Application Questions

Since there's some compelling was too slow, I can only do the screen shot for the problems, but not knit it to pdf.

```
---
title: "cs539 hw4 Application Questions"
author: "Enbo Tian"
date: '2022-03-25'
output: pdf_document
---
# Viterbi Decoding Algorithm
## a)
```{r}
library(HMM)
# Viterbi algorithm
Viterbi=function(v,a,b,initial_distribution) {

  T = length(v)
  M = nrow(a)
  prev = matrix(0, T-1, M)
  omega = matrix(0, M, T)

  omega[, 1] = log(initial_distribution * b[, v[1]])
  for(t in 2:T){
    for(s in 1:M) {
      probs = omega[, t - 1] + log(a[, s]) + log(b[s, v[t]])
      prev[t - 1, s] = which.max(probs)
      omega[s, t] = max(probs)
    }
  }

  S = rep(0, T)
  last_state=which.max(omega[,ncol(omega)])
  S[1]=last_state

  j=2
  for(i in (T-1):1){
    S[j]=prev[i,last_state]
    last_state=prev[i,last_state]
    j=j+1
  }

  S[which(S==1)]='A'
  S[which(S==2)]='B'

```
```

```
  S[which(S==1)]='A'
  S[which(S==2)]='B'

  S=rev(S)

  return(S)

}

#repeat the result presented in the pdf file

hmm = initHMM(c("H","L"), c("A","C","G","T"),startProbs = c(0.5,0.5),
transProbs=matrix(c(0.5,0.5,0.4,0.6),2),
  emissionProbs=matrix(c(0.2,0.3,0.3,0.2,0.3,0.2,0.2,0.3),2))

observations = c("G","G","C","A","C","T","G","A","A")
viterbi = viterbi(hmm,observations)
viterbi
```

```
[1] "H" "H" "H" "L" "L" "L" "L" "L" "L"
```

```
## b)
```{r}
#the observed sequence of: AGTCGTA
observations = c("A","G","T","C","G","T","A")
viterbi = viterbi(hmm,observations)
viterbi
```
```

```
[1] "L" "L" "L" "H" "H" "L" "L"
```

```
# Bayesian Filtering
## a)
```{r}
# create X and Y
x0 <- rnorm(1,0,1)
x = x0
X <- rep(0,100)
Y <- rep(0,100)
for(i in 1:100){
  X[i] <- rnorm(1,0.99*x+0.1,0.1)
  x <- X[i]
  Y[i] <- rnorm(1,-2*x+1,0.4)
}
X <- c(x0,X)
head(X)
head(Y)
```
```

```
[1] 1.082742 1.113829 1.268545 1.397311 1.623790 1.670091
[1] -1.596619 -1.715995 -1.250474 -2.215364 -2.754681 -2.480468
```


```
## b)
```{r}
pygx <- pnorm(Y,-2*X[-1],0.4)
pxgx1 <- pnorm(X[-1],0.99*X[1:100]+0.1,0.1)
pxkgy1k1 <- pygx*pxgx1
z <- pygx[100]*pxkgy1k1[100]
pxkgy1k <- 1/z * pygx *pxkgy1k1
head(pxkgy1k)
```
```

```
[1] 0.2707539 0.7765434 0.7172082 0.9888070 0.3311852 0.6504082
```

```
## c)
```{r}
post <- qnorm(pxkgy1k,0.99*X[1:101]+0.1,0.1)
post = post[-100]
mean(post)
conf <- post/X[-1]
alpha <- 1-conf
length(alpha[abs(alpha)<0.05])
```
```

```
[1] 2.51459
[1] 1.0160477 0.9991755 0.9986455 0.9975084 1.0124402 0.9994709
[1] -0.0160477017  0.0008244506  0.0013545006  0.0024916007 -0.0124402129
0.0005290819
[1] 95
```

```
## d)
```{r}
library("readxl")
data<-read_excel("filter_problem.xlsx", col_names = c("index","Xk","Yk"))
Y <- data$Yk[-1]
X <- data$Xk
pygx <- pnorm(Y,-2*X[-1],0.4)
pxgx1 <- pnorm(X[-1],0.99*X[1:100]+0.1,0.1)
pxkgy1k1 <- pygx*pxgx1
z <- pygx[100]*pxkgy1k1[100]
pxkgy1k <- 1/z * pygx *pxkgy1k1
head(pxkgy1k)
post <- qnorm(pxkgy1k,0.99*X[1:101]+0.1,0.1)
post = post[-100]
mean(post)
conf <- post/X[-1]
alpha <- 1-conf
length(alpha[abs(alpha)<0.05])
```
```

```
[1] 58.38498 73.13255 74.34234 73.62419 69.11370 67.46193
```

Alpha = [1] 96

```
## e)
```{r}
x0 <- rnorm(1,0,1)
x = x0
X <- rep(0,100)
for(i in 1:100){
  X[i] <- rnorm(1,x,0.2)
  x <- X[i]
}
X <- c(x0,X)
pygx <- pnorm(Y,-2*X[-1],0.4)
pxgx1 <- pnorm(X[-1],X[1:100],0.2)
pxkgy1k1 <- pygx*pxgx1
z <- pygx[100]*pxkgy1k1[100]
pxkgy1k <- 1/z * pygx *pxkgy1k1
head(pxkgy1k)
post <- qnorm(pxkgy1k,X[1:100],0.2)
post = post[-100]
mean(post)
conf <- post/X[-1]
alpha <- 1-conf
length(alpha[abs(alpha)<0.05])
```
```

```
[1] 3.752094e+89 7.791352e+88 4.688298e+88 4.846695e+86 2.044845e+79 3.585929e+80
Warning in qnorm(pxkgy1k, X[1:100], 0.2) : 产生了NaNs
[1] NaN
Warning in post/X[-1] :
  longer object length is not a multiple of shorter object length
[1] 33
```

```
## f)
Since there is NA value in d and e, y may not follow the distribution which we provided.
In addition,c d have a good confidence but e doesn't.

# Sequential MCMC
```{r}
Y <- data$Yk[-1]
X <- data$Xk
pygx <- pnorm(Y,-2*X[-1],0.4)
pxgx1 <- pnorm(X[-1],0.99*X[1:100]+0.1,0.1)
pxkgy1k1 <- pygx*pxgx1
z <- pygx[100]*pxkgy1k1[100]
pxkgy1k <- 1/z * pygx *pxkgy1k1
#a)
mean(sample(pxkgy1k,100))
#b)
mean(sample(pxkgy1k,1000,replace = TRUE))
```
```

```
[1] 31.32629
[1] 32.46146
```

GP and Linear Regression
##a)
```{r}
library(GPfit)
library(brms)
data <- read.csv("synchronous_machine.csv")
### It is too slow for my computer to do the chain with sampling in the model,
### so I can only pick one parameter for Gp.(nearly 10 min once)
fitgp <- brm(qIy~gp(PF)+e+dIf+If,chain = 4, data = data)
summary(fitgp)
```

```
Warning: Parts of the model have not converged (some Rhats are > 1.05). Be careful
when analysing the results! We recommend running more iterations and/or setting
stronger priors.
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: qIy ~ gp(PF) + e + dIf + If
   Data: data (Number of observations: 557)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Gaussian Process Terms:
             Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sdgp(gpPF)       0.73      0.33     0.38     1.73 1.63        8       18
lscale(gpPF)     0.11      0.10     0.01     0.30 2.56        5       19

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept -1091.42    544.66 -2392.03  -350.81 1.75        6       15
e            -7.07      3.61   -10.51     2.58 1.60        7       13
dIf        -920.50    460.75 -2020.75  -293.37 1.75        6       15
If          927.63    461.14   300.54  2028.76 1.75        6       15

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.45      0.09     0.34     0.57 1.84        6       46

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

b)

```{r}
fit <- lm(qIy~PF+e+dIf+If,data = data)
summary(fit)
```

```
Call:
lm(formula = qIy ~ PF + e + dIf + If, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.70820 -0.39040 -0.03484  0.31406  2.14232

Coefficients: (2 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -7.0398     0.4654  -15.13   <2e-16 ***
PF           10.8102     0.4628   23.36   <2e-16 ***
e                NA         NA      NA       NA
dIf           7.4658     0.2663   28.03   <2e-16 ***
If               NA         NA      NA       NA
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5767 on 554 degrees of freedom
Multiple R-squared:  0.5872,    Adjusted R-squared:  0.5857
F-statistic:   394 on 2 and 554 DF,  p-value: < 2.2e-16
```

c)

MSE of gp is much less than linear regression.