Homework 2

1.

a. 

\[
X^T X = \begin{bmatrix}
    n & \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} z_i \\
    \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i z_i \\
    \sum_{i=1}^{n} z_i & \sum_{i=1}^{n} x_i z_i & \sum_{i=1}^{n} z_i^2 \\
\end{bmatrix}
\]

When the upper triangular elements are suppressed due to symmetry,

\[
X^T X = \begin{bmatrix}
    n & \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i z_i \\
    \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} z_i \\
\end{bmatrix}
\]
Let \( \hat{A} = \begin{bmatrix} X'X & X'Y \\ Y'X & Y'Y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \\ \sum_{i=1}^{n} z_i & \sum_{i=1}^{n} x_i z_i & \sum_{i=1}^{n} z_i^2 \\ \sum_{i=1}^{n} y_i & \sum_{i=1}^{n} x_i y_i & \sum_{i=1}^{n} z_i y_i & \sum_{i=1}^{n} y_i^2 \end{bmatrix}. \)

(i) Sweep on the first column to obtain:

\[
\hat{A} = \begin{bmatrix}
-\frac{1}{n} \\
\frac{\sum_{i=1}^{n} x_i}{n} \\
\frac{\sum_{i=1}^{n} x_i^2}{n} - \left( \frac{\sum_{i=1}^{n} x_i}{n} \right)^2 \\
\frac{\sum_{i=1}^{n} z_i}{n} \\
\frac{\sum_{i=1}^{n} x_i z_i}{n} - \left( \frac{\sum_{i=1}^{n} x_i}{n} \right) \left( \frac{\sum_{i=1}^{n} z_i}{n} \right) \\
\frac{\sum_{i=1}^{n} z_i^2}{n} - \left( \frac{\sum_{i=1}^{n} z_i}{n} \right)^2 \\
\frac{\sum_{i=1}^{n} y_i}{n} \\
\frac{\sum_{i=1}^{n} x_i y_i}{n} - \left( \frac{\sum_{i=1}^{n} x_i}{n} \right) \left( \frac{\sum_{i=1}^{n} y_i}{n} \right) \\
\frac{\sum_{i=1}^{n} z_i y_i}{n} - \left( \frac{\sum_{i=1}^{n} z_i}{n} \right) \left( \frac{\sum_{i=1}^{n} y_i}{n} \right) \\
\frac{\sum_{i=1}^{n} y_i^2}{n} - \left( \frac{\sum_{i=1}^{n} y_i}{n} \right)^2 
\end{bmatrix}
\]

by changing the notation (e.g. \( \sum (z_i - \overline{z})(y_i - \overline{y}) = S_{ZY} \) and \( \sum (x_i - \overline{x})^2 = S_{XX} \) etc.)

\[
= \begin{bmatrix}
-\frac{1}{n} \\
\bar{x} \\
\bar{y} \\
\frac{1}{n} S_{XX} \\
\frac{1}{n} S_{xz} & \frac{1}{n} S_{zz} \\
\bar{y} S_{xy} & \bar{y} S_{zy} & \bar{y} S_{yy} 
\end{bmatrix}
\]
(ii) Sweep on the second column to obtain:

\[
\hat{A} = \begin{bmatrix}
-\frac{1}{n} - \frac{(\bar{X})^2}{S_{XX}} & 0 & 0 \\
\bar{X} S_{XX} & -S_{XX} S_{Z} - S_{Z}^2 & S_{ZY} - S_{XY} S_{XZ} \\
\bar{Y} S_{XX} & 0 & S_{YY} - S_{XY} S_{XX} \\
\bar{Y} - \bar{X} & S_{XY} & S_{YY} - S_{XY} S_{XX}
\end{bmatrix}
\]

(iii) Sweep on the third column to obtain:

\[
\hat{A} = \begin{bmatrix}
-\frac{1}{n} - \frac{(\bar{X})^2}{S_{XX}} - \frac{S_{XX}}{S_{XX} S_{Z} S_{Z} - S_{Z}^2} \left[ \bar{X} S_{XX} - \bar{X} S_{XZ} \right] \left[ \bar{X} S_{XX} S_{Z} S_{Z} - S_{Z}^2 \right] \\
\bar{Y} S_{XX} & S_{XX} S_{Z} S_{Z} - S_{Z}^2 & S_{ZY} - \frac{S_{XY} S_{XZ}}{S_{XX}} \\
\bar{Y} S_{XX} & S_{XX} S_{Z} S_{Z} - S_{Z}^2 & S_{YY} - \frac{S_{XY} S_{XX}}{S_{XX}} \\
\bar{Y} - \bar{X} & S_{XY} & S_{YY} - \frac{S_{XY} S_{XX}}{S_{XX}} \\
\frac{S_{XY}}{S_{XX} S_{Z} S_{Z} - S_{Z}^2} & S_{YY} - \frac{S_{XY} S_{XX}}{S_{XX}} & S_{YY} - \frac{S_{XY} S_{XX}}{S_{XX}}
\end{bmatrix}
\]

Although looking a bit ugly (it’s tedious to simplify them), it actually is the following:

\[
\begin{bmatrix}
-\left( X^T X \right)^{\dagger} X^T Y \\
\left( X^T X \right)^{\dagger} X^T Y
\end{bmatrix}
\]

And \( \left( X^T X \right)^{\dagger} X^T Y \) is the same \( \hat{\beta} \) that can be obtained from the normal equation

\[
\left( X^T X \right) \hat{\beta} = X^T Y \quad \text{and} \quad Y^T Y - Y^T X \left( X^T X \right)^{\dagger} X^T \text{ is } n \hat{\sigma}^2.
\]

Thus the final results of the sweep
procedure are equivalent to the solutions of the normal equations. Checking one by one is quite a tedious job and is omitted here due to the limit of space.

When \( z_i = ax_i \) for some positive \( a \), \( \bar{z} = a\bar{x} \), \( S_{ZZ} = a^2 S_{XX} \), \( S_{XZ} = a S_{XY} \), and \( S_{ZY} = a S_{XY} \).

By plugging in these, we can see the divide-by-zero problems in the third Sweep procedure. Therefore, this algorithm fails to work as in the other algorithms. For example, in the algorithm using the normal equations, matrices cannot be inverted due to rank deficiency. This is the multicollinearity problem in a multiple linear regression.

2.

a.

Let \( \tilde{y} \) and \( \tilde{\varepsilon} \) be column vectors, that is, \( \tilde{y} = (y_i) \), \( \tilde{\varepsilon} = (\varepsilon_i) \).

Then, \( \tilde{y} = X\beta + \tilde{\varepsilon} \) \hspace{1cm} (1)

And let \( W = (w_{ij}) \), a matrix defined as given, whose elements \( w_{ij} \) are weights of variable variances for the units such that \( \text{var}(\varepsilon_i) = \sigma^2 w_{ii} \) (or \( \text{var}(\varepsilon) = \sigma^2 W \) in matrix terms), where \( \sigma^2 \) is a constant.

We can decompose the matrix \( W \) such that \( W = PP^T \).

By left multiplying (1) by \( P^{-1} \) we have

\[ P^{-1}\tilde{y} = P^{-1}X\beta + P^{-1}\tilde{\varepsilon} \] \hspace{1cm} (2)

Let \( P^{-1}\tilde{y} = B \), \( P^{-1}X = A \), and \( P^{-1}\tilde{\varepsilon} = \eta \), and (2) becomes:

\[ B = A\beta + \eta \] \hspace{1cm} (3)
Here \[ \text{var}(\eta) = P^{-1} \text{var}(\varepsilon) (P^{-1})' = P^{-1} \sigma^2 W (P^{-1})' = \sigma^2 P^{-1} PP^T (P^T)' = \sigma^2 I_n \], where \( I_n \) is \((n \times n)\) identity matrix.

Therefore, under the assumption of normality, (3) can be solved for \( \hat{\beta}_{\text{MLE}} \) using the normal equation (This procedure was shown in class, and thus omitted due to limited space): \( A^T A \hat{\beta} = A^T B \)  

\[ \begin{align*}
A^T A \hat{\beta} &= A^T B \\
\Rightarrow \quad \left(P^{-1} X\right) (P^{-1} X)^T \hat{\beta} &= \left(P^{-1} X\right) (P^{-1} y) \\
\Rightarrow \quad X^T (P^{-1}) P^{-1} X \hat{\beta} &= X^T (P^{-1}) P^{-1} y \\
\Rightarrow \quad X^T W^{-1} X \hat{\beta} &= X^T W^{-1} y
\end{align*} \]  

\[ \begin{align*}
(5) \text{ is the same as } **.
\end{align*} \]

\textbf{b.}  

First, obtain a matrix \( P \) such that \( PP^T = W \). Since \( W \) is a diagonal matrix, \( P \) can easily be obtained by taking a square root of each diagonal element. Then, left-multiply \( X \) by the inverse of \( P \) (is called A for convenience). Conduct QR decomposition on A. Finally, solve for \( \hat{\beta} \) from \( R \hat{\beta} = Q^T P^{-1} y \) by back-solving. This algorithm is illustrated in the following R example. Although this may not be the best way to implement this algorithm (e.g., we can use backsolving instead of inverting matrices), it shows that the obtained solution is the same as MLE shown in (a).

\begin{verbatim}
W <- diag(c(2,4,6,8),nrow=4) # 4*4 diagonal matrix
### This W matrix is to see whether the square matrix check is working.
# W <- matrix(c(1,1,1,2,2,2),nrow=2)
y <- c(4,7,8,9) # y vector
\end{verbatim}
X <- cbind(c(1,1,1,1),c(1,2,3,4)) # design matrix with one IV

### This is a function to obtain beta.hat
### for weighted least squares estimation
### using QR decomposition.
library(MASS) # To use ginv function for generalized inverse
beta.qr <- function(W,X,y)
{
  if (dim(W)[1]!=dim(W)[2]) stop("W is not a square matrix")
  else {
    n <- dim(W)[1]
    P <- matrix(0,nrow=n,ncol=n)
    for (i in 1:n)
      P[i,i] <- sqrt(W[i,i])
    A <- ginv(P)**X
    qr <- qr(A)
    Q <- qr.Q(qr)
    R <- qr.R(qr)
    beta.hat <- ginv(R)**t(Q)**ginv(P)**y
  }
  beta.hat
}

### The result is:
beta.qr(W,X,y)
# 
# [1,] 2.517241
# [2,] 1.793103

### Which should be the same as the following
beta.wls <- ginv(t(X)**ginv(W)**X)**t(X)**ginv(W)**y
# 
# [1,] 2.517241
# [2,] 1.793103

3.
The given lr.r file contains all the necessary functions and examples, so the functions
were tested on the provided examples simply by running the given file. And it was
checked whether the two algorithms produce equivalent results like the following (I do
not see anything further required to do in this question):

source("c:\\chanho\\lr.r")
4.

Necessary functions are given in the given zip.r file, and these functions were adopted for this example. Newton’s method and Fisher-scoring method were fastest by taking 6 iterations from the initial value of 3, while Simple scaling took as many as 72 iterations. It was also checked whether the 4 methods reached maximum, and every method except Simple scaling did so. Although Simple scaling method forcibly stopped iterations after maximum (100), the 100\textsuperscript{th} value is close to the tolerance. So we may conclude that the 4 methods were more or less successfully estimated the parameter. The R code for these and the resulting plots are as follows:

```r
source("c:\\chanho\\zip.r")
xx <- c(0,0,0,0,0,0,2,2,4,5,8,9)
n0=sum(xx==0)
nn=length(xx)
sumx=sum(xx)

# Newton's method
test1 <- root(xx,theta0=3, score=dl, dscore=d2l)
length(test1)
# [1] 6

# Fisher-scoring
test2 <- root(xx,theta0=3, score=dl, dscore=fi )
length(test2)
# [1] 6

# Simple scaling (start from above thetahat)
test3 <- root(xx,theta0=3, score=dl, dscore=bo )
length(test3)
# [1] 72
```
# Secant method

test4 <- secant(xx, theta0=3, theta1=2.9, score=dl)
length(test4)
# [1] 9

### Checking whether the iterations reached maximum
### TRUE if maximum is reached

f1 <- dl(test1, n0, nn, sumx)
(f1[length(test1)] < 1e-6)
# [1] TRUE

f2 <- dl(test2, n0, nn, sumx)
(f2[length(test2)] < 1e-6)
# [1] TRUE

f3 <- dl(test3, n0, nn, sumx)
(f3[length(test3)] < 1e-6)
# [1] FALSE

f3[length(test3)]
# [1] 3.414067e-05

f4 <- dl(test4, n0, nn, sumx)
(f4[length(test4)] < 1e-6)
# [1] TRUE

### Necessary for plotting

supp <- seq(1, 6, length=70)
ll <- -nn*log(2) + n0*log(1 + exp(-supp)) - supp*(nn-n0) + sumx*log(supp)
lld <- dl(supp, n0, nn, sumx)
ll2d <- d2l(supp, n0, nn, sumx)

# a plot

par(mfrow=c(1,3), mar=c(3,3,1,1), mgp=c(2,1,0))

plot( supp, ll, type="b", ylab="loglikelihood" )
plot( supp, lld, type="b", ylab="score" )
abline(h=0)
plot( supp, ll2d, type="b", ylab="observed info" )
5. 

\[ Y_i \mid x_i \sim \text{Poisson}(\exp\{\theta x_i\}) = \text{Poisson}(\mu_i) \] 

by letting \( \exp\{\theta x_i\} = \mu_i \).

Then \( P(Y_i = y_i \mid x_i) = \frac{\exp\{\mu_i\}^{y_i}}{y_i!} \).

\[ L(\bar{\mu}) = \prod_{i=1}^{n} \frac{\exp\{\mu_i\}^{y_i}}{y_i!}, \] where \( \bar{\mu} \) is a vector for \( \mu_i \).

The log likelihood function is:

\[ \ell(\bar{\mu}) = \sum_{i=1}^{n} \left[ (-\mu_i) + y_i \log \mu_i - \log(y_i!) \right] \]

The score function for \( \theta \) is:

\[ S(\theta) = \frac{\partial \ell(\bar{\mu})}{\partial \theta} = \sum_{i=1}^{n} \left[ -x_i \exp\{\theta x_i\} + y_i x_i \right] \]

\[ \text{………… (1)} \]
Since $E[Y_i] = \exp\{\theta x_i\}$, $E[S(\theta)] = 0$. Therefore, the expected score equals zero.

The Fisher information for $\theta$ is:

$$I(\theta) = E \left[ \frac{\partial^2 S(\theta)}{\partial \theta^2} \right] = \sum_{i=1}^{n} \left[ x_i \exp\{\theta x_i\} \right]^2$$

The MLE for $\theta$ is obtained by solving $S(\hat{\theta}) = 0$ for $\hat{\theta}$. Then, an expression for the observed information for $\theta$ is:

$$J_n(\hat{\theta}) = \frac{\partial^2 \ell(\theta)}{\partial \theta^2} \bigg|_{\theta = \hat{\theta}} = \sum_{i=1}^{n} \left[ x_i \exp\{\theta x_i\} \right]^2$$

The MLE and its standard error are computed to be 0.940894 and 0.3353144, respectively using the following R code. Plots are also shown below.

```r
loglik <- function(theta) {
  x <- c(-2,-1,0,1,2)
  y <- c(0,3,2,8,5)
  mu <- exp(theta*x)
  sum <- 0
  for (i in 1:5)
    sum <- sum + mu[i] + y[i]*log(mu[i]) - log(factorial(y[i]))
  return(sum)
}

score <- function(theta) {
  x <- c(-2,-1,0,1,2)
  y <- c(0,3,2,8,5)
  mu <- exp(theta*x)
  sum <- 0
  for (i in 1:5)
    sum <- sum - x[i]*mu[i] + y[i]*x[i]
  return(sum)
}

se <- function(theta) {
  x <- c(-2,-1,0,1,2)
  y <- c(0,3,2,8,5)
  mu <- exp(theta*x)
  info <- 0
  for (i in 1:5)
    info <- info + x[i]^2*mu[i]
  sterr <- 1/info
  return(sterr)
}
```
```
supp <- seq(0,2,length=1e+4) 
ll <- sco <- sterr <- rep(0,1e+4) 
for (i in 1:1e+4) 
  ll[i] <- loglik(supp[i])
for (i in 1:1e+4) 
  sco[i] <- score(supp[i])
for (i in 1:1e+4) 
  sterr[i] <- se(supp[i]) 
par(mfrow=c(1,3))
plot(supp, ll, xlab="theta", ylab="logLik", main="log likelihood", type="l")
plot(supp, sco, xlab="theta", ylab="score", main="score", type="l")
abline(h=0)
plot(supp, sterr, xlab="theta", ylab="sterr", main="standard error", type="l")

### MLE for theta
supp[which.max(ll)]
# [1] 0.940894

### Standard Error
se(supp[which.max(ll)])
# [1] 0.03353144
```

6.

Finished reading the chapter.