A8 & A9

A8 R excercise. Suppose that the logistic model holds in which x is uniformly distributed between 0 and 100, and logit(π_i)=-2.0+0.04x_i

(a) Set the seed to 2, viz. set.seed(2) in R (find out what set.seed does if you don't know what this means) and randomly generate n = 100 independent observations from this model. This can be done as follows:

(i) Generate n independent variables from the (continuous) uniform distribution on (0, 100). Store these in a vector x.

```
# random number generator
set.seed(2)
x<-runif(100,0,100)
```

(ii)

```
p<-exp(-2+0.04*x)+1
p=1-1/p
p
```

```
##   [1] 0.2208928 0.6920021 0.5728051 0.2095248 0.8551226 0.8549420 0.1849198
##   [8] 0.7914677 0.4680621 0.5498179 0.5524801 0.2602979 0.7392460 0.2181091
##  [15] 0.4063992 0.8044266 0.8705233 0.2503591 0.4450323 0.1544545 0.6564683
##  [22] 0.3894079 0.7937299 0.1981346 0.3518514 0.4887751 0.1973385 0.3608332
##  [29] 0.8641947 0.1868648 0.1236466 0.2072748 0.7756978 0.8138835 0.5142779
##  [36] 0.6245216 0.7986239 0.2972312 0.6612574 0.1981145 0.8729072 0.3074736
##  [43] 0.1765842 0.2063291 0.8552230 0.7648499 0.8697502 0.3535099 0.5019699
##  [50] 0.7758405 0.1222210 0.1255134 0.6755978 0.8479846 0.2893804 0.7768562
##  [57] 0.7583245 0.8760569 0.6120201 0.6986217 0.7465151 0.8246155 0.6225738
##  [64] 0.2771185 0.8078799 0.4378117 0.3899741 0.4615770 0.2450297 0.1497889
##  [71] 0.2896271 0.3189773 0.1380837 0.2207491 0.2198558 0.7353327 0.2999027
##  [78] 0.8132668 0.4038548 0.5721773 0.3549320 0.6655209 0.1301313 0.4023709
##  [85] 0.2314585 0.8062929 0.8683058 0.3306802 0.7176361 0.3453082 0.8706841
##  [92] 0.3984484 0.3822511 0.5600957 0.4638713 0.2291896 0.4274589 0.1641183
##  [99] 0.1767151 0.4403176
```

(iii) For each i, draw a Bernoulli observation with probability π_i. Fit the logistic model with the intercept and x as predictors and report the parameter estimates.

```
library(Rlab)
```

```
## Rlab 2.15.1 attached.
```

```
##
## Attaching package: 'Rlab'
```

```
## The following objects are masked from 'package:stats':
##
##     dexp, dgamma, dweibull, pexp, pgamma, pweibull, qexp, qgamma,
##     qweibull, rexp, rgamma, rweibull
```

```
## The following object is masked from 'package:datasets':
##
##     precip
```

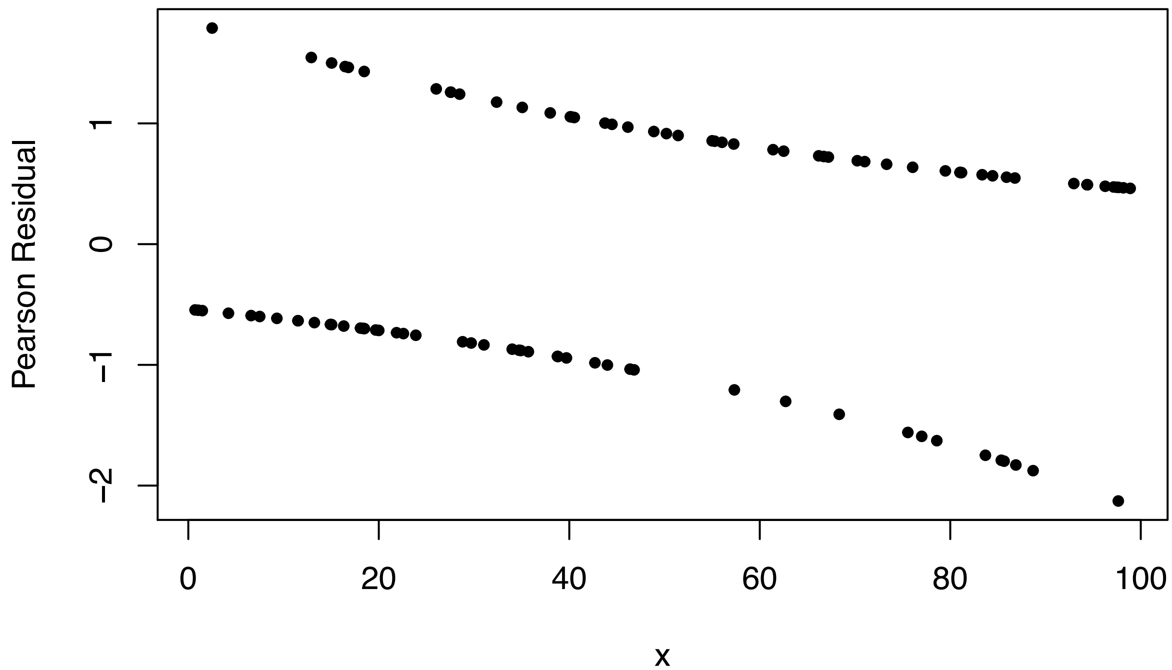
```
y<-0
for(i in 1:100){
  y[i]<-rbern(1,p[i])
}
f0<-glm(y~x,family=binomial(link=logit))
summary(f0)
```

```
##
## Call:
## glm(formula = y ~ x, family = binomial(link = logit))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8491  -1.0057   0.6324   1.0262   1.6945
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.234223   0.425755  -2.899 0.003745 **
## x            0.028105   0.007794   3.606 0.000311 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 138.27  on 99  degrees of freedom
## Residual deviance: 123.27  on 98  degrees of freedom
## AIC: 127.27
##
## Number of Fisher Scoring iterations: 4
```

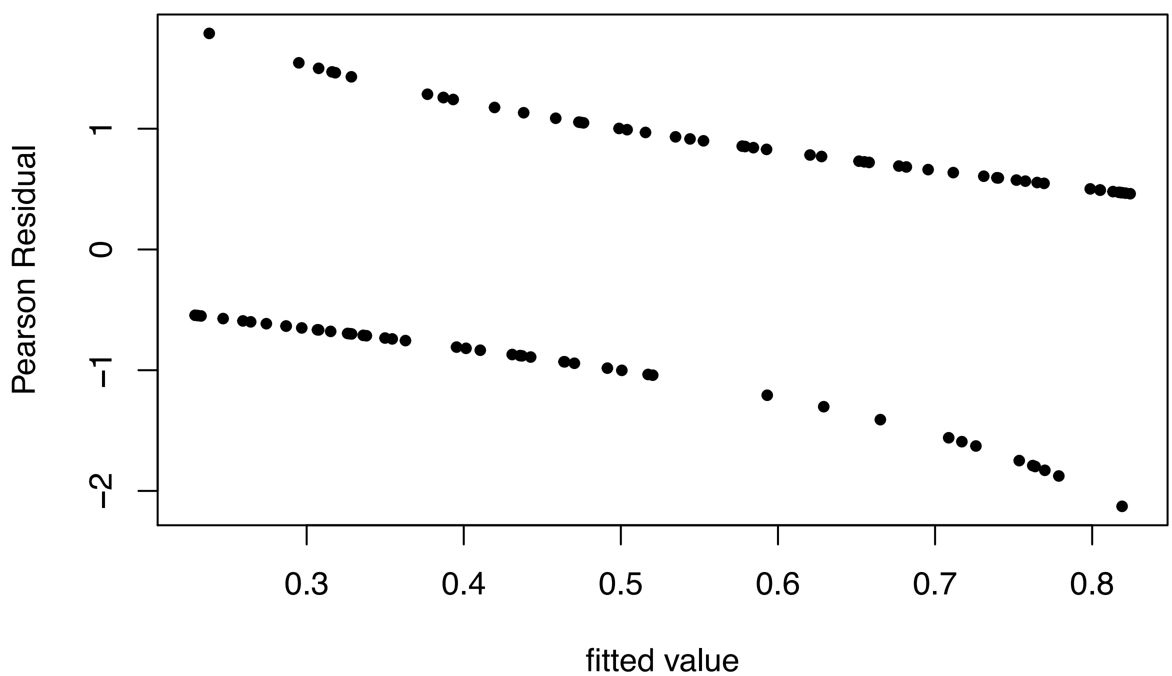
It shows that the estimated parameters is 0.028 and the intercept is -1.23

(b) For the sample generated in part (a), plot the Pearson residuals agains x and against the fitted values. Why do the residuals have this appearance? The points in the residual plots seem to lie on lines – find out what these lines are and display them on each residual plot.

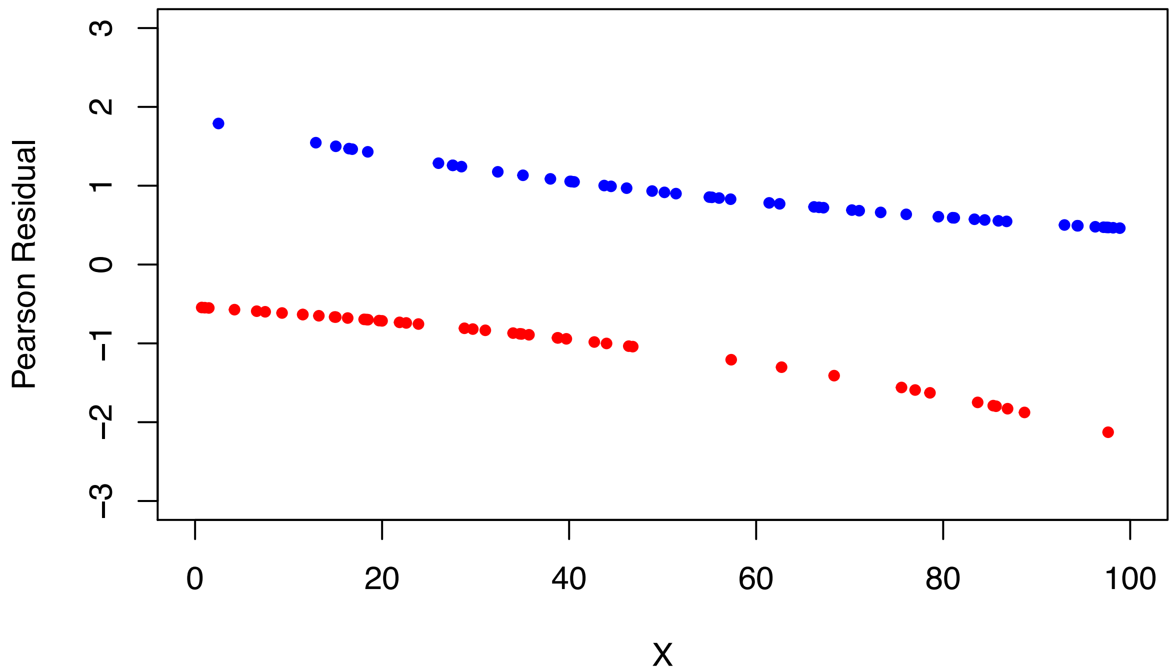
```
rp0<-residuals(f0,"pearson")
matplot(x,rp0,xlab="x",ylab="Pearson Residual",pch=20)
```




```
fitted<-f0$fitted.values
matplot(fitted,rp0,xlab="fitted value",ylab="Pearson Residual",pch=20)
```



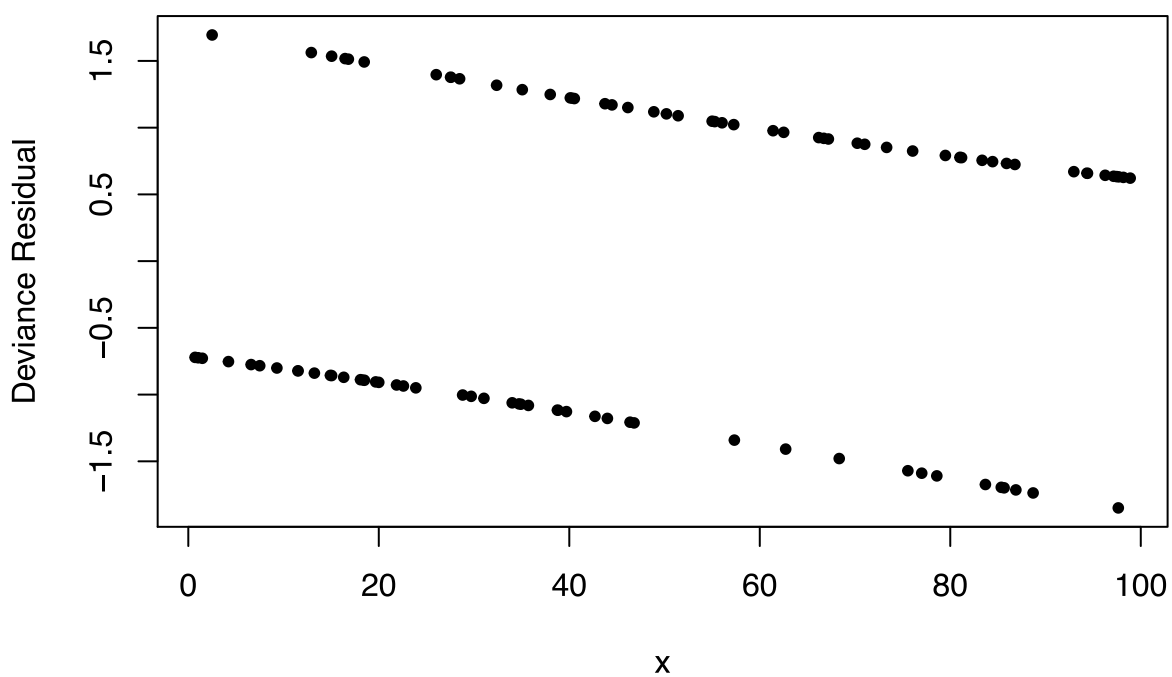

```
#investigate them
I1<-y==0
matplot(x[I1],rp0[I1],xlab="X",ylab="Pearson Residual",pch=20,col="red",xlim=c(0,100),ylim=c(-3,3))

I2<-y==1
points(x[I2],rp0[I2],xlab="X",ylab="Pearson Residual",pch=20,col="blue")
```



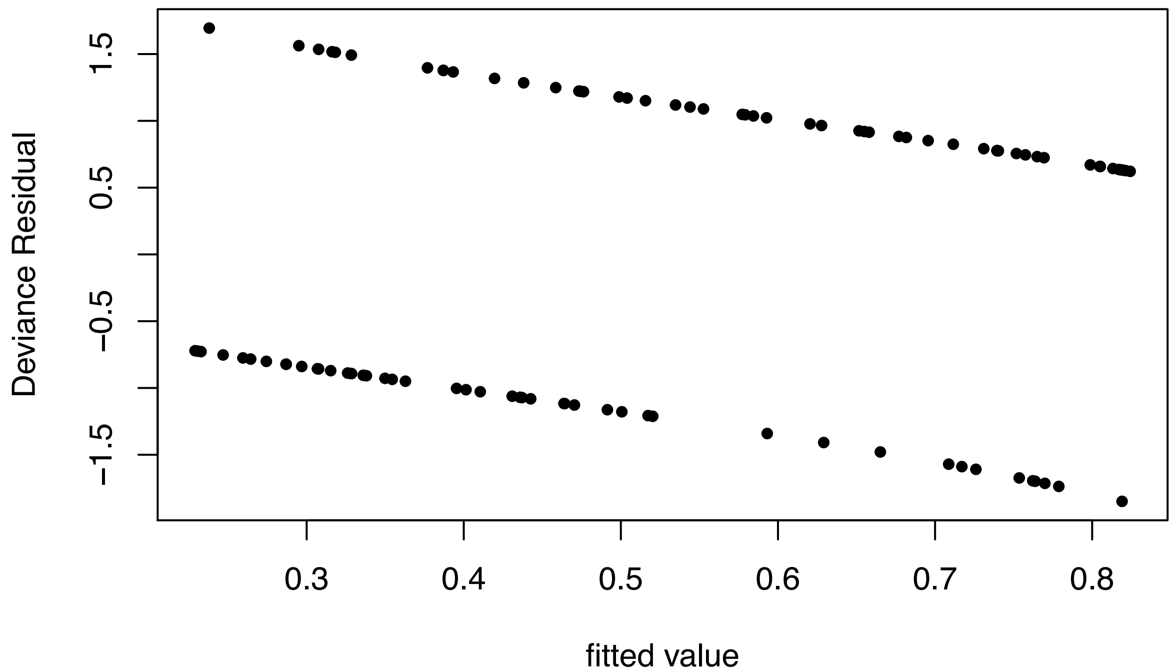
The above line showing that the data is really discrete and the above blue line representing the the case of y=1 and the below line show the case when y=0.

(c) Redo part (b) for deviance residuals.

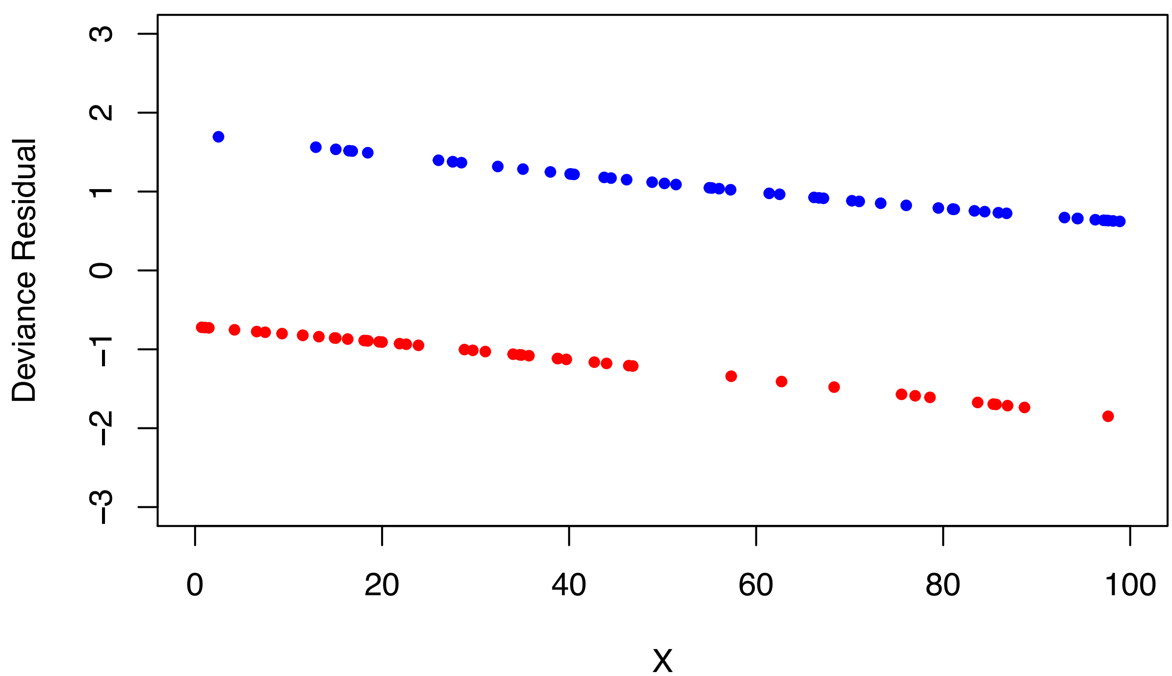
```r
rp1<-residuals(f0,"deviance")
matplot(x,rp1,xlab="x",ylab="Deviance Residual",pch=20)
```




```r
fitted<-f0$fitted.values
matplot(fitted,rp1,xlab="fitted value",ylab="Deviance Residual",pch=20)
```




```
#investigate them
I1<-y==0
matplot(x[I1],rp1[I1],xlab="X",ylab="Deviance Residual",pch=20,col="red",xlim=c(0,100),ylim=c(-3,3))
I2<-y==1
points(x[I2],rp1[I2],xlab="X",ylab="Deviance Residual",pch=20,col="blue")
```



The above line result of Deviance Residuals showing that the data is really discrete and the above line representing the the case of response when y=0 and y=1.