Class 4 (Solution)

Yubo Rasmussen

## Introduction

This notebook covers the Binomial Model with the Uniform Distribution of Deaths (UDD) and Constant Force of Mortality (CFM) assumptions. Additionally, it implements the Newton-Raphson method for numerical optimization.

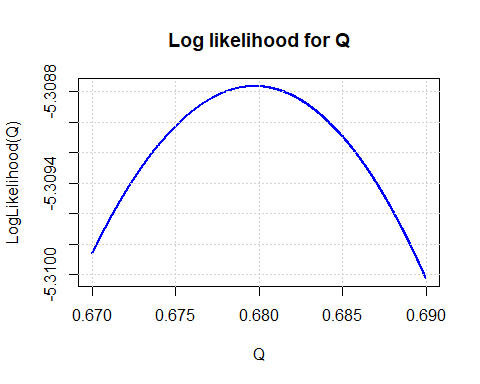
## Binomial Model: Uniform Distribution of Deaths (UDD)

Write down the loglikelihood function from Example 4.7 in terms of q and assign it as a named variable under the UDD assumption.

# Log likelihood function in terms of q  
LogQ <- function(Q) 3\*log(Q) +2\*log(1-Q) - log(4-Q) - log(4-3\*Q)

Write a function to plot the log-likelihood function above and call it.

# Plot log likelihood  
PlotLogQ <- function(Left, Right){  
 Q <- seq(Left, Right, length = 1000)  
 plot(Q, LogQ(Q), type = "l", col = "blue", lwd = 2,  
 main = "Log likelihood for Q", ylab = "LogLikelihood(Q)")  
 grid()  
}  
  
# Plotting  
PlotLogQ(0.67, 0.69)



What are we looking for in this plot? - The maxima of this curve What do you think is a reasonable assumption for the value of Qhat? Assign and store it.

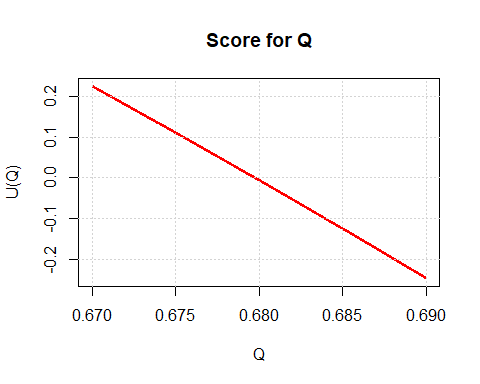
QHat <- 0.68

Write down the score function of Q. Can you explain how the score function is derived?

# Score for Q  
ScoreQ <- function(Q) 3/Q - 2/(1-Q) + 1/(4-Q) + 3/(4-3\*Q)

Now write a function to plot the Score function and call it.

# Plot of Score function  
PlotScoreQ <- function(Left, Right){  
 Q <- seq(Left, Right, length = 1000)  
 plot(Q, ScoreQ(Q), type = "l", col = "red", lwd = 2,  
 main = "Score for Q", ylab = "U(Q)")  
 grid( )  
}  
PlotScoreQ(0.67, 0.69)



What are we looking for in this plot? - The value of Q when U(Q) is at 0 What do you think is a reasonable assumption for the value of QHat? - 0.68

# And QHat = 0.68 again  
QHat = 0.68

Now, it is time to compute the standard error of the estimated parameter QHat.

Use a numerical approximation by computing the difference quotient.

# SE numerical  
Slope <- (ScoreQ(0.68+0.001) - ScoreQ(0.68-0.001))/0.002  
SE <- sqrt(-1/Slope); SE

[1] 0.2059085

Use an analytical second derivative approach.

# And with second derivative  
DUQ <- function(Q) -3/Q^2 - 2/(1-Q)^2 + 1/(4-Q)^2 + 9/(4-3\*Q)^2   
Slope2 <- DUQ(0.68)  
SE2 <- sqrt(-1/Slope2); SE2

[1] 0.2059094

Do the two estimates of SE aggree? - Yes

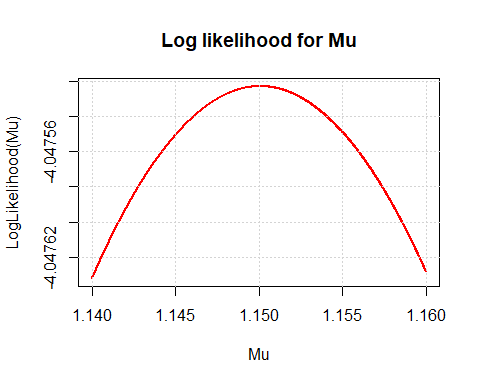
## Binomial Model: Constant Force of Mortality (CFM)

Write down the loglikelihood function from Example 4.7 in terms of and assign it as a named variable under the CFM assumption.

# Log likelihood function in terms of mu  
LogMu <- function(Mu) - 1.75\*Mu + 2\*log(1-exp(-0.5\*Mu)) + log(1-exp(-Mu))

Write a function to plot the log-likelihood function above and call it.

# Plot log likelihood  
PlotLogMu <- function(Left, Right){  
 Mu <- seq(Left, Right, length = 1000)  
 plot(Mu, LogMu(Mu), type = "l", col = "red", lwd = 2,  
 main = "Log likelihood for Mu", ylab = "LogLikelihood(Mu)")  
 grid()  
}  
  
# Plotting  
PlotLogMu(1.14, 1.16)



What are we looking for in this plot? - The maxima of this curve What do you think is a reasonable assumption for the value of MuHat? Assign and store it.

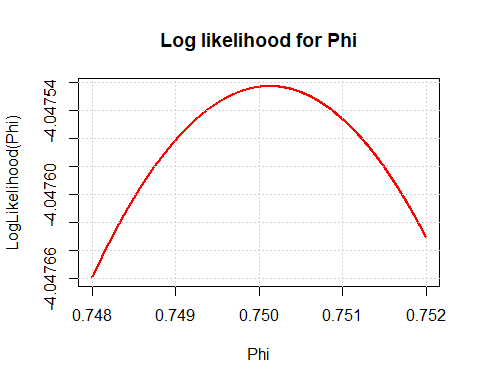
MuHat <- 1.15

Write down the loglikelihood function from Example 4.7 in terms of and assign it as a named variable under the CFM assumption.

# Log likelihood function in terms of phi  
LogPhi <- function(Phi) 7\*log(Phi) + 2\*log(1-Phi^2) + log(1-Phi^4)

Write a function to plot the log-likelihood function above and call it.

# Plot log likelihood for Phi  
PlotLogPhi <- function(Left, Right){  
 Phi <- seq(Left, Right, length = 1000)  
 plot(Phi, LogPhi(Phi), type = "l", col = "red", lwd = 2,  
 main = "Log likelihood for Phi", ylab = "LogLikelihood(Phi)")  
 grid()  
}  
PlotLogPhi(0.748, 0.752)



What are we looking for in this plot? - The maxima of this curve What do you think is a reasonable assumption for the value of PhiHat? Assign and store it.

PhiHat <- 0.750

Now check the consistency, i.e. is close to

exp(-0.25\*MuHat)

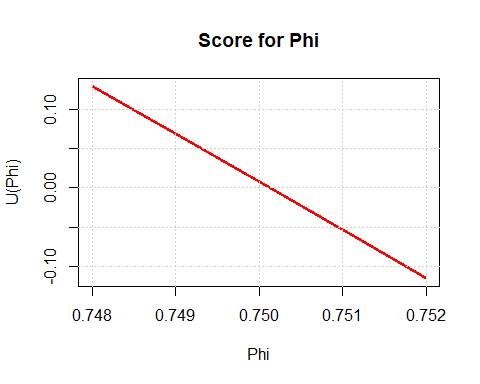
[1] 0.7501366

Write down the score function for Phi. Can you explain how the score function is derived for Phi?

# Score for Phi  
ScorePhi <- function(Phi) 7/Phi - 4\*Phi/(1-Phi^2) - 4\*Phi^3/(1-Phi^4)

Now write a function to plot the Score function and call it.

# Plot of Score function  
PlotScorePhi <- function(Left, Right){  
 Phi <- seq(Left, Right, length = 1000)  
 plot(Phi, ScorePhi(Phi), type = "l", col = "red", lwd = 2,  
 main = "Score for Phi", ylab = "U(Phi)")  
 grid( )  
}  
PlotScorePhi(0.748, 0.752)



What are we looking for in this plot? - The value of Phi when U(Phi) is at 0 What do you think is a reasonable assumption for the value of PhiHat? - 0.75

# And PhiHat = 0.75 again  
PhiHat = 0.75

Now, it is time to compute the standard error of the estimated parameter PhiHat.

Use a numerical approximation by computing the difference quotient.

# SE numerical  
Slope <- (ScorePhi(0.75+0.001) - ScorePhi(0.75-0.001))/0.002  
Slope

[1] -61.06643

SE <- sqrt(-1/Slope); SE

[1] 0.1279672

Use an analytical second derivative approach.

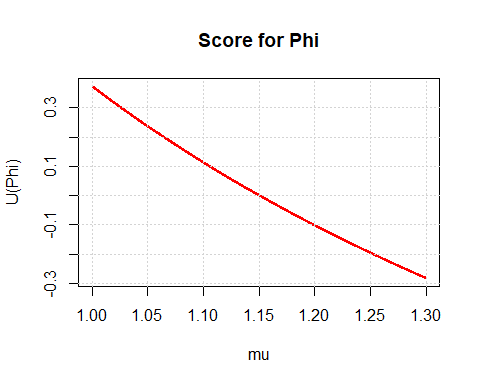
# And with second derivative  
DUPhi <- function(Phi) -7/Phi^2 - 4/(1-Phi^2) - 8\*Phi^2/(1-Phi^2)^2 -  
 12\*Phi^2/(1-Phi^4) - 16\*Phi^6/(1-Phi^4)^2  
Slope2 <- DUPhi(0.75)  
SE2 <- sqrt(-1/Slope2); SE2

[1] 0.1279681

Do the two estimates of SE aggree? - Yes

Repeat the above and write down the Score function and SE for mu, including a plot of the Score function.

# Score function and SE for mu  
ScoreMu <- function(mu) {  
 phi = exp(-0.25\*mu)  
 ScorePhi(phi) \* phi \*(-.25)  
}  
PlotScoreMu <- function(Left, Right){  
 mu <- seq(Left, Right, length = 1000)  
 plot(mu, ScoreMu(mu), type = "l", col = "red", lwd = 2,  
 main = "Score for Phi", ylab = "U(Phi)")  
 grid( )  
}  
PlotScoreMu(1, 1.3)



SlopeMu <- (ScoreMu(MuHat+0.001) - ScoreMu(MuHat-0.001))/0.002  
SlopeMu

[1] -2.149347

SE.Mu <- sqrt(-1/SlopeMu); SE.Mu

[1] 0.682098

What can you say about the relationship between mu and phi? Why do we compute both of them? Which score function do you think is easier to derive analytically?

## Newton-Raphson Method

Newton-Raphson is an iterative method used to solve equations of the form . The update rule for finding a better estimate of the root is given by:

Since computing the analytical derivative is not always convenient, we use a numerical approximation.

### Implementing the Newton-Raphson Method

# Newton-Raphson iteration using numerical differentiation  
Newton <- function(f, x0, Delta){  
 Derivative <- (f(x0+Delta/2) - f(x0-Delta/2))/Delta  
 x0 - f(x0)/Derivative  
}

### Example: Finding the Square Root of 2

Square <- function(x) x^2 - 2  
x1 <- Newton(Square, 1.5, 0.0001)  
x2 <- Newton(Square, x1, 0.0001)  
c(x1, x2, x2^2)

[1] 1.416667 1.414216 2.000006

### Generalizing Newton-Raphson with a Loop

We can now define a function that iteratively applies the Newton-Raphson method until convergence:

Solve <- function(f, x0, Delta, Tol, Max.Iter){  
 TOL <- abs(f(x0))  
 Iter <- 0  
 while((TOL > Tol) && (Iter < Max.Iter)) {  
 Iter <- Iter + 1  
 x1 <- Newton(f, x0, Delta)  
 TOL <- abs(f(x1))  
 x0 <- x1  
 }  
 x0  
}

### Finding Roots of a Function

Root <- Solve(Square, 1.5, 0.0001, 10^(-6), 10)  
c(Root, Root^2)

[1] 1.414214 2.000000

### Applying Newton-Raphson to the Score Function of Q

Score <- function(Q) 3/Q - 2/(1 - Q) + 1/(4 - Q) + 3/(4 - 3\*Q)  
q.hat <- Solve(Score, 0.5, 0.0001, 10^(-6), 10)  
q.hat

[1] 0.6797277

### Alternative Root Finding Using uniroot()

The uniroot() function in R can also find the root of a function over a specified interval:

x.lower <- 0.001; x.upper <- 0.999  
uniroot(Score, c(x.lower, x.upper))

$root  
[1] 0.6797251  
  
$f.root  
[1] 6.191831e-05  
  
$iter  
[1] 6  
  
$init.it  
[1] NA  
  
$estim.prec  
[1] 6.576709e-05

### Numerical Derivative of the Log-Likelihood Function

Log.L <- function(Q) 3\*log(Q) + 2\*log(1-Q) - log(4-Q) - log(4-3\*Q)  
Score.Num <- function(x) (Log.L(x + Delta/2) - Log.L(x - Delta/2))/Delta  
Delta <- 0.0001  
uniroot(Score.Num, c(x.lower, x.upper))

$root  
[1] 0.6797251  
  
$f.root  
[1] 6.191399e-05  
  
$iter  
[1] 6  
  
$init.it  
[1] NA  
  
$estim.prec  
[1] 6.576218e-05

### Checking the Score Function at the Estimated Root

Score(q.hat)

[1] -1.79412e-13

Score(uniroot(Score.Num, c(x.lower, x.upper))$root)

[1] 6.195374e-05

* How does the Newton-Raphson method compare to uniroot()?
* How does step size ( ) affect convergence?
* What are the advantages of using Newton-Raphson in maximum likelihood estimation?