MATH 534 HOMEWORK 6

Gustavo Esparza

3/24/2019

1.)

A

```
Time = c(0.25, 0.5, 0.75, 1, 1.5, 2, 3, 4, 6, 12, 24, 48)
Concentration = c(215.6, 189.2, 176, 162.8, 138.6, 121, 101.2, 88, 61.6, 22, 4.4, 0)

model = function(alpha_1, lambda_1, alpha_2, lambda_2, t){
  E_1 = exp(lambda_1*t)
  E_2 = exp(-lambda_2*t)
  f = alpha_1*E_1 + alpha_2*E_2
  grad = cbind(E_1, alpha_1*t*E_1, E_2, -alpha_2*t*E_2)
  attr(f,'gradient') = grad
  return(f)
}
```

```
Result = nls(Concentration ~ model(alpha_1, lambda_1, alpha_2, lambda_2, Time),
         start=list(alpha_1=100, lambda_1=0.05, alpha_2=100, lambda_2=0.05),trace = TRUE,
         nls.control(maxiter = 50, tol = 1e-5, minFactor = 1/1024, printEval=TRUE))
```

```
## 1467426 :  1e+02 5e-02 1e+02 5e-02
## 291337.6 :  117.70889622   0.02792691  78.34186855   0.36480855
## 14578.49 :   15.72534624   0.02504509 201.31255034   0.14968154
## 1555.431 :   19.699747699  -0.001236943 189.572837396   0.240197105
## 550.3467 :   25.09567158  -0.03041565 190.82812021   0.29130563
## 416.1091 :   56.00672585  -0.09272928 163.86795105   0.35424356
## 395.9537 :  133.3465269  -0.1686788  90.4777954   0.4934618
## 373.9764 :  156.4320010  -0.1784388  68.9979824   0.6033732
## 333.6786 :  166.3855802  -0.1812592  60.7703647   0.7203197
## 256.5963 :  173.6974436  -0.1810699  56.9454141   0.9306454
## 97.12377 :  171.743499  -0.173778  65.120558   1.207145
## 35.62475 :  162.5897018  -0.1618363  81.2745180   1.3302877
## 34.38149 :  162.5295380  -0.1617149  81.2564401   1.3032926
## 34.38027 :  162.6081936  -0.1618016  81.2367152   1.3063819
## 34.38026 :  162.5968351  -0.1617894  81.2418058   1.3060291
## 34.38026 :  162.5981307  -0.1617908  81.2412402   1.3060701
```

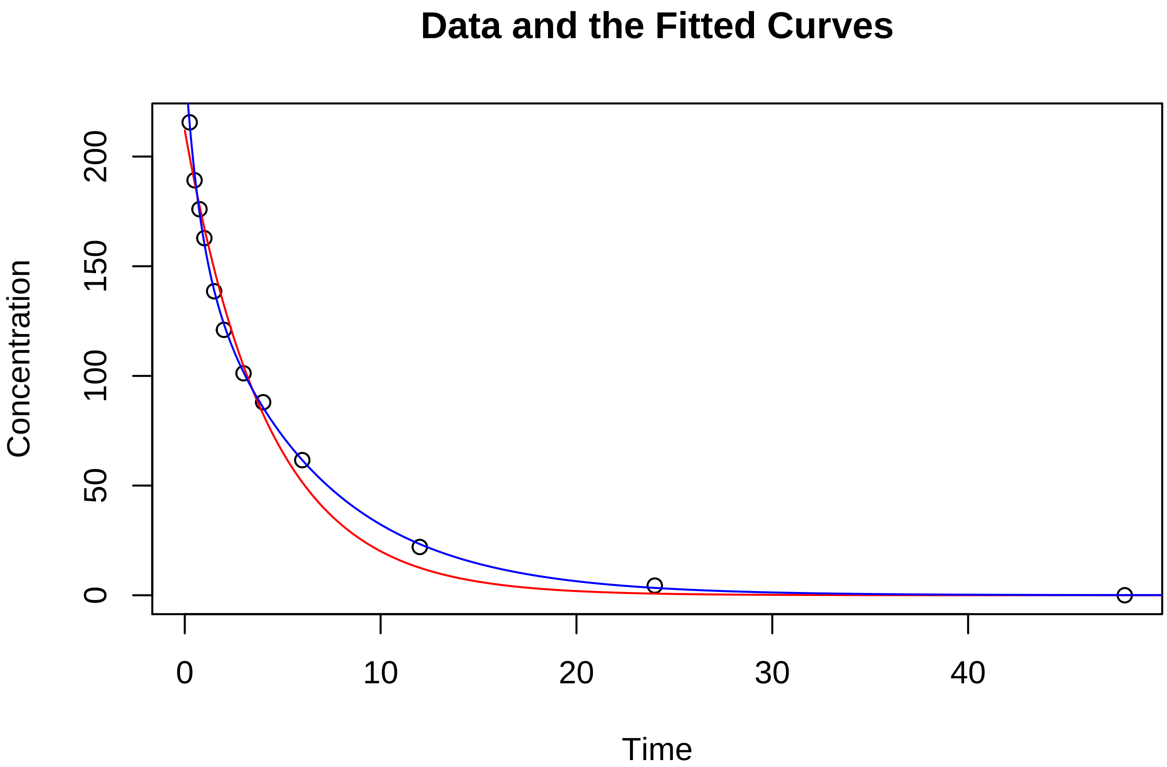
We can see that our estimates our α_1 = 162.59, λ_1 = -.1618, α_2 = 81.24, and λ_2 = 1.306.

Now, we will plot our one component (RED line) and two component (BLUE line) models against the data provided.

```
plot(Time, Concentration,main='Data and the Fitted Curves')
t = seq(0,50, 0.1)

lines(t,211.9203*exp(-0.2357*t),col='red')

lines(t, 162.6*exp(-0.1618*t) + 81.24*exp(-1.306*t), col = 'blue')
```



By comparing the two models, we can observe that the two component model fits the data better than our one component model.

2.)

A

```
dose=c(0,2,10,50,250)
animals=c(111,105,124,104,90)
tumor =c(4,4,11,13,60)

data=cbind(dose,animals,tumor)
data=as.matrix(data)
```

The IRLS algorithm is derived as follows:

```
# Start of the IRLS algorithm
X = data[,1:2]
y = as.matrix(data[,3])

JacfW_Binom = function(beta,X){
  x=X[,1]
  n=X[,2]
  p =(1-exp(-beta[1]-x*beta[2]-(x^2)*beta[3]))
  q=1-p
  f=n*p

  db0 = n*exp(-beta[1]-x*beta[2]-(x^2)*beta[3])
  db1 = n*x*exp(-beta[1]-x*beta[2]-(x^2)*beta[3])
  db2 = n*(x^2)*exp(-beta[1]-x*beta[2]-(x^2)*beta[3])

  J =cbind(db0,db1,db2)
  W = diag(1/(n*p*q))
  list(f=f, J=J, W=W) }
```

```
GN = function(y,X, beta0, Jac, Wt = 1, maxit,IRLS = TRUE){
cat("Iteration", "         b0", "            b1",
    "         b2", "           MRE",    "           Digits\n")

  for (it in 1: maxit){
    a = do.call(Jac, list(beta0, X))
    J = a$J
    f = a$f
    if (IRLS==TRUE){
      Wt = a$W }
    JW = t(J) %*% Wt
    dir = solve(JW %*% J) %*% JW %*% (y-f)
    beta1 = beta0 + dir

    relerr = norm(beta1-beta0)/max(1,norm(beta1))
    digits = -log10(norm(beta1-beta_star)/norm(beta_star))

cat(sprintf(' %2.0f              %6.6f     %6.6f     %6.6f     %1.1e          %1.1e\n'
                 ,it,beta0[1],beta0[2],beta0[3],relerr, digits))
    beta0=beta1
  }
  MLE_beta<<-beta0
  }
```

```r
maxit=20
beta0 = matrix(c(.1,.001,.00001),3,1)
#found beta star by outputting more digits.
beta_star = matrix(c(0.045945, 0.001627, 0.00001019))

mle = GN(y,X, beta0, 'JacfW_Binom', Wt = 1, maxit, IRLS = TRUE)
```

```
## Iteration          b0          b1          b2          MRE          Digits
##    1          0.100000    0.001000    0.000010    5.4e-02       1.7e+00
##    2          0.046765    0.001434    0.000011    7.1e-04       2.2e+00
##    3          0.046228    0.001605    0.000010    2.6e-04       3.0e+00
##    4          0.045987    0.001624    0.000010    3.9e-05       3.9e+00
##    5          0.045951    0.001627    0.000010    6.0e-06       4.9e+00
##    6          0.045946    0.001627    0.000010    9.3e-07       5.2e+00
##    7          0.045945    0.001627    0.000010    1.4e-07       5.1e+00
##    8          0.045945    0.001627    0.000010    2.2e-08       5.0e+00
##    9          0.045945    0.001627    0.000010    3.5e-09       5.0e+00
##   10          0.045945    0.001627    0.000010    5.4e-10       5.0e+00
##   11          0.045945    0.001627    0.000010    8.3e-11       5.0e+00
##   12          0.045945    0.001627    0.000010    1.3e-11       5.0e+00
##   13          0.045945    0.001627    0.000010    2.0e-12       5.0e+00
##   14          0.045945    0.001627    0.000010    3.1e-13       5.0e+00
##   15          0.045945    0.001627    0.000010    4.8e-14       5.0e+00
##   16          0.045945    0.001627    0.000010    7.4e-15       5.0e+00
##   17          0.045945    0.001627    0.000010    1.2e-15       5.0e+00
##   18          0.045945    0.001627    0.000010    1.9e-16       5.0e+00
##   19          0.045945    0.001627    0.000010    3.6e-17       5.0e+00
##   20          0.045945    0.001627    0.000010    7.2e-18       5.0e+00
```

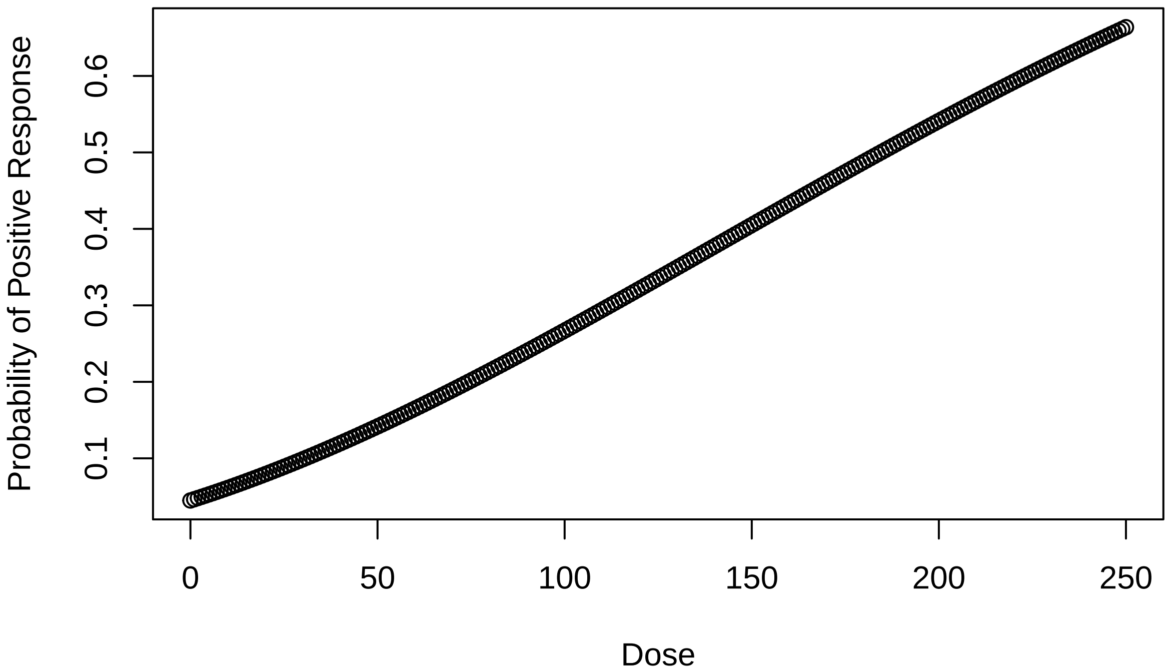
Thus, our MLE estimates are $\beta_0 = 0.0459$, $\beta_1 = 0.0016$, and $\beta_2 = 0.00001$. We can observe that the estimates are found before the 20 iterations are met.

B

Now, we will plot the probability of positive response, using our MLE estimates.

```r
x = seq(0,250, length=250)
y= 1 - exp(-MLE_beta[1] - MLE_beta[2]*x - MLE_beta[3]*x^2)

plot(x,y, xlab="Dose", ylab="Probability of Positive Response")
```



From the plot provided, we can note that as the dosage of DDT exposure is increased, the probability of positive response of an animal having a tumor also increases.

3.)

A

```
TreeData = read.table('/Users/gustavo/Desktop/Math 534/8/trees.txt', header=TRUE)

D=TreeData$D
S=TreeData$S

X = as.matrix(cbind(1, log(D), S))
y=TreeData$y
```

The IRLS algorithm is derived as follows:

```
JacfW_Bern = function(beta,X){

  xb = X%*%beta
  exb =  exp(xb)
  p = exb/(1+exb)
  q=1-p
  f=p
  frac = exb/((1+exb)^2)

  J= (diag(as.vector(frac))) %*% X
  W = diag(1/as.vector(p*q))
  list(f=f, J=J, W=W)}
```

```
maxit = 20
beta_0=matrix(c(0,0,0))
#found beta star by outputting more digits.
beta_star = matrix(c(-9.562084874971, 3.197563503535, 4.508593181753))

MLE_beta = GN(y, X, beta_0, 'JacfW_Bern', Wt = 1, maxit, IRLS = TRUE)
```

```
## Iteration          b0          b1          b2          MRE           Digits
##    1            0.000000     0.000000     0.000000     1.0e+00         4.4e-01
##    2            -6.124979     2.035718     2.857737     3.0e-01         1.0e+00
##    3            -8.673430     2.895867     4.076634     8.7e-02         2.1e+00
##    4            -9.492693     3.173968     4.474456     7.3e-03         4.3e+00
##    5            -9.561642     3.197413     4.508373     4.7e-05         8.7e+00
##    6            -9.562085     3.197563     4.508593     1.9e-09         1.3e+01
##    7            -9.562085     3.197564     4.508593     1.8e-16         1.3e+01
##    8            -9.562085     3.197564     4.508593     1.8e-16         1.3e+01
##    9            -9.562085     3.197564     4.508593     2.1e-16         1.3e+01
##   10            -9.562085     3.197564     4.508593     3.6e-16         1.3e+01
##   11            -9.562085     3.197564     4.508593     3.3e-16         1.3e+01
##   12            -9.562085     3.197564     4.508593     1.0e-16         1.3e+01
##   13            -9.562085     3.197564     4.508593     7.7e-17         1.3e+01
##   14            -9.562085     3.197564     4.508593     2.1e-16         1.3e+01
##   15            -9.562085     3.197564     4.508593     3.6e-16         1.3e+01
##   16            -9.562085     3.197564     4.508593     3.3e-16         1.3e+01
##   17            -9.562085     3.197564     4.508593     1.0e-16         1.3e+01
##   18            -9.562085     3.197564     4.508593     7.7e-17         1.3e+01
##   19            -9.562085     3.197564     4.508593     2.1e-16         1.3e+01
##   20            -9.562085     3.197564     4.508593     3.6e-16         1.3e+01
```

Thus, our MLE estimates are β_0 = -9.5621, β_1 = 3.1976 , β_2 = 4.508. We can again observe that our estimates are obtained in very few interations.

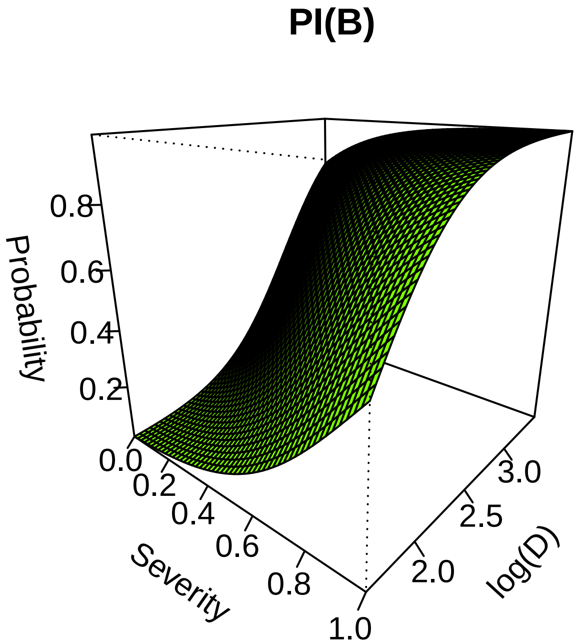
B

In order to plot our graph, we need to define two preliminary functions that compute our probability at a specific value and over a general interval.

```
#single value
pi_bern =function(s,logd){
  x= c(1,logd,s)
  xb = x%*%beta_star
  exb =  exp(xb)
  p = exb/(1+exb)
  as.numeric(p)}
#interval
pi_bern2 = function(x1,x2){
  n=length(x1)
  pi=numeric(n)
  for(i in 1:n){
    pi[i] = pi_bern(x1[i],x2[i])}
    pi}
```

Now, we will establish the interval at which we wish to plot our function.

```
x1 = seq(0,1, length=50)
x2 = seq(min(X[,2]), max(X[,2]), length=50)
x3 = outer(x1, x2, pi_bern2)
#plot
persp(x1,x2,x3, theta=40, phi=20, axes=TRUE, box=TRUE,
      ticktype="detailed" , nticks= 5, col="lawngreen",
      xlab="\n\nSeverity", ylab="\n\nlog(D)",
      zlab="\n\nProbability", main='PI(B)')
```



From the 3-D plot, we can observe a full perspective of how wind severity and tree diameter affect the probability of a tree being blown down.

C We will use our previously derived function for the prediction.

```
pi_bern(.3,log(10))
```

```
## [1] 0.3000951
```

Thus, a tree with a diameter of 10 located in an area with wind severity of 0.3 has a probability of 0.30 of being blown down.

4.)

A.) If the response distribution is normal, then our Likelihood function for β is as follows:

$$L(\beta) = (2\pi\sigma^2)^{\frac{-n}{2}} * exp\Big(\frac{-\sum_{i=1}^{n}(y_i - \mu_i)^2}{2\sigma^2}\Big)$$

Since our exponent is being raised to a negative value, we can note that our Likelihood function is a decreasing function. Thus, our MLE occurs when the varying terms (those affected by the index i) are minimized. That is, when the following is minimized:

$$\sum_{i=1}^{n}(y_i - \mu_i)^2 = \sum_{i=1}^{n}(y_i - X_i^T\beta)^2$$

Which is equivalent to the $L - 2$ norm.

B If the response distribution is double exponential, then our Likelihood function for β is as follows:

$$L(\beta) = (2\sigma)^{-n} * exp\Big(\frac{-\sum_{i=1}^{n}|y_i - \mu_i|}{2\sigma^2}\Big)$$

Again, since our exponent is being raised to a negative value, we can note that our Likelihood function is a decreasing function. Thus, our MLE occurs when the varying terms (those affected by the index i) are minimized. That is, when the following is minimized:

$$\sum_{i=1}^{n}|y_i - \mu_{i|} = \sum_{i=1}^{n}|y_i - X_i^T\beta|$$

Which is equivalent to the $L - 1$ norm.

C If the response distribution is uniform, then our density, f(y_i), can be expressed in the following manner:

$$\begin{aligned} f(y_i) &\propto (1_{(\mu_i - \sigma_i, \mu_i + \sigma_i)})^{(y_i)} \\ &\propto 1 \text{ if } \mu_i - \sigma_i < y_i < \mu_i + \sigma_i \text{ , 0 o.w.} \\ &\propto 1 \text{ if } |y_i - \mu_i| < \sigma_i \text{ , 0 o.w.} \end{aligned}$$

Thus, we can observe that the likelihood function of β is simply the product of the indicator function provided above over all y_i values. Then, the likelihood function is maximized when the likelihood function is equivalent to 1.This happens when all of our y_i values satisfy our indicator function, and more importantly, this can only occur when the maximum y_i value satisfies the following condition:

$$max_i|y_i - \mu_i| = max_i|y_i - X_i^T\beta| < \sigma_i$$

This is equivalent to minimizing the $L - \infty$ norm.