

Solutions:

b) we calculate

> (1.636 + 0.374 + 0.534 + 3.015 + 0.932 + 0.179)/6

[1] 1.111667

Solving for

> 1/1.111667

[1] 0.89955

Hence analytical MLE is

= 1/ = 0.8995

Using R we find the value of through numerical optimization

> # NUMERICAL MLE FORMULA

> nmle<- function(x) - sum(log(dexp(c(1.636, 0.374, 0.534, 3.015, 0.932, 0.179),x)))

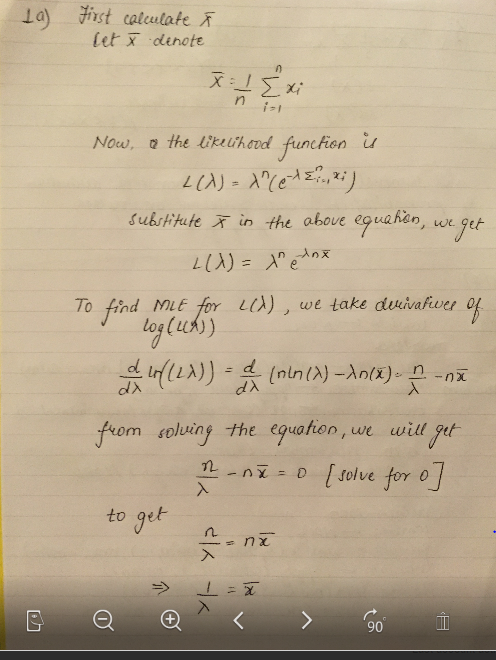
> nmle.results<-optim(1, nmle)

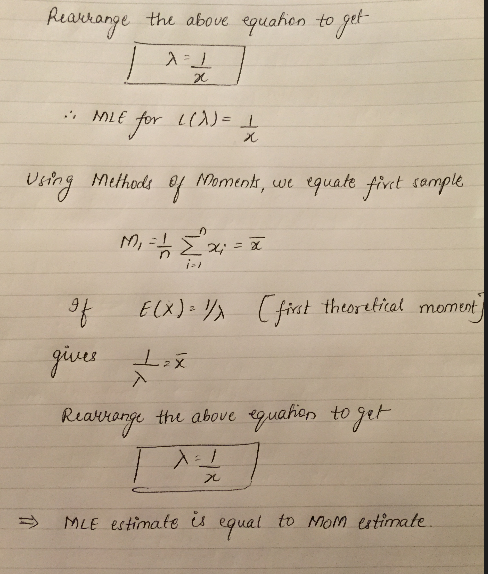
We get the output

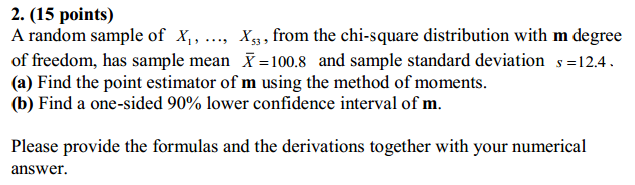
|  |
| --- |
| > print(nmle.results$par)  [1] 0.8996094 |
|  |
| |  | | --- | |  | |

Here the value of = 0.8996094

Thus by both ways the value of is 0.8995 (approx)







Solution:

a) To find point estimator m, equate the first sample moment about the origin

M1 =

First theoretical moment = E(X) = m = 100.8

Therefore our point estimator is 100.8

b) To find one sided 90% lower confidence interval

we have = 100.8

sample sd = s = 12.4, when = 0.1



( se(),∞)

100.8 + t.1,52

Below is the R script for 90% CI

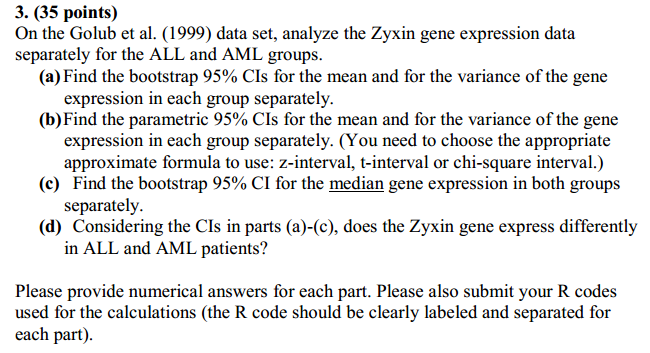
> # mean = 100.8, sd= 12.4

> LC<- 100.8 + (qt(0.1, df=53-1)\*(12.4/sqrt(53)))

> LC

[1] 98.58908

The lower 90% CI is ( 98.59 , ∞)



Solution:

1. The R script output for bootstrap 95% CI’s for mean and for varience of gene expression in each group is

For ALL

> print("Mean expression of Zyxin for ALL group")

[1] "Mean expression of Zyxin for ALL group"

> print(mean(ZALL))

[1] -0.2947926

> print("95% Bootstrap CI for ALL group Zyxin mean expression")

[1] "95% Bootstrap CI for ALL group Zyxin mean expression"

> print(CI.ALL.mean)

2.5% 97.5%

-0.55225881 -0.01809128

> print("variance expression of zyxin for ALL group")

[1] "variance expression of zyxin for ALL group"

> print(var(ZALL))

[1] 0.5224983

> print("95% Bootstrap CI for ALL group zyxin variance")

[1] "95% Bootstrap CI for ALL group zyxin variance"

> print(CI.ALL.var)

2.5% 97.5%

0.3419506 0.6431232

For AML

|  |
| --- |
| > print("Mean expression of Zyxin for AML group")  [1] "Mean expression of Zyxin for AML group"  > print(mean(ZAML))  [1] 1.586668  > print("95% Bootstrap CI for AML group Zyxin mean expression")  [1] "95% Bootstrap CI for AML group Zyxin mean expression"  > print(CI.AML.mean)  2.5% 97.5%  1.380538 1.790513  > print("variance expression of zyxin for AML group")  [1] "variance expression of zyxin for AML group"  > print(var(ZAML))  [1] 0.1351442  > print("95% Bootstrap CI for AML group zyxin variance")  [1] "95% Bootstrap CI for AML group zyxin variance"  > print(CI.AML.var)  2.5% 97.5%  0.04868182 0.20032541 |
|  |
| |  | | --- | | > | |

95% Bootstrap CI for ALL group Zyxin mean expression is ( -0.5522, -0.01809)

95% Bootstrap CI for ALL group Zyxin variance expression is ( 0.3419, 0.6431)

95% Bootstrap CI for AML group Zyxin mean expression is ( 1.38053 , 1.7905)

95% Bootstrap CI for AML group Zyxin varience expression is (0.0486, 0.2003)

b)

The parametric CI for the means were computed using t-interval



The parametric CI for the variance were computed using chi-square



> ci.mean.ALL<-mean(ZALL)+qt(c(0.025,0.975), df=nALL-1)\*sd(ZALL)/sqrt(nALL)

> print("95% CI's (t-interval) for the mean for All" )

[1] "95% CI's (t-interval) for the mean for All"

> print(ci.mean.ALL)

[1] -0.580738750 -0.008846435

> ci.var.ALL<-((nALL-1)\*var(ZALL))/qchisq(c(0.975,0.025), df=nALL-1)

> print("95% CI's (chi-square) for variance for ALL")

[1] "95% CI's (chi-square) for variance for ALL"

> print(ci.var.ALL)

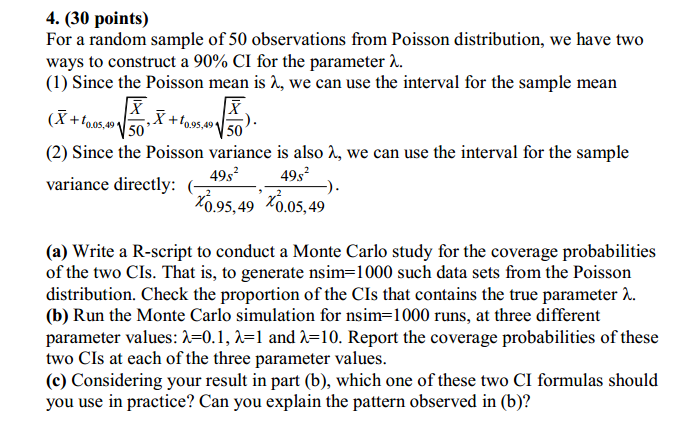
[1] 0.3240441 0.9812951

|  |
| --- |
| The parametric CI for the means computed using t-interval for ALL (-0.5807, -0.0088)  The parametric CI for the variance computed using chi-square for ALL (0.3240, 0.9812)    > print("95% CI's (t-interval) for the mean for AML" )  [1] "95% CI's (t-interval) for the mean for AML"  > print(ci.mean.AML)  [1] 1.339698 1.833638  > ci.var.AML<-((nAML-1)\*var(ZAML))/qchisq(c(0.975,0.025), df=nAML-1)  > print("95% CI's (chi-square) for variance for AML")  [1] "95% CI's (chi-square) for variance for AML"  > print(ci.var.AML)  [1] 0.06597815 0.41621602 |
| The parametric CI for the means computed using t-interval for AML (1.3396, 1.8336)  The parametric CI for the variance computed using chi-square for AML (0.0659, 0.4162)   1. The R script output is :   > print("Median expression of zyxin for ALL")  [1] "Median expression of zyxin for ALL"  > print(CI.ALL.median)  2.5% 97.5%  -0.73507 0.31432  > print("Median expression of zyxin for AML")  [1] "Median expression of zyxin for AML"  > print(CI.AML.median)  2.5% 97.5%  -1.36832 0.25025 |
| |  | | --- | |  | |

the bootstrap 95% CI for the median gene expression for ALL (-0.7350, 0.3142)

the bootstrap 95% CI for the median gene expression for AML (-1.3683, 0.2502)

d) observing the gene expression in ALL and AML patients we say that Zyxin gene expresses differently between AML and ALL patient



Solution

a) R script for 4a)

> # 4(a) Write a R-script to conduct a Monte Carlo study for the coverage probabilities

> # of the two CIs. That is, to generate nsim=1000 such data sets from the Poisson

> # distribution. Check the proportion of the CIs that contains the true parameter ??.

>

> # finding no of simulations and mean for formula 1

>

> #Number of simulations and generate dataset

> nsim <- 1000

> lambda <-

+

+ getdata <- matrix(rpois(50\*nsim,10),nrow=nsim)

> new.lambda <- (apply(getdata,1,mean))

> tdist <- qt(.05,49) \* sqrt(new.lambda/50)

>

> #90% CI for sample mean

> method1.low = new.lambda+tdist

> method1.High = new.lambda-tdist

>

> #Check coverage probabilities

> sum(method1.low<lambda & lambda<method1.High)/1000

[1] 9.38

>

> # finding no of simulations and mean for formula 2

>

> #Number of simulations and generate dataset

> nsim <- 1000

> lambda <-

+ getdata <- matrix(rpois(50\*nsim,10),nrow=nsim)

> new.lambda <- (apply(getdata,1,mean))

>

> #90% CI for sample mean

> method2.low = 49 \*(new.lambda)/qchisq(.95,49)

> method2.High = 49 \*(new.lambda)/qchisq(.05,49)

>

> #Check coverage probabilities

> coverage<-sum(method2.low<lambda & lambda<method2.High)/1000

> coverage

[1] 35.292

Solution 4b)

> print("For nsim = 1000 & lambda = 0.1 coverage prabability of lambda using method 1 is")

[1] "For nsim = 1000 & lambda = 0.1 coverage prabability of lambda using method 1 is"

> print (coverage1)

[1] 0.853

> print("For nsim = 1000 & lambda = 0.1 coverage prabability of lambda using method 2 is")

[1] "For nsim = 1000 & lambda = 0.1 coverage prabability of lambda using method 2 is"

> print (coverage2)

[1] 0.49

> print("For nsim = 1000 & lambda = 1 coverage prabability of lambda using method 1 is")

[1] "For nsim = 1000 & lambda = 1 coverage prabability of lambda using method 1 is"

> print (coverage3)

[1] 0.903

> print("For nsim = 1000 & lambda = 1 coverage prabability of lambda using method 2 is")

[1] "For nsim = 1000 & lambda = 1 coverage prabability of lambda using method 2 is"

> print (coverage4)

[1] 0.972

> print("For nsim = 1000 & lambda = 10 cverage prabability of lambda using method 1 is")

[1] "For nsim = 1000 & lambda = 10 cverage prabability of lambda using method 1 is"

> print (coverage5)

[1] 0

> print("For nsim = 1000 & lambda = 10 cverage prabability of lambda using method 2 is")

[1] "For nsim = 1000 & lambda = 10 cverage prabability of lambda using method 2 is"

> print (coverage6)

[1] 1

1. Observing the pattern we shoub follow formula 2 as its more stable than formula 1