SML HW3

Problem 3

3.1

```r
myOLS <- function(Y, X, is1 = TRUE) {
# Inputs:
# * Y is the vector of length n of response variables
# * X is an n-by-p matrix of numerical covariates (in columns); p < n
# ** assume the columns of X are linearly independent and
# ** do not include the column for the intercept as a part of the X matrix
# * is1 is a logical "flag" whether the intercept is included; is1 = TRUE by default
# Output:
# the function must return a list L with two elements:
# L[1] will contain the vector of OLS/MLE coefficients, betahat
# L[2] will contain standard errors (i.e., estimated standard deviations) for betahat

  n_y = length(Y)
  n_x = nrow(X)
  if(n_x != n_y){
    print("Error: X and Y not of same length")
  }else{
    n = n_y
  }

  p = ncol(X)

  if(is1==TRUE){
    # add intercept to matrix
    X0 = rep(1, n)
    X = cbind(X0, X)
  }

  betahat = solve(t(X)%*%X)%*%t(X)%*%Y
  pred = X%*%betahat # prediction
  sigma_sq <- sum((Y - pred)^2)/(n-p)  # estimate of sigma-squared
  var_covar <- sigma_sq*solve(t(X)%*%X) # variance covariance matrix
  std_err <- sqrt(diag(var_covar)) # standard error

  return(list(betahat, std_err))
}
```

```r
n = 30
set.seed(0)
p = 3
X = matrix(runif(n*p),nrow=n)*2-1
b = seq(1,p,by=1)
Y = X%*%b + rnorm(n)
fit1 = lm(Y ~ X); summary(fit1)
```

```
##
## Call:
## lm(formula = Y ~ X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3701 -0.3304  0.1082  0.4938  2.3930
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.1291     0.1708  -0.756  0.45648
## X1            0.9214     0.2987   3.085  0.00478 **
## X2            2.4021     0.3468   6.926 2.36e-07 ***
## X3            2.7482     0.3452   7.960 1.94e-08 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.9235 on 26 degrees of freedom
## Multiple R-squared:  0.802,  Adjusted R-squared:  0.7792
## F-statistic: 35.11 on 3 and 26 DF,  p-value: 2.711e-09
```

```r
myOLS(Y,X,TRUE)
```

```
## [[1]]
##          [,1]
## X0 -0.1291359
##     0.9214173
##     2.4020865
##     2.7482181
##
## [[2]]
##        X0
## 0.1676341 0.2930771 0.3403307 0.3387858
```

3.2

```r
myPolyReg1 <- function(Y, X1, deg=1) {
# Inputs: same as for myOLS, except
# * X1 is a vector of length n that contain the covariate values (numerical)
# * deg is the degree k (i.e., largest power) of the polynomial fit; k < n ; deg=1 by default.
# Outputs: same as for myOLS
  X = rep(1, n)
  for (i in seq(1, deg)){
    X = cbind(X, X1**i)
  }

  return(myOLS(X=X, Y=Y, is1=FALSE))
}
```

```
n = 30
set.seed(0)
X = runif(n)*4-2 # X is uniformly distributed on [-2,2]
Y = 1 + 3*X -2*X^2 + 1*X^3 + rnorm(n)
fit0 = lm(Y ~ X + I(X^2) + I(X^3))
summary(fit0)
```

```
##
## Call:
## lm(formula = Y ~ X + I(X^2) + I(X^3))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3159 -0.5052 -0.1633  0.5612  1.6267
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.1415     0.2385   4.787 5.90e-05 ***
## X             2.9127     0.3166   9.199 1.17e-09 ***
## I(X^2)       -2.0916     0.1318 -15.866 6.88e-15 ***
## I(X^3)        1.0150     0.1221   8.313 8.56e-09 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.8295 on 26 degrees of freedom
## Multiple R-squared:  0.9856, Adjusted R-squared:  0.984
## F-statistic: 594.4 on 3 and 26 DF,  p-value: < 2.2e-16
```

```
myPolyReg1(Y,X,deg=3)
```

```
## [[1]]
##        [,1]
## X  1.141535
##    2.912750
##   -2.091594
##    1.015042
##
## [[2]]
##         X
## 0.2384849 0.3166353 0.1318304 0.1221091
```

3.3

```
myAnova1 <- function(Y, XF, is1=TRUE) {
# Inputs: same as for myOLS, except
# * XF is a vector of length n that contain the covariate values (categorical or "factor")
# Outputs: same as for myOLS
  uniq_var <- unique(XF)
  X <- +outer(XF, uniq_var, `==`)
  colnames(X) <- uniq_var

  if(is1==TRUE){
    X = X[,-1]
  }
  return(myOLS(X=X, Y=Y, is1=is1))
}
```

```
n = 30
set.seed(0)
XF = rep(c("A","B","C"),each=10)
Y = rnorm(n) + rep(c(1,2,3),each=10)
fit1 = lm(Y ~ XF)
summary(fit1) # with an intercept
```

```
##
## Call:
## lm(formula = Y ~ XF)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.89887 -0.62259  0.01652  0.70476  2.04573
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.3589     0.2828   4.805 5.15e-05 ***
## XFB           0.2786     0.4000   0.697 0.492057
## XFC           1.7105     0.4000   4.276 0.000212 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8944 on 27 degrees of freedom
## Multiple R-squared:  0.4382, Adjusted R-squared:  0.3966
## F-statistic: 10.53 on 2 and 27 DF,  p-value: 0.0004163
```

```r
myAnova1(Y, XF, is1=TRUE)
```

```
## [[1]]
##         [,1]
## X0 1.3589240
## B  0.2785947
## C  1.7104858
##
## [[2]]
##        X0         B         C
## 0.2777328 0.3927735 0.3927735
```

```r
fit0 = lm(Y ~ -1 + XF)
summary(fit0) # without an intercept
```

```
##
## Call:
## lm(formula = Y ~ -1 + XF)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.89887 -0.62259  0.01652  0.70476  2.04573
##
## Coefficients:
##     Estimate Std. Error t value Pr(>|t|)
## XFA   1.3589     0.2828   4.805 5.15e-05 ***
## XFB   1.6375     0.2828   5.790 3.69e-06 ***
## XFC   3.0694     0.2828  10.853 2.39e-11 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.8944 on 27 degrees of freedom
## Multiple R-squared:  0.8659, Adjusted R-squared:  0.851
## F-statistic: 58.13 on 3 and 27 DF,  p-value: 6.599e-12
```

```r
myAnova1(Y, XF, is1=FALSE)
```

```
## [[1]]
##       [,1]
## A 1.358924
## B 1.637519
## C 3.069410
##
## [[2]]
##         A         B         C
## 0.2828292 0.2828292 0.2828292
```

Problem 4

4.1

```
myFullObj <- function(par) {
# Inputs:
# * b is the vector of regression coefficients, b=[b0,b1];
# * sig is the standard dev of errors; sig > 0
# Output: the negative log-likelihood of the observed data (Y given X) evaluated at b and sig
  b = par[1:2]
  sig = par[3]
  miu <- b[1]+mean(X)*b[2]
  (-1)*sum(dnorm(Y, mean=miu, sd = sig, log=TRUE))
}
```

4.2

```
sigKnown = 2
myObj1 <- function(b){
  myFullObj(c(b,sigKnown))
}

optim(par=c(1,3),myObj1,method="BFGS")
```

```
## $par
## [1] 1.676241 3.076002
##
## $value
## [1] 53.16802
##
## $counts
## function gradient
##        4        3
##
## $convergence
## [1] 0
##
## $message
## NULL
```

4.3

```
optim(par = c(1,3,1), myFullObj,method="L-BFGS-B",lower=c(-Inf, -Inf, 10^(-5)))
```

```
## $par
## [1] 1.676241 3.076002 1.132013
##
## $value
## [1] 46.28807
##
## $counts
## function gradient
##       12       12
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```