STA442 Assignment 1

Alexander Tran

2/7/2023

Question 1

a)

The bounds of Wald's 95% confidence interval are given by $\hat{\pi} \pm 1.96\sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}$, where in this case $\hat{\pi} = \frac{27}{30} = 0.9$ and $n = 30$. Thus, we compute the bounds:

```
## Wald Confidence Interval ##
pihat <- 0.9
n <- 30
alph <- 0.05

L.wald <- pihat - qnorm(1-alph/2)*sqrt((pihat*(1-pihat))/n)
U.wald <- pihat + qnorm(1-alph/2)*sqrt((pihat*(1-pihat))/n)
L.wald
```

```
## [1] 0.7926484
```

```
U.wald
```

```
## [1] 1.007352
```

Resulting in the confidence interval [0.7926484, 1.0073516].

The bounds of 95% score confidence interval are given by
$\left[\hat{\pi}\left(\frac{n}{n+1.96^2}\right) + \frac{1}{2}\left(\frac{1.96^2}{n+1.96^2}\right)\right] \pm 1.96\sqrt{\frac{1}{n+1.96^2}\left[\hat{\pi}(1-\hat{\pi})\left(\frac{n}{n+1.96^2}\right) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{1.96^2}{n+1.96^2}\right)\right]}$,
where again $\hat{\pi} = \frac{27}{30} = 0.9$ and $n = 30$. We compute the bounds:

```
## Scores Confidence Interval ##
nprime <- n + (qnorm(1-alph/2))^2
w1 <- n/nprime
w2 <- ((qnorm(1-alph/2))^2)/nprime
midpoint <- pihat*w1+0.5*w2
L.score <- midpoint - qnorm(1-alph/2)*sqrt((1/nprime)*(pihat*(1-pihat)*w1+0.25*w2))
U.score <- midpoint + qnorm(1-alph/2)*sqrt((1/nprime)*(pihat*(1-pihat)*w1+0.25*w2))
L.score
```

```
## [1] 0.7437892
```

```
U.score
```

```
## [1] 0.9654001
```

Resulting in the confidence interval [0.7437892, 0.9654001].

b)

```r
### Calculating Wald and Score Confidence Interval with Simulation ##
set.seed(1006314089)
N <- 100000
pie <- 0.9 ## pi is already defined in R thus we use "pie"
n <- 30
alph <- 0.05 # For 95% Confidence Interval
y <- rbinom(N, n, pie)
pihat <- y/n

## Wald Confidence Interval ##
L.wald <- pihat - qnorm(1-alph/2)*sqrt((pihat*(1-pihat))/n)
U.wald <- pihat + qnorm(1-alph/2)*sqrt((pihat*(1-pihat))/n)
pi_in_wald_CI <- (pie > L.wald)*(pie < U.wald)
observedConfLevel_WaldCI <- mean(pi_in_wald_CI)
observedConfLevel_WaldCI
```

```
## [1] 0.80917
```

```r
## Scores Confidence Interval ##
nprime <- n + (qnorm(1-alph/2))^2
w1 <- n/nprime
w2 <- ((qnorm(1-alph/2))^2)/nprime
midpoint <- pihat*w1+0.5*w2
L.score <- midpoint - qnorm(1-alph/2)*sqrt((1/nprime)*(pihat*(1-pihat)*w1+0.25*w2))
U.score <- midpoint + qnorm(1-alph/2)*sqrt((1/nprime)*(pihat*(1-pihat)*w1+0.25*w2))
pi_in_score_CI <- (pie >  L.score)*(pie <  U.score)
observedConfLevel_scoreCI <- mean(pi_in_score_CI)
observedConfLevel_scoreCI
```

```
## [1] 0.9737
```

The proportion of the calculated Wald intervals that contain 0.9 is 0.80917. The proportion of the calculated score intervals that contain 0.9 is 0.9737. Comparing these results, the proportion of score intervals that contain 0.9 is significantly higher than the proportion of Wald intervals that contain 0.9. Furthermore, the proportion of score intervals meets expectations and is much closer to the expected 0.95 proportion. On the other hand, the Wald intervals significantly underperform. Given this, the score interval appears to be the more reliable CI.

Question 2

a)

Given $Y \sim Bin(30, \pi)$ and $y = 27$, the likelihood function for estimating π is given by $\ell(\pi) = \pi^{27}(1-\pi)^3$. It follows that the log-likelihood function is given by $L(\pi) = 27\ln(\pi) + 3\ln(1-\pi)$

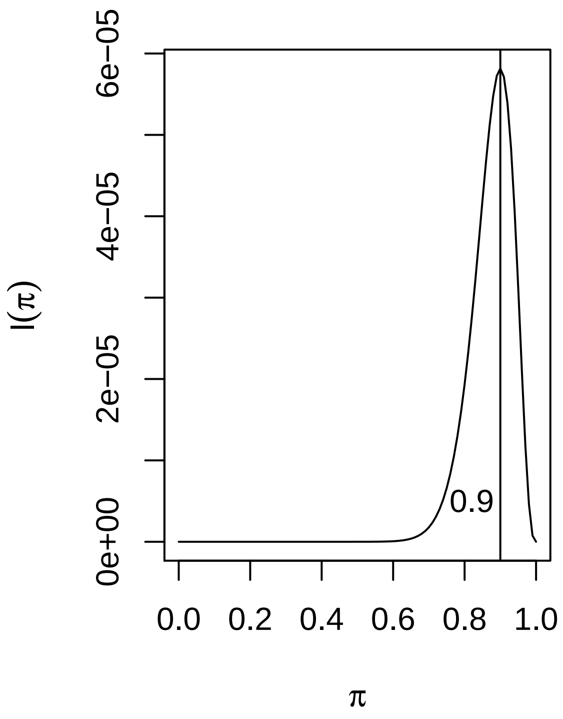
b)

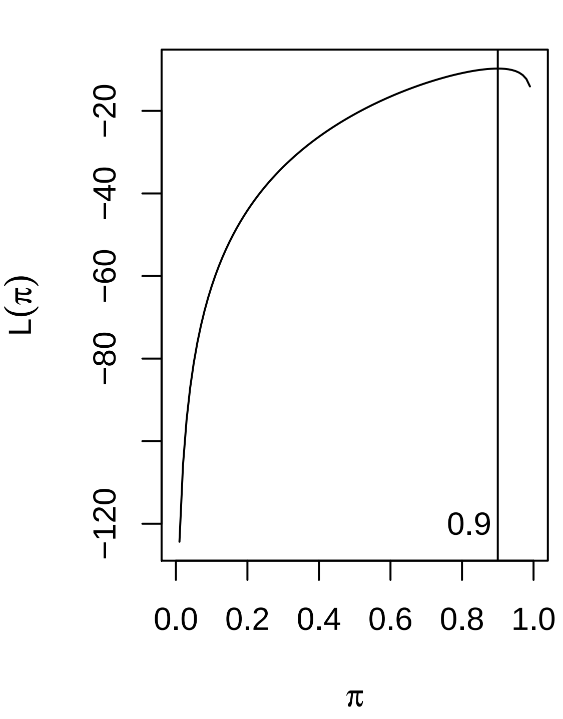
```
#R code for finding the MLE of pi where Y~Bin(30, pi) and observed y = 27

par(mfrow = c(1,2))

## Using Likelihood ##
likelihood <- function(pi) { (pi^27)*((1-pi)^3) }
opt.lik <- optimize(likelihood, interval=c(0, 1), maximum=TRUE)
curve(likelihood, from=0, to=1, xlab=expression(pi), ylab=expression(l(pi)))
abline(v = opt.lik$maximum)
text(x = opt.lik$maximum - 0.08, y = 0.000005, label = round(opt.lik$maximum, 4))

## Using log-likelihood
log_likelihood <- function(pi) { 27*log(pi) + 3*log(1-pi) }
curve(log_likelihood, from=0, to=1, xlab=expression(pi), ylab=expression(L(pi)))
opt.log_lik <- optimize(log_likelihood, interval=c(0, 1), maximum=TRUE)
abline(v = opt.log_lik$maximum)
text(x = opt.lik$maximum - 0.08, y = -120, label = round(opt.lik$maximum, 4))
```





It can be seen from the plots that the likelihood and log-likelihood functions are maximized when $\pi = 0.9$. Thus, 0.9 is the maximum likelihood estimate of π.

c)

The likelihood ratio test statistic is given by $-2\log(\Lambda) = 2\left[y\log\left(\frac{y}{n\pi_0}\right) + (n-y)\log\frac{n-y}{n-n\pi_0}\right]$.

Calculating the test statistic:

```r
n <- 30
y <- 27
phat <- y/n

f1 <- function(pi0) {2*(y*log(phat/pi0) + (n-y)*log((n-y)/(n-n*pi0)))}
f1(0.5)
```

```
## [1] 22.08385
```

```r
qchisq(0.95,df=1)
```

```
## [1] 3.841459
```

The calculated test statistic, 22.08385, is greater than the critical value at the 95% confidence level, 3.841459. Thus, we reject the null hypothesis, $H_0 : \pi_0 = 0.5$.

d)

We can find the boundaries of the 95% likelihood ratio confidence interval by solving the equation $-2\log(\Lambda) = 2\left[y\log\left(\frac{y}{n\pi_0}\right) + (n-y)\log\frac{n-y}{n-n\pi_0}\right] = \chi_1^2(\alpha)$, where $\alpha = 0.95$.

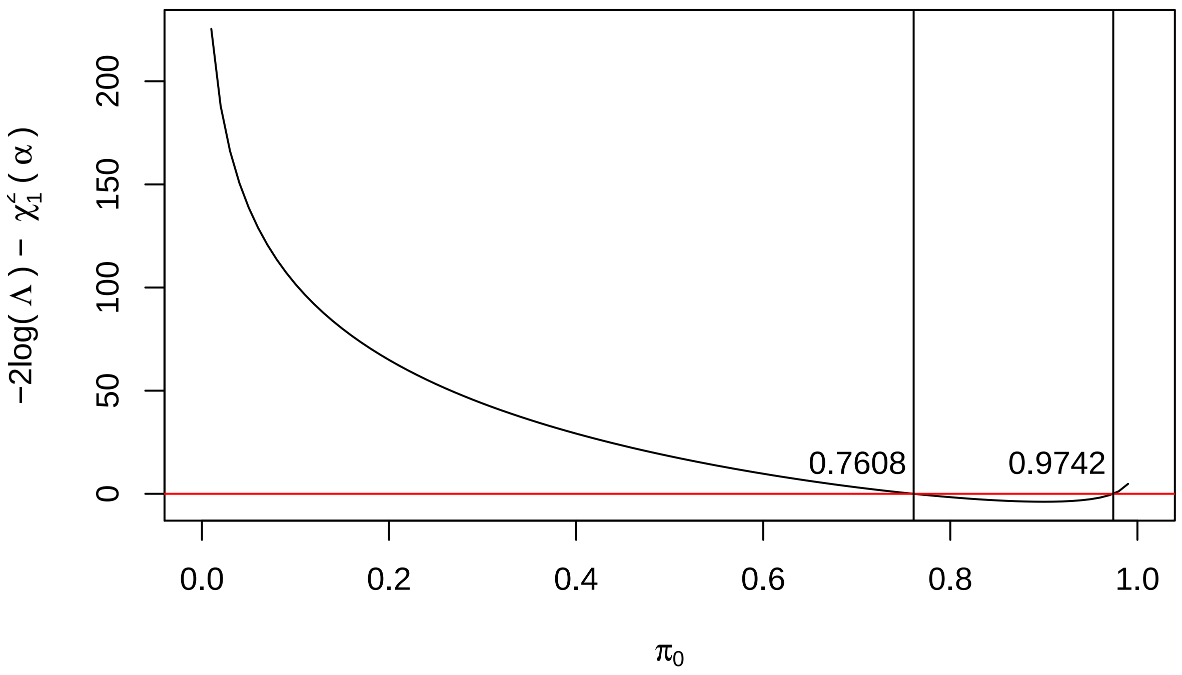
Solving:

```r
library(rootSolve)

f2 <- function(pi0) {2*(y*log(phat/pi0) + (n-y)*log((n-y)/(n-n*pi0))) - qchisq(0.95,df=1)}
bounds <- uniroot.all(f=f2, interval=c(0.000001,0.999999))
bounds
```

```
## [1] 0.7607957 0.9741571
```

```r
curve(f2, from=0, to=1,
      xlab=expression(pi[0]),
      ylab=expression(paste("-2log(" ) ~ Lambda ~ paste( ") - ") ~ chi[1]^2 ~
                           paste( "(" )   ~ alpha ~ paste( ")" ) ))
abline(h=0, col="red")
abline(v = bounds[1])
abline(v = bounds[2])
text(x = bounds[1] - 0.06, y = 15, label = round(bounds[1], 4))
text(x = bounds[2] - 0.06, y = 15, label = round(bounds[2], 4))
```



Resulting in the confidence interval [0.7607957, 0.9741571].

Question 3

a)

```r
set.seed(1006314089)

## Simulate the data
n <- 500
X1 <- runif(n,-10,10)
X2 <- rnorm(n, 0, 4)
X3 <- sample(c(0,1), n, replace = T, prob = c(0.3, 0.7))
Xmat <- cbind(rep(1,n), X1, X2, X3)
b0 <- -0.8
b1 <- 0.1
b2 <- 0.2
b3 <- 0.3
eta0 <- b0 + b1*X1 + b2*X2 + b3*X3
Y <- rpois(n, lambda = exp(eta0))
```

b)

The Iteratively Weighted Least Square (IRLS) method is used to estimate the parameters (βs) of a Generalized Linear Model (GLM). Given observations y_i and x_{ij}, it is done by the following steps:
1. Initialize $\hat{\beta}_j^{(0)} = 0$ for $j = 0, 1, ..., P$, and calculate $\eta_i^{(0)} = \sum_j x_{ij}\hat{\beta}_j^{(0)}$, as well as $\hat{\mu}_i^{(0)} = \hat{\pi}_i^{(0)}$, which is equal to $e^{\eta_i^{(0)}}$ for Poisson.
2. Set the matrix $W^{(1)}$ to be the diagonal matrix with diagonal elements $\frac{1}{v_i}$, where $v_i = (g'(\mu_i))^2 Var(Y_i) = \frac{1}{\mu_i^2}\mu_i = \frac{1}{\mu_i} = \frac{1}{e^{\eta_i^{(0)}}}$, given $g(\mu_i) = \ln \mu_i$ for Poisson. Thus, we simply have a diagonal matrix with diagonal elements $e^{\eta_i^{(0)}} = \hat{\pi}_i^{(0)}$.
3. Set $z_i^{(1)} = \hat{\eta}_i^{(0)} + (W^{(1)})^{-1}(y_i - \hat{\pi}_i^{(0)})$.
4. Estimate $\hat{\beta}^{(1)}$ using the weighted least squares formula: $\hat{\beta}^{(1)} = (X^T W^{(1)} X)^{-1} X^T W^{(1)} z^1$.
5. Set $\eta_i^{(1)} = \sum_j x_{ij}\hat{\beta}_j^{(1)}$ and $\hat{\pi}_i^{(1)} = e^{\eta_i^{(1)}}$.
6. Repeat steps 2-4 to obtain $W^{(2)}$, $z_i^{(2)}$, and $\hat{\beta}^{(2)}$.
7. Repeat steps 2-6 until either $|\hat{\eta}_i^{(t)} - \hat{\eta}_i^{(t-1)}| < \epsilon_\eta$, $|\hat{\pi}_i^{(t)} - \hat{\pi}_i^{(t-1)}| < \epsilon_\pi$, or $\frac{\partial L(\beta)}{\partial \beta}_{|\hat{\beta}} \geq \delta$.

```r
glm.IRLS <- function(Y, Xmat, tol = 1e-8){
  #Initialization
  beta <-rep(0, ncol(Xmat))
  eta <- Xmat %*% beta
  pivec <- exp(eta)
  W <- diag(as.vector(pivec))
  Z <- eta + solve(W) %*% (Y - pivec)

  eqns<-sum(t(Xmat) %*% (Y-pivec))
  istep <- 0

  # Set the threshold for the NP algorithm #
  while(eqns^2 > tol){
    #   print(beta)
    beta <- solve(t(Xmat) %*% W %*% Xmat) %*% t(Xmat) %*% W %*% Z
    eta <- Xmat %*% beta
    pivec <- exp(eta)
    W <- diag(as.vector(pivec))
```

```
      Z <- eta + solve(W) %*% (Y - pivec)
      eqns <- sum(t(Xmat) %*% (Y - pivec))
      istep <- istep +1
   }
   #print(beta)

   SE <- sqrt(diag(solve(t(Xmat) %*% W %*% Xmat)))
   z  = beta/SE

   res.mat <- data.frame(Estimates = beta,
                         SE = SE,
                         z  = z,
                         p_value = ifelse(z < 0, 2 * pnorm(z), 2 * (1 - pnorm(z)))
   )
   rownames(res.mat)[1] <- "Intercept"
   results <- list(Table = res.mat, Iteration = istep)

   return(results)

}

glm.IRLS(Y = Y, Xmat = Xmat) # Default tolerance is 1e-8
```

```
## $Table
##            Estimates          SE         z      p_value
## Intercept -0.6927651 0.107987659 -6.415225 1.406150e-10
## X1         0.1025799 0.008882594 11.548415 0.000000e+00
## X2         0.1997573 0.012298117 16.242921 0.000000e+00
## X3         0.1423773 0.112103440  1.270053 2.040658e-01
##
## $Iteration
## [1] 5
```

```
glm.p<-glm(Y ~ X1+X2+X3,family=poisson(link = log))
summary(glm.p)
```

```
##
## Call:
## glm(formula = Y ~ X1 + X2 + X3, family = poisson(link = log))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1268  -0.8320  -0.5164   0.4051   2.9001
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.692765   0.107984  -6.415  1.4e-10 ***
## X1           0.102580   0.008882  11.549  < 2e-16 ***
## X2           0.199757   0.012298  16.243  < 2e-16 ***
## X3           0.142377   0.112101   1.270    0.204
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
```

```
##
##     Null deviance: 857.90  on 499  degrees of freedom
## Residual deviance: 454.85  on 496  degrees of freedom
## AIC: 1028.6
##
## Number of Fisher Scoring iterations: 5
```

The results from the IRLS method are very close to the results from the glm function in R. The estimates, standard errors, z-scores, and p-values are all consistent, and the number of iterations is 5 in both.