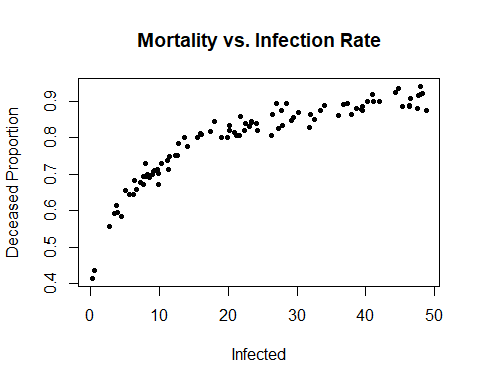
Assignment3

Fenglin Chen

14/11/2021

### Question 1 a)

disease <- read.csv("Infectious.csv", header=TRUE)  
plot(Deceased.Prop ~ Infected, data=disease, pch=19, cex=0.7,   
 xlab="Infected", ylab="Deceased Proportion",   
 main="Mortality vs. Infection Rate")

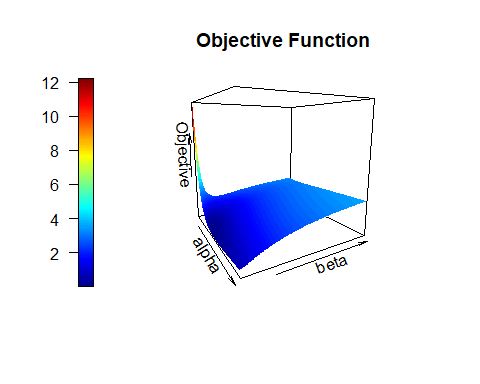


The deceased proportion (mortality rate) appears to increase monotonically as the number of infections increase. This indicates that the deadliness of the disease is related to its ability to spread. The public might see that sicker patients are more infectious.

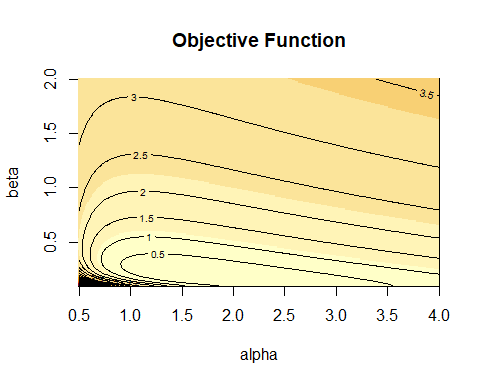
The points are relatively compact along the line of best fit, which means the behaviour of the disease is consistent. In addition, the rate increases slower as mortality nears 100%, as it can never surpass 1 no matter how many people are infected.

### Question 1 b)

CreateObjectiveFunction <- function(X, Y) {  
 function(alpha, beta){  
 sum((Y - 1 + 1/(alpha + beta\*X))^2)  
 }  
}  
rho <- CreateObjectiveFunction(disease$Infected, disease$Deceased.Prop)  
  
alpha <- seq(0.5, 4, 0.01)  
beta <- seq(0.1, 2, 0.01)  
mat <- outer(alpha, beta, FUN=Vectorize(rho))  
  
# 3D plot of alpha vs. beta  
library(plot3D)  
colours <- colorRampPalette(c("blue", "red"))(100)  
persp3D(x=alpha, y=beta, z=mat, xlab="alpha", ylab="beta", zlab="Objective",  
 main="Objective Function", theta=60, phi=10, ticktype="simple",   
 col.palette=heat.colors, colkey=list(side=2))



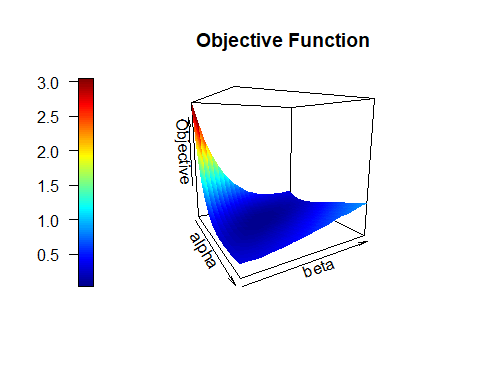
# Heat map with superimposed contour plot  
image(x=alpha, y=beta, z=mat, xlab="alpha", ylab="beta",   
 main="Objective Function")  
contour(x=alpha, y=beta, z=mat, nlevels=20, add=TRUE)



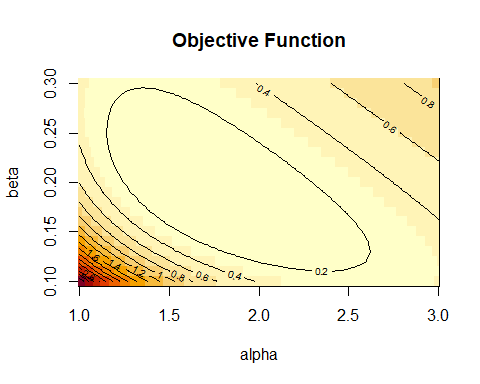
### Question 1 c)

and makes the neighbourbood of the minima more visible.

# Narrow the ranges of alpha and beta  
alpha <- seq(1, 3, 0.01)  
beta <- seq(0.1, 0.3, 0.01)  
mat <- outer(alpha, beta, FUN=Vectorize(rho))  
  
# 3D plot of alpha vs. beta  
library(plot3D)  
colours <- colorRampPalette(c("blue", "red"))(100)  
persp3D(x=alpha, y=beta, z=mat, xlab="alpha", ylab="beta", zlab="Objective",  
 main="Objective Function", theta=60, phi=10, ticktype="simple",   
 col.palette=heat.colors, colkey=list(side=2))



image(x=alpha, y=beta, z=mat, xlab="alpha", ylab="beta",   
 main="Objective Function")  
contour(x=alpha, y=beta, z=mat, nlevels=20, add=TRUE)



### Question 1 d)

The partial derivatives of the objective function are:

The second partial derivatives of the objective function are:

CreatePsiFn <- function(X, Y){  
 psiFn <- function(theta) {  
 bracket = (theta[1]+theta[2]\*X)  
 c(sum( -2\*(Y-1 + 1/bracket)\* bracket^-2 ),  
 sum( -2\*(Y-1 + 1/bracket)\*X\*bracket^-2 ))  
 }  
}  
  
CreatePsiPrimeFn <- function(X, Y){  
 function(theta) {  
 val = matrix(0, nrow=length(theta), ncol=length(theta))  
 bracket = (theta[1]+theta[2]\*X)  
 val[1, 1] = sum( 2\* bracket^-4 + 4\* (Y-1 + 1/bracket)\*bracket^-3 )  
 val[1, 2] = sum( 2\*X\* bracket^-4 + 4\*X\* (Y-1 + 1/bracket)\*bracket^-3 )  
 val[2, 1] = sum( 2\*X\* bracket^-4 + 4\*X\* (Y-1 + 1/bracket)\*bracket^-3 )  
 val[2, 2] = sum( 2\*X^2\*bracket^-4 + 4\*X^2\*(Y-1 + 1/bracket)\*bracket^-3 )  
 return(val)  
 }  
}  
  
psiFn = CreatePsiFn(disease$Infected, disease$Deceased.Prop)  
psiPrimeFn = CreatePsiPrimeFn(disease$Infected, disease$Deceased.Prop)

Using the definition of the Newton-Raphson algorithm from class:

testConvergence <- function(thetaNew, thetaOld, tolerance = 1e-10, relative = FALSE) {  
 sum(abs(thetaNew - thetaOld)) < if (relative)   
 tolerance \* sum(abs(thetaOld)) else tolerance  
}  
  
NewtonRaphson <- function(theta, psiFn, psiPrimeFn, dim, testConvergenceFn = testConvergence, maxIterations = 200, tolerance = 1e-06, relative = FALSE) {  
 if (missing(theta)) {  
 if (missing(dim)) {  
 dim <- length(psiFn())  
 }  
 theta <- rep(0, dim)  
 }  
 converged <- FALSE  
 i <- 0  
 while (!converged & i <= maxIterations) {  
 thetaNew <- theta - solve(psiPrimeFn(theta), psiFn(theta))  
 converged <- testConvergenceFn(thetaNew, theta, tolerance = tolerance,   
 relative = relative)  
 theta <- thetaNew  
 i <- i + 1  
 }  
 list(theta = theta, converged = converged, iteration = i, fnValue = psiFn(theta))  
}

#### i) Initial values :

NRresult1 <- NewtonRaphson(theta=c(2, 3), psiFn=psiFn, psiPrimeFn=psiPrimeFn, dim=2)  
print(NRresult1)

## $theta  
## [1] 1.549544e+36 6.440326e+35  
##   
## $converged  
## [1] FALSE  
##   
## $iteration  
## [1] 201  
##   
## $fnValue  
## [1] 1.399999e-72 5.450558e-72

#### ii) Initial values :

NRresult2 <- NewtonRaphson(theta=c(3, 0.2), psiFn=psiFn, psiPrimeFn=psiPrimeFn, dim=2)  
print(NRresult2)

## $theta  
## [1] -9.190532e+35 4.111109e+34  
##   
## $converged  
## [1] FALSE  
##   
## $iteration  
## [1] 201  
##   
## $fnValue  
## [1] 1.88750e-66 4.21769e-65

#### iii) Estimate of the minima from part c) with :

NRresult3 <- NewtonRaphson(theta=c(1.75, 0.2), psiFn=psiFn, psiPrimeFn=psiPrimeFn, dim=2)  
print(NRresult3)

## $theta  
## [1] 1.705989 0.190827  
##   
## $converged  
## [1] TRUE  
##   
## $iteration  
## [1] 4  
##   
## $fnValue  
## [1] -1.544056e-15 -2.522949e-14

rho(NRresult1$theta[1], NRresult1$theta[2])

## [1] 5.331619

rho(NRresult2$theta[1], NRresult2$theta[2])

## [1] 5.331619

rho(NRresult3$theta[1], NRresult3$theta[2])

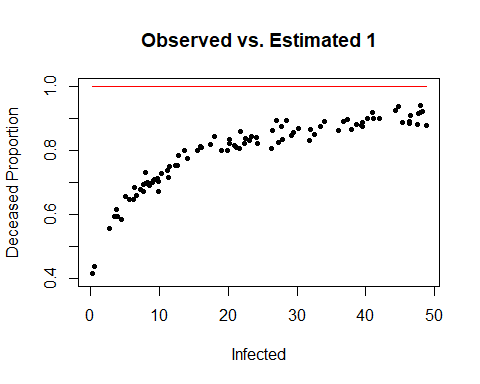
## [1] 0.03656048

Since the objective function is lowest for the third pair of parameters, the initial value of is the most appropriate.

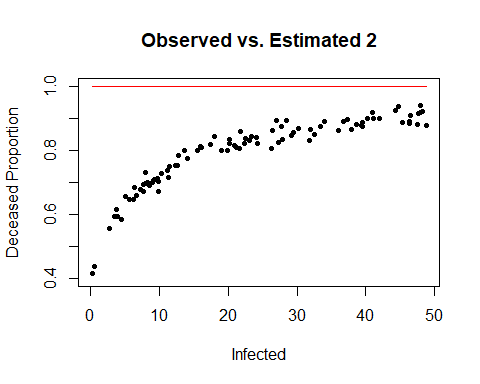
### Question 1 e)

regression <- function(alpha, beta){  
 function(x){  
 1 - 1/(alpha + beta\*x)  
 }  
}  
  
infectedRange = seq(min(disease$Infected), max(disease$Infected), 0.01)  
reg1 = regression(NRresult1$theta[1], NRresult1$theta[2])  
reg2 = regression(NRresult2$theta[1], NRresult2$theta[2])  
reg3 = regression(NRresult3$theta[1], NRresult3$theta[2])

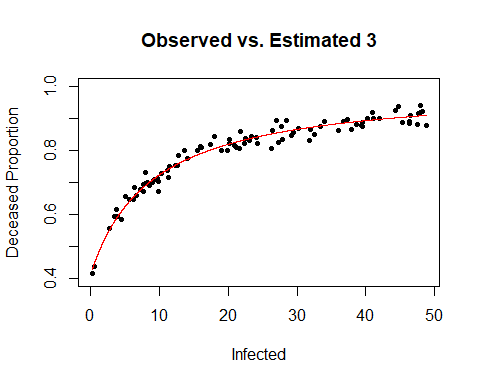
plot(Deceased.Prop ~ Infected, data=disease, pch=19, cex=0.7,   
 xlab="Infected", ylab="Deceased Proportion",   
 main="Observed vs. Estimated 1", ylim=c(0.4, 1))  
lines(infectedRange, reg1(infectedRange), col="red")



plot(Deceased.Prop ~ Infected, data=disease, pch=19, cex=0.7,   
 xlab="Infected", ylab="Deceased Proportion",   
 main="Observed vs. Estimated 2", ylim=c(0.4, 1))  
lines(infectedRange, reg2(infectedRange), col="red")



plot(Deceased.Prop ~ Infected, data=disease, pch=19, cex=0.7,   
 xlab="Infected", ylab="Deceased Proportion",   
 main="Observed vs. Estimated 3", ylim=c(0.4, 1))  
lines(infectedRange, reg3(infectedRange), col="red")



The plots show that the first two parameter estimates weren’t able to converge. Since the Newton-Raphson method only finds the roots of the function, it is possible that they approached the maximum instead of the minimum.

On the other hand, the hand-picked parameters (set 3) were able to converge to the minimum and demonstrate a close fit of the observed values.

### Question 1 f)

Transform the model so and can be fitted linearly:

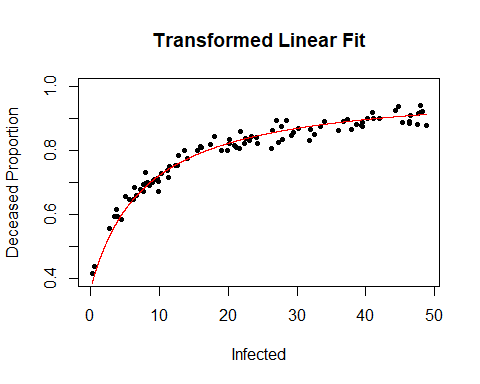
#### i) Summary of the fit results

disease$transformed <- 1 / (1-disease$Deceased.Prop)  
coefs <- lm(transformed ~ Infected, data=disease)  
summary(coefs)

##   
## Call:  
## lm(formula = transformed ~ Infected, data = disease)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.3280 -0.5327 0.0053 0.3270 5.7475   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.583036 0.250936 6.309 8.17e-09 \*\*\*  
## Infected 0.201425 0.009125 22.074 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.317 on 98 degrees of freedom  
## Multiple R-squared: 0.8326, Adjusted R-squared: 0.8308   
## F-statistic: 487.3 on 1 and 98 DF, p-value: < 2.2e-16

#### ii) Plot fitted results

new.reg = regression(coefs$coefficients[1], coefs$coefficients[2])  
plot(Deceased.Prop ~ Infected, data=disease, pch=19, cex=0.7,   
 xlab="Infected", ylab="Deceased Proportion",   
 main="Transformed Linear Fit", ylim=c(0.4, 1))  
lines(infectedRange, new.reg(infectedRange), col="red")



The plot shows that the model fits very well to the data. When the observations were transformed using the proposed model, the coefficients could be fitted linearly, and the resultant re-transformation provided a good estimate.

This demonstrates that assumptions made in the model were correct.

### Question 2 a)

The HT estimate of the total death in all 486 communities/towns/cities is the number of deaths in each sample divided by the chance of selecting that sample.

In SRSWR, the inclusion probability of one sample is , and the includion probability of any two samples is .

# Inclusion probability for any one sample  
srswr.incl <- function(u) {  
 N <- 486  
 n <- 100  
 1 - ((N-1)/N)^n  
}  
  
# Joint inclusion probability for any two samples  
srswr.joint <- function(u, v) {  
 N <- 486  
 n <- 100  
 if (u == v) {1 - ((N-1)/N)^n}   
 else {1 - 2\*((N-1)/N)^n + ((N-2)/N)^n}  
}  
  
# HT estimator from class  
createHTestimator <- function(pi\_u\_fn) {  
 function(samp, variateFn) {  
 Reduce(`+`,   
 Map(function(u) {variateFn(u)/ pi\_u\_fn(u)}, samp),  
 init = 0  
 )  
 }  
}  
  
# HT variance estimator from class  
createHTVarianceEstimator <- function(pop, pi\_u\_fn, pi\_uv\_fn) {  
 function(samp, variateFn) {  
 Reduce(`+`,  
 Map(function(u) {  
 pi\_u <- pi\_u\_fn(u)  
 y\_u <- variateFn(u)  
 Reduce(`+`,   
 Map(function(v) {  
 pi\_v <- pi\_u\_fn(v)  
 pi\_uv <- pi\_uv\_fn(u, v)  
 y\_v <- variateFn(v)  
 Delta\_uv <- pi\_uv - pi\_u \* pi\_v  
 result <- (Delta\_uv \* y\_u \* y\_v)   
 result <- result/(pi\_uv \* pi\_u \* pi\_v)  
 result  
 },   
 samp),  
 init = 0)   
 },  
 samp  
 ),  
 init = 0)  
 }  
}  
  
HT.est <- createHTestimator(srswr.incl)  
HT.var <- createHTVarianceEstimator(1:486,  
 pi\_u\_fn=srswr.incl,  
 pi\_uv\_fn=srswr.joint)  
  
createVariateFn <- function(pop, variate1, variate2){  
 function(u) {pop[u, variate1] \* pop[u, variate2]}  
}  
deaths <- createVariateFn(disease, "Infected", "Deceased.Prop")  
  
tot.mean <- HT.est(1:100, deaths) \* 12.5  
tot.mean

## [1] 134137.6

tot.sdn <- sqrt(HT.var(1:100, deaths)) \* 12.5  
tot.sdn

## [1] 8304.263

The number of deaths in country A due to the disease is around per , so the total number is (or if rounded to the nearest person).

The standard error of the estimate is per , or in total.

# Assume normal distribution  
confidence <- 0.95  
c.val <- qnorm((confidence+1)/2)  
conf.interval <- c(tot.mean + c.val \* c(-tot.sdn, tot.sdn))  
conf.interval

## [1] 117861.5 150413.7

The 95% confidence interval for the total number of deaths is [117861.5, 150413.7].

### Question 2 b)

The unweighted inclusion probabilities can be rewritten as:

and

Where is the weight for each unit. Therefore, the inclusion and joint inclusion probabilities of weighted simple random sampling with replacement (WSRSWR) can be derived by replacing the weight variable:

### Question 2 c)

# Inclusion probability for any one sample  
wsrswr.incl <- function(u) {  
 N <- 486  
 n <- 100  
 1 - (1-w[u])^n  
}  
  
# Joint inclusion probability for any two samples  
wsrswr.joint <- function(u, v) {  
 N <- 486  
 n <- 100  
 if (u == v) {1 - ((N-1)/N)^n}   
 else {1 - (1-w[u])^n - (1-w[v])^n + (1-w[u]-w[v])^n}  
}  
  
HT.est.weighted <- createHTestimator(wsrswr.incl)  
HT.var.weighted <- createHTVarianceEstimator(1:486,  
 pi\_u\_fn=wsrswr.incl,  
 pi\_uv\_fn=wsrswr.joint)  
  
tot.mean.weighted <- HT.est.weighted(1:100, deaths) \* 12.5  
tot.mean.weighted

## [1] 92713.64

tot.sdn.weighted <- sqrt(HT.var.weighted(1:100, deaths)) \* 12.5  
tot.sdn.weighted

## [1] 12444.1

The total number of deaths in country A estimated using WSRSWR is 92,713.64 (or 92714 when rounded), and the standard error of the estimate is 12,444.1.

# Assume normal distribution  
confidence <- 0.95  
c.val <- qnorm((confidence+1)/2)  
conf.interval <- c(tot.mean.weighted + c.val \* c(-tot.sdn.weighted, tot.sdn.weighted))  
conf.interval

## [1] 68323.65 117103.62

The confidence interval for WRSRWR is [68,323.65, 117,103.62].

### Question 2 d)

The sampling protocol in WSRSWR makes more sense than SRSWR. In SRSWR, the weight of each unit is , which assumes that small towns have the same probability of being sampled than big cities.

However, larger locations have more residents. Assuming that each person is sampled at random, the cumulative probability of sampling people from larger cities is bigger than sampling people small towns. Since WSRSWR reflects the reality of population size differences, it is the better sampling protocol.