Estimating model parameters by maximum likelihood*
Measles in Niamey, Niger

John M. Drake & Pejman Rohani

with contributions from
Ben Bolker, Matt Ferrari, Aaron King and Dave Smith

1 Introduction

In the previous exercise we saw that the trajectory of an epidemic contains information not only about R_0, but also about the fixed paramters that deter ine transmission (i.e., β and γ). We found least-squares fitting to be a a powerful, general and straightforward approach to estimating parameters. There are several problems with elast squares fitting, however, including:

- There is an element of arbitrariness in the choice of objective function
- Although we could fairly easily obtain point estimates of model parameters using least-squares, it was not clear how we could obtain concomitant estimates of parameter uncertainty (e.g., confidence intervals)
- Under many conditions there are limits to parameter estimability (i.e., *identifiability*).

This exercise explores these issues and introduces maximum likelihood as an attractive resolution to the first and second of these. The third is a fundamental challenge.

2 The likelihood

Likelihood theory has many advantages:

1. Efficiency
2. A deep and general theory
3. Sound theoretical basis for confidence intervals and model selection
4. Fidelity to model

General definition

Likelihood is the *probability of a given set of data D having occurred under a particular hypothesis (or model) H*:

$$\mathcal{L}(H, D) = D|H$$

A simple example: suppose n individuals participate in a serological survey and k of these individuals are found to be seropositive. One parameter of interest is the true fraction, p, of the population that has seroconverted. Assuming the sample was drawn at random and the population is large, then the probability of the data (m of n individuals seropositive) given the hypothesis that the true probability is p is

$$D|H = \binom{n}{k} p^k (1-p)^{n-k}$$

If the true seroprevalence was, say, $p = 0.3$, then Fig. 1 shows the probability of observing k seropositives in a sample of size $n = 50$.

```
p <- 0.3
n <- 50
k <- seq(0,50,by=1)
prob <- dbinom(x=k,size=n,prob=p)
plot(k,prob,type='h',lwd=5,lend=1,
     ylab="probability")
```

Thus, the likelihood is a function of p:

$$\mathcal{L}(p) = \binom{n}{k} p^k (1-p)^{n-k}$$

Often, it will be convenient to work with the logarithm of this function, which we call "log-liklihood". Looking at this function for each of two different surveys:

```
k1 <- 18
n1 <- 50
p <- seq(0,1,by=0.001)
plot(p,dbinom(x=k1,size=n1,prob=p,log=TRUE),
     ylim=c(-10,-2),ylab="log-likelihood",
     type='l')
abline(h=dbinom(x=k1,size=n1,prob=k1/n1,log=TRUE)-
       0.5*qchisq(p=0.95,df=1),col='red')
abline(v=k1/n1,col='blue')

k2 <- 243
n2 <- 782
p <- seq(0,1,by=0.001)
plot(p,dbinom(x=k2,size=n2,prob=p,log=TRUE),
     ylim=c(-10,-2),ylab="log-likelihood",
     type='l')
abline(h=dbinom(x=k2,size=n2,prob=k2/n2,log=TRUE)-
       0.5*qchisq(p=0.95,df=1),col='red')
abline(v=k2/n2,col='blue')
```

The results of this are shown in Fig. 2.

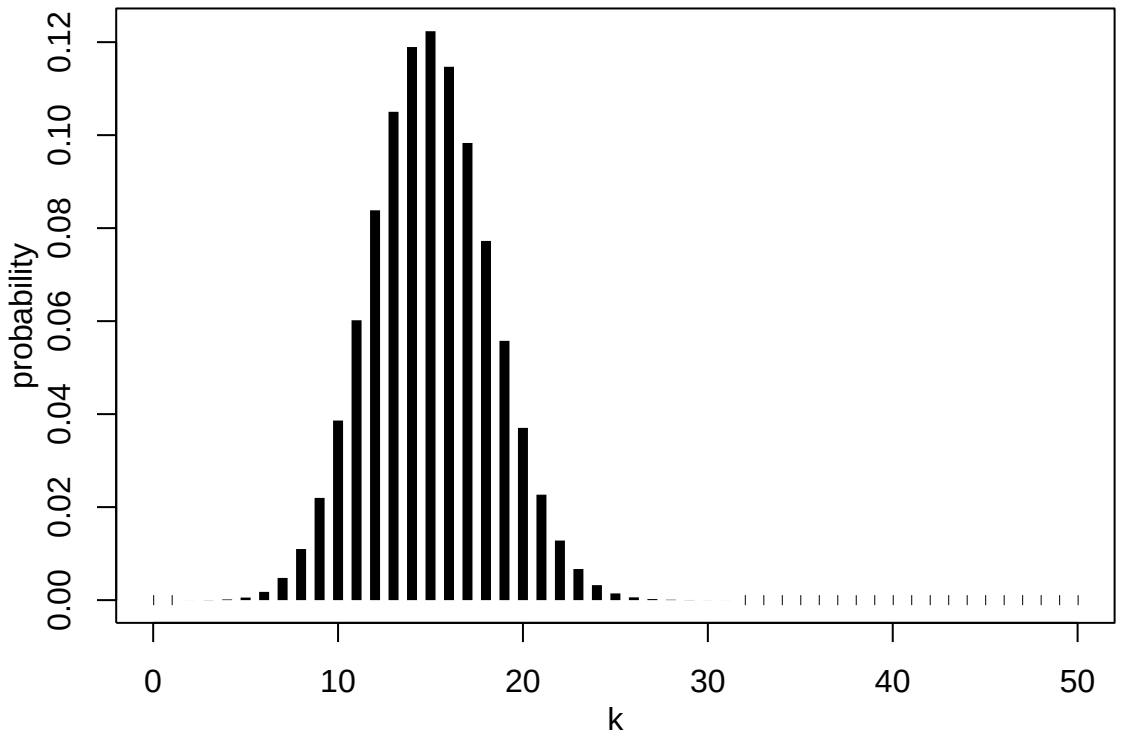


Figure 1: Probability of observing k seropositive individuals in a sample of size 50 when the true seroprevalence in a large population is 0.3.

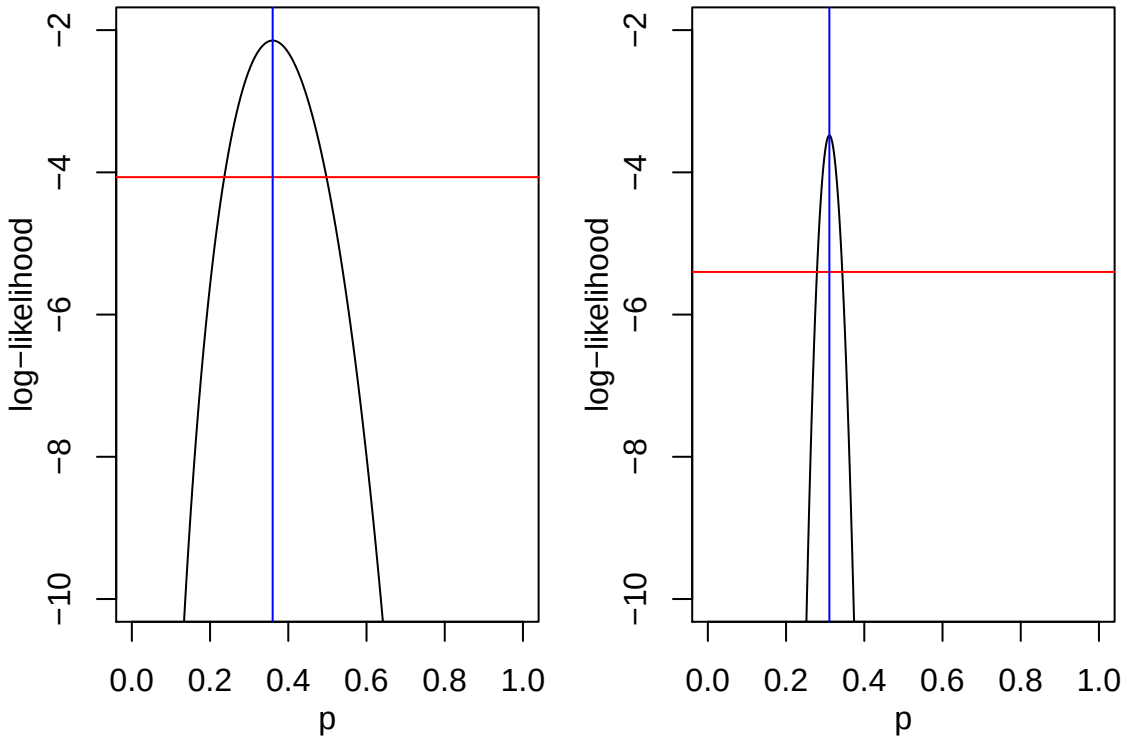


Figure 2: Binomial likelihood for two serological surveys. The likelihood is a function of the model parameters, in this case p. Vertical lines show the maximum likelihood estimate (MLE) of p. Horizontal lines show the critical likelihoods for the likelihood ratio test at the 95% confidence level.

From data points to data sets

Let's suppose we have three samples, D_1, D_2, D_3, taken by three different researchers, for the same large population. If these samples are *independent*, then

$$D|H = D_1|H \times D_2|H \times D_3|H$$

which means that the likelihood of the full data set is the product of the likelihoods from each of the samples. In other words, the likelihood gives a general recipe for combining data from different studies. We'd compute the likelihood as follows:

```
n <- c(13,484,3200)
k <- c(4,217,1118)
dbinom(x=k,size=n,prob=0.2,log=TRUE)
```

```
[1]   -1.873761  -79.243371 -197.561806
```

```
sum(dbinom(x=k,size=n,prob=0.2,log=TRUE))
```

```
[1] -278.6789
```

```
ll.fn <- function (p) {
  sum(dbinom(x=k,size=n,prob=p,log=TRUE))
}
p <- seq(0,1,by=0.001)
loglik <- sapply(p,ll.fn)

plot(p,loglik,type='l',ylim=max(loglik)+c(-10,1))
```

3 Fitting SIR to an epidemic curve using likelihood

Let's revisit the least squares model-fitting we did for the case of measles in Niger. Let's simplify the model slightly to eliminate some unnecessary elements. Our SIR model is

$$\frac{dS}{dt} = -\beta S I$$
$$\frac{dI}{dt} = \beta SI - \gamma I$$
$$\frac{dR}{dt} = \gamma I$$

The R code for this model is:

```
require(deSolve)
sir.model.closed <- function (t, x, params) {    #here we begin a function with three arguments
  S <- x[1]                                      #create local variable S, the first element of x
  I <- x[2]                                      #create local variable I
  R <- x[3]                                      #create local variable R
  with(                                          #we can simplify code using "with"
       as.list(params),                          #this argument to "with" lets us use the variable names
       {                                         #the system of rate equations
         dS <- -beta*S*I
         dI <- beta*S*I-gamma*I
         dR <- gamma*I
         dx <- c(dS,dI,dR)                       #combine results into a single vector dx
         list(dx)                                #return result as a list
       }
       )
}
```

Here we use likelihood instead. Let's suppose that, when we record cases, we make errors that are normal. First, we write an auxiliary function that will return a vector of the model predictions.

```
prediction <- function (params, times) {
  xstart <- params[c("S.0","I.0", "R.0")]
  out <- ode(
             func=sir.model.closed,
             y=xstart,
             times=c(0,times),
             parms=params
             )
  out[-1,3]     # return the I variable only
}
```

Now we can compute the likelihood of the data given the model and its parameters:

```
loglik <- function (params, data) {
  times <- data$biweek/26
  pred <- prediction(params,times)
  sum(dnorm(x=data$measles,mean=pred,sd=params["sigma"],log=TRUE))
}
```

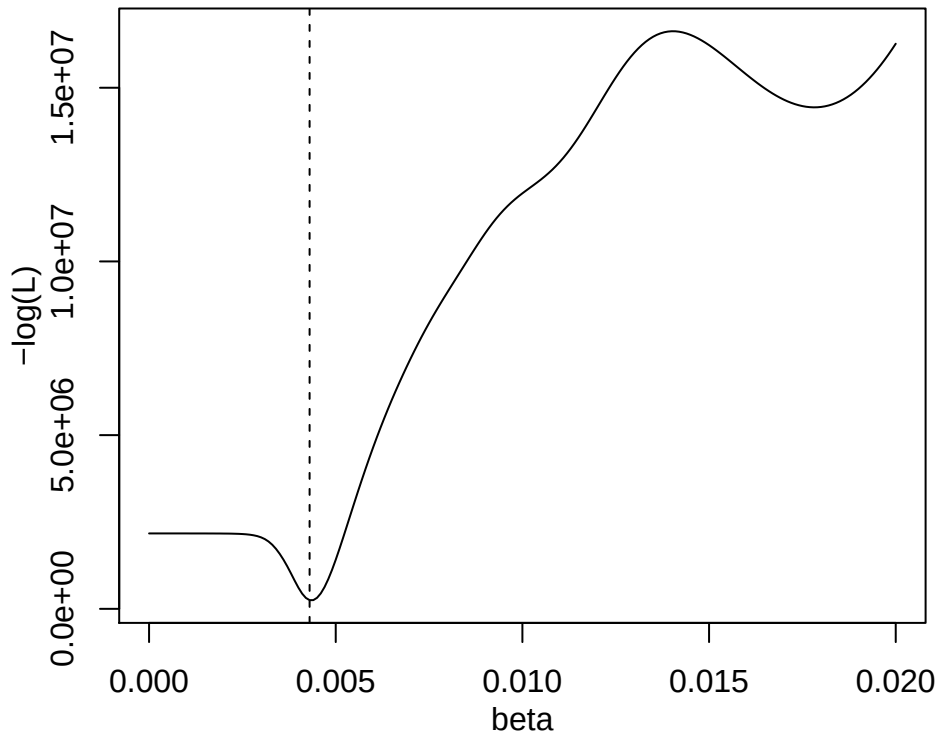


Figure 3: Log likelihood of the SIR model for community "A" of Niamey assuming normal errors with constant error standard deviation $\sigma = 1$.

```
load('data.RData')
dat <- data.frame(biweek=seq(1:dim(niamey)[1]), measles=niamey[,1])
#niamey <- read.csv(file="niamey_measles.csv")
#dat <- subset(niamey,community=="A")

params <- c(S.0=10000,I.0=10,gamma=365/13,beta=NA,sigma=1)
f <- function (beta) {
  par <- params
  par["beta"] <- beta
  loglik(par,dat)
}
beta <- seq(from=0,to=0.02,by=0.0001)
ll <- sapply(beta,f)
plot(beta,-ll,type='l',ylab="-log(L)")
beta.hat <- beta[which.max(ll)]
abline(v=beta.hat,lty=2)
```

We plot the results in Fig. 3. The great similarity in the likelihood estimate to our first least-squares estimate is no accident. Why is this? Let Y_t be the observed number of infectives at time t and I_t be

the model's prediction. Then the key component of the likelihood is

$$\begin{aligned}\log Y_t|I_t &= \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\,\exp\left(-\frac{(Y_t-I_t)^2}{2\sigma^2}\right)\right)\\ &= -\tfrac{1}{2}\,\log 2\pi\sigma^2 - \tfrac{1}{2}\,\frac{(Y_t-I_t)^2}{\sigma^2}\end{aligned}$$

and

$$\log\mathcal{L} = -\tfrac{1}{2}\left(\frac{1}{\sigma^2}\,\sum_t (Y_t-I_t)^2 + \log\left(\sigma^2\right) + \log\left(2\pi\right)\right)$$

So MLE and least-squares are essentially identical if the errors are normal with constant variance!

4 Modeling the noise

All this raises the question of what the best model for the errors is. Suppose that the data represent Poisson samples with expectation $p\,I$, where p is the reporting probability and I represents the true prevalence:

$$Y_t \;\sim\; \text{Poisson}(p\,I_t)$$

This leads to the following likelihood function

```
poisson.loglik <- function (params, data) {
  times <- data$biweek/26
  pred <- prediction(params,times)
  sum(dpois(x=data$measles,lambda=params["p"]*pred[-1],log=TRUE))
}
```

Let's see what the MLE parameters are for this model. We'll start by estimating just one parameter, using the function `mle2` from the bbmle package.

Now, we must have $beta > 0$. This is a *constraint* on the parameter. As before, we enforce this constraint is by log-transformation.

```
require(bbmle)
#niamey <- read.csv(file="niamey_measles.csv")
#dat <- subset(niamey,community=="A")

params <- c(S.0=20000,I.0=1, R.0=0, gamma=365/13,b=NA,p=0.2)
## objective function
f <- function (log.beta) {
  par <- params
  par[c("beta")] <- exp(log.beta)
  -poisson.loglik(par,dat)
}
guess <- list(log.beta=log(220/50000))
fit0 <- mle2(f,start=guess); fit0
```

```
Call:
mle2(minuslogl = f, start = guess)
```

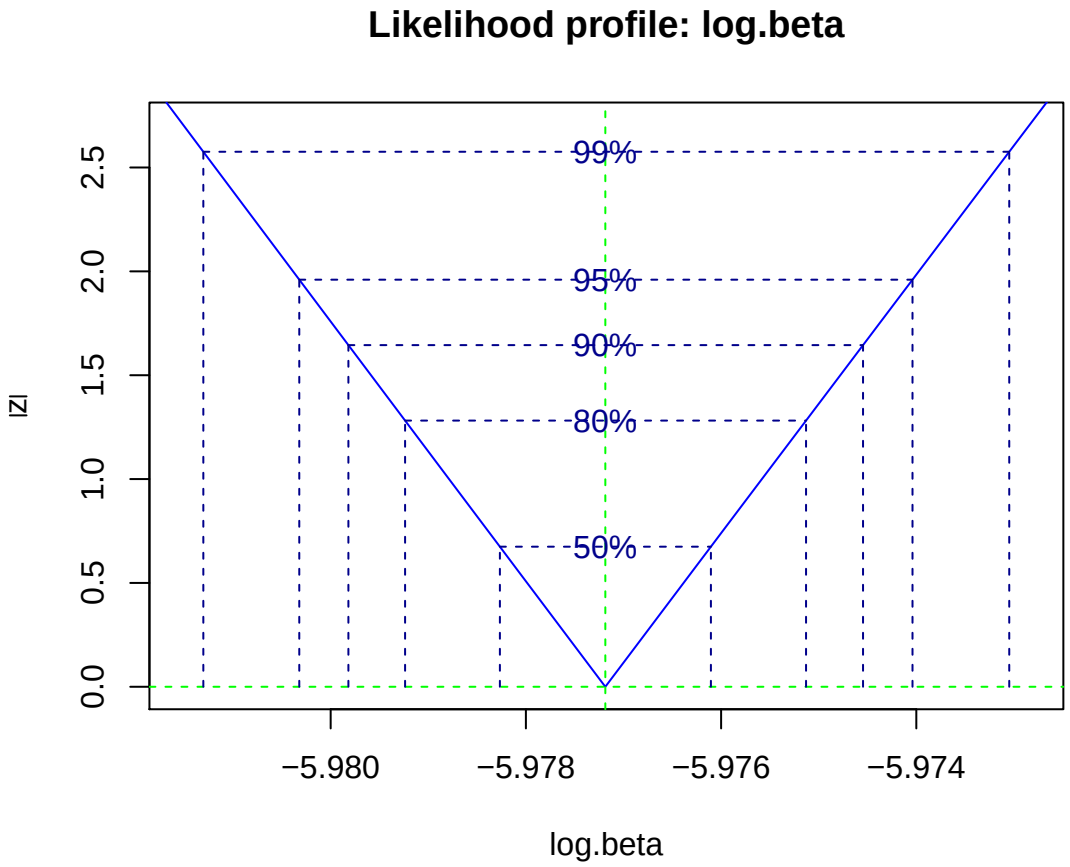
```
Coefficients:
 log.beta
-5.977186

Log-likelihood: -1963.32
```

```
 fit <-  mle2(f,start=as.list(coef(fit0))); fit
```

```
Call:
mle2(minuslogl = f, start = as.list(coef(fit0)))

Coefficients:
 log.beta
-5.977186

Log-likelihood: -1963.32
```

We can get an idea about the uncertainty and in particular obtain confidence intervals using the *profile likelihood*. In bblme, this is easy to obtain.

```
 prof.beta <- profile(fit)
 plot(prof.beta, col.conf='darkblue')
```


Likelihood profile: log.beta
|z|
0.0
0.5
1.0
1.5
2.0
2.5
99%
95%
90%
80%
50%
−5.980
−5.978
−5.976
−5.974
log.beta

Now let's try to estimate both β and the reporting probability p. Since we have constraints on p ($0 \leq p \leq 1$), we'll transform it as well. However, in this case we'll use the *logit* function (and its inverse for back-transformation):

$$\text{logit}(p) = \log \frac{p}{1-p}$$
$$\text{ilogit}(x) = \frac{1}{1+\exp(-x)}$$

```
#niamey <- read.csv(file="niamey_measles.csv")
#dat <- subset(niamey,community=="A")

params <- c(S.0=20000,I.0=1, R.0=0, gamma=365/13,b=NA,p=NA)
logit <- function (p) log(p/(1-p))      # the logit transform
ilogit <- function (x) 1/(1+exp(-x))    # inverse logit
f <- function (log.beta, logit.p) {
  par <- params
  par[c("beta","p")] <- c(exp(log.beta),ilogit(logit.p))
  -poisson.loglik(par,dat)
}
guess <- list(log.beta=log(0.005),logit.p=logit(0.2))
fit0 <- mle2(f,start=guess); fit0
```

```
Call:
mle2(minuslogl = f, start = guess)

Coefficients:
  log.beta    logit.p
-5.9918687 -0.1470358

Log-likelihood: -394.2
```

```
fit <-  mle2(f,start=as.list(coef(fit0))); fit
```

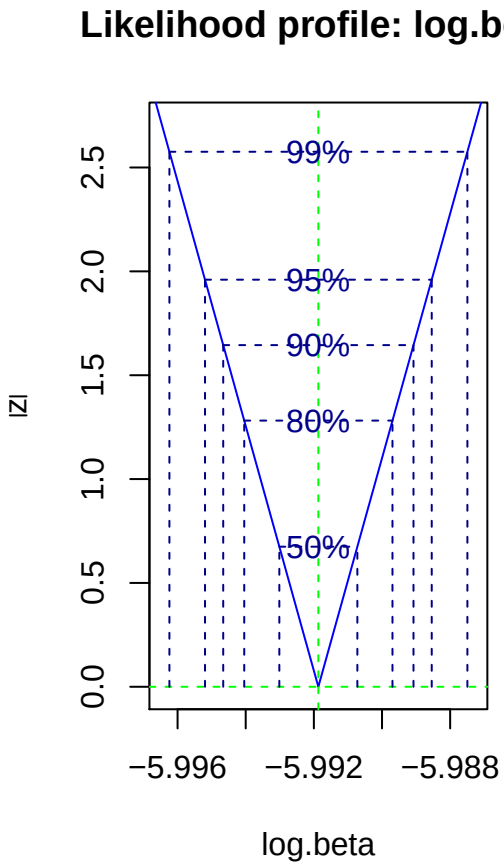
```
Call:
mle2(minuslogl = f, start = as.list(coef(fit0)))

Coefficients:
  log.beta    logit.p
-5.9918687 -0.1470358

Log-likelihood: -394.2
```

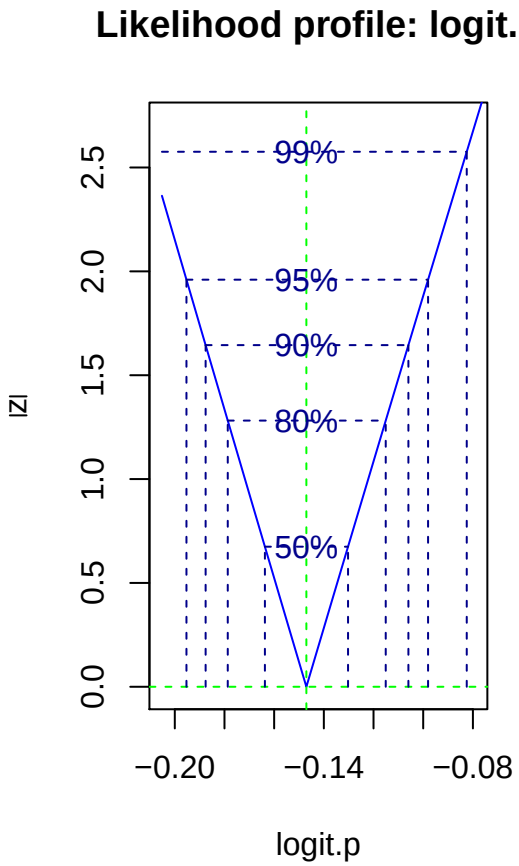
```
prof2 <- profile(fit)
```

```
DLSODA-  Warning..Internal T (=R1) and H (=R2) are
      such that in the machine, T + H = T on the next step
     (H = step size). Solver will continue anyway.
In above message, R1 = 0, R2 = 0

DINTDY-  T (=R1) illegal
```

```
In above message, R1 = 0.0384615

      T not in interval TCUR - HU (= R1) to TCUR (=R2)
In above message, R1 = 0, R2 = 0

DINTDY-  T (=R1) illegal
In above message, R1 = 0.0769231

      T not in interval TCUR - HU (= R1) to TCUR (=R2)
In above message, R1 = 0, R2 = 0

DLSODA-  Trouble in DINTDY.  ITASK = I1, TOUT = R1
In above message, I1 = 1

In above message, R1 = 0.0769231

DLSODA-  Warning..Internal T (=R1) and H (=R2) are
      such that in the machine, T + H = T on the next step
     (H = step size). Solver will continue anyway.
In above message, R1 = 0, R2 = 0

DINTDY-  T (=R1) illegal
In above message, R1 = 0.0384615

      T not in interval TCUR - HU (= R1) to TCUR (=R2)
In above message, R1 = 0, R2 = 0

DINTDY-  T (=R1) illegal
In above message, R1 = 0.0769231

      T not in interval TCUR - HU (= R1) to TCUR (=R2)
In above message, R1 = 0, R2 = 0

DLSODA-  Trouble in DINTDY.  ITASK = I1, TOUT = R1
In above message, I1 = 1

In above message, R1 = 0.0769231
```

```
plot(prof2, col.conf='darkblue')
```


Likelihood profile: log.beta
|z|
0.0
0.5
1.0
1.5
2.0
2.5
99%
95%
90%
80%
50%
−5.996
−5.992
−5.988
log.beta


Likelihood profile: logit.p
|z|
0.0
0.5
1.0
1.5
2.0
2.5
99%
95%
90%
80%
50%
−0.20
−0.14
−0.08
logit.p

We can also get confidence intervals:

```
 confint(prof2)
```

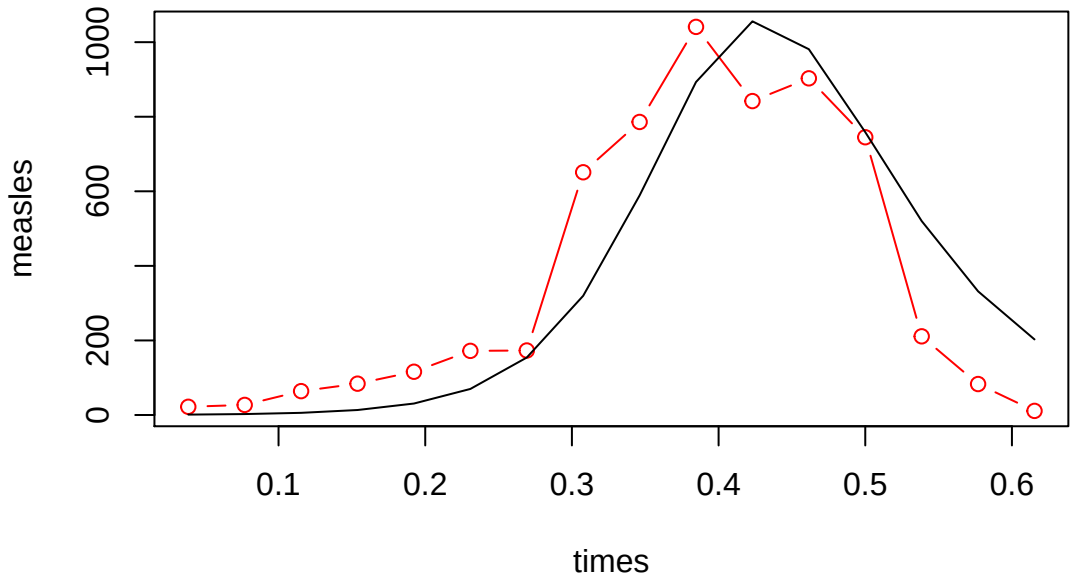
```
             2.5 %      97.5 %
log.beta -5.995195 -5.98854083
logit.p  -0.195306 -0.09803483
```

```
 ci <- confint(prof2)
 ci[1,] <- exp(ci[1,])
 ci[2,] <- ilogit(ci[2,])
 rownames(ci) <- c("b","p")
 ci
```

```
        2.5 %     97.5 %
b 0.002490692 0.00250732
p 0.451328117 0.47551090
```

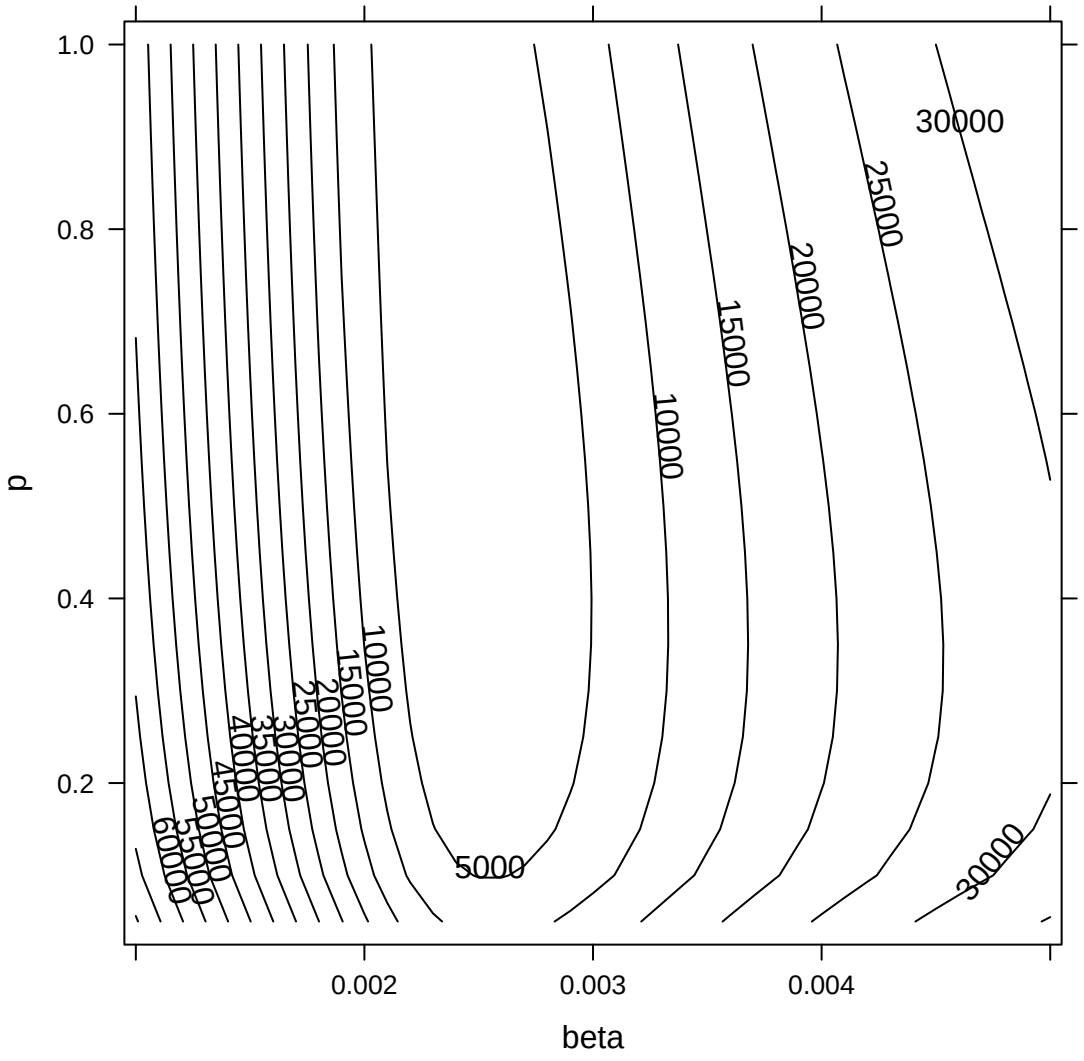
Let's look at the model's predictions at the MLE:

```
 params["beta"] <- exp(coef(fit)["log.beta"])
 params["p"] <- ilogit(coef(fit)["logit.p"])
 times <- c(dat$biweek/26)
 model.pred <- prediction(params,times)
 plot(measles~times,data=dat,type='b',col='red')
 lines(times,params["p"]*model.pred,type='l')
```



Let's revisit the contour diagram we drew before.

```
#niamey <- read.csv(file="niamey_measles.csv")
#dat <- subset(niamey,community=="A")

## this time the objective function has to
## take a vector argument
f <- function (pars) {
  par <- params
  par[c("beta","p")] <- as.numeric(pars)
  -poisson.loglik(par,dat)
}
beta <- seq(from=0.001,to=0.005,by=0.0001)
p <- seq(0,1,by=0.05)
grid <- expand.grid(beta=beta,p=p)
grid$loglik <- apply(grid,1,f)
grid <- subset(grid,is.finite(loglik))
require(lattice)
contourplot(loglik~beta+p,data=grid,cuts=20)
```



Exercise 1. Revisit the other communities of Niamey and/or the British boarding school influenza data using the Poisson model and `bbmle`.

***Exercise 2.** Try to fit p, b, and S_0 simultaneously.