# hw3code

Han-Yuan Hsu

2022-10-04

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

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

Reference: lecture code

## 1

A noisy measurement device is being examined for understanding the distribution of the errors that are being produced by it. Suppose that ten measurements led to the following observations on the errors made by the device:

```
err = c(-0.69, -4.26, 0.14, -0.86, 0.42, 24.21, 0.51, -1.23, 2.30, 4.15)
```

### a

Calculate the evidences of each of the above three models, given the observed data.

```
lik1 = function(sigma) {
  if (sigma==0) {return(0)} # the formula below is undefined for sigma=0, but the limit is 0

  1/(sqrt(2*pi)*sigma)^10 * exp(-sum(err^2)/2/sigma^2)
}

lik2 = function(sigma) {
  if (sigma==0) {return(0)}
  1/(2*sigma)^10 * exp(-sum(abs(err))/sigma)
}

lik3 = function(sigma) {
  prod = 1
  for (i in 1:10) {
    term = sigma / (err[i]^2 + sigma^2) / pi
    prod = prod * term
  }
  prod
}
```

The range of the integrals is $(e^{-15}, e^{15})$, but $e^{15}$ is too large for numerical integration. The following shows integrating up to 100 does a good approximation:

```
lik1(100)
```

```
## [1] 9.89512e-25
```

```
lik2(100)
```

```
## [1] 6.627108e-24
```

```
lik3(100)
```

```
## [1] 1.004296e-25
```

```
res = .001
my.grid = seq(exp(-15), 100+exp(-15), by=res)

lik1.vals = sapply(my.grid, FUN=lik1)
evid1 = 1/30 * sum(lik1.vals / my.grid) * res

lik2.vals = sapply(my.grid, FUN=lik2)
evid2 = 1/30 * sum(lik2.vals / my.grid) * res

lik3.vals = sapply(my.grid, FUN=lik3)
evid3 = 1/30 * sum(lik3.vals / my.grid) * res
```

The evidences:

```
evid1
```

```
## [1] 1.317392e-17
```

```
evid2
```

```
## [1] 1.539515e-15
```

```
evid3
```

```
## [1] 3.538157e-14
```

## b

Normalize the three evidences to obtain posterior probabilities for the three models. Which model has the highest posterior probability? Comment on whether your results seem intutively sensible. (2 points).

```
normalizing.const = evid1 + evid2 + evid3
evid1 / normalizing.const
```

```
## [1] 0.0003566857
```

```
evid2 / normalizing.const
```

```
## [1] 0.04168257
```

```
evid3 / normalizing.const
```

```
## [1] 0.9579607
```

Model 3 has the highest posterior probability. That makes sense – Cauchy distribution has the fattest tail compared with the distrbutions of the other two models that decay exponentially. The error data has an outlier 24.21, so a heavy-tailed distribtuion makes this outlier more possible.

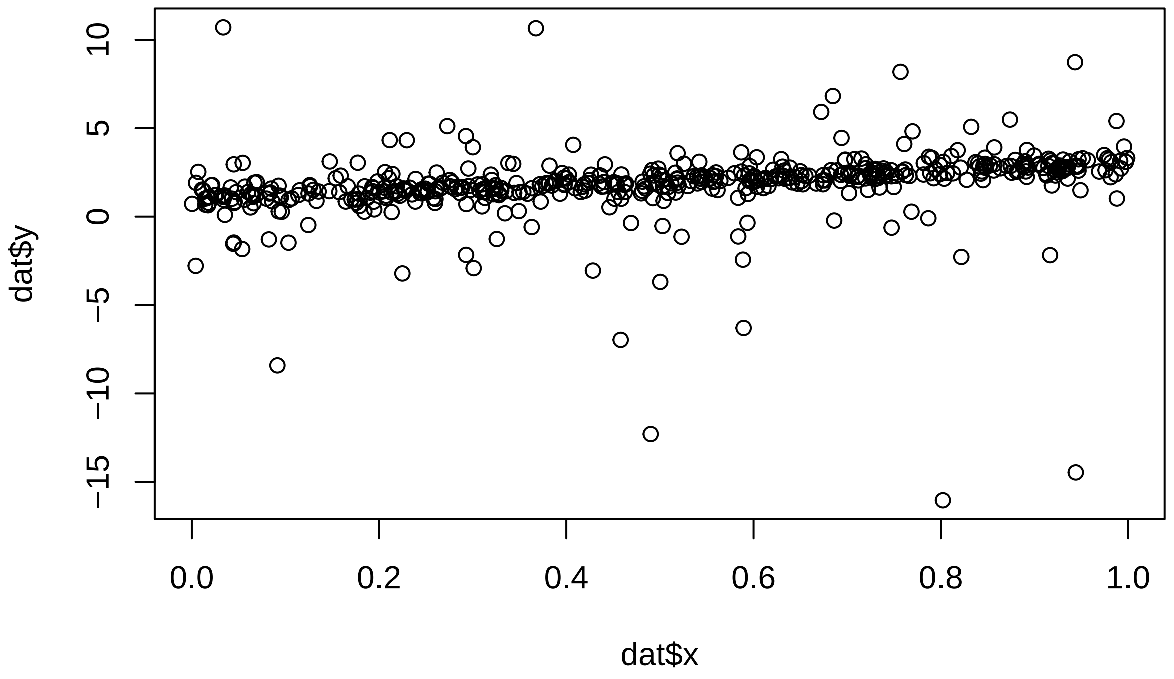
## 2

```r
dat = read.csv('HW3Data153Fall2022.csv')
head(dat)
```

```
##   X            x          y
## 1 1 0.0003764606  0.7235082
## 2 2 0.0042100137 -2.7847222
## 3 3 0.0045304559  1.9114261
## 4 4 0.0069880036  2.5319608
## 5 5 0.0108175443  1.5236083
## 6 6 0.0123043244  1.4784446
```

```r
n = length(dat$x)
```

```r
plot(dat$x, dat$y)
```



### a

Numerically calculate the Evidence for each of these models given the observed data. Report the normalized evidences. Which model has the highest evidence and what is the value of this highest evidence? (6 points)

```
beta0.vals = seq(-10, 10, by=.1)
beta1.vals = seq(-10, 10, by=.1)
sig.vals = exp(seq(-10, 10, by=.1))
vals.length = length(beta0.vals)

loglik1 = function(param) {
  b0 = param[1]; b1 = param[2]; sig = param[3]
  n*(log(sig)-log(pi)) - sum(log((dat$y - b1*dat$x - b0)^2 + sig^2))
}

loglik2 = function(param) {
  b0 = param[1]; b1 = param[2]; sig = param[3]
  -n*log(2*sig) - sum(abs(dat$y - b1*dat$x - b0))/sig
}

loglik3 = function(param) {
  b0 = param[1]; sig = param[2]
  n*(log(sig)-log(pi)) - sum(log((dat$y - b0)^2 + sig^2))
}

loglik4 = function(param) {
  b0 = param[1]; sig = param[2]
  -n*log(2*sig) - sum(abs(dat$y - b0))/sig
}
```

```
vals = expand.grid(beta0.vals, beta1.vals, sig.vals)
```

We apply our log likelihood functions to the possible values (`vals` or `vals2`) to get `loglik1.vals`, `loglik2.vals`, etc.

```
loglik1.vals = apply(vals, MARGIN=1, FUN=loglik1) # takes 3 minutes
save(loglik1.vals, file='loglik1.vals.RData')
```

```
load(file='loglik1.vals.RData')
```

```
loglik1.vals.max = max(loglik1.vals)
loglik1.vals = loglik1.vals - loglik1.vals.max
lik1.vals = exp(loglik1.vals) # the true likelihood values should be this times exp(loglik1.vals.max)
```

Below, the true evidence of model 1 is proportional to `evid1.scaled` times exp(`loglik1.vals.max`), where the proportionality constant is the same for all four models. So, to choose the best model, we find the largest `evidm.scaled` times exp(`loglikm.vals.max`) for m=1,2,3,4.

```
evid1.scaled = sum(lik1.vals) # I didn't multiply the prior here because it is the same for all 4 model
evid1.scaled
```

```
## [1] 1.990494
```

```
loglik2.vals = apply(vals, MARGIN=1, FUN=loglik2)
save(loglik2.vals, file='loglik2.vals.RData')
```

```
load(file='loglik2.vals.RData')
```

```
loglik2.vals.max = max(loglik2.vals)
loglik2.vals = loglik2.vals - loglik2.vals.max
lik2.vals = exp(loglik2.vals) # the true likelihood values should be this times exp(loglik2.vals.max)
```

```
evid2.scaled = sum(lik2.vals)
evid2.scaled
```

```
## [1] 2.724382
```

Now for model 3 and 4:

```
vals2 = expand.grid(beta0.vals, sig.vals)
```

```
loglik3.vals = apply(vals2, MARGIN=1, FUN=loglik3)
save(loglik3.vals, file='loglik3.vals.RData')
loglik4.vals = apply(vals2, MARGIN=1, FUN=loglik4)
save(loglik4.vals, file='loglik4.vals.RData')
```

```
load('loglik3.vals.RData')
load('loglik4.vals.RData')
```

```
loglik3.vals.max = max(loglik3.vals)
loglik3.vals = loglik3.vals - loglik3.vals.max
lik3.vals = exp(loglik3.vals) # the true likelihood values should be this times exp(loglik3.vals.max)
```

```
evid3.scaled = sum(lik3.vals)
evid3.scaled
```

```
## [1] 1.844927
```

```
loglik4.vals.max = max(loglik4.vals)
loglik4.vals = loglik4.vals - loglik4.vals.max
lik4.vals = exp(loglik4.vals) # the true likelihood values should be this times exp(loglik3.vals.max)
```

```
evid4.scaled = sum(lik4.vals)
evid4.scaled
```

```
## [1] 2.132421
```

Below, we calculate the normalized evidences. Note that among `loglik1.vals.max`, `loglik2.vals.max`, `loglik3.vals.max`, and `loglik4.vals.max`, `loglik1.vals.max` is the largest:

```
loglik1.vals.max
```

```
## [1] -600.1249
```

```r
loglik2.vals.max
```

```
## [1] -728.8744
```

```r
loglik3.vals.max
```

```
## [1] -827.7335
```

```r
loglik4.vals.max
```

```
## [1] -856.9092
```

So when we calculate `evidm.scaled` times exp(`loglikm.vals.max`) for m=1,2,3,4 to choose the best model, we also divide all these numbers by exp(`-loglik1.vals.max`) to avoid issues in numerical computation.

```r
evid1 = evid1.scaled
evid2 = evid2.scaled * exp(loglik2.vals.max - loglik1.vals.max)
evid3 = evid3.scaled * exp(loglik3.vals.max - loglik1.vals.max)
evid4 = evid4.scaled * exp(loglik4.vals.max - loglik1.vals.max)
normalizing.const = evid1 + evid2 + evid3 + evid4
evid1 = evid1 / normalizing.const
evid2 = evid2 / normalizing.const
evid3 = evid3 / normalizing.const
evid4 = evid4 / normalizing.const
```

```r
evid1 # should be almost 1
```

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

```r
evid2
```

```
## [1] 1.663856e-56
```

```r
evid3
```

```
## [1] 1.311779e-99
```

```r
evid4
```

```
## [1] 3.235247e-112
```

Model 1 has the highest evidence, which is almost 1 because the evidences of other models are very small.

### b

For your chosen model, describe your best estimates of $\beta_0$, $\sigma$ and $\beta_1$ (if $\beta_1$ exists in the model) along with appropriate uncertainty quantification. Plot your best estimate of $\beta_0 + \beta_1 x$ on a plot of the data. (4 points)

My estimators for the parameters are maximum likelihood estimators:

```
ind.max = which.max(loglik1.vals)
beta0.hat = vals[ind.max, 1]
beta1.hat = vals[ind.max, 2]
sig.hat = vals[ind.max, 3]
```

```
beta0.hat
```

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

```
beta1.hat
```

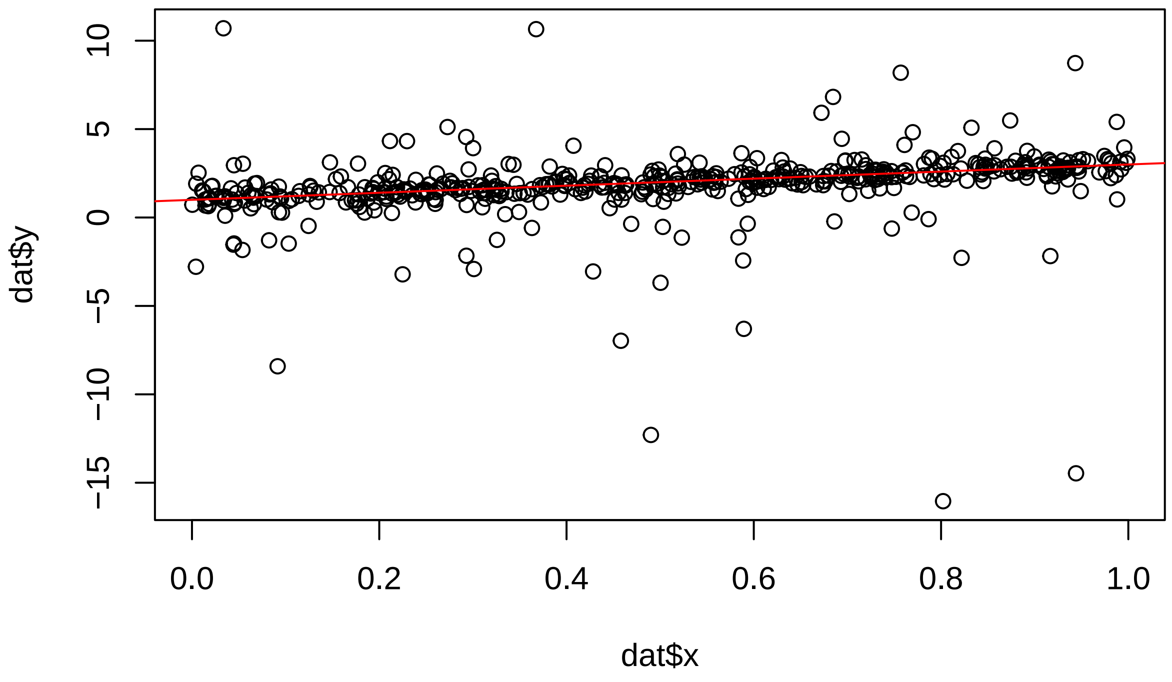
```
## [1] 2
```

```
sig.hat
```

```
## [1] 0.2725318
```

Below, the red line is $\hat{\beta}_0 + \hat{\beta}_1 x$:

```
plot(dat$x, dat$y)
abline(beta0.hat, beta1.hat, col='red')
```



To get uncertainty quantification for the parameters, we sample from the posterior distribution:

```
N = 1000000
post.pdf = lik1.vals / sum(lik1.vals) # posterior pdf of beta0, beta1, sigma
ind.samples = sample(1:nrow(vals), N, replace=T, prob=post.pdf)
beta0.samples = vals[ind.samples, 1]
beta1.samples = vals[ind.samples, 2]
sig.samples = vals[ind.samples, 3]
```

Below are the probabilities of each parameter equaling its estimator:

```
sum(beta0.samples == beta0.hat) / length(beta0.samples)
```

```
## [1] 0.949599
```

```
sum(beta1.samples == beta1.hat) / length(beta1.samples)
```

```
## [1] 0.921611
```

```
sum(sig.samples == sig.hat) / length(sig.samples)
```

```
## [1] 0.539836
```

Below is a 95% credibility interval for sigma:

```
quantile(sig.samples, c(.025, .975))
```

```
##      2.5%     97.5%
## 0.2465970 0.3011942
```

## 3

R has an inbuilt dataset called state which gives some data on 50 states in America from the 1970s. You can access this dataset via (see help(state) for information about the data) We want to fit a linear model to this dataset with life expectancy as the response variable and some subset of the remaining seven explanatory variables (including the intercept term). Your goal is to figure out which subset of the explanatory variables provides the best explanation for the response variable in a linear model.

```
data(state); dt = data.frame(state.x77, row.names = state.abb)
head(dt)
```

```
##    Population Income Illiteracy Life.Exp Murder HS.Grad Frost   Area
## AL       3615   3624        2.1    69.05   15.1    41.3    20  50708
## AK        365   6315        1.5    69.31   11.3    66.7   152 566432
## AZ       2212   4530        1.8    70.55    7.8    58.1    15 113417
## AR       2110   3378        1.9    70.66   10.1    39.9    65  51945
## CA      21198   5114        1.1    71.71   10.3    62.6    20 156361
## CO       2541   4884        0.7    72.06    6.8    63.9   166 103766
```

### a

Use the Evidence-based Bayesian model selection method from Lecture 11 to calculate the evidences for each of the 2^7 = 128 models (obtaining by taking all possible subsets of the explanatory variables along with the intercept term). How many of the 128 models get nontrivial Evidences (after normalization so that the Evidences sum to one)? Describe the models getting high evidences? Do the results of your analysis seem sensible? (6 points)

```
#y = dt$Life.Exp
g <- lm(Life.Exp ~ ., data=dt)
summary(g)
```

```
##
## Call:
## lm(formula = Life.Exp ~ ., data = dt)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.48895 -0.51232 -0.02747  0.57002  1.49447
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.094e+01  1.748e+00  40.586  < 2e-16 ***
## Population   5.180e-05  2.919e-05   1.775   0.0832 .
## Income      -2.180e-05  2.444e-04  -0.089   0.9293
## Illiteracy   3.382e-02  3.663e-01   0.092   0.9269
## Murder      -3.011e-01  4.662e-02  -6.459 8.68e-08 ***
## HS.Grad      4.893e-02  2.332e-02   2.098   0.0420 *
## Frost       -5.735e-03  3.143e-03  -1.825   0.0752 .
## Area        -7.383e-08  1.668e-06  -0.044   0.9649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.7448 on 42 degrees of freedom
## Multiple R-squared:  0.7362, Adjusted R-squared:  0.6922
## F-statistic: 16.74 on 7 and 42 DF,  p-value: 2.534e-10
```

all01 below indexes subsets of the features in a 1/0 fashion.

```
fullX = model.matrix(g)
numvar = 7
all01 = expand.grid(replicate(numvar, 0:1, simplify = FALSE))
all01 = cbind(rep(1, nrow(all01)), all01)
colnames(all01) = colnames(fullX)
logEvid = rep(-1, nrow(all01))
```

```
head(all01)
```

```
##   (Intercept) Population Income Illiteracy Murder HS.Grad Frost Area
## 1           1          0      0          0      0       0     0    0
## 2           1          1      0          0      0       0     0    0
## 3           1          0      1          0      0       0     0    0
## 4           1          1      1          0      0       0     0    0
## 5           1          0      0          1      0       0     0    0
## 6           1          1      0          1      0       0     0    0
```

The following calculates the log evidence for each model mm in all01:

```
for(mm in 1:nrow(all01))
{
    inds = all01[mm,]
    Xmat = fullX[,(inds == 1)]
    if(mm == 1) {
      # this is the model with the intercept only, which is a trivial case
      # if we don't run this if statement, weird bugs happen
      Xmat = as.matrix(rep(1, nrow(dt)), nrow=nrow(dt), ncol=1)
    }
    p = ncol(Xmat)
    n = nrow(Xmat)
    md = lm(dt$Life.Exp ~ -1 + Xmat)
    # the following log evidence formula for model mm comes from lecture 11 notes.
    logEvid[mm] = (lgamma(p/2)) - ((p/2)*(log((sum(md$fitted.values^2))))) + (lgamma((n-p-1)/2)) - (((n
}
```

The first 10 models with highest evidence:

```
logEvid.sorted = sort(logEvid, decreasing = T, index.return=T)
r = 10 # show first r models
high.evid.ind = logEvid.sorted$ix[1:r]
all01[high.evid.ind, ]
```

```
##    (Intercept) Population Income Illiteracy Murder HS.Grad Frost Area
## 9            1          0      0          0      1       0     0    0
## 25           1          0      0          0      1       1     0    0
## 10           1          1      0          0      1       0     0    0
## 41           1          0      0          0      1       0     1    0
## 11           1          0      1          0      1       0     0    0
## 57           1          0      0          0      1       1     1    0
## 73           1          0      0          0      1       0     0    1
## 13           1          0      0          1      1       0     0    0
## 26           1          1      0          0      1       1     0    0
## 43           1          0      1          0      1       0     1    0
```

All of these 10 models include the murder feature, which makes sense because in the summary of the model g that uses all the features, we see that only murder is very significant. Also, it turns out models with fewer features are preferred, which makes sense because the evidence formula we use here penalizes models with a large number of features. Indeed, the best model only has one feature, which is murder of course. The second-best model includes HS.Grad feature, which also makes sense because the feature has the second best significance level, as shown in the summary of g.

Get normalized evidences from log evidences:

```
evid = logEvid - max(logEvid)
evid = exp(evid)
evid = evid / sum(evid)
```

## b and c

Calculate the best model (among the 128 possible models) using the BIC. Compare this model with the models obtaining high evidence from part (a). (3 points) Calculate the best model (among the 128 possible models) using the AIC. Compare this model with the models obtaining high evidence from part (a). (3 points)

```
bic = rep(-9999, nrow(all01))
aic = rep(-9999, nrow(all01))
for(mm in 1:nrow(all01))
{
    inds = all01[mm,]
    Xmat = fullX[,(inds == 1)]
    if(mm == 1) {
      # this is the model with the intercept only, which is a trivial case
      # if we don't run this if statement, weird bugs happen
      Xmat = as.matrix(rep(1, nrow(dt)), nrow=nrow(dt), ncol=1)
    }
    p = ncol(Xmat)+1 # number of parameters, betas AND sigma
    n = nrow(Xmat)
    md = lm(dt$Life.Exp ~ -1 + Xmat)
    sig.hat.squared = sum((md$residuals)^2) / n
    bic[mm] = n + n*log(2*pi*sig.hat.squared) + p*log(n)
    aic[mm] = n + n*log(2*pi*sig.hat.squared) + p*2
}
```

```
bic.sorted = sort(bic, decreasing = T, index.return=T)
high.bic.ind = bic.sorted$ix[1:10]
all01[high.bic.ind, ]
```

```
##     (Intercept) Population Income Illiteracy Murder HS.Grad Frost Area
## 98            1          1      0          0      0       0     1    1
## 66            1          1      0          0      0       0     0    1
## 34            1          1      0          0      0       0     1    0
## 2             1          1      0          0      0       0     0    0
## 97            1          0      0          0      0       0     1    1
## 65            1          0      0          0      0       0     0    1
## 100           1          1      1          0      0       0     1    1
## 36            1          1      1          0      0       0     1    0
## 4             1          1      1          0      0       0     0    0
## 68            1          1      1          0      0       0     0    1
```

BIC chooses completely different models compared with Bayesian Evidence. In particular, the top models chosen by BIC do not use Murder.

```
aic.sorted = sort(aic, decreasing = T, index.return=T)
high.aic.ind = aic.sorted$ix[1:r]
all01[high.aic.ind, ]
```

```
##    (Intercept) Population Income Illiteracy Murder HS.Grad Frost Area
## 66           1          1      0          0      0       0     0    1
## 2            1          1      0          0      0       0     0    0
## 98           1          1      0          0      0       0     1    1
## 65           1          0      0          0      0       0     0    1
## 34           1          1      0          0      0       0     1    0
## 1            1          0      0          0      0       0     0    0
## 97           1          0      0          0      0       0     1    1
## 33           1          0      0          0      0       0     1    0
## 36           1          1      1          0      0       0     1    0
## 4            1          1      1          0      0       0     0    0
```

The models chosen by AIC are still completely different from those chosen by Bayesian evidence, but they are similar to those chosen by BIC. Again, no Murder.

## 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 use model selection to figure out the best polynomial trend model among the degrees k = 1, 2, 3, 4, 5, 6, 7, 8. To prevent numerical instability issues, take x_i to be some scaled version of time (for example, take x_i = i/n).

```
yahoo <- read.csv('../hw1/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) # time series length
```

```
x = c(1:n) / n # scaled time
y = yahoo$y
```

### a

Use the Evidence-based Bayesian model selection method from Lecture 11 to calculate the evidences for each of the above 8 models. Report the normalized evidences of each of the 8 models. Which model has the highese evidence? Does this model selection method select models that seemings overfit? (6 points)

```
num.models = 8
logEvid = rep(-1, num.models)

Xmat = matrix(rep(1, n), nrow=n)
for(mm in 1:num.models)
{
    Xmat = cbind(Xmat, x^mm)
    p = ncol(Xmat) # here, p is the number of betas only
    md = lm(y ~ -1 + Xmat)
    # the following log evidence formula for model mm comes from lecture 11 notes.
    logEvid[mm] = (lgamma(p/2)) - ((p/2)*(log((sum(md$fitted.values^2))))) + (lgamma((n-p-1)/2)) - (((n

}
```

```
which.max(logEvid)
```

```
## [1] 7
```

7 is chosen, which is very large for polynomial regression, so there may be overfitting. A degree-7 polynomial is definitely unable to forcast well because common sense tells us that the search popularity for a keyword should not keep growing as fast as $x^7$ as time goes on.

## b and c

Calculate the best model using the BIC and the AIC. Compare these models with the models obtaining high evidence from part (a).

```r
bic = rep(-9999, num.models)
aic = rep(-9999, num.models)
Xmat = matrix(rep(1, n), nrow=n)
for(mm in 1:num.models)
{
    Xmat = cbind(Xmat, x^mm)
    md = lm(y ~ -1 + Xmat)
    p = ncol(Xmat)+1 # number of parameters, betas AND sigma
    sig.hat.squared = sum((md$residuals)^2) / n
    bic[mm] = n + n*log(2*pi*sig.hat.squared) + p*log(n)
    aic[mm] = n + n*log(2*pi*sig.hat.squared) + p*2
}
```

```r
which.min(bic)
```

```
## [1] 7
```

```r
which.min(aic)
```

```
## [1] 7
```

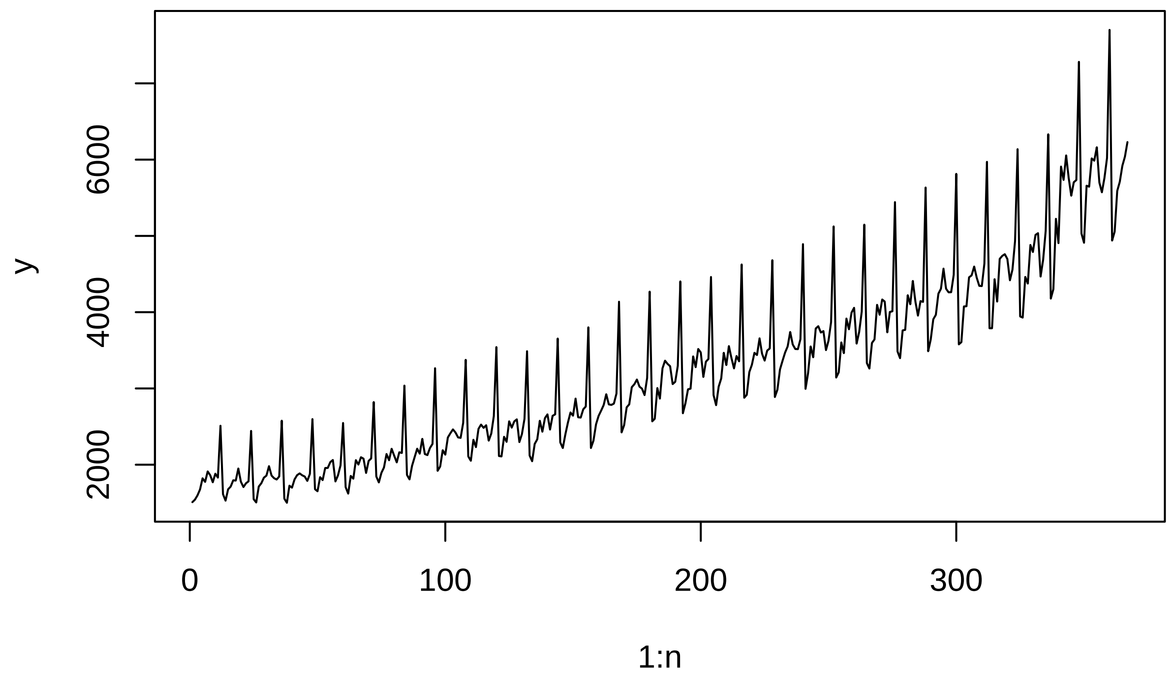
Both BIC and AIC choose the same model as Bayesian Evidence does.

## 5

Download the FRED dataset on "Retail Sales: Beer, Wine, and Liquor Stores" from https://fred.stlouisfed.org/series/MRTSSM4453USN. This is a monthly dataset (the units are millions of dollars) and is not seasonally adjusted. To this data, we want to fit one of the models $M_{kl}$ where k ranges in 0, 1, 2, 3, 4, 5 and l ranges in 0, 1, 2, 3, 4, 5.

```r
alc = read.csv('alc.csv')
y = alc[,2]
n = length(y)
```

```r
plot(1:n, y, type='l')
```



### a

Use the Evidence-based Bayesian model selection method from Lecture 11 to calculate the evidences for each of the above 36 models. Report the normalized evidences of each of the 36 models. Which models have high evidence? Does this model selection method favor models that seemingly overfit? (6 points)

```r
k.vals = 0:5
l.vals = 0:5
logEvid = rep(-9999, 36)
ind = 1 # index for accessing elements of logEvid
```

```
Poly = matrix(rep(1, n), ncol=1) # polynomial part of the model matrix
for (k in k.vals) {
  Xmat = Poly
  for (l in l.vals) {
    p = ncol(Xmat)
    md = lm(y ~ -1 + Xmat)
    # the following log evidence formula for model mm comes from lecture 11 notes.
    logEvid[ind] = (lgamma(p/2)) - ((p/2)*(log((sum(md$fitted.values^2))))) + (lgamma((n-p-1)/2)) - (((

    ind = ind + 1
    Xmat = cbind(Xmat, cos(2*pi*(l+1)/12 * (1:n)))
    Xmat = cbind(Xmat, sin(2*pi*(l+1)/12 * (1:n)))
  }
  Poly = cbind(Poly, ((1:n)/n)^(k+1))
}
```

Below are the top 5 models with high evidence:

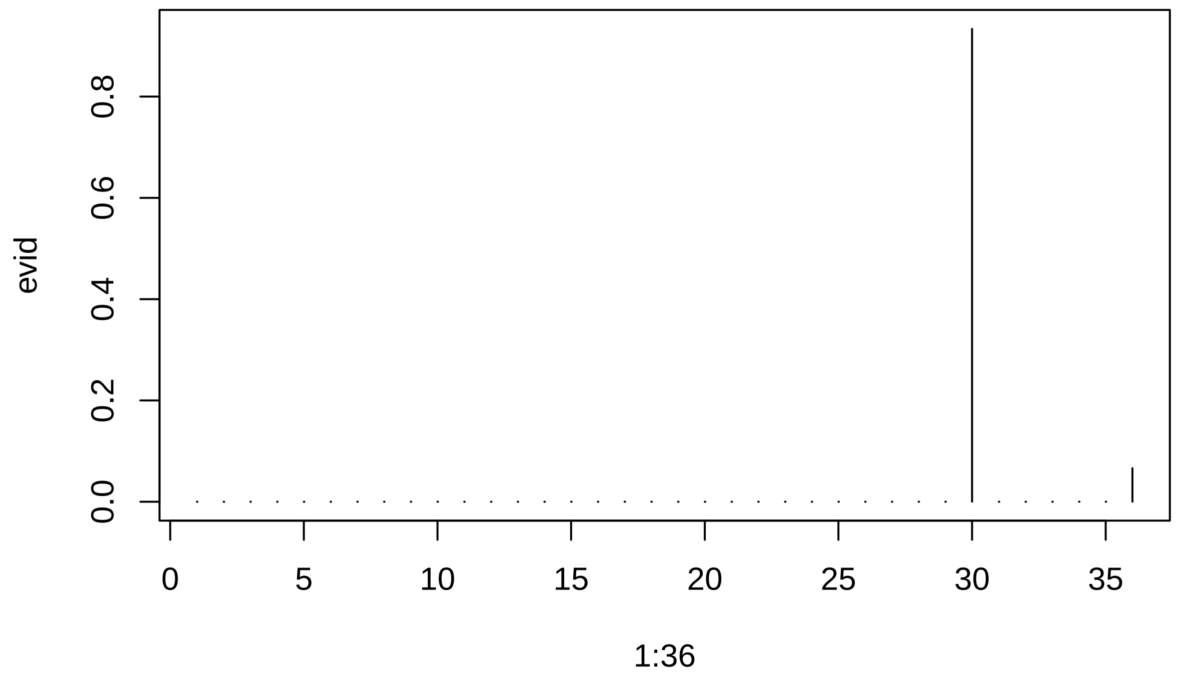
```
logEvid.sorted = sort(logEvid, decreasing = T, index.return=T)
ind = logEvid.sorted$ix[1:5]
k.and.l = expand.grid(l.vals, k.vals)
colnames(k.and.l) = c('l', 'k')
k.and.l[ind,]
```

```
##    l k
## 30 5 4
## 36 5 5
## 24 5 3
## 18 5 2
## 29 4 4
```

The best model, chosen by Bayesian evidence, has k=4 and l=5.

Calculate evidences:

```
logEvid.max = max(logEvid)
logEvid = logEvid - logEvid.max
evid = exp(logEvid)
evid = evid / sum(evid)
plot(1:36, evid, type='h')
```



The normalized evidence of the models with the highest evidence:

```
evid[ind[1]]
```

```
## [1] 0.9337331
```

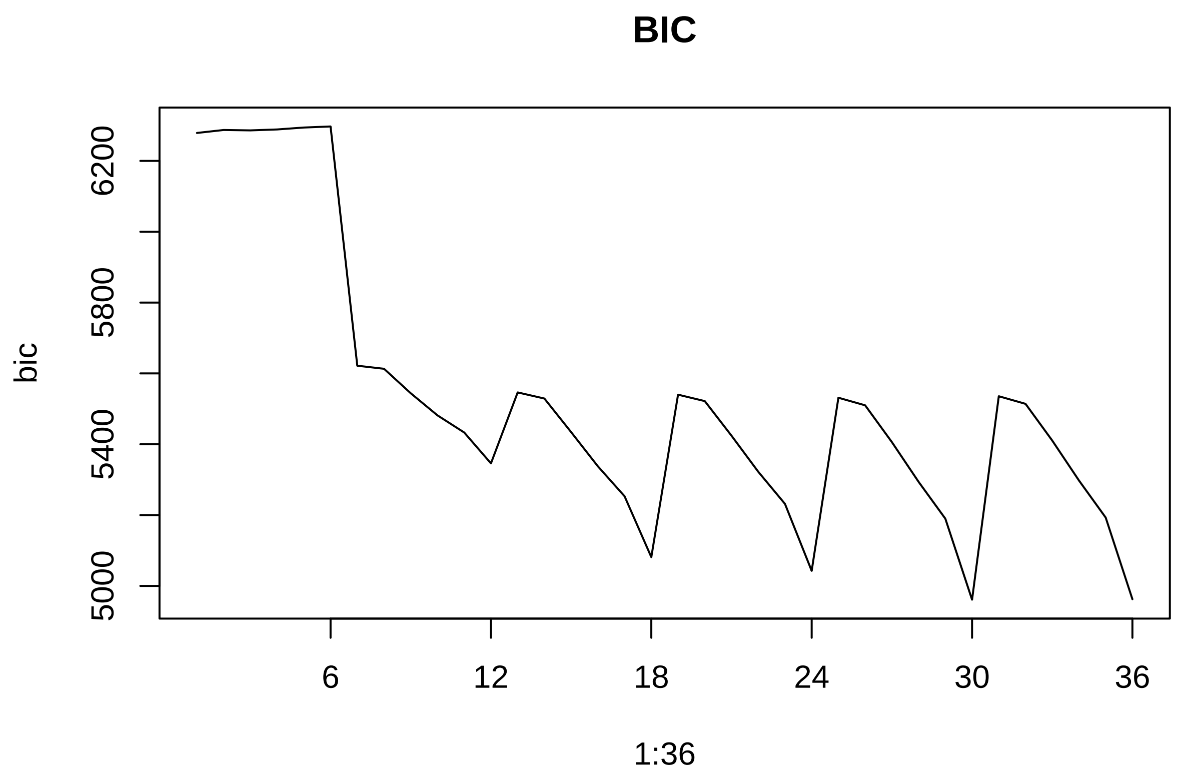
Again, this chosen model may be overfitting because k=4 means the model contains a degree-4 polynomial. The degree is still too high for the model to make reasonable predictions.

### b and c

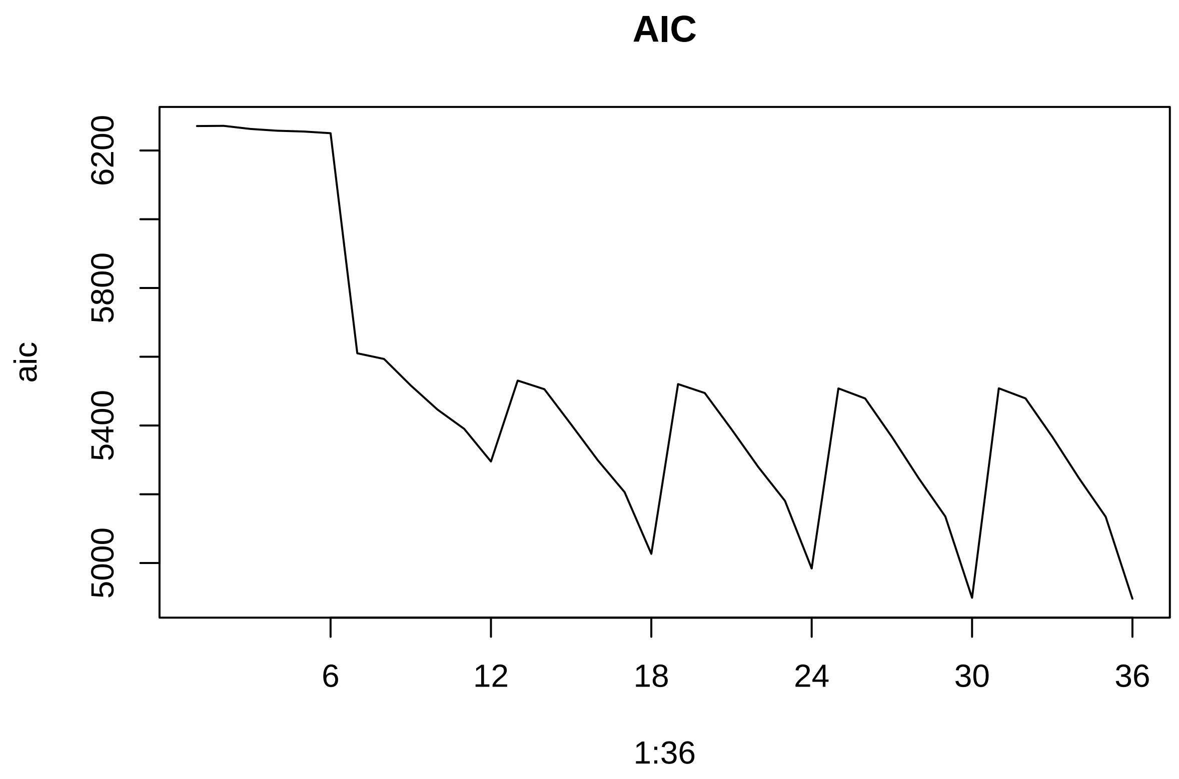
Calculate the best model using the BIC and AIC. Compare with the models obtaining high evidence from part (a).

```
k.vals = 0:5
l.vals = 0:5
bic = rep(-9999, 36)
aic = rep(-9999, 36)
ind = 1 # index for accessing elements of logEvid

Poly = matrix(rep(1, n), ncol=1) # polynomial part of the model matrix
for (k in k.vals) {
  Xmat = Poly
  for (l in l.vals) {
    p = ncol(Xmat)+1 # number of parameters, betas AND sigma
    md = lm(y ~ -1 + Xmat)
    sig.hat.squared = sum((md$residuals)^2) / n
    bic[ind] = n + n*log(2*pi*sig.hat.squared) + p*log(n)
    aic[ind] = n + n*log(2*pi*sig.hat.squared) + p*2

    ind = ind + 1
    Xmat = cbind(Xmat, cos(2*pi*(l+1)/12 * (1:n)))
    Xmat = cbind(Xmat, sin(2*pi*(l+1)/12 * (1:n)))
  }
  Poly = cbind(Poly, ((1:n)/n)^(k+1))
}
```

```
plot(1:36, bic, type='l', xaxt='n', main='BIC')
axis(side=1, at=(1:6)*6)
```



```
plot(1:36, aic, type='l', xaxt='n', main='AIC')
axis(side=1, at=(1:6)*6)
```



```
k.and.l[which.min(bic), ] # best model chosen by bic
```

```
##    l k
## 30 5 4
```

```
k.and.l[which.min(aic), ] # best model chosen by aic
```

```
##    l k
## 36 5 5
```

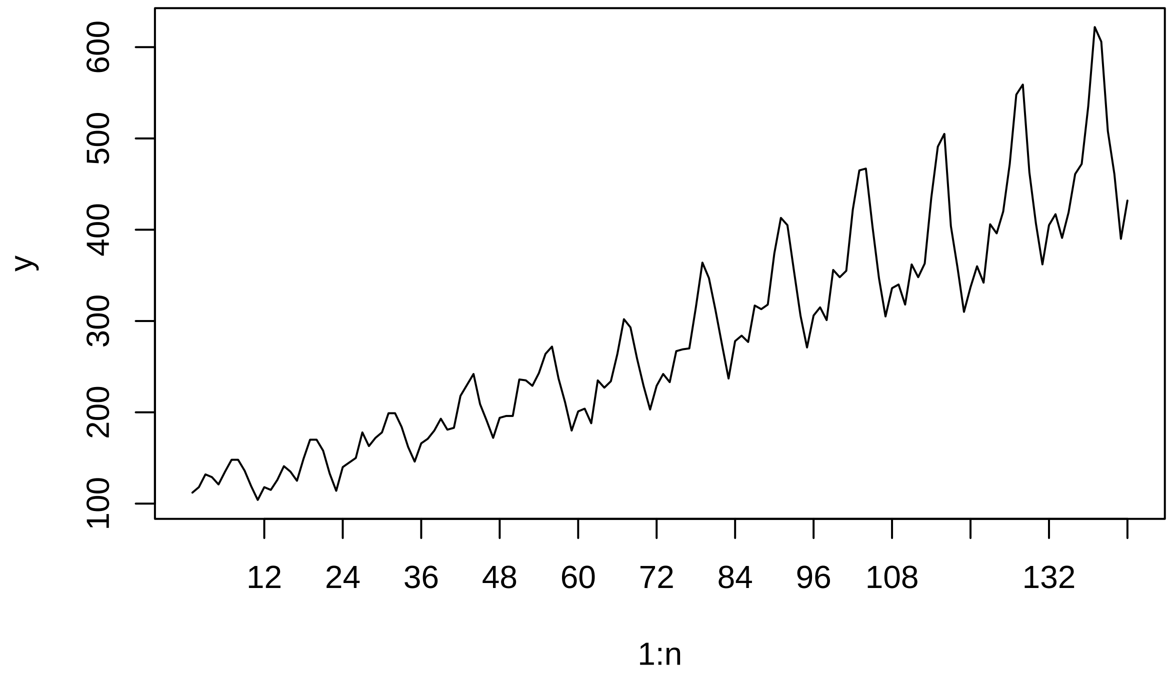
We see that BIC chooses the same model as Bayesian evidence does, whereas AIC chooses the model with a larger l, i.e. l=5. This makes sense because BIC penalizes models with many parameters. From the plots of BIC and AIC above, we see that BIC and AIC both favor large l, just like Bayesian Evidence.

## 6

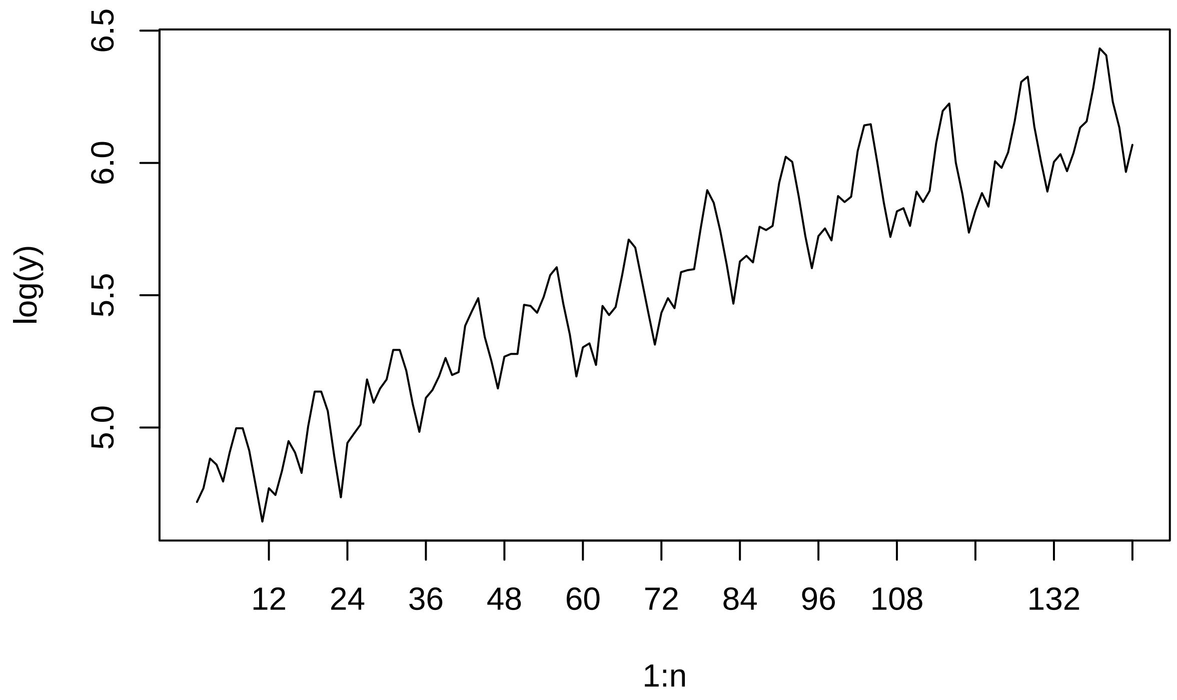
A classic time series dataset is the Box and Jenkins airline passenger data (can be accessed in R via data(AirPassengers)). This gives monthly totals of international airline passengers from 1949 to 1960. There are n = 144 observations in total corresponding to 12 years.

```
data(AirPassengers)
```

```
y = as.numeric(AirPassengers)
n = length(y)
plot(1:n, y, type='l', xaxt='n')
axis(side=1, at=(1:12)*12)
```



```
plot(1:n, log(y), type='l', xaxt='n')
axis(side=1, at=(1:12)*12)
```



## a Use the Evidence-based Bayesian model selection method from Lecture 11 to calculate the evidences for each of the above $2^{19}$ models. Which models have high evidences? Do the frequencies k/n appearing in the high evidence models have any intuitive meaning? (8 points)

```r
fullX = matrix(1, nrow=n, ncol=38) # 38 = 2 + 18*2
fullX[,2] = 1:n
for (k in 1:18) {
  fullX[,2*k+1] = cos(2*pi*k/n*(1:n))
  fullX[,2*k+2] = sin(2*pi*k/n*(1:n))
}
all01 = expand.grid(replicate(18, 0:1, simplify = FALSE))

logEvid.M = rep(-9999, nrow(all01)) # log evidences for the models in M_S
logEvid.L = rep(-9999, nrow(all01)) # log evidences for the models in LM_S
```

```r
# takes 5-10 minutes
inds.init = rep(F, 38) # initial indices
inds.init[1] = T; inds.init[2] = T
for(mm in 1:100) # nrow(all01)
{
  inds = inds.init
  if(mm >= 2) { # excluding the case when all01 is all FALSE
    where.true = (1:18)[as.logical(all01[mm,])]
    inds[2 + 2*where.true-1] = T
    inds[2 + 2*where.true] = T
    # I later knew that using rep(each=2) is easier
  }
  Xmat = fullX[,inds]
  p = ncol(Xmat)
  md = lm(y ~ -1 + Xmat)
  # the following log evidence formula for model mm comes from lecture 11 notes.
  logEvid.M[mm] = (lgamma(p/2)) - ((p/2)*(log((sum(md$fitted.values^2))))) + (lgamma((n-p-1)/2)) - (((n

  md = lm(I(log(y)) ~ -1 + Xmat)
  logEvid.L[mm] = (lgamma(p/2)) - ((p/2)*(log((sum(md$fitted.values^2))))) + (lgamma((n-p-1)/2)) - (((n
}
```

```
save(logEvid.M, file='logEvid.M.RData')
save(logEvid.L, file='logEvid.L.RData')
```

```
load('logEvid.M.RData')
load('logEvid.L.RData')
```

Calculate evidences from log evidences:

```
logEvid.max = max(logEvid.M, logEvid.L)
evid.M = exp(logEvid.M - logEvid.max)
evid.L = exp(logEvid.L - logEvid.max)
normalizing.const = sum(evid.M) + sum(evid.L)
evid.M = evid.M / normalizing.const
evid.L = evid.L / normalizing.const
```

It turns out that the log evidences of models in $LM_S$ are significantly higher than those of $M_S$ (see part b). So the high evidence models must be in $LM_S$, and so we only sort `evid.L` to get our top 10 models below:

```
ind = sort(evid.L, decreasing = T, index.return=T)$ix[1:10]
all01[ind,]
```

```
##      Var1 Var2 Var3 Var4 Var5 Var6 Var7 Var8 Var9 Var10 Var11 Var12 Var13 Var14
## 2049    0    0    0    0    0    0    0    0    0     0     0     1     0     0
## 2050    1    0    0    0    0    0    0    0    0     0     0     1     0     0
## 2054    1    0    1    0    0    0    0    0    0     0     0     1     0     0
## 2053    0    0    1    0    0    0    0    0    0     0     0     1     0     0
## 2051    0    1    0    0    0    0    0    0    0     0     0     1     0     0
## 3073    0    0    0    0    0    0    0    0    0     0     1     1     0     0
## 6145    0    0    0    0    0    0    0    0    0     0     0     1     1     0
## 2081    0    0    0    0    0    1    0    0    0     0     0     1     0     0
## 3074    1    0    0    0    0    0    0    0    0     0     1     1     0     0
## 2052    1    1    0    0    0    0    0    0    0     0     0     1     0     0
##      Var15 Var16 Var17 Var18
## 2049     0     0     0     0
## 2050     0     0     0     0
## 2054     0     0     0     0
## 2053     0     0     0     0
## 2051     0     0     0     0
## 3073     0     0     0     0
## 6145     0     0     0     0
## 2081     0     0     0     0
## 3074     0     0     0     0
## 2052     0     0     0     0
```

All of the top 10 models have frequency 12/n, which corresponds to period n/12=12 months. That makes sense because from the plot of the data, we can already see clear yearly seasonality.

### b

For fixed S, which of the two models $M_S$ and $LM_S$ generally has higher evidence? (3 points)

Models in $LM_S$ have higher evidence, as shown below:

```
min(logEvid.L)
```

```
## [1] 31.88992
```

```
max(logEvid.M)
```

```
## [1] -594.1739
```

This makes sense. Note that all the models considered here cannot fit data with varying oscillation amplitudes well; they are designed to fit data with constant oscillation amplitudes. The amplitude of the oscillations in the original data increases as time goes on, whereas in the logged data, the increase of amplitude is much less noticeable due to taking logarithm.