acfs will lie within the blue thresholds if the residuals were to follow a Gaussian White Noise model).

5

Download the google trends time series dataset for the query frisbee. This should be a monthly time series dataset that indicates the search popularity of this query from January 2004 to August 2022.

a

Describe a model that is appropriate for estimating the trend in this dataset. Explain how you arrived at your model. (4 points)

```
fris <- read.csv('frisbee.csv', header=T, skip=1)
colnames(fris) = c('Month', 'y')
fris <- fris[1:(nrow(fris)-1), ] # drop last row, which corresponds to Sep 2022
head(fris)
```

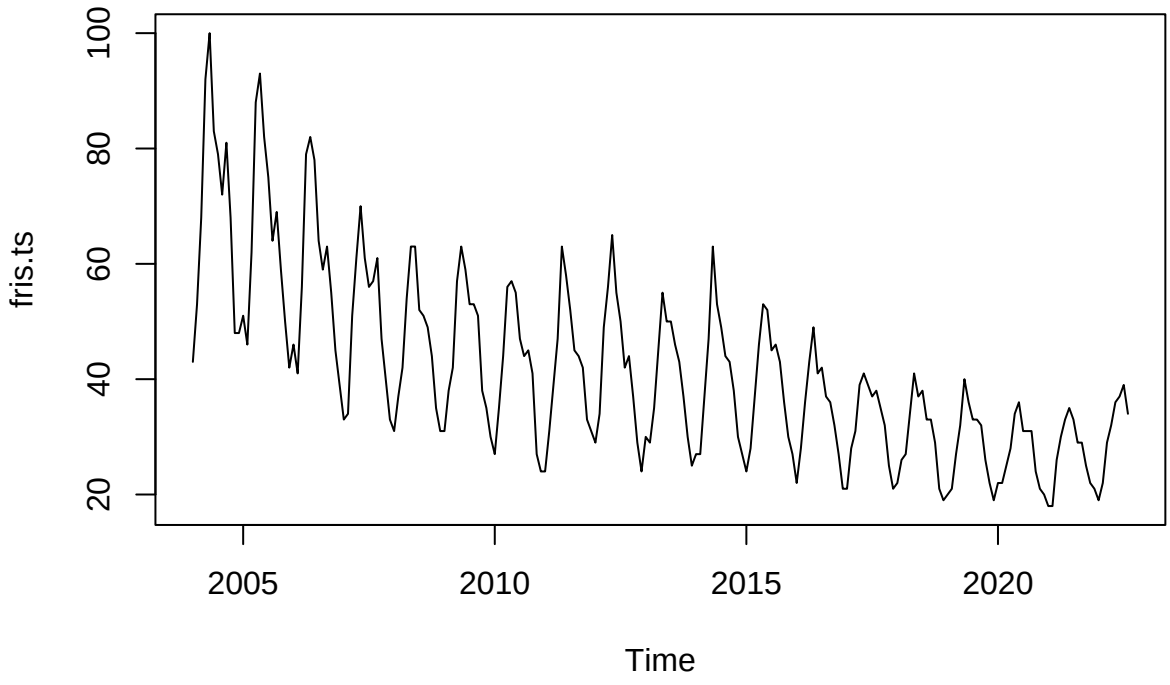
```
##     Month   y
```

```
## 1 2004-01  43
## 2 2004-02  53
## 3 2004-03  68
## 4 2004-04  92
## 5 2004-05 100
## 6 2004-06  83
```

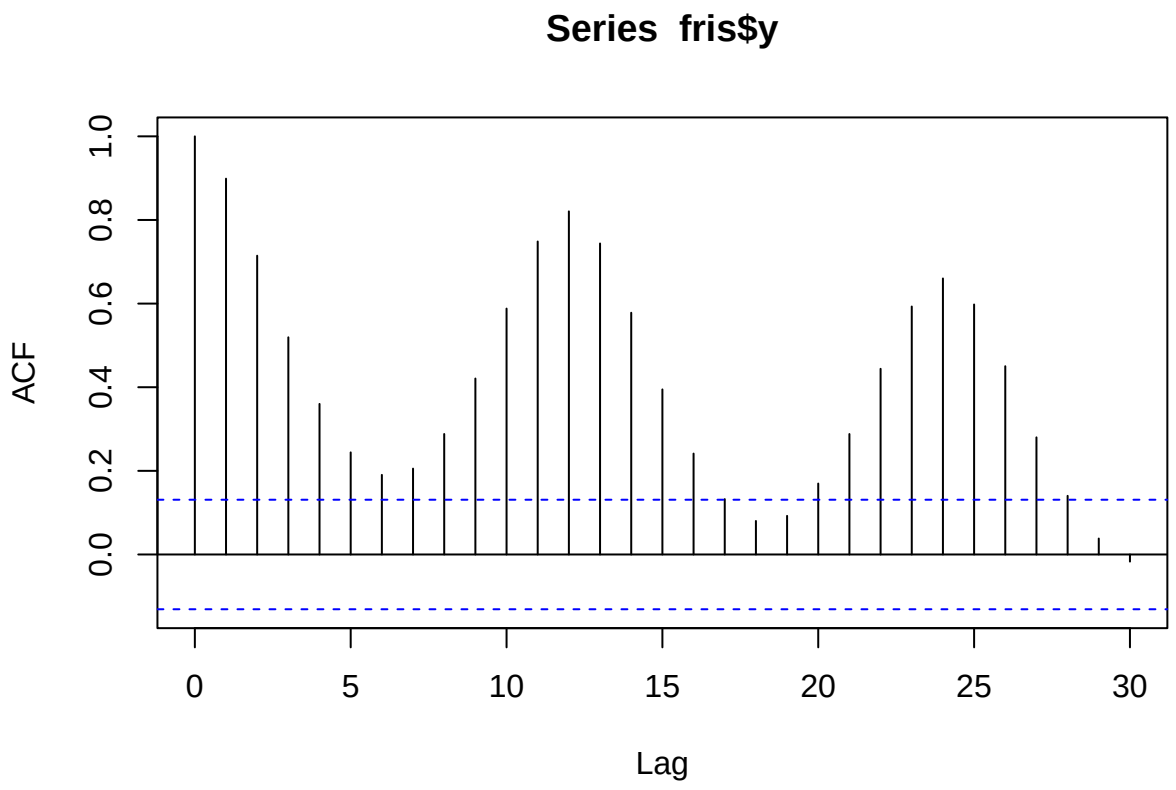
```
n = length(fris$y) # number of timepoints

fris.ts <- ts(fris$y, start = c(2004, 1), end = c(2022, 8), frequency = 12)
```

If we plot the frisbee time series, we can already see a strong seasonal oscillation and guess that the period of the oscillation is a year. This guess can be supported from the acf plot. The peaks of autocorrelation at lag k occur when k is a multiple of 12, which suggests that the data points versus those that shift 12 months ahead are in phase and so 12 is likely to be the period.

```
t <- 1:n
#plot(t, fris$y, type='l')
plot(fris.ts)
```




```
acf(fris$y, lag.max=30)
```



Thus, I will use the following model to fit the time series:

$$y_i = \beta_0 + \beta_1 t_i + \beta_2 t_i^2 + \beta_3 \cos(2\pi t_i/12) + \beta_4 \sin(2\pi t_i/12) + z_i,$$

where the z_i's are iid following $N(0, \sigma^2)$. The quadratic part $\beta_0 + \beta_1 t_i + \beta_2 t_i^2$ models the long-term trend, while the cos and sin model the oscillation.

b

Estimate the trend in this dataset by fitting your model. Quantify the uncertainty in your trend estimation. Explain your methodology briefly (5 points).

Since the model is linear in the parameters, use linear regression to get the least square estimate of $\vec{\beta}$.

```
t <- 1:n
c <- cos(2*pi/12*t)
s <- sin(2*pi/12*t)
my.model <- lm(fris$y ~ 1 + t + I(t^2) + c + s)
summary(my.model)
```

```
##
## Call:
## lm(formula = fris$y ~ 1 + t + I(t^2) + c + s)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -14.4417  -3.9410   0.1282   3.3420  22.9847
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.787e+01  1.208e+00  56.165  < 2e-16 ***
## t           -3.390e-01  2.480e-02 -13.667  < 2e-16 ***
## I(t^2)       7.121e-04  1.068e-04   6.670 2.06e-10 ***
## c           -1.316e+01  5.645e-01 -23.315  < 2e-16 ***
## s           -1.148e+00  5.645e-01  -2.034   0.0432 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.969 on 219 degrees of freedom
## Multiple R-squared:  0.8664, Adjusted R-squared:  0.864
## F-statistic: 355.1 on 4 and 219 DF,  p-value: < 2.2e-16
```

The posterior $\vec{\beta} = (\beta_0, ..., \beta_4)^{\mathrm{T}}$ conditioned on data $\vec{y}$ follows

$$t_{n-5}\big(\hat{\beta},\ \frac{S(\hat{\beta})}{n-5}(X'X)^{-1}\big),$$

where X is

$$\begin{bmatrix} 1 & t_1 & t_1^2 & \cos(2\pi t_1/12) & \sin(2\pi t_1/12) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_n & t_n^2 & \cos(2\pi t_n/12) & \sin(2\pi t_n/12) \end{bmatrix}$$

Thus, each β_i follows $t_{n-5}\big(\hat{\beta}_i,\ \frac{S(\hat{\beta})}{n-5}((X'X)^{-1})_{ii}\big)$, and we can get a 95% confidence interval for it from this t distribution.

```
bhat = as.vector(my.model$coefficients)
S = sum(my.model$residuals^2) # residual sum of squares
X = model.matrix(my.model)
Sigma = S / (n-5) * solve(t(X) %*% X)
```

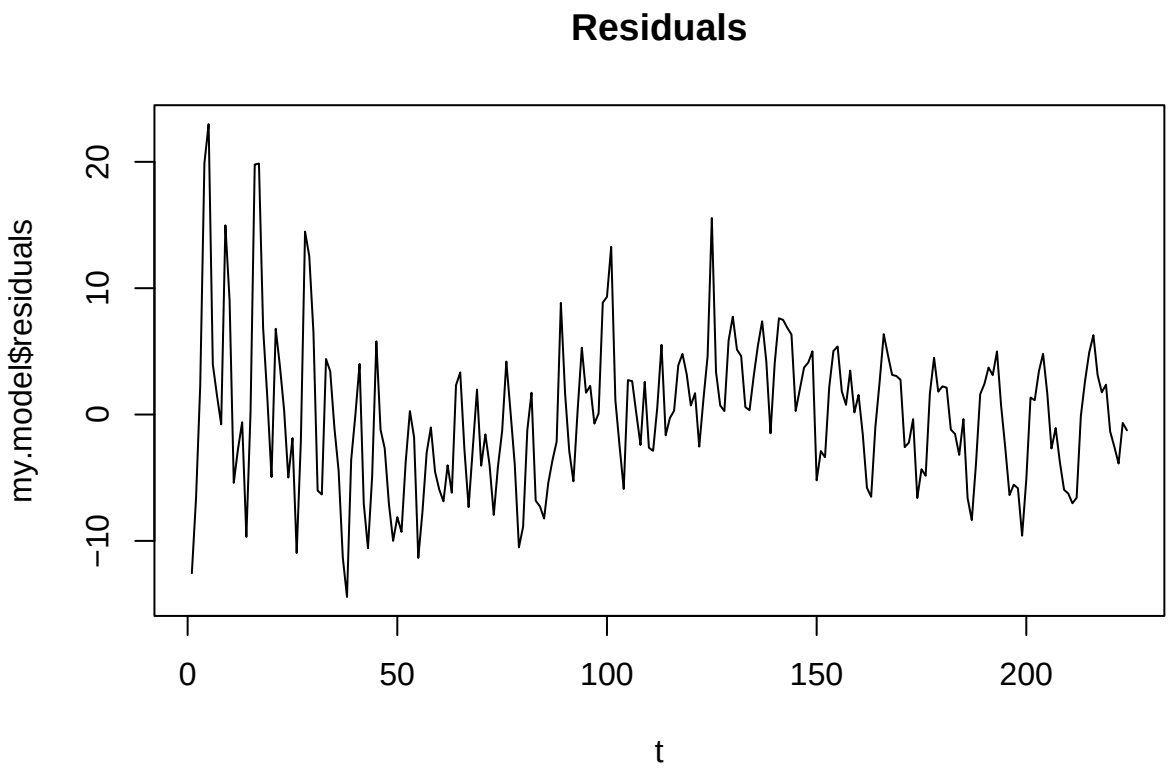
```
for(i in 1:5) {
  conf_int = c(bhat[i] + sqrt(Sigma[i,i])*qt(p=.025, df=n-5),
               bhat[i] + sqrt(Sigma[i,i])*qt(p=.975, df=n-5))
  print(sprintf('CI for beta hat %i: (%f, %f)', i-1, conf_int[1], conf_int[2]))
}
```

```
## [1] "CI for beta hat 0: (65.487151, 70.250264)"
## [1] "CI for beta hat 1: (-0.387837, -0.290080)"
## [1] "CI for beta hat 2: (0.000502, 0.000923)"
## [1] "CI for beta hat 3: (-14.273440, -12.048425)"
## [1] "CI for beta hat 4: (-2.260693, -0.035678)"
```

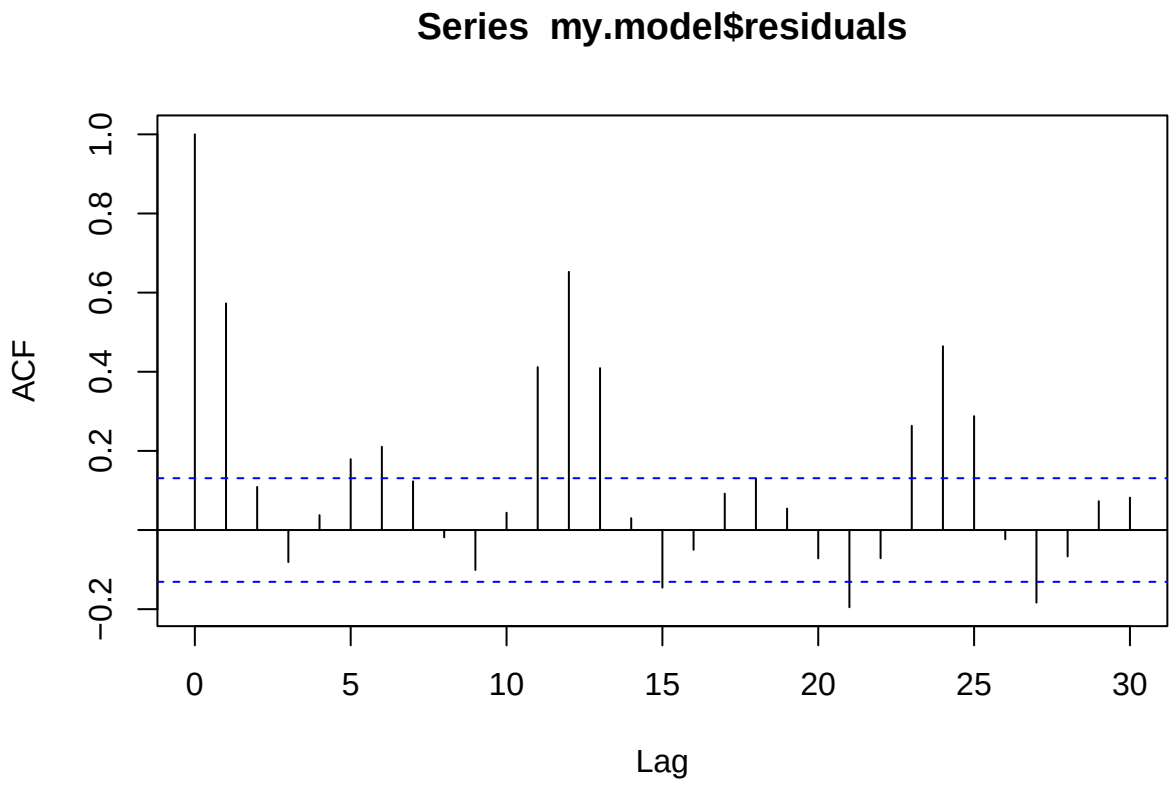
c

Plot the residuals obtained after fitting your model. Also plot the correlogram of the residuals. Is Gaussian White Noise suitable as a model for the residuals? (4 points)

```
plot(t, my.model$residuals, type='l', main='Residuals')
```




```
acf(my.model$residuals, lag.max=30)
```



No, Gaussian White Noise is still not a suitable model for the residuals, as there are still peaks at multiples of 12 and smaller ones at multiples of 6. Adding $\cos(2\pi mt_i/12)$ and $\sin(2\pi mt_i/12)$ into the model, where $m = 2, 3, 4, ...$, may help cancel out the peaks. After all, the shape of the oscillation graph need not look perfectly sinusoidal, and it is known that any continuous periodic function can be approximated by many sinusoids of the form $\cos(2\pi mt_i/12)$ and $\sin(2\pi mt_i/12)$.