Universitat Autònoma de Barcelona

Facultat de Ciències

Homework 3: Generalized Linear Models

Ona Sánchez & Andrea González

1601181 — 1603921

15/05/2022

Índex

1 Problem Statement

The number of registered cases of COVID-19 in Catalonia, day by day, from 27/02/2020 to 31/03/2020, are the following:

y = c(2, 3, 5, 6, 15, 15, 15, 24, 24, 24, 49, 75, 124, 156, 260, 316, 509, 715, 903, 1394, 1866, 2702, 3270, 4203, 4704, 5925, 7864, 9937, 11592, 12940, 1423, 15026, 16157, 18773)

We want to analyze the evolution of the number of affected individuals as a function of time $A(t)$.

1.1 Part 1

Fit the data using the following exponential functions,

$$A_1(t) = e^{\beta_0+\beta_1 \cdot t}, \quad A_2(t) = e^{\beta_0+\beta_1 \cdot t+\beta_2 \cdot t^2} \tag{1}$$

To do this assume that the observations are Poisson distributed. Fit the models using glm and nlm functions in R.

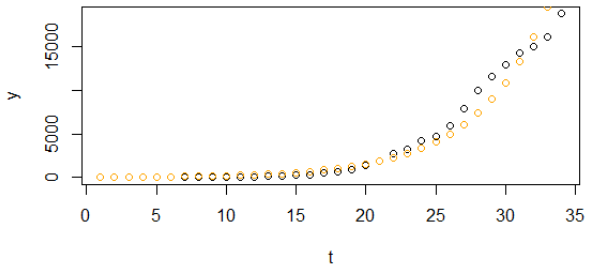
- Which model is better? Create a plot to illustrate the fitted models and comment on the results.

  General code:

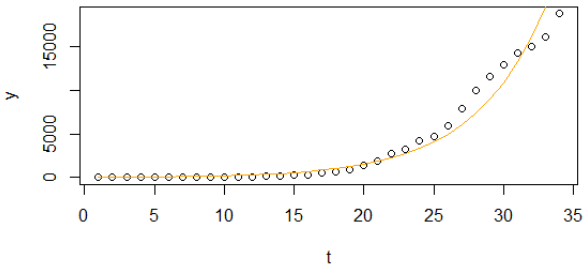
```
#Introduction of the data
y = c(2, 3, 5, 6, 15, 15, 15, 24, 24, 24, 49, 75, 124, 156, 260, 316,
509, 715, 903, 1394, 1866, 2702, 3270, 4203, 4704, 5925, 7864, 9937,
11592, 12940, 14230, 15026, 16157, 18773)

t = seq(1,34)
t2 = seq(1,34)*seq(1,34)
```

1.2 A1

```
#GLM
fit = glm(formula = y~t, family=poisson)
summary(fit)
plot(t,y)
lines(t,fit\$fitted.values,type="p", col="orange")
-----------------------------------------------------------------
#NLM
loglik<-function(beta){
  mu=exp(-beta[1]-beta[2]*t)
  loglik=-sum(-mu + y*log(mu))
  }

llike = nlm(loglik,p=c(0,0), hessian=T)
llike$estimate = -llike$estimate

plot1 = function(t){
  y = exp(llike$estimate[1]+llike$estimate[2]*t)
  return(y)
}

plot(t, y)
points(t,plot1(t), type = 'l', col='orange')
```



(a) a_1 glm



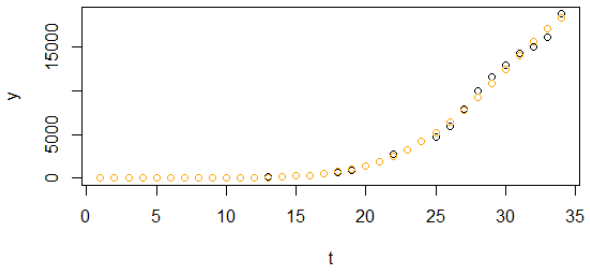
(b) a_1 nlm

Executing the code shown above using glm, we can estimate the values of β_0 and β_1, the results are: 3.4273221 and 0.1955985 respectively. On the other hand, using nlm, the results are: 3.4273763 and 0.1955967. Comparing the values, we can see that are really close.

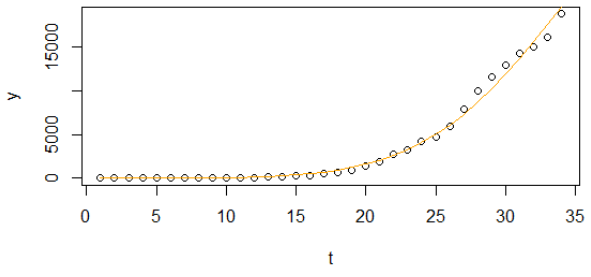
So that, when we plot the functions, they are almost indistinguishable. The black points in the plot indicate the real values of the observed data whereas the orange ones are the fitted data.

1.3 A2

```
#GLM
fit = glm(formula = y~t+t2, family=poisson)
summary(fit)
plot(t,y)
lines(t,fit$fitted.values,type="p", col="orange")

-----------------------------------------------------------------

#NLM
loglik<-function(beta){
  mu=exp(beta[1]+beta[2]*t+beta[3]*t2)
  loglik=-sum(-mu + y*log(mu))
  (loglik)}

llike = nlm(loglik,p=c(0,0,0), hessian=T)
llike

plot1 = function(t){
  y = exp(llike$estimate[1]+llike$estimate[2]*t+llike$estimate[3]*t2)
  return(y)
}

plot(t, y)
points(t,plot1(t), type = 'l', col='orange')
```



(a) a_2 glm



(b) a_2 nlm

Executing the code shown above using glm, we can estimate the values of β_0, β_1 and β_2, the results are: -2.3189932, 0.6527898 and -0.0087034 respectively. On the other hand, using nlm, the results are: 0.033420390, 0.478314608 and -0.005551174. Comparing the values, we can see that excepting β_0 are not so different.

Although the values of β_0 calculated with the shown codes using *glm* and *nlm* are a bit different, we can observe in the graphic that they fit fairly pretty well to real values.
The black points in the plot indicate the real values of the observed data whereas the orange ones are the fitted data.

◦ Answering to the previous question: *Which model is better?*
Comparing the graphics of A_1 and A_2, we can see that the exponential function $A_2(t)$ fits better our data. We can deduce this due to observing that the fitted and observed data are equal for a longer time during the plot (we can see that the orange points are overlapping the black ones only for 6 days with A_1, while A_2 doubles the time). This is easily explained taking into account that, when in the plot of the function only appears the orange color, means that both functions (black and orange in the graphic) have the same values.

- For the second model $A_2(t)$ what is the predicted day for which the speed of growing will be zero?
To do this section, we have to consider that the points where the slope of a function is zero, is in a maximum or minimum of a function. Taking this into account, the first step that we have done has been to calculate the derivate of our function and then, we have proceeded to set it equal to zero. The calculations have been the following:

$$f = e^{\beta_0+\beta_1 t+\beta_2 t^2}$$
$$df = e^{\beta_0+\beta_1 t+\beta_2 t^2} \cdot (\beta_1 + 2 \cdot \beta_2 \cdot t)$$
$$df = 0 \rightarrow e^{\beta_0+\beta_1 t+\beta_2 t^2} \cdot (\beta_1 + 2 \cdot \beta_2 \cdot t) = 0 \rightarrow \beta_1 + 2 \cdot \beta_2 \cdot t = 0$$
$$t = -\frac{\beta_1}{\beta_2}$$

1.4 Part 2

Fit the data using a two-parameters sigmoid function of the form,

$$A(t) = \frac{A_0 \cdot C}{A_0 + (C - A_0) \cdot e^{-\beta \cdot t}}, \quad \text{where } A(0) = A_0, \beta > 0 \tag{2}$$

- Is it glm model?

  Our function $A(t)$ is logistic, that is to say that has a common S-shaped curve, and our samples are *Poisson* distributed, it can be studied by the *glm* function in *R*. Although it works correctly, it's not the most efficient way.

- Fit the data using nlm assuming again that the observations are Poisson distributed.

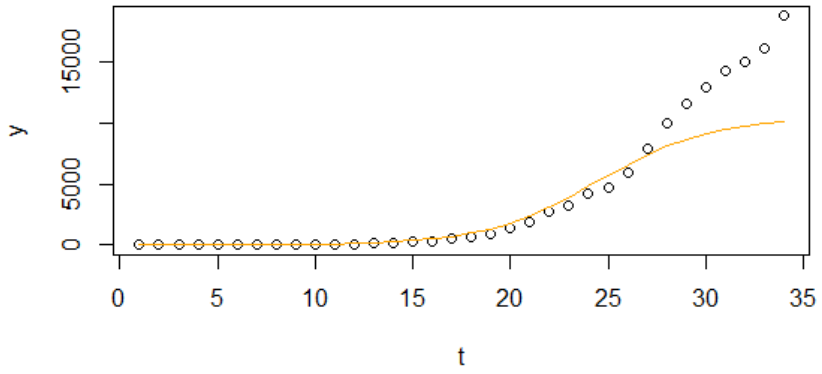
  Code used:

```
loglik<-function(param){
mu= ((2*param[2])/(2+(param[2]-2)*exp(-param[1]*t)))
loglik=-sum(-mu + y*log(mu))
}

llike = nlm(loglik,p=c(1,1), hessian=T)
llike
```

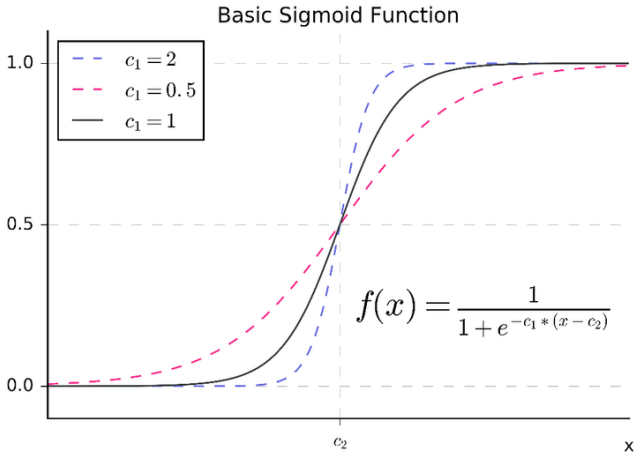
In order to fit the data, we assumed that A_0 was the first value of the vector of observations y, meaning $A_0 = 2$. By fitting the data, we calculated the values β and C, which are 0.3578941 and $11705,62$ respectively.

We also created a plot to illustrate the fitted model, using the following code:

```
param = llike$estimate
param

plot1 = function(t){
y = ((2*param[2])/(2+(param[2]-2)*exp(-param[1]*t)))
return(y)
}

plot(t, y)
points(t,plot1(t), type = 'l', col='orange')
```



We can appreciate in the graphic above that has a curious form, because the observed data, tends to infinity and the fitted values tednd to a value between 5000 and 15000.As stated in the statement, we are dealing with a sigmoidal function, so that explains the behaviour of the orange ploted line. So, we have done the same as the previous exercice but changing the μ function used, in this case we have used $A(t) = \frac{A_0 \cdot C}{A_0 + (C - A_0) \cdot e^{-\beta \cdot t}}$. We can see that predicts pretty well the y values. The sigmoid function has the property that tends to different values in the infinities $(+and-)$. We can see that our function distributed as a *poisson* keeps growing, but we know that a big quantity of *poissons* act such as a *normal* distribution. That is to say that the *poisson* keeps growing but in a certain point will start to decrease. If this has to happen, the orange plotted function can't go all the way up because otherwise it would get stuck, so it tends to a mean value and producing the Gibbs phenomenon.



- Which is the limit of A(t) when t tends to infinity?

$$A(t) = \frac{A_0 \cdot C}{A_0 + (C - A_0) \cdot e^{-\beta \cdot t}} \tag{3}$$

For this section of the exercise we used a function in R which returns the value of a function $f(t)$ when $t \to \infty$.

Nevertheless, it can be seen with the naked eye that if $t \to \infty$, $e^{-\beta \cdot t}$ *tends to zero. This causes the fraction to reduce to* $\frac{A_0 \cdot C}{A_0}$. *We can simplify this equation by eliminating* A_0 *which is found in both parts of the fraction(denominator and numerator). So we can finally deduce that* $\lim_{t \to \infty} A(t) = C$

The code is the following:

```
x = seq(1,34)
param = c(2,llike$estimate[1],llike$estimate[2])

lim = function(f,x=1,tol=0.0000000000001){
  next.diff=tol
  while(next.diff>=tol){
    next.diff = abs(f(x)-f(x+1))
    x = x + 1
  }
  return(list("Iterations"=x,"Limit"=f(x),"Next Value"=f(x+1)))
}

my.fun <- function(x){
  (param[1]*param[3])/(param[1]+(param[3]-param[1])*exp(-param[2]*x))}
lim(my.fun,1,1e-6)
```

With the code above, we see the following:

$$\lim_{t \to \infty} f = \lim_{t \to \infty} A(t) = \lim_{t \to \infty} \frac{A_0 \cdot C}{A_0 + (C - A_0) \cdot e^{-\beta \cdot t}} = C = 11705,62$$

- Estimate C giving also a 95% confidence interval and interpret the results.

  The code used to calculate de confidence interval for C is the following:

```
out = nlm(loglik, p=c(1,1), hessian = TRUE)
c = out$estimate[2] #C

hess = out$hessian
hess
# Confidence Intervals
conf.level = 0.95
crit = qnorm((1 + conf.level)/2)
inv.hess = solve(hess)
#param[1] + c(-1, 1) * crit * sqrt(inv.hess[1,1]) #estimar beta
c + c(-1, 1) * crit * sqrt(inv.hess[2,2]) #estimar C
```

  Which returns the following interval for C:

$$C \in [10388.87, 10532.30]$$

  Interpretation: The 95% confidence interval is a range of values that you can be 95% confident contains the true mean of the population $(11705, 62)$. Due to natural sampling variability, the sample mean (center of the CI) will vary from sample to sample.