Michael Leibert

Math 504

Homework 2

2. Write a function called **MySqrt(a)** that computes the square root of a scalar a, i.e. $\sqrt{a}$. Solve this problem by finding the root of $f(x) = x^2 - a$. The function should return $\sqrt{a}$ and the number of iterations needed to reach the answer. For each method below, compute $\sqrt{1000}$ and show the number of iterations.

   (a) Implement **MySqrt**(a) using Newton's method.

```
MySqrt<-function(a){

    if (a == 0 ) { stop("Root is 0") }
    if (a < 0 ) { stop("Positive Numbers only") }

    fx<-function(x ){(x^2)-a}
    dfdx<-function(x){2*x}

    Xi<-a;
    iterations<-0

    repeat{
        xi<-Xi
        Xi<-Xi-(fx(Xi)/dfdx(Xi))
        iterations<-iterations+1
        if( abs(Xi-xi) < 10^-9 ) {break}
    }

    print(Xi);print(iterations)
    print(paste0("R sqrt function: ",sqrt(a))) }

MySqrt(1000)
```

```
## [1] 31.62278
## [1] 10
## [1] "R sqrt function: 31.6227766016838"
```

   (b) Implement **MySqrt**(a) using bisection method.

```
MySqrt<-function(a){

    if (a == 0 ) { stop("Root is 0") }
    if (a < 0 ) { stop("Positive Numbers only") }

    fx<-function(x ){(x^2)-a}

    XL<-0
    XR<-a+1
    iterations<-0
```

```
    repeat{
        XM<- (.5 * (XL + XR))
        ifelse(fx(XM) > 0,XR<-XM,XL<-XM)
        iterations<-iterations+1
        if( abs(XR-XL) < 10^-9 ) {break}
    }

    print(XM);print(iterations)
    print(paste0("R sqrt function: ",sqrt(a))) }

MySqrt(1000)
```

```
## [1] 31.62278
## [1] 40
## [1] "R sqrt function: 31.6227766016838"
```

(c) Implement **MySqrt**(a) using the R function **uniroot**.

```
MySqrt<-function(a){

    if (a == 0 ) { stop("Root is 0") }
    if (a < 0 ) { stop("Positive Numbers only") }

    fx<-function(x ){(x^2)-a}

    print(uniroot( fx, c(0,a) )$root)
    print(uniroot(  fx, c(0,a)  )$iter)
    print(paste0("R sqrt function: ",sqrt(a))) }

MySqrt(1000)
```

```
## [1] 31.62277
## [1] 15
## [1] "R sqrt function: 31.6227766016838"
```

3. Let $f(x, y) = 100(y - x^2)^2 + (1 - x)^2$. This is the famed banana function (see wikipedia). The minimum of f is $(1, 1)$ (why?).

Both terms will be nonnegative because they are positive number (100 and 1) multiplied by a square. The first term will be 0 when $x = y$ and the second term will be 0 when $x = 1$. So both terms will be 0 when $x = 1$ and $y = 1$.

(a) Starting at the point $(4, 4)$ use a fixed step size to locate the minimum. Here, since we are minimizing, your direction should be $d^{(i)} = -\nabla f(x^{(i)})$. How many iterations before you find the minimum?

```
fxy<-function(x,y){ 100*(y-x^2)^2 + (1-x)^2 }
dfdx<-function(x,y){ -400*x*y+400*x^3  + 2*x - 2 }
dfdy<-function(x,y){ 200*y - 200 * x^2  }
```

```r
x<-y<-4
i=1
repeat{
    x1<-x-dfdx(x,y)*.0003
    y1<-y-dfdy(x,y)*.0003
    x<-x1;y<-y1
    i=i+1
    if( (1-10^-4 < x & x < 1+10^-4) & (1-10^-4 < y & y < 1+10^-4) ){break}
}
x;y;i
```

```
## [1] 0.9999501
## [1] 0.9999
## [1] 83265
```

(b) Starting at the point $(4,4)$ use steepest descent with backtracking to "find" the minimum. How many iterations before you find the mininum?

```r
x<-y<-4;
i=1
repeat{
    di<-c(dfdx(x,y) ,dfdy(x,y))
    di<- (  di/(sqrt(sum(di^2)) ) )
    si<-1
    while( fxy(x,y) <  fxy(  x-si*di[1], y-si*di[2])  ) {
    si<-si/2 }
    x<-x-di[1]*si
    y<-y-di[2]*si
    i=i+1
    if( (1-10^-4 < x & x < 1+10^-4) & (1-10^-4 < y & y < 1+10^-4) ){break}
}

x;y;i
```

```
## [1] 1.00005
## [1] 1.0001
## [1] 17184
```

(b) Use the R function **nlm** to find the minimum starting at $(4,4)$. How many iterations? Compare to (a) and (b).

```r
fxy<-function(xy){
    x<-xy[1]
    y<-xy[2]
    100*(y-x^2)^2 + (1-x)^2 }

nlm(fxy,c(4,4))$estimate;nlm(fxy,c(4,4))$iterations
```

```
## [1] 0.999998 0.999996
## [1] 55
```

nlm is considerably more efficient than both (a) and (b). **nlm** took only 55 iterations compared with 17,183 with backtracking and 83,264 with a fixed step size.

4. Attached you will find the o-ring dataset discussed in class. Let y be the o-ring failure variable and x the temperature. Then logistic regression, as discussed in class, assumes the model

$$P(y=1|x,\alpha_0,\alpha_1)=\frac{1}{1+\exp(-\alpha_0-\alpha_1 x)}$$

(a) Let

$$L(\alpha)=\prod_{i=1}^{N}(y_i P(y=1|x_i,\alpha_0,\alpha_1)+(1-y_i)(1-P(y=1|x_i,\alpha_0,\alpha_1)))$$

Explain why this is the product of the probability of each datapoint given α. $L(\alpha)$ is the likelihood function.

$P(y=1|x,\alpha_0,\alpha_1)$ is the probability of failure given the temperature and α_0,α_1. Because y_i is binary, the ith observation records the probability of failure or the probability of success, $1-P(y=1|x,\alpha_0,\alpha_1)$. The likelihood estimates the parameters through the given data, and because the model assumes the observations are independent we take their product to generate the likelihood for α_0,α_1.

(b) Show that

$$\log L(\alpha)=\sum_{i=1}^{N}((1-y_i)(-\alpha_0-\alpha_1 x_i)-\log(1+\exp(-\alpha_0-\alpha_1 x_i)))$$

by considering $y_i=1$ and then $y_i=0$.

$$\begin{aligned}\log L(\alpha) &= \log\left[\prod_{i=1}^{N} y_i\ P(y=1|x_i,\alpha_0,\alpha_1)+(1-y_i)\,(1-P(y=1|x_i,\alpha_0,\alpha_1))\right]\\ &= \sum_{i=1}^{N}\ \log\left[y_i\ P(y=1|x_i,\alpha_0,\alpha_1)+(1-y_i)(1-P(y=1|x_i,\alpha_0,\alpha_1))\right]\\ &= \sum_{i=1}^{N}\ \log\left[y_i\ \frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}+(1-y_i)\left(1-\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right)\right]\end{aligned}$$

$y_i = 0$

$$\begin{aligned}
\log L(\alpha) &= \sum_{i=1}^{N} \log\left[(0)\,\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)} + (1-0)\left(1-\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right)\right] \\
&= \sum_{i=1}^{N} \log\left(1-\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right) \\
&= \sum_{i=1}^{N} \log\left(\frac{\exp(-\alpha_0-\alpha_1 x_i)}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right) \\
&= \sum_{i=1}^{N} \log\Big[\exp(-\alpha_0-\alpha_1 x_i)\Big] - \log\Big[1+\exp(-\alpha_0-\alpha_1 x_i)\Big] \\
&= \sum_{i=1}^{N} \big(-\alpha_0-\alpha_1 x_i\big) - \log\Big[1+\exp(-\alpha_0-\alpha_1 x_i)\Big]
\end{aligned}$$

$y_i = 1$

$$\begin{aligned}
\log L(\alpha) &= \sum_{i=1}^{N} \log\left[(1)\,\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)} + (1-1)\left(1-\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right)\right] \\
&= \sum_{i=1}^{N} \log\left(\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right) \\
&= \sum_{i=1}^{N} -\log\Big[1+\exp(-\alpha_0-\alpha_1 x_i)\Big]
\end{aligned}$$

The difference between $y_i = 0$ and $y_i = 1$ is the $\big(-\alpha_0-\alpha_1 x_i\big)$ term when $y_i = 0$. So we can capture both cases by writing:

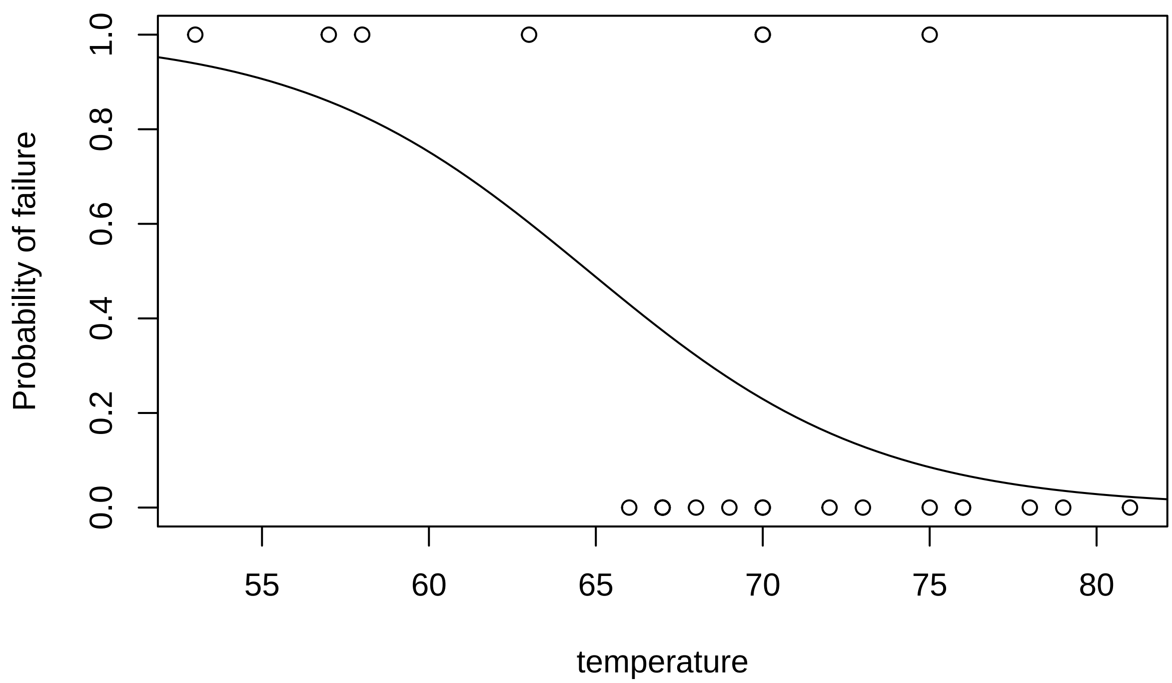
$$\begin{aligned}
\log L(\alpha) &= \sum_{i=1}^{N} \log\left[y_i\,\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)} + (1-y_i)\left(1-\frac{1}{1+\exp(-\alpha_0-\alpha_1 x_i)}\right)\right] \\
&= \sum_{i=1}^{N} (1-y_i)\big(-\alpha_0-\alpha_1 x_i\big) - \log\Big[1+\exp(-\alpha_0-\alpha_1 x_i)\Big]
\end{aligned}$$

(c) Use steepest ascent to find α that solves $\max_{\alpha\in\mathbb{R}^2} \log L(\alpha)$. Once you find α plot $P(y=1|x,\alpha_0,\alpha_1)$ for a range of x values and include the datapoints in the plot. Is the logistic curve a good fit?

```
#likelihood
l<-function(a0,a1,x,y){ sum(    (1-y)*(-a0-a1*x) - log(1+exp(-a0-a1*x)) )}

#Gradient components
dlda0<-function(a0,a1,x,y){  sum( ((1)/(exp(a0+a1*x)+1)) + y -1   )}

dlda1<-function(a0,a1,x,y){  sum( ((x)/(exp(a0+a1*x)+1))    + x * (y-1)  )}

#starting point
```

```
a0<-1;a1<-0

for( i in 1:5500000) {
    di<-c( dlda0(a0,a1,x,y) , dlda1(a0,a1,x,y) )
    di<- (  di/(sqrt(sum(di^2)) ) )
    si<-1
    while(  l(a0,a1,x,y) >  l( a0+di[1]*si ,    a1+di[1]*si  ,x,y) ){
    si<-si/2 }
    a0<-a0+di[1]*si
    a1<-a1+di[1]*si
}
```

```
OR<-read.table("g:/504/o_ring_data.txt",header=T)
x<-OR[,1];y<-OR[,2]
plot(x,y,xlab="temperature",ylab="Probability of failure")
curve( 1/(1+exp(-15.0429+0.2322*x)), from=50, to=85, add=TRUE)
```



Our simple model looks like a good fit. For lower teperatures it models the probability of failure high and as the teperature rises the probability decreases. It still provides a useful probability at intermediate levels of temperatures as well. However, there are some high temperature points (> 70) that are failures, so we may need more observations to construct a better model.

(d) Check your answers using R's **glm** function

```
glm(Failure~Temp, data=OR, family = "binomial")
```

```
##
## Call:  glm(formula = Failure ~ Temp, family = "binomial", data = OR)
##
## Coefficients:
## (Intercept)         Temp
##     15.0429      -0.2322
##
## Degrees of Freedom: 22 Total (i.e. Null);  21 Residual
## Null Deviance:       28.27
## Residual Deviance: 20.32     AIC: 24.32
```