Lec 4: Maximum Likelihood Estimation with examples

Toshihide Kitakado

May 29, 2019

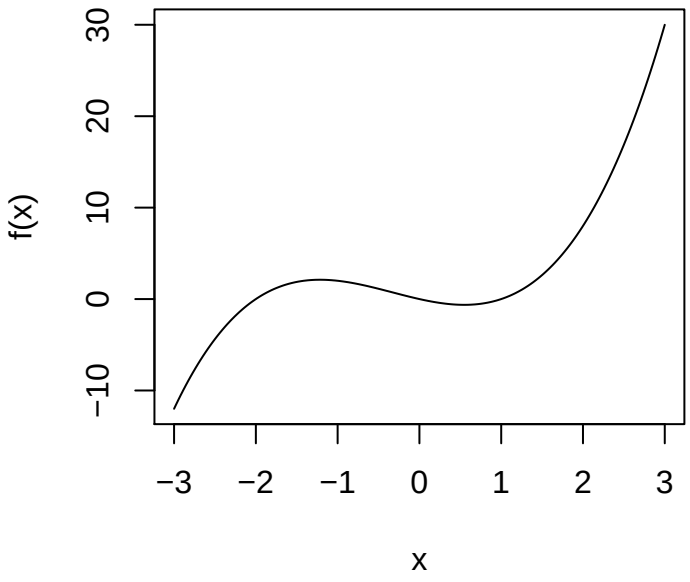
Optimization

Definition of convenient function

```
logit <- function(p) log(p/(1-p))
logitinv <- function(x) 1/(1+exp(-x))
```

Optimization of a simple function

```
f <- function(x){x^3 + x^2 - 2*x }
x <- seq(-3,3,0.01)
plot(x, f(x), type="l")
```




```
#Minimization
#Syntax: optim(Initial value, function name, optimization method)
optim(-2, f, method="BFGS")
```

```
## $par
## [1] -2.612088e+21
##
## $value
## [1] -1.782228e+64
```

```
##
## $counts
## function gradient
##        6        6
##
## $convergence
## [1] 0
##
## $message
## NULL
```

```
optim(0, f, method="BFGS")
```

```
## $par
## [1] 0.5485839
##
## $value
## [1] -0.6311303
##
## $counts
## function gradient
##       12        6
##
## $convergence
## [1] 0
##
## $message
## NULL
```

```
#optim(1, f, method="BFGS")
```

Optimization of a simple function in a limited domain

```
#Optimization in a domain (0,1)
optim(-2, f, method="L-BFGS-B", lower=0, upper=1)
```

```
## $par
## [1] 0.5485836
##
## $value
## [1] -0.6311303
##
## $counts
## function gradient
##        7        7
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

```
#Optimization in a domain (0,1)
f <- function(logitx){
x <- 1/(1+exp(-logitx))
```

```
x^3 + x^2 - 2*x
}
res <- optim(-5, f, method="BFGS")
res$par
```

```
## [1] 0.1949502
```

```
exp(res$par) /(1+exp(res$par))
```

```
## [1] 0.5485838
```

Occupancy model (1) -simple poisson model-

Description

- Case 1: Suppose that you are studying occupation of a species in a study area. At first, let us assume a quite ideal situation that you can observe counts of animals without missing.
- Notation: $Y_1, Y_2, \ldots, Y_k \sim (iid)Pois(\lambda)$
- Task: to estimate λ
- simulation data

```
Ns <- 20
lambda <- 3
yy <- rpois(Ns, lambda)
yy
```

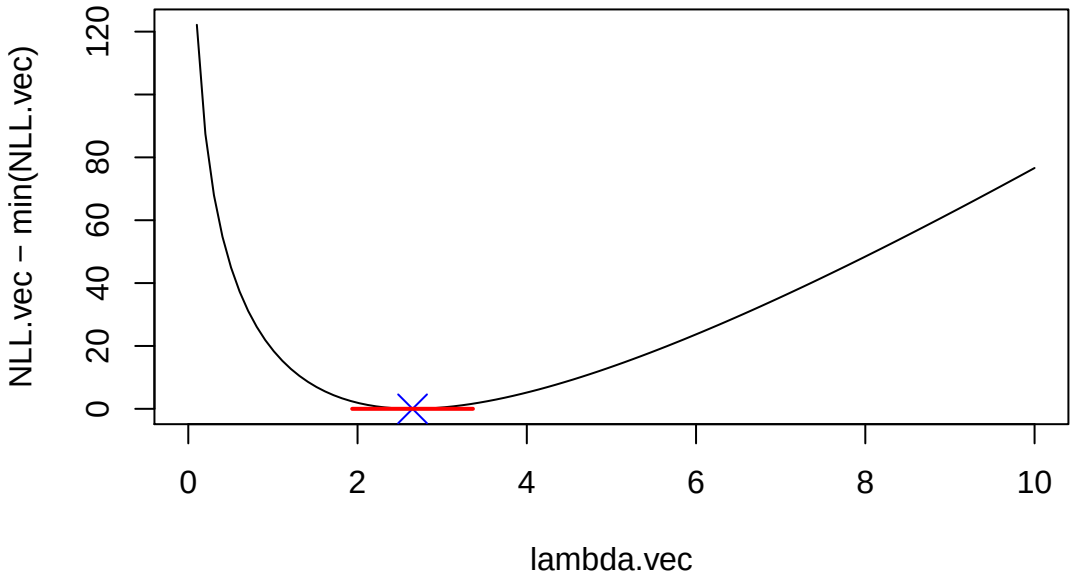
```
##  [1] 4 3 2 3 2 1 2 3 2 4 3 5 0 1 3 3 3 2 4 3
```

Definition of likelihood function and implementation of optimization-1

```
#Def of negative loglikelihood
NLL.pois <- function(lambda){
 logPF <- dpois(yy, lambda, log=T)
 NLL <- (-1.0)*sum(logPF)
 return(NLL)
}

lambda.vec <- seq(0, 10, length.out=100)
NLL.vec <- sapply(lambda.vec, NLL.pois)
plot(lambda.vec, NLL.vec-min(NLL.vec), type="l")

Res.pois <- optim(5, NLL.pois, method="BFGS", hessian=T)
lambda.est <- Res.pois$par
lambda.se <- sqrt(1/Res.pois$hessian)
data.frame(lambda.est,lambda.se)
```

```
##   lambda.est lambda.se
## 1   2.649994 0.3640046
```

```
points(Res.pois$par, 0, pch=4, col="blue", cex=2)
lines(lambda.est+c(-1.96,1.96)*lambda.se, c(0,0), col="red", lwd=2)
```

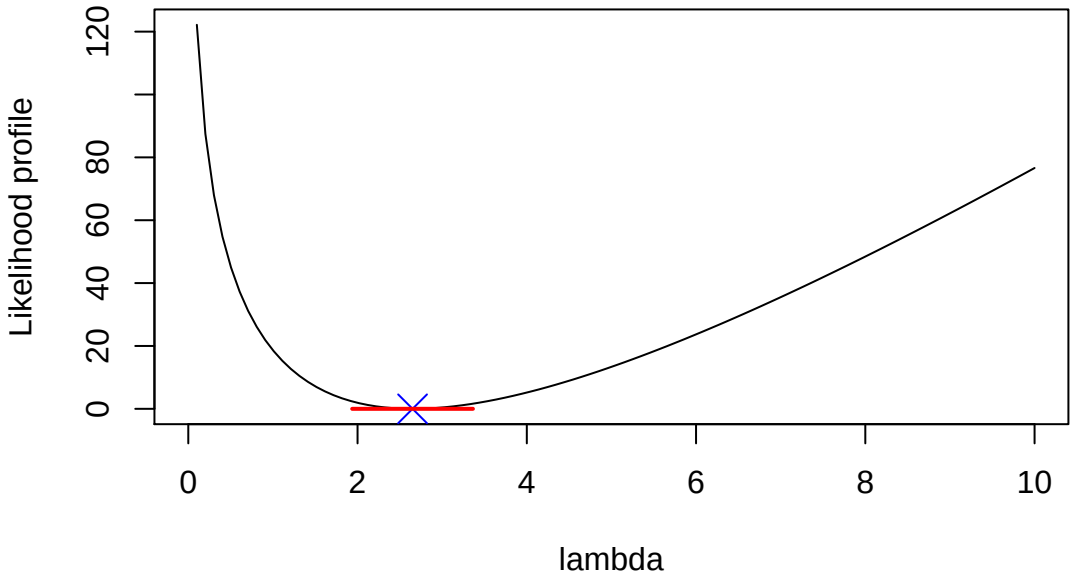


Definition of likelihood function and implementation of optimization-2

```
#Def of negative loglikelihood-2
NLL.pois <- function(loglambda){
 lambda <- exp(loglambda)
 logPF <- dpois(yy, lambda, log=T)
 NLL <- (-1.0)*sum(logPF)
 return(NLL)
}

lambda.vec <- seq(0, 10, length.out=100)
NLL.vec <- sapply(log(lambda.vec), NLL.pois)
plot(lambda.vec, NLL.vec-min(NLL.vec), type="l", xlab="lambda", ylab="Likelihood profile")

Res.pois <- optim(0, NLL.pois, method="BFGS", hessian=T)
lambda.est <- exp(Res.pois$par)
lambda.se <- lambda.est*sqrt(1/Res.pois$hessian)
data.frame(lambda.est,lambda.se)
```

```
##   lambda.est lambda.se
## 1   2.650007 0.3640059
```

```
points(lambda.est, 0, pch=4, col="blue", cex=2)
lines(lambda.est+c(-1.96,1.96)*lambda.se, c(0,0), col="red", lwd=2)
```



Occupancy model (2) -Presence-Absence model-

Description

- Case 2: Like in Case 1, suppose that you are studying occupation of a species in a study area. Here, let us assume that you can observe only "presence/absence" without missing (0 means absence). (In this case, you can increase the number of sample locations (k))
- Notation: $Z_1, Z_2, \ldots, Z_k \sim (iid)Bin(1, p(\lambda))$, where $p(\lambda) = what?$.
- Task: to estimate λ
- simulation data

```
Ns <- 50
lambda <- 3
yy <- rpois(Ns, lambda)
zz <- as.numeric(yy>0)
zz
```

```
##  [1] 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [36] 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
```

Definition of likelihood function and implementation of optimization

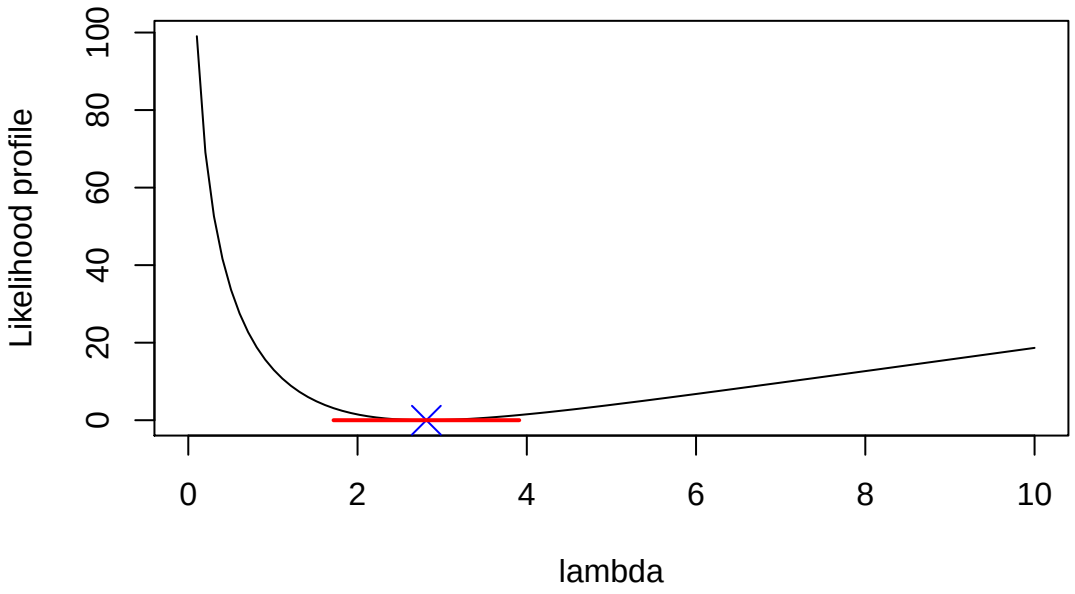
```
#Def of negative loglikelihood-2
NLL.PA <- function(loglambda){
 lambda <- exp(loglambda)
 pp <- 1-exp(-lambda)
```

```
 logPF <- dbinom(sum(zz), Ns, pp, log=T)
 NLL <- (-1.0)*sum(logPF)
 return(NLL)
}

lambda.vec <- seq(0, 10, length.out=100)
NLL.vec <- sapply(log(lambda.vec), NLL.PA)
plot(lambda.vec, NLL.vec-min(NLL.vec), type="l", xlab="lambda", ylab="Likelihood profile")

Res.PA <- optim(0, NLL.PA, method="BFGS", hessian=T)
lambda.est <- exp(Res.PA$par)
lambda.se <- lambda.est*sqrt(1/Res.PA$hessian)
data.frame(lambda.est,lambda.se)
```

```
##   lambda.est lambda.se
## 1    2.81341 0.5597618
```

```
points(lambda.est, 0, pch=4, col="blue", cex=2)
lines(lambda.est+c(-1.96,1.96)*lambda.se, c(0,0), col="red", lwd=2)
```

```
## Warning in c(-1.96, 1.96) * lambda.se: Recycling array of length 1 in vector-array arithmetic is depr
##   Use c() or as.vector() instead.
```



Occupancy model (3) - Rather realistic occupancy model-

Description

- Case 3: Like in the previous cases, suppose that you are studying occupation of a species in a study area. Again, suppose that you visit each sampling location several day and observe if at least one individual exist of not ("presence/absence) every visit", but your detection process is subject to missing.

The probability of detecting presence depends on the number of individuals in a sampling unit and detection probability per individual. Note that the number of individuals in each location does not change during the study period.

- Notation:

$$N_1, N_2, \ldots, N_k \sim (iid) Pois(\lambda)$$
$$Z_{it}|N_i \sim (iid) Bin(1, p(N_i)), \quad \text{where } p(N_i) = 1-(1-p)^{N_i} \quad (t = 1, 2, ..., T_i; i = 1, 2, \ldots, k)$$
$$Y_i|N_i \sim Bin(T_i, p(N_i)) \quad (i = 1, 2, \ldots, k)$$

- Task: to estimate λ and p (λ is of your interest)
- simulation data

```
lambda <- 3
pp <- 0.2
Ns <- 50
Tmax <- 10
Tvec <- sample(1:Tmax, Ns, replace=T)
zz <- array(NA, c(Ns, Tmax))
yy <- numeric(Ns)

for(i in 1:Ns){
NN <- rpois(Ns, lambda)
zz[i,1:Tvec[i]] <- rbinom(Tvec[i], 1, 1-(1-pp)^NN[i])
yy[i] <- sum(zz[i,], na.rm=T)
}

head(zz, 10)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    1    0    0    1   NA   NA   NA    NA
##  [2,]    0    0    0    1    0    1    1    0    1     0
##  [3,]    0   NA   NA   NA   NA   NA   NA   NA   NA    NA
##  [4,]    1    1    1    1    1    1    1    1    1    NA
##  [5,]    1    0    0    0    1    0    0   NA   NA    NA
##  [6,]    1   NA   NA   NA   NA   NA   NA   NA   NA    NA
##  [7,]    1    1    0    1    1    0    1    1   NA    NA
##  [8,]    1    0    0    0    0    1    1    0    0    NA
##  [9,]    1    1    0   NA   NA   NA   NA   NA   NA    NA
## [10,]    1    0    1    0    1    0    1    1    0    NA
```

```
yy
```

```
##  [1] 3 4 0 9 2 1 6 3 2 5 0 8 5 1 0 3 4 7 1 2 4 5 5 2 4 1 3 4 0 6 0 0 5 3 3
## [36] 1 0 5 2 1 6 0 0 1 0 1 7 2 5 3
```

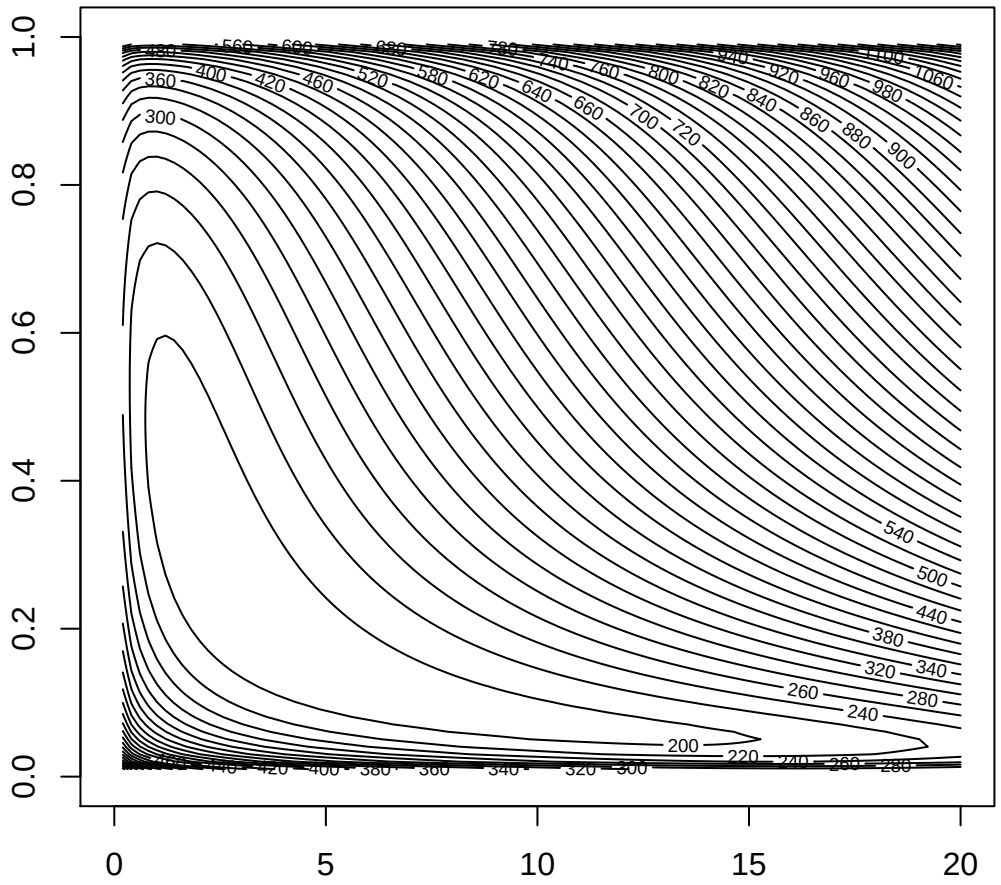
Defining likelihood with respect to $\log\lambda$ and logit$p$

```
NLL <- function(par, Kmax=20){
 lambda <- exp(par[1])
 pp <- 1/(1+exp(-par[2]))
 Detectvec <<- 1-(1-pp)^(0:Kmax)

 Tmp1 <- dpois(0:Kmax, lambda)
 Likevec <- numeric(Ns)
```

```
 for(i in 1:Ns){
  Tmp2 <- Detectvec^yy[i]*(1-Detectvec)^(Tvec[i]-yy[i])
  Likevec[i] <- sum(Tmp1*Tmp2)
 }
 obj <- (-1.0)*sum(log(Likevec))
}
```

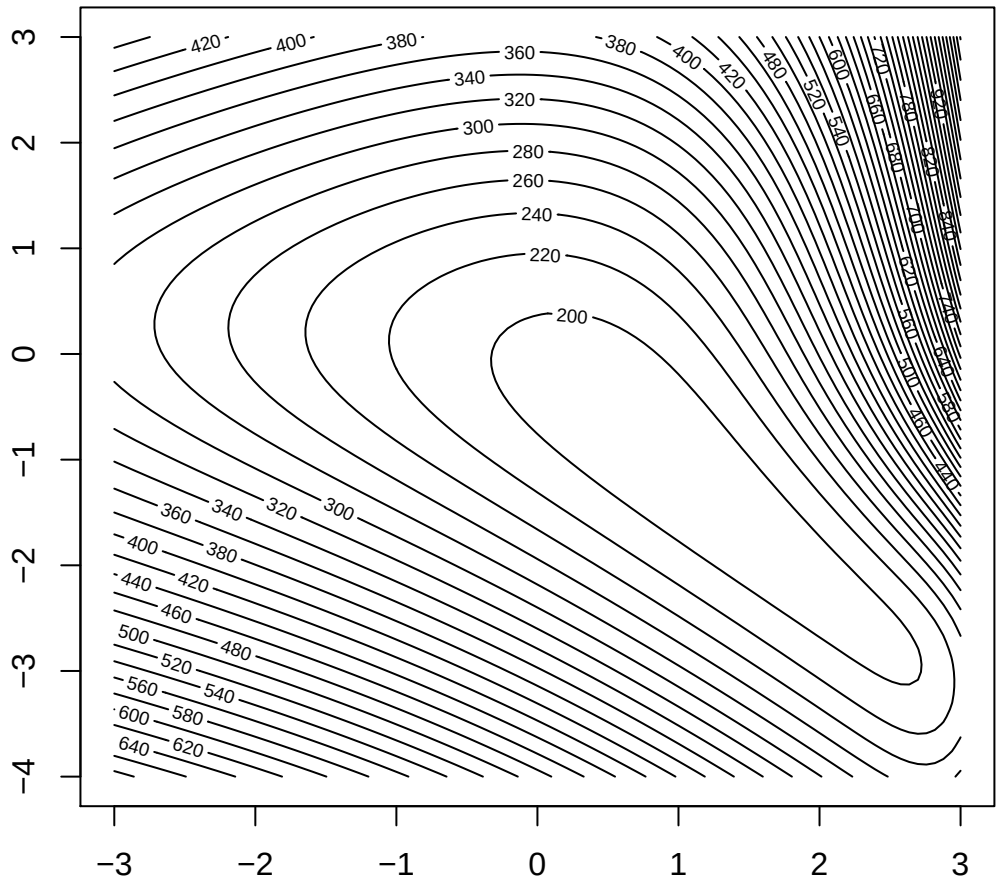
Contour of likelihood with respect to λ and p

```
Len <- 100
LL <- array(0, c(Len,Len))
lvec <- seq(0,20,length.out=Len)
pvec <- seq(0,1,length.out=Len)

for(i in 1:Len){
  for(j in 1:Len){
    LL[i,j] <- NLL(c(log(lvec[i]), logit(pvec[j])), Kmax=20)
  }
}
contour(lvec,pvec,LL,nlevels=50)
```



Contour of likelihood with respect to $log\lambda$ and logit$p$

```
Len <- 100
LL <- array(0, c(Len,Len))
loglvec <- seq(-3,3,length.out=Len)
logitpvec <- seq(-4,3,length.out=Len)
for(i in 1:Len){
  for(j in 1:Len){
    LL[i,j] <- NLL(c(loglvec[i], logitpvec[j]), Kmax=20)
  }
}
contour(loglvec,logitpvec,LL,nlevels=50)
```



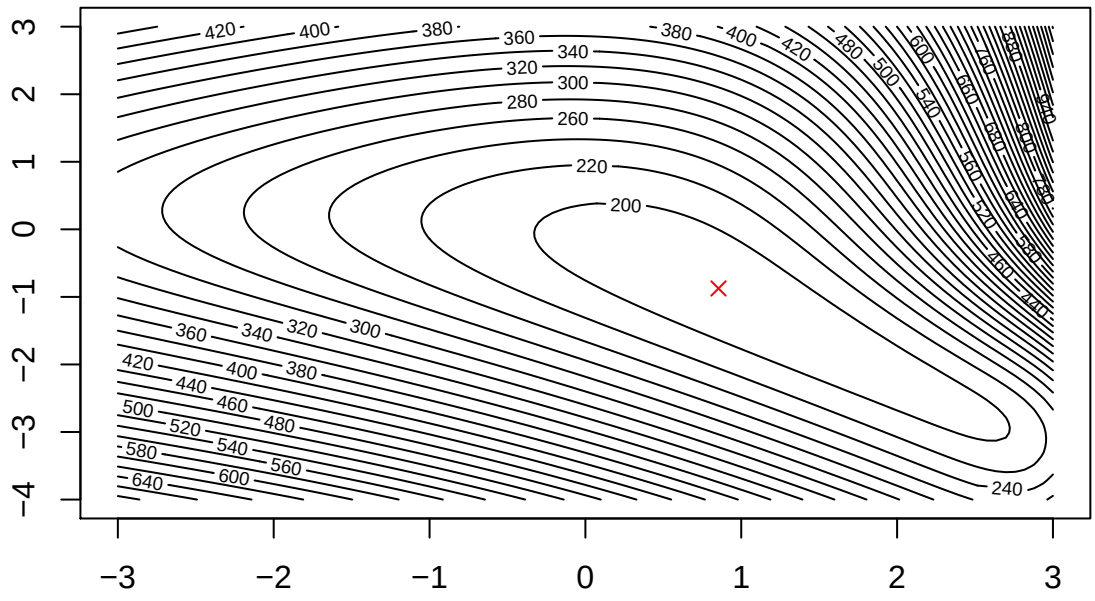
Optimization with respect to $\log \lambda$ and logit$p$

```
init <- c(0,0)
Res <- optim(init, NLL, method="BFGS", hessian=T, Kmax=20)
Res
```

```
## $par
## [1]  0.8541628 -0.8748722
##
## $value
## [1] 186.7679
##
## $counts
## function gradient
##       28        9
##
## $convergence
```

```
## [1] 0
##
## $message
## NULL
##
## $hessian
##          [,1]     [,2]
## [1,] 61.08435 43.98288
## [2,] 43.98288 40.34284
```

```
contour(loglvec,logitpvec,LL,nlevels=50)
points(Res$par[1],Res$par[2], col="red", pch=4)
```



Estimates and their standard errors

More accurate CIs can be given by profile likelihood methods, bootstrapping and Bayesian methods.

```
lambda.est <- exp(Res$par[1])
p.est <- 1/(1+exp(-Res$par[2]))
Avar <- solve(Res$hessian)
lambda.se <- lambda.est*sqrt(Avar[1,1])
p.se <- p.est*(1-p.est)*sqrt(Avar[2,2])
data.frame(lambda.est, lambda.se, p.est, p.se)
```

```
##   lambda.est lambda.se     p.est       p.se
## 1   2.349407 0.6483002 0.2942415 0.07051147
```