STAT471_Project

Group7:Yuzhuo Kang

4/25/2020

a. Abstract:

This project is the application and visualization of bootstraps with simulation. Bootstraps method belongs to nonparametric Monte Carlo methods, which is used to estimate the distribution of the population data by simulating and resampling. This project will use four different data sets and two bootstrap methods to visualize the bootstrap simulation by Shinyapp. For the bootstrap on discrete, continuous, and data with unknown distribution, when the simulation time increases, the bootstrap method gives a more precise estimation on the population data.For the residual bootstrap, when the simulation time increases, the distribution of the impact of all the predictors is close to the normal distribution.

b. Introduction:

This project will conduct a bootstrap analysis of different kinds of data. In order to show proximity and accuracy of estimation, this project will adopt visualization methods such as histogram and density plots via Shinyapp to show relevant statistical parameters of bootstrap samples. The motivation is that after learning bootstrap in class, our group finds that it might be a little bit difficult for someone learning bootstrap to understand it with only theoretical formulas and numerical results truly. Based on this consideration, this project aims to show applications and visualization results on the real data for the learners of simulation studies to help audiences better understand its properties and strength.

This program's objective is to provide an intuitive representation of bootstrap results. Various types of visualizations, such as Shinyapp, are adapted in order to show the relationship between the bootstrap results and the results from actual data directly to audiences. The goal is to show whether the estimation results from bootstrap results have proximity to the actual value when the data have different distributions and show how the distribution of estimator changes when the total simulation time changes or the asample

This project includes diverse types of diagrams to show how the bootstraps sampling method creates a confidence interval about the population from samples. We expect to show that bootstrap is a reliable way to approximate the population by showing the bootstrap estimated confidence intervals. The expected outcome is that the estimation results from bootstrap will close to the real value when the sample size and simulation time increases.

For the methods, this project will use nonparametric bootstrap methods to treat subgroups of population data as finite observed samples. The selected sample will be made inference to estimate the characteristics, distribution, and other parameters of the whole population. The type of estimators include mean and variance. This project uses resampling methods to compare the mean, variance, and confidence intervals of the data with the true theoretical value. It will also use a comprehensive calculation to show the standard error and estimated bias. To make the bootstrap analysis more comprehensive and conclusive, the data type of populations includes both discrete variables, continuous variables, variables with unknown distribution. The goal is to see whether these three types of variables show similar results. This project will also conduct

residual bootstrap using life expectancy data. The goal is to compare the estimated impacts of three factors using residual bootstrap and a linear regression model. Using the ShinyApp, it shows how the distribution of predictor varies when the simulation time increases. We hypothesis that the bootstrap with more simulation is better. The estimator is more accurate.

c. Statistical Analysis

Bootstrap on data following poisson distribution

This part tends to use Bootrap Method to estimate parameters for discrete population, and for this example, a 2-dimension Poisson distribution.

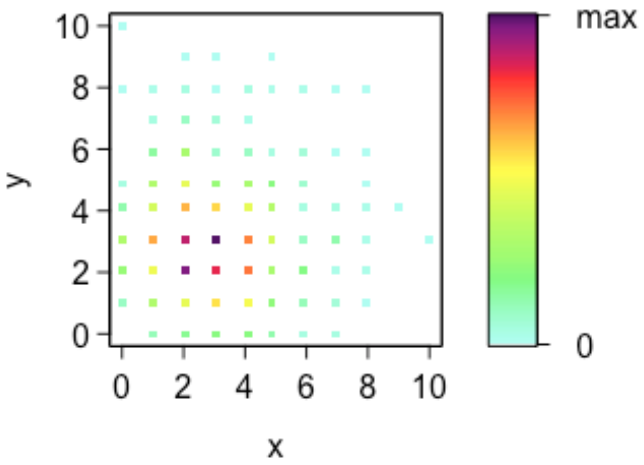
The data is from a 2-dimension Poisson distribution simulated by code. And then we can use bootstrap method to estimate its mean and variance. By theoretical calculation, we know: $p(k;\lambda)=e^{-\lambda}\frac{\lambda^k}{ k!}$

$EX=\sum_{n=0}^\infty nP(X=n)=e^{-\lambda}\sum_{n=0}^\infty\frac{n\lambda^n}{ n!}=e^{-\lambda}\sum_{n=0}^\infty\frac{\lambda^n}{ (n-1)!}=\lambda e^{-\lambda}\sum_{n=0}^\infty\frac{\lambda^n}{ n!}=\lambda$

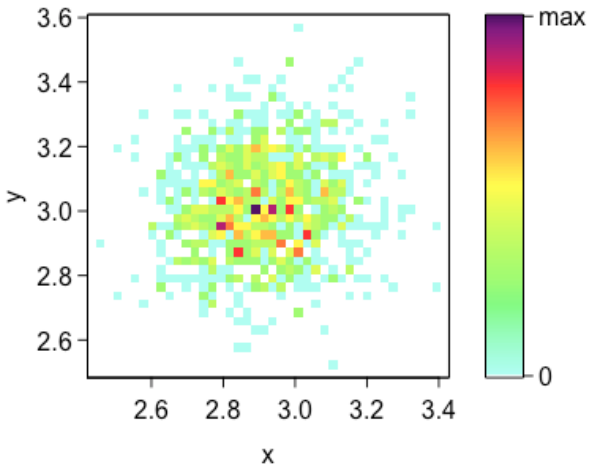
$EX^2=\sum_{n=0}^\infty n^2P(X=n)=\sum_{n=1}^\infty\frac{n^2e^{-\lambda}\lambda^n}{ n!}=e^{-\lambda}\sum_{n=2}^\infty\frac{n(n-1)\lambda^n}{ n!}+e^{-\lambda}\sum_{n=1}^\infty\frac{n\lambda^n}{ n!}=\lambda^2 e^{-\lambda}\sum_{n=2}^\infty\frac{\lambda^{n-2}}{ (n-2)!}+\lambda=\lambda^2+\lambda$

$Var(X)=EX^2-(EX)^2=\lambda^2+\lambda-\lambda^2=\lambda$

So if bootstrap method can give a right estimation, mean and variance should all be approximately λ.



From this plot we can see the most density dot is (3,3). However, since data from Poisson distribution are all integers, this plot does not reflect the random situation very well.

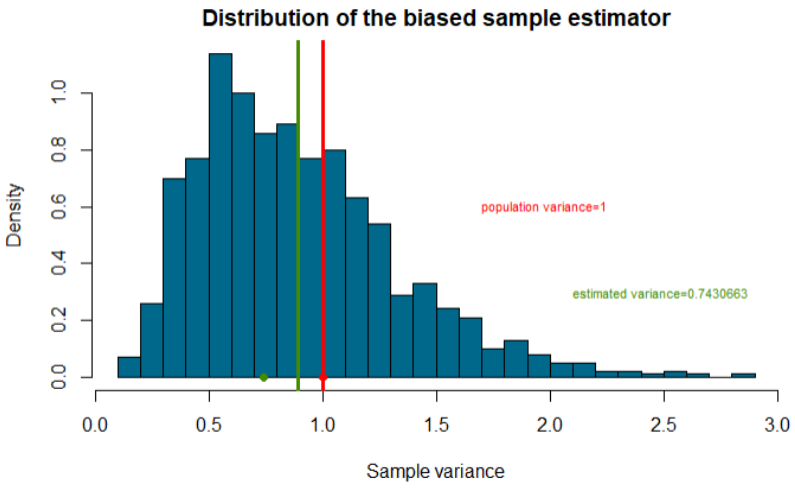


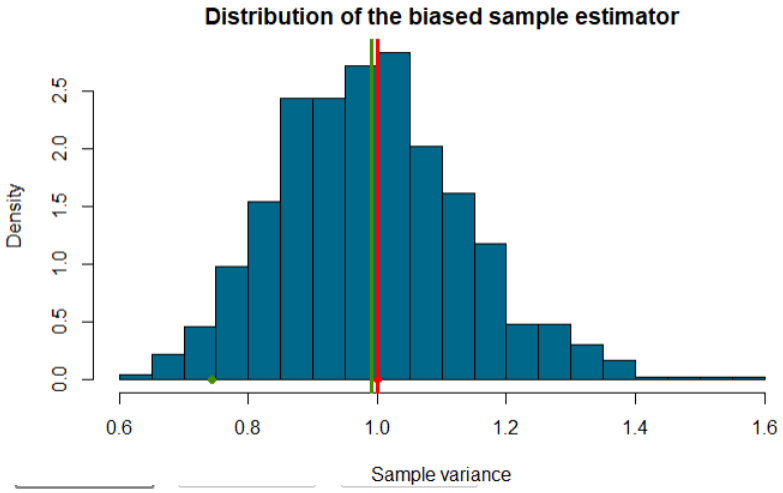
According to the graph, we can see the data assemble around 3. According to all the 95% confidence intervals, we can see 3 right in them. So Bootstrap Method gives a quite precise estimation of corresponding statistics of Poisson distribution.

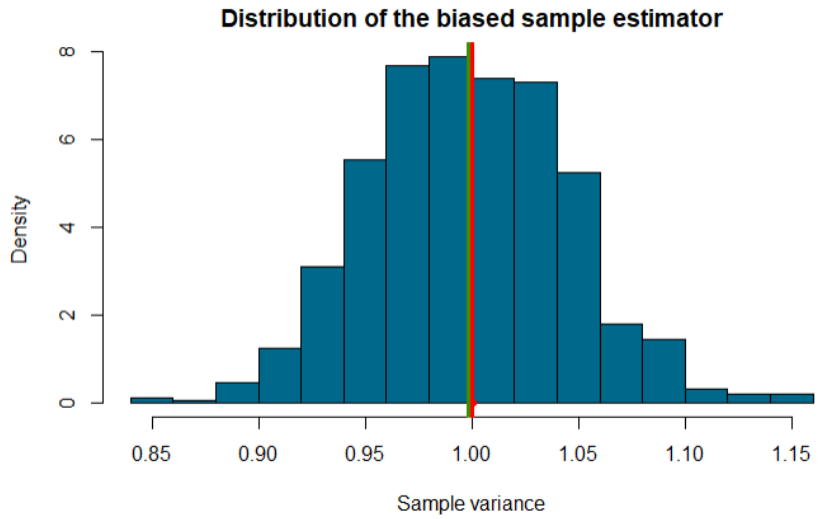
Bootstrap on data following continuous distribution

This part use the normal distribution *We first evaluate how sample size n affects the boostrap method* We simulate a random sample of size of size n=10,100,1000 from a standard normal distribution: μ=0, σ^2=1 when the simulation time=1000.

The following plots are for n=10, 100, 1000



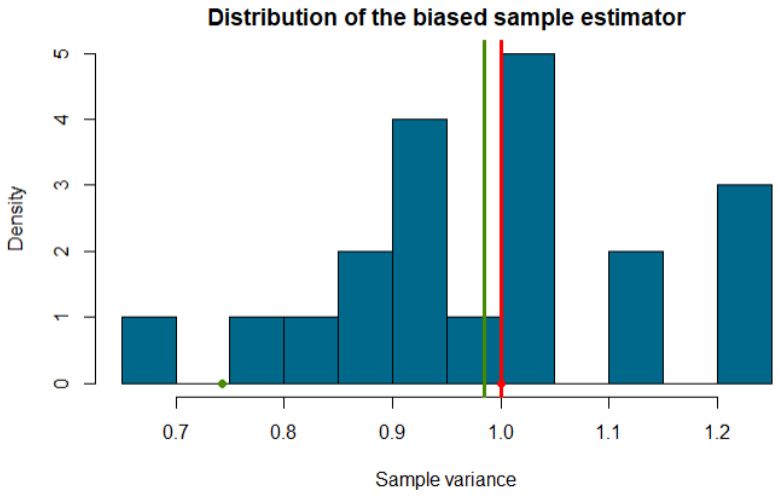


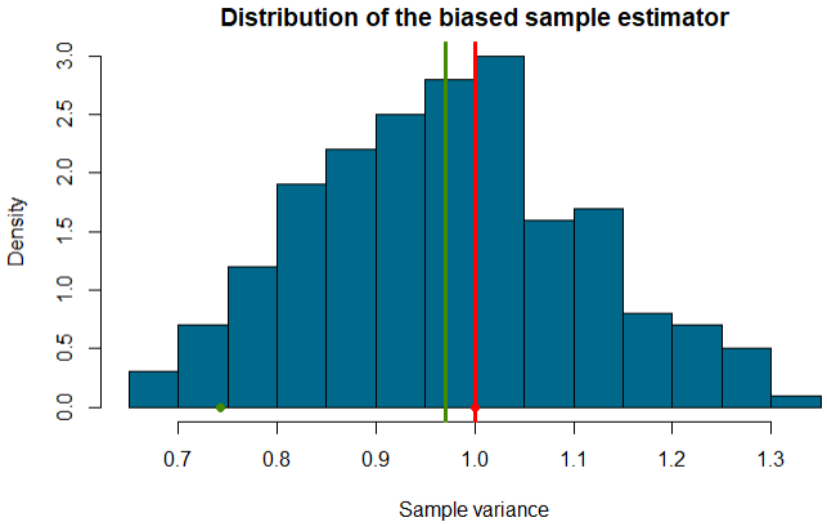


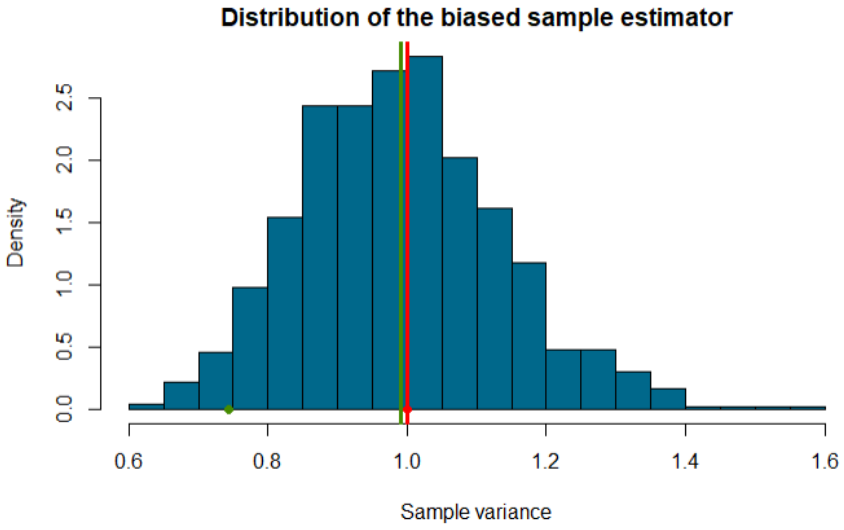
We observe that, as the sampling size increase: 1.the estimated variance becomes closer and closer to the population variance 2.the distribution histogram changeas from a right-skewed one to a bell-shaped one. Note the reason it is right-skewed is because our data. estimated sample variance, can't be smaller than 0 3.the length all types of confidence interval shrink; the changes is dramatic at first, then tends to stablize

We then evaluate how simulation times affect the bootsrap method* We simulate a random sample of size n=100 from a standard normal distribution: μ=0, σ^2=1 simulation times=20,200,1000

The following plots are for simulation times=20,200,1000







We observe that, as the bootstrap simualtion time increase: The distribution histogram tends to become more normally distributed. Therefore, this verifies what we studied in class, and we suggect using large boostrp times

Bootstrap on faithful data (unknown distribution)

This part tends to find out the relationship between simulation times and the corresponding bootstrap confidence interval.

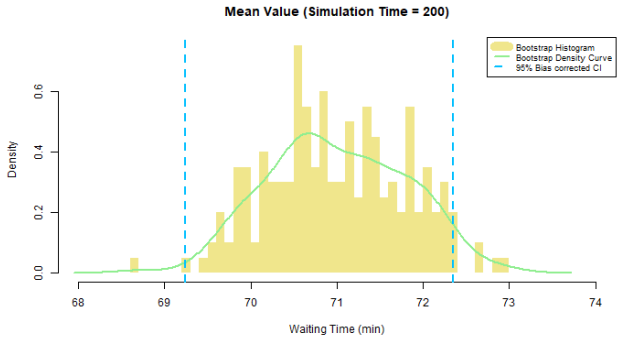
The data we use is the ‘Old Faithful Geyser Data’ which is stored in R already. The data,with 272 observations, tells the waiting time between eruptions for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.

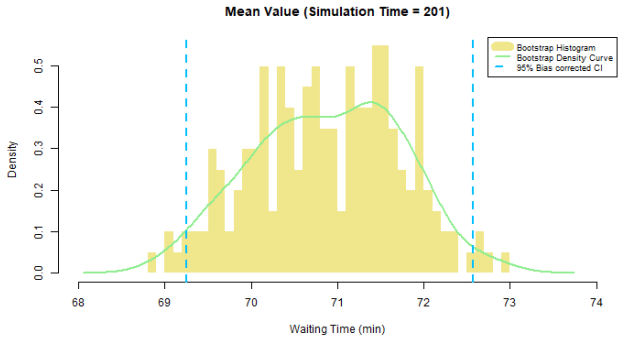
The following is the example focusing on the mean and standard variance value of the bootstraped data.

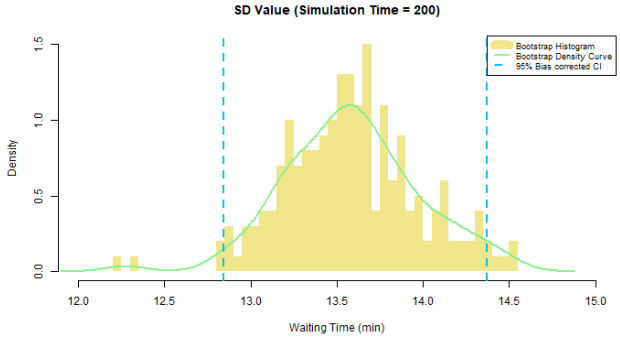
We made a little application to visualize how the statistics(mean and standard variance) change when simulation time changes.

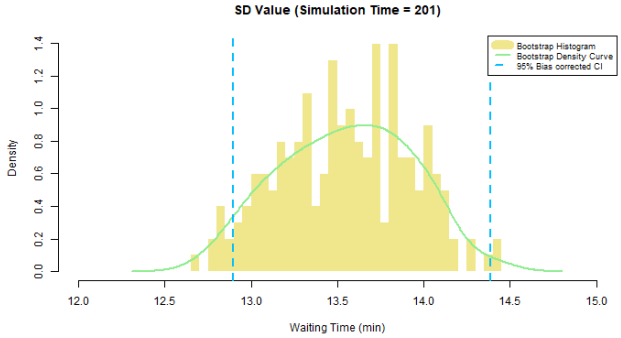
After trying different simulation times, we came to the conclusion that when simulation times are small, the confidence intervals are unstable – they change dramaticly when simulation time grows. On the contray, the confidence intervals become stable when simulation times are large.

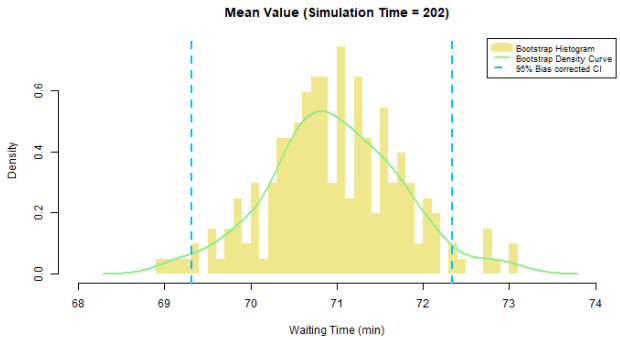
For example, when simulation times changes from 200 to 202, we have:

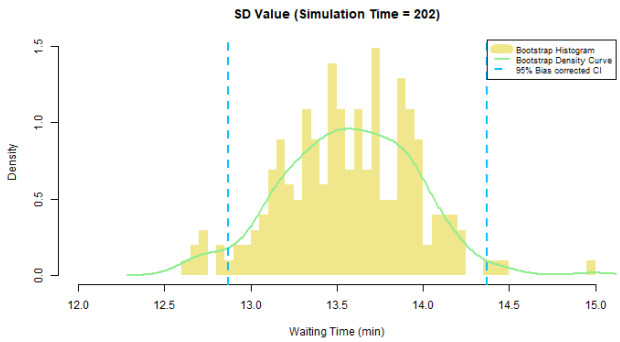




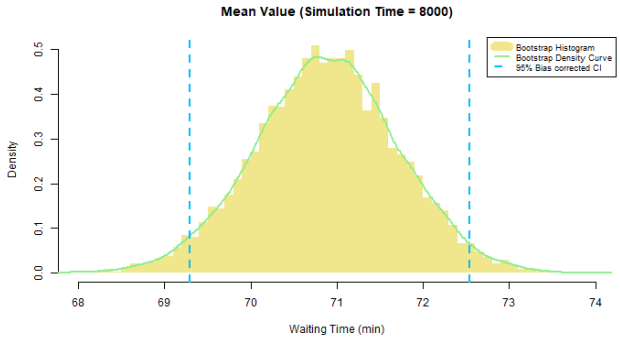


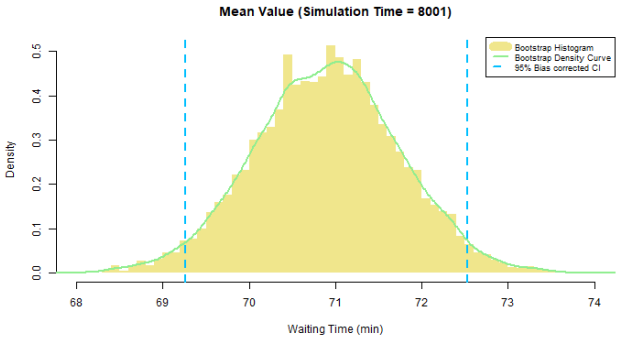


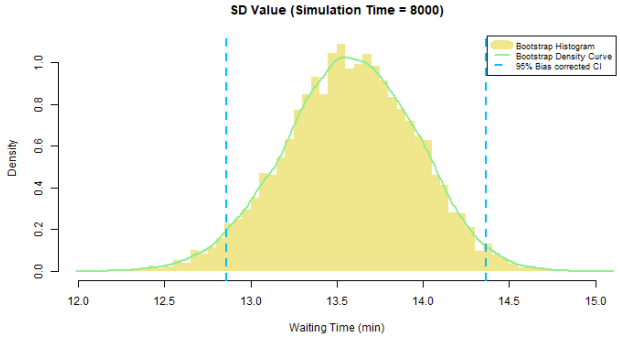


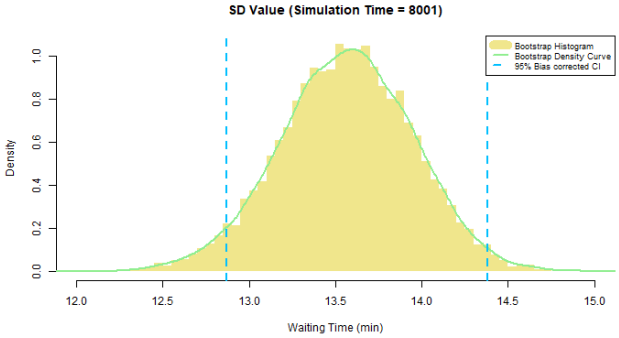


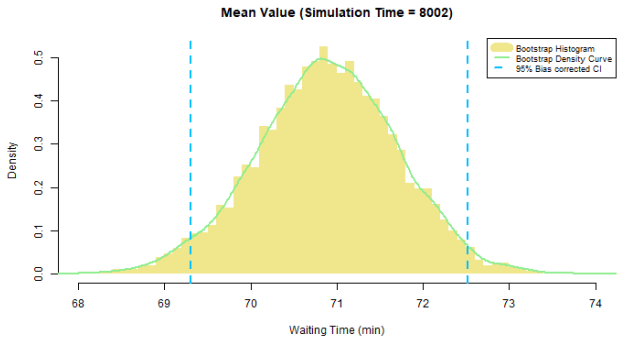
When simulation times changes from 8000 to 8002, we have:

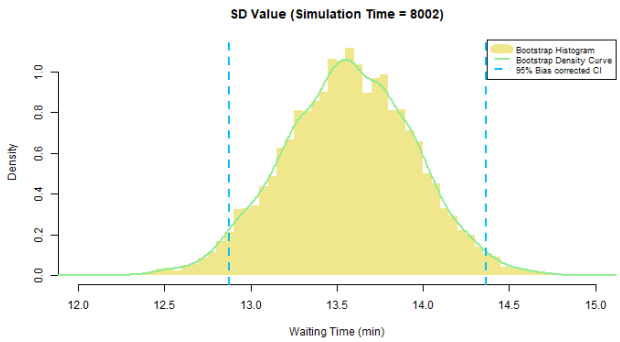












Hence, if we want to get a more precise result, we may better choose simulation times as much as possible, as shown in the shiny app

Residual bootstrap on life expectancy data

This part shows the visualization of the residual bootstrap.The aim of this part of the project is to do the data analysis with residual bootstrap on the life expentacy data from WHO.

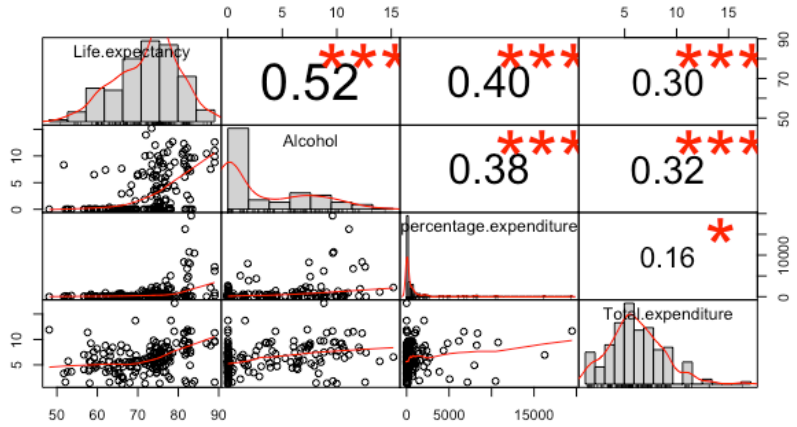
The dataset is the Life Expectancy Data, which is collected from WHO and United Nations website. The data contains the health status data for all the countires in 2014. The selected factors include resident health expeditures, government total health expenditure, and alcohol consumption. The response variable is the average life expectancy in age. In the data, the health expediture measures the expenditure on health as a percentage of Gross Domestic Product per capita(%). The alcohol consumption records per capita consumption (in litres of pure alcohol) for individuals who are older than 15. The government health expenditure measures the general government expenditure on health as a percentage of total government expenditure (%).

The model is defined as follows:

\(Life=\beta_0+\beta_1Achohol+\beta_2Health+\beta_3Government+\epsilon\) The Achohol is the alcohol consumption. Health represents the proportion of total health expenditure as total GDP. The government is the total proportion of government health expenditure. \(\beta\) is estimated impacts of three factors. The error term is assumed to be normally distributed with mean 0.

First conduct a classic analysis on this MLR model. The average value of life expectancy,alcohol consumption,health expenditure, total government health expenditure is 71.71, 3.308,1018.36,6.22. From the following correlation matrix plot, it shows that the the life expectancy and alcohol consumption, health expenditure by GDP, and total government health expenditure are highly correlated. This plot shows the

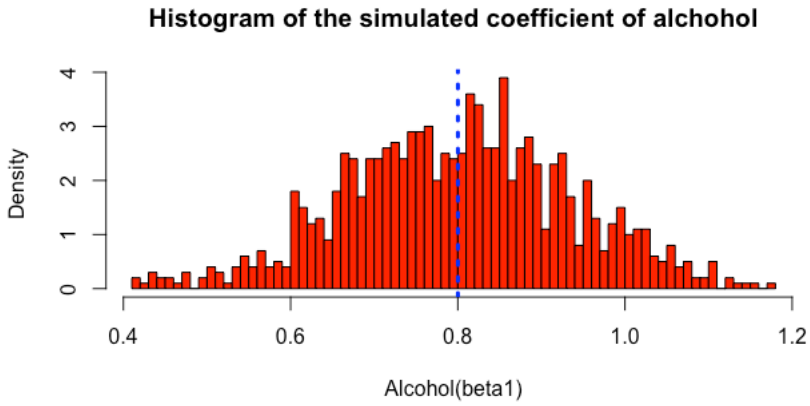
density plot of the four variables. It shows that the life expectancy and total government health expenditure is close to normal distribution, and the distribution of alcohol consumption and health expenditure are highly skewed.

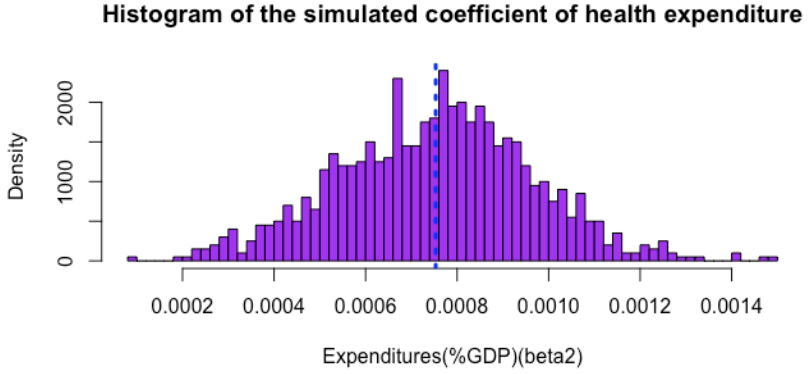


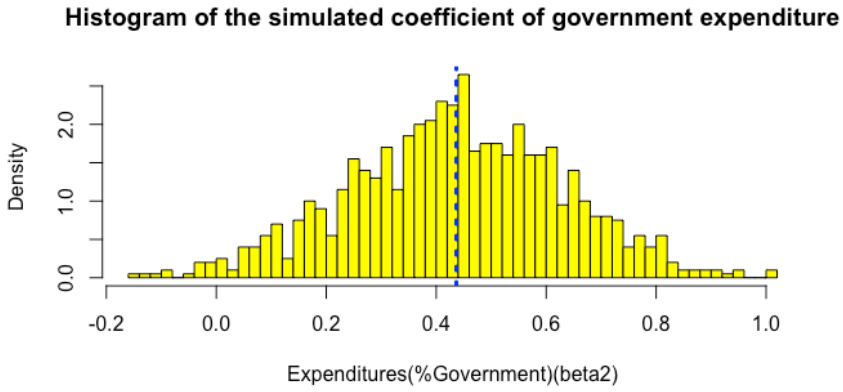
The linear regression model shows that the R squared value is 0.3398. The estimated impacts of alcohol consumption, health expenditure by GDP, and total government health expenditure are 0.7965, 0.000754, and 0.0436. Alcohol consumption, total health expenditure as a percentage as GDP, government health expenditures have significant impacts of the life expectancy. Use the estimated coefficients and residual standard error from the regression model to simulate at the observed value of each covariate. The residual standard error is 6.932. The confidence intervals for the three factors using linear regression model are (0.52,1.07),(0.00032,0.0012),(0.042,0.83).

Next step is to simulate on the life expectancy data. In the simulation matrix, for each column, the response vector of the life expectancy is simulated. It contains the predicted values. The total number of column is same as the total number of simulations, 1000. The total row number is the total observation of the life expectancy data, 180. In addtion, a matrix of simulated residuals is also generated by the random variables \(\epsilon\), which is assumed to be normally distributed with mean 0 and variance 6.932. The simulation on the coefficient of three factors is shown in the following graphs. The simulated coefficients of alcohol, health expenditure by GDP, and total government health expenditure are 0.80014, 0.0007531, 0.4365329, which are close to the estimated value in the regression model. Also, the figure shows that the distribution of the simulated coefficients are close to normal distribution.

The plots of the simulation on the impacts of these three factors:







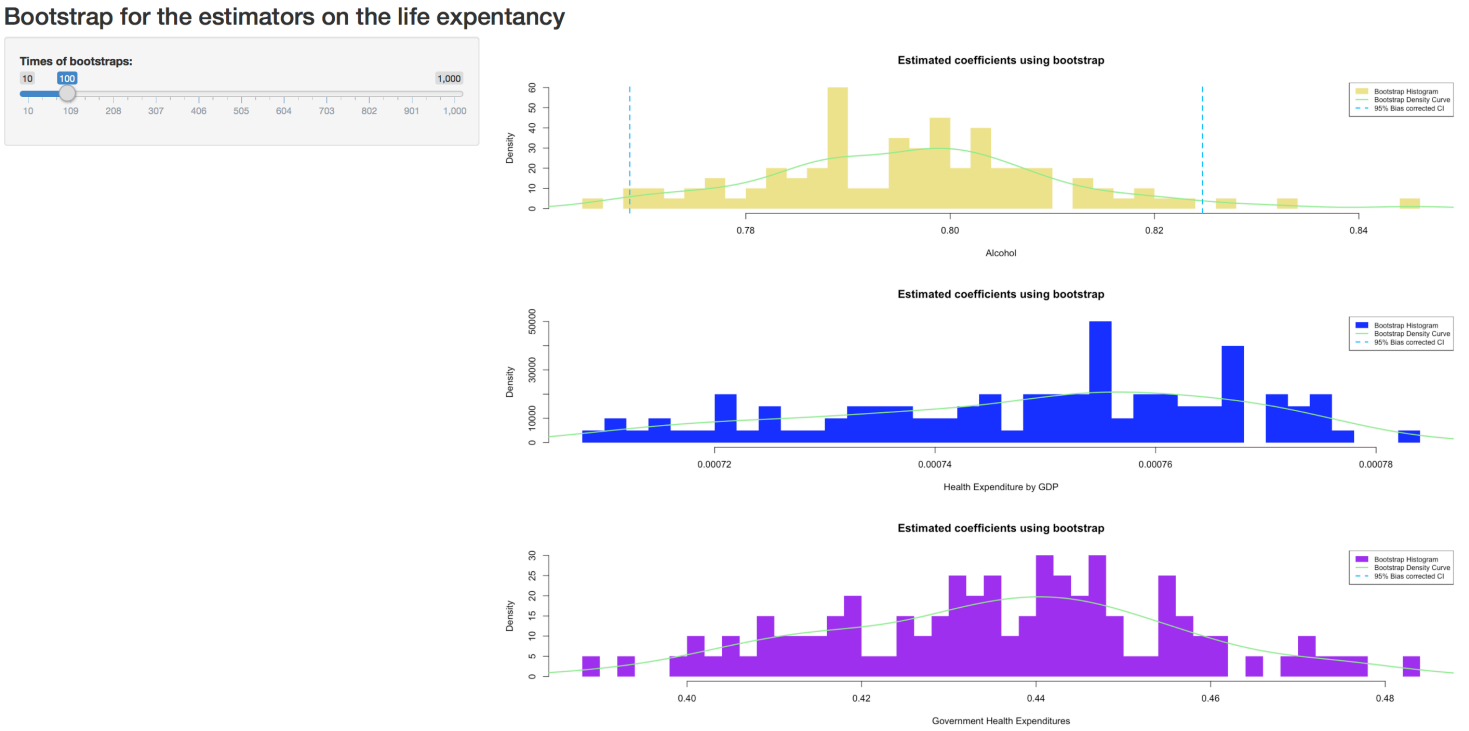
Then the residual bootstrap is conducted on this multiple linear regression model because of the possible linear correlation between the covariate. Compared to other bootstrap methods, it resamples the error term. Also, it is more stable and handle the problem of tie points in the data, and it is more stable because it does not change the matrix design. It conducts bootstrap on the residuals. The bootstrap value of life expectancy is calculated by the predicted life expectancy and residuals. Then the estimated impacts of three factors is calculated by matrix multiplication. The result of bootstrap estimation shows that the bias of three coefficients is very small, which is all close to 0. The standard error of three effects are 0.142, 0.00022,and 0.1933 when the simulation time is 1000. The estimated impacts of three factors are 0.7958,0.000756,and 0.4311. These values are closer to the OLS estimators.

The 95 % bias corrected bootstrap basic confidence interval of the impact of the alcohol,health expenditures by GDP, and governmental health Expenditure is (0.522, 1.091), (0.000318,0.00116),(0.0728,0.8275). The bootstrap percentile confidence intervals are (0.5035,1.07),(0.00034,0.0011),(0.055,0.81). The bootstrap

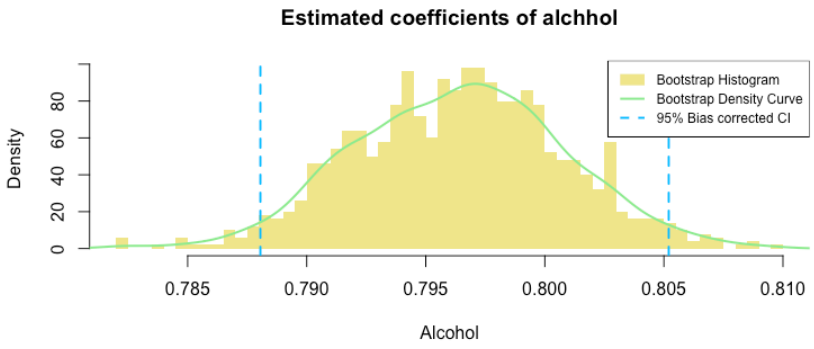
confidence interval using standard normal approximations are (0.518, 1.076),(-0.000325,0.00118),(0.063, 0.82). It shows that the percentile confidence interval and basic confidence interval is better because they are shorter.

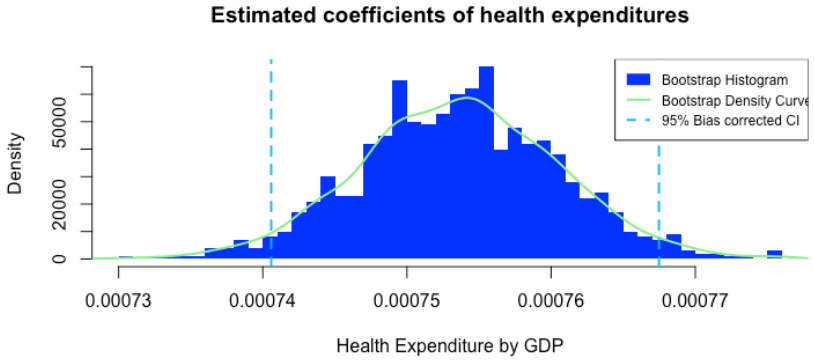
Next, use the shiny app to show how the estimator on the life expentancy changes when the total simulation changes. The shiny app includes the total simulation times, the bootstrap histogram, the bootstrap density curve, and 95% confidence interval using standard normal approximation.

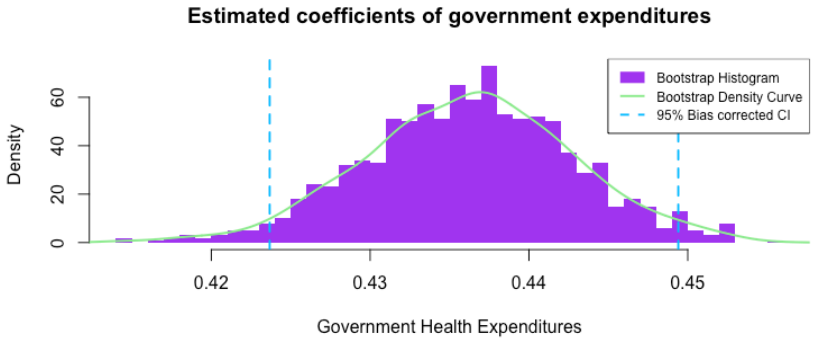
It is shown in the following plots when the simulation time is 100.



We could also compare the distribution of estimators when the simulation time differs. It shows that when the simulation time rises from 100 to 1000, the distribution of the estimators of three factors are close to normal distibution, and the estiamted result is better.







d. Conclusion and discussion

For the bootstrap on the data with discrete, continuous, and unknown distribution, our project finds that when the total simulation time increases, the estimator using bootstrap is closer to the observed value of the estimator using population data. It has a more accurate estimation on the population data. The distribution of the bootstrap estimator is closer to the distribution of population data as the total simulation time increases. The curve evolves from skewed distribution to bell-shaped in the continuous and unknown data distribution

For the residual bootstrap on the WHO health data, when the total simulation time is 1000, the bootstrap estimated impacts of alcohol consumption, health expenditures by GDP, governmental health Expenditure are 0.7958,0.000756,and 0.4311. which are all very close to the coefficients in the linear regression model, and the bias is very close to 0. The visulization using shiny app shows that when the simulation time is small, the distribution of estimated impacts of three factors are skewed to one side or distributed unevenly. When the simulation time increases, the distribution of estimated impacts of three factors is close to normal distribution.

The advantage of our project is that it does extensive simulation on different types of data. it also uses shiny app with histograms to show the distribution of estimator at different values of simulation. The limitation parts that we can further improve include trying simulations on more types of distribution to get a general

conclusion, such as the gamma and uniform distribution, and plots could be further improved by adding text and number. Also, in the residual bootstrap, we could try use PCA to make our results more accurate.

e.Bibliography

Mario L. Rizzo, Statistical computing with R, Computer Science and Data Analysis Series

f.Appendix: The compelete R code

Code for data following discrete distribution

```
set.seed(123)
library(bootstrap)
x=rpois(1000,3) #data
y=rpois(1000,3)
mydata=rbind(x,y)
library(IDPmisc)
iplot(t(mydata),xlab ="x",ylab = "y")
```

```
#From this plot we can see the most density dot is (3,3). However, since data from Po
isson distribution are all integers, this plot does not reflect the random situation
very well.

m.obs=apply(mydata,1,mean)# observed mean
v.obs=apply(mydata,1,var)#observed variance
B=1000
mboot=matrix(rep(0,2000),nrow=2)
vboot=matrix(rep(0,2000),nrow=2)
for(i in 1:B){
  z=sample(1:1000,size=1000,replace=TRUE)#sample the columns with replacement
  newsam<-mydata[,z]#reordered data
  mboot[,i]=apply(newsam,1,mean)#new mean
  vboot[,i]=apply(newsam,1,var)#new variance
}

#install.packages("IDPmisc")

iplot(t(mboot),xlab ="x",ylab = "y")
```

```
iplot(t(vboot),xlab ="x",ylab = "y")
```

```
se.boot=apply(mboot,1,sd)#standard deviacian
m.c=2*m.obs-apply(mboot,1,mean)#bias correction
(rbind(m.c-se.boot*1.96,m.c+se.boot*1.96))#Standard normal approximated CI
```

```
##             x        y
## [1,] 2.860203 2.886508
## [2,] 3.068945 3.100256
```

```
(c(2*m.obs[1]-quantile(mboot[1,],0.975),2*m.obs[1]-quantile(mboot[1,],0.025)))
```

```
##        x        x
## 2.856975 3.068000
```

```
(c(2*m.obs[2]-quantile(mboot[2,],0.975),2*m.obs[2]-quantile(mboot[2,],0.025)))
```

```
##        y        y
## 2.880875 3.100025
```

```
#The basic interval

(c(quantile(mboot[1,],0.025),quantile(mboot[1,],0.975)))
```

```
##     2.5%    97.5%
## 2.866000 3.077025
```

```
(c(quantile(mboot[2,],0.025),quantile(mboot[2,],0.975)))
```

```
##     2.5%    97.5%
## 2.883975 3.103125
```

```
#The Percentile interval


se.boot=apply(vboot,1,sd)#standard deviacian
v.c=2*v.obs-apply(vboot,1,mean)#bias correction
(rbind(v.c-se.boot*1.96,v.c+se.boot*1.96))#Standard normal approximated CI
```

```
##             x        y
## [1,] 2.650190 2.728361
## [2,] 3.193004 3.326082
```

```
(c(2*v.obs[1]-quantile(vboot[1,],0.975),2*v.obs[1]-quantile(vboot[1,],0.025)))
```

```
##        x        x
## 2.655261 3.195589
```

```
(c(2*v.obs[2]-quantile(vboot[2,],0.975),2*v.obs[2]-quantile(vboot[2,],0.025)))
```

```
##        y        y
## 2.734817 3.328610
```

```
#The basic interval

(c(quantile(vboot[1,],0.025),quantile(vboot[1,],0.975)))
```

```
##     2.5%    97.5%
## 2.638067 3.178395
```

```
(c(quantile(vboot[2,],0.025),quantile(vboot[2,],0.975)))
```

```
##     2.5%    97.5%
## 2.729320 3.323113
```

```
#The Percentile interval
```

Residual bootstrap on regression data(WHO data)

```
suppressMessages(library(dplyr))
suppressMessages(library(PerformanceAnalytics))
suppressMessages(library(kableExtra))
# Read the data
data=read.csv("Life Expectancy Data.csv")
# Select the desired variables
data1=data%>% filter(Year==2014)%>%select(Life.expectancy,Alcohol,percentage.expendit
ure,Total.expenditure)
# Delete the observations which have missing values
data1= na.omit(data1)
# Show the structure
str(data1)
```

```
## 'data.frame':    180 obs. of  4 variables:
##  $ Life.expectancy       : num  59.9 77.5 75.4 51.7 76.2 76.2 74.6 82.7 81.4 72.5
...
##  $ Alcohol               : num  0.01 4.51 0.01 8.33 8.56 ...
##  $ percentage.expenditure: num  73.5 428.7 54.2 24 2423 ...
##  $ Total.expenditure     : num  8.18 5.88 7.21 3.31 5.54 ...
##  - attr(*, "na.action")= 'omit' Named int  45 149 151
##   ..- attr(*, "names")= chr  "45" "149" "151"
```

```
#summarize all the variables
summary(data1)
```

```
##  Life.expectancy    Alcohol       percentage.expenditure Total.expenditure
##  Min.   :48.10   Min.   : 0.010   Min.   :    0.00       Min.   : 1.210
##  1st Qu.:65.85   1st Qu.: 0.010   1st Qu.:   11.96       1st Qu.: 4.487
##  Median :73.85   Median : 0.390   Median :  157.04       Median : 5.860
##  Mean   :71.71   Mean   : 3.308   Mean   : 1018.36       Mean   : 6.220
##  3rd Qu.:76.97   3rd Qu.: 6.790   3rd Qu.:  716.80       3rd Qu.: 7.750
##  Max.   :89.00   Max.   :15.190   Max.   :19479.91       Max.   :17.140
```

```
# Show the correlation between variables
chart.Correlation(data1,col="dodgerblue3")
```

```
mod = lm(Life.expectancy~Alcohol+percentage.expenditure+Total.expenditure,data=data1)
summary(mod)
```

```
##
## Call:
## lm(formula = Life.expectancy ~ Alcohol + percentage.expenditure +
##     Total.expenditure, data = data1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22.6927  -3.6836   0.8295   5.0127  17.2977
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            6.559e+01  1.286e+00  51.011  < 2e-16 ***
## Alcohol                7.965e-01  1.401e-01   5.687  5.3e-08 ***
## percentage.expenditure 7.538e-04  2.177e-04   3.462 0.000672 ***
## Total.expenditure      4.364e-01  2.000e-01   2.182 0.030455 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.932 on 176 degrees of freedom
## Multiple R-squared:  0.3398, Adjusted R-squared:  0.3286
## F-statistic:  30.2 on 3 and 176 DF,  p-value: 8.441e-16
```

```
# Build the confidence interval for each variables
confint(mod, level=.95)
```

```
##                                2.5 %       97.5 %
## (Intercept)            6.305303e+01 68.128271430
## Alcohol                5.200905e-01  1.072942097
## percentage.expenditure 3.241426e-04  0.001183497
## Total.expenditure      4.164161e-02  0.831122635
```

```
#estimated coefficients
(beta = mod$coef)
```

```
##            (Intercept)                 Alcohol percentage.expenditure
##           6.559065e+01           7.965163e-01           7.538197e-04
##      Total.expenditure
##           4.363821e-01
```

```
# Build a design matrix of predictor
X = as.matrix(cbind(rep(1,nrow(data1)),data1[,2:4]))
# Calculate the predicted value from the model
pred = X%*%beta
sigma = summary(mod)$sigma
sigma
```

```
## [1] 6.932004
```

```
# Do the simulation on the data
set.seed(2020)
# number of simulations
nsim = 1000
nobs = dim(data1)[1]
Ysim = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) +
          matrix(data=rnorm(nobs*nsim,mean=0,sd=sigma),nrow=nobs,ncol=nsim,byrow=FALS
E)

# Simulate on the impacts of threee factors
beta.sim = solve( t(X)%*%X, t(X)%*%Ysim,na.rm=TRUE )
beta.simT<-t(beta.sim)
#Alchohol
hist(beta.simT[,2],breaks=60,freq=FALSE,col="red",xlab = "Alcohol(beta1)",main="Histo
gram of the simulated coefficient of alchohol")
abline(v=c(mean(beta.simT[,2])), lwd=3,lty=3,col="blue")
```

```
# Expenditures by GDP
hist(beta.simT[,3],breaks=60,freq=FALSE,col="purple",xlab = "Expenditures(%GDP)(beta2
)",main="Histogram of the simulated coefficient of health expenditure")
abline(v=c(mean(beta.simT[,3])), lwd=3,lty=3,col="blue")
```

```
# Government Expenditures by GDP
hist(beta.simT[,4],breaks=60,freq=FALSE,col="yellow",xlab = "Expenditures(%Government
)(beta2)",main="Histogram of the simulated coefficient of government expenditure")
abline(v=c(mean(beta.simT[,4])), lwd=3,lty=3,col="blue")
```

```
c(mean(beta.simT[,2]),mean(beta.simT[,3]),mean(beta.simT[,4]))
```

```
## [1] 0.800139960 0.000753181 0.436532962
```

```
# residual bootstrap
res.data1 = data1$Life.expectancy - pred

res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot

beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
# estimated impacts using the bootstrap estimation
apply(X=beta.boot,MARGIN=1,FUN=mean)
```

```
##    rep(1, nrow(data1))                 Alcohol percentage.expenditure
##           6.564001e+01           7.958300e-01           7.559505e-04
##      Total.expenditure
##           4.311429e-01
```

```
# Bias
(biases = apply(X=beta.boot,MARGIN=1,FUN=mean) - beta)
```

```
##    rep(1, nrow(data1))                 Alcohol percentage.expenditure
##           4.936250e-02          -6.862856e-04           2.130818e-06
##      Total.expenditure
##          -5.239198e-03
```

```
#se
se.boot = apply(X=beta.boot,MARGIN=1,FUN=sd)
se.boot
```

```
##    rep(1, nrow(data1))                 Alcohol percentage.expenditure
##           1.2292440614           0.1422299126           0.0002178562
##      Total.expenditure
##           0.1932772234
```

```
# bias corrected estimator
beta.c = beta-biases
beta.c
```

```
##            (Intercept)                Alcohol percentage.expenditure
##           6.554129e+01           7.972026e-01           7.516889e-04
##      Total.expenditure
##           4.416213e-01
```

```
# Lower and Upper Bounds
betaq.low = apply(beta.boot,1,quantile,0.025)
betaq.up = apply(beta.boot,1,quantile,0.975)

# The 95% basic Confidence Interval
cbind(2*beta.c - betaq.up, 2*beta.c-betaq.low)
```

```
##                                [,1]         [,2]
## (Intercept)            6.297990e+01 67.972523216
## Alcohol                5.218357e-01  1.090886490
## percentage.expenditure 3.183948e-04  0.001156294
## Total.expenditure      7.285002e-02  0.827548137
```

```
# 95% quantile CI
qci1=c(betaq.low[2],betaq.up[2])
qci2=c(betaq.low[3],betaq.up[3])
qci3=c(betaq.low[4],betaq.up[4])

# 95% CI with standard normal approximation
snci1=beta.c[2] + c(-1,1)*qnorm(0.975)*se.boot[2]
snci2=beta.c[3] + c(-1,1)*qnorm(0.975)*se.boot[3]
snci3=beta.c[4] + c(-1,1)*qnorm(0.975)*se.boot[4]


# Do the distribution of bootstrap estimator when n=1000
nsim=1000
u=rep(0,nsim)
for(i in 1:nsim){
          res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
          res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
          Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot
          beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
            u[i]=mean(beta.boot[3,])
        }
        hist(u,breaks=50,freq=F,col="blue",border=NA,xlab='Health Expenditure by GDP'
,main="Estimated coefficients of health expenditures")
        lines(density(u),col='lightgreen',lwd=2)
        obs=mod$coef[3]
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('blue','lightgreen','deepskyblue'),legen
d=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.75
,lwd=c(10,2,2))
```

```
#Alcohol
u=rep(0,nsim)
        for(i in 1:nsim){
          res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
          res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
          Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot
          beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
            u[i]=mean(beta.boot[2,])
        }
        hist(u,breaks=50,freq=F,col="khaki",border=NA,xlab='Alcohol',main="Estimated
coefficients of alchhol")
        lines(density(u),col='lightgreen',lwd=2)
        obs=mod$coef[2]
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('khaki','lightgreen','deepskyblue'),lege
nd=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.7
5,lwd=c(10,2,2))
```

```
# Gov estimation
     u=rep(0,nsim)
        for(i in 1:nsim){
          res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
          res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
          Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot
          beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
            u[i]=mean(beta.boot[4,])
        }
        hist(u,breaks=50,freq=F,col="purple",border=NA,xlab='Government Health Expend
itures',main="Estimated coefficients of government expenditures")
        lines(density(u),col='lightgreen',lwd=2)
        obs=mod$coef[4]
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('purple','lightgreen','deepskyblue'),leg
end=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.
75,lwd=c(10,2,2))
```

Code for shiny app on unknown data

```
library(shiny)
```

```
##
## Attaching package: 'shiny'
```

```
## The following object is masked from 'package:IDPmisc':
##
##     hr
```

```
ui=fluidPage(
    titlePanel("Bootstrap for faithful data"),
    sidebarLayout(
        sidebarPanel(
            sliderInput("nsim",
                        "Times of bootstraps:",
                        min=10,
                        max=10000,
                        value=4000)
        ),
        mainPanel(
           plotOutput("distPlot")
        )
    )
)
server=function(input, output) {
    output$distPlot=renderPlot({
        x=faithful[,2]
        nsim=input$nsim
        u=rep(0,nsim)
        v=rep(0,nsim)
        for(i in 1:nsim){
            temp=sample(x,size=length(x),replace=TRUE)
            u[i]=mean(temp)
            v[i]=sd(temp)
        }
        par(mfrow=c(2,1))
        hist(u,breaks=50,freq=F,col="khaki",border=NA,xlim=c(68,74),xlab='Waiting Tim
e (min)',main=paste(sep='','Mean Value (Simulation Time = ',nsim,')'))
        lines(density(u),col='lightgreen',lwd=2)
        obs=mean(x)
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('khaki','lightgreen','deepskyblue'),lege
```

```
nd=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.7
5,lwd=c(10,2,2))

        hist(v,breaks=50,freq=F,col="khaki",border=NA,xlim=c(12,15),xlab='Waiting Tim
e (min)',main=paste(sep='','SD Value (Simulation Time = ',nsim,')'))
        lines(density(v),col='lightgreen',lwd=2)
        obs=sd(x)
        boot=v
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('khaki','lightgreen','deepskyblue'),lege
nd=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.7
5,lwd=c(10,2,2))
    },height = 800, width = 700 )
}
#shinyApp(ui=ui,server=server)
```

Code for shiny app on the estiamted impacts

```
library(shiny)
ui=fluidPage(
    titlePanel("Bootstrap for the estimators on the life expentancy"),
    sidebarLayout(
        sidebarPanel(
            sliderInput("nsim",
                        "Times of bootstraps:",
                        min=10,
                        max=1000,
                        value=100)
        ),
        mainPanel(
           plotOutput("dist1Plot",height="300px"),
           plotOutput("dist2Plot",height="300px"),
           plotOutput("dist3Plot",height="300px"),
        )
    )
)

server=function(input, output) {
    output$dist1Plot=renderPlot({
        nsim=input$nsim
        u=rep(0,nsim)
        for(i in 1:nsim){
          res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
          res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
          Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot
```

```
          beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
            u[i]=mean(beta.boot[2,])
        }
        hist(u,breaks=50,freq=F,col="khaki",border=NA,xlab='Alcohol',main="Estimated
coefficients using bootstrap")
        lines(density(u),col='lightgreen',lwd=2)
        obs=mod$coef[2]
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('khaki','lightgreen','deepskyblue'),lege
nd=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.7
5,lwd=c(10,2,2))
    })
    output$dist2Plot=renderPlot({
        nsim=input$nsim
        u=rep(0,nsim)
        for(i in 1:nsim){
          res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
          res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
          Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot
          beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
            u[i]=mean(beta.boot[3,])
        }
        hist(u,breaks=50,freq=F,col="blue",border=NA,xlab='Health Expenditure by GDP'
,main="Estimated coefficients using bootstrap")
        lines(density(u),col='lightgreen',lwd=2)
        obs=mod$coef[3]
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('blue','lightgreen','deepskyblue'),legen
d=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.75
,lwd=c(10,2,2))
    })
    output$dist3Plot=renderPlot({
        nsim=input$nsim
        u=rep(0,nsim)
        for(i in 1:nsim){
          res.boot = res.data1[sample(x=1:nobs,size=nsim*nobs,replace=TRUE)]
          res.boot = matrix(data=res.boot,nrow=nobs,ncol=nsim,byrow=FALSE)
          Y.boot = matrix(data=pred,nrow=nobs,ncol=nsim,byrow=FALSE) + res.boot
```

```
          beta.boot = solve( t(X)%*%X, t(X)%*%Y.boot)
            u[i]=mean(beta.boot[4,])
        }
        hist(u,breaks=50,freq=F,col="purple",border=NA,xlab='Government Health Expend
itures',main="Estimated coefficients using bootstrap")
        lines(density(u),col='lightgreen',lwd=2)
        obs=mod$coef[4]
        boot=u
        se=sd(boot)
        bias=mean(boot)-obs
        obs.corrected=obs-bias
        CI=obs.corrected+c(-1,1)*qnorm(0.975)*se
        abline(v=CI[1],col='deepskyblue',lty=2,lwd=2)
        abline(v=CI[2],col='deepskyblue',lty=2,lwd=2)
        legend('topright',lty=c(1,1,2),col=c('purple','lightgreen','deepskyblue'),leg
end=c('Bootstrap Histogram','Bootstrap Density Curve','95% Bias corrected CI'),cex=0.
75,lwd=c(10,2,2))
    })
}

#shinyApp(ui=ui,server=server)
```

Data following normal distribution

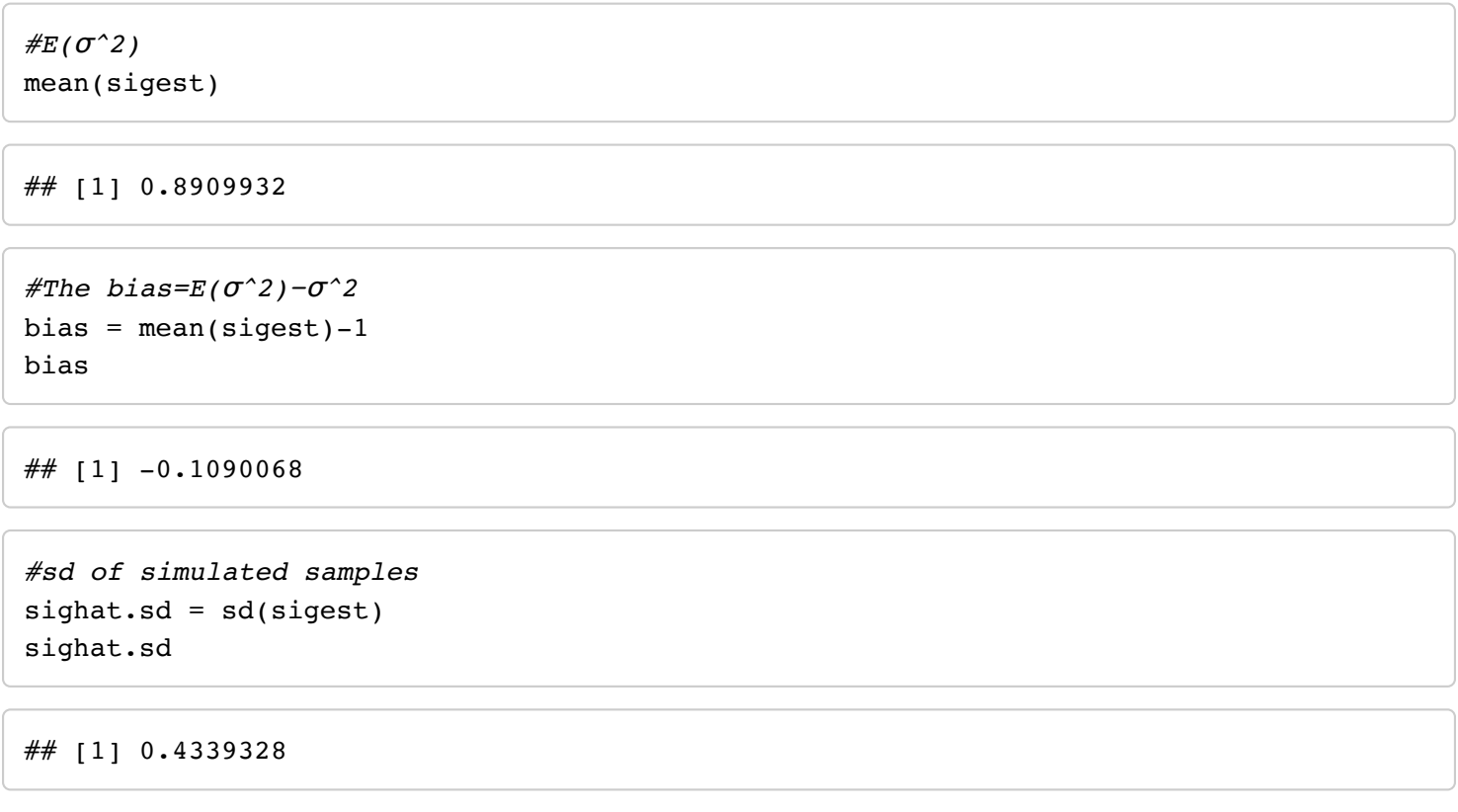
We simulate a random sample of size n=10 from a standard normal distribution: μ=0, σ^2=1

```
# We simulate a random sample of size n=10 from a standard normal distribution: μ=0, σ^2=1
# simulation time=1000
set.seed(26)
nsim = 1000
sigest = rep(0,nsim)

for(i in 1:nsim){
  X = rnorm(10,mean=0,sd=1)
  sigest[i] = mean( (X-mean(X))^2 )
}

hist(sigest, 20, freq = F,col = "deepskyblue4",main = "Distribution of the biased sam
ple estimator",xlab="Sample variance")
points(1,0,pch=16,col="red")
points(0.7430663,0,pch=16,col="chartreuse4")
abline(v=mean(sigest),col="chartreuse4",lwd=3)
abline(v=1,col="red",lwd=3)

#arrows(1.6,0.6,t.obs,0.5, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="red"
)
text(x=1.7,y=0.6, labels = "population variance=1", adj = 0,offset = 2, cex = 0.7, co
l = "red", font = NULL)
#arrows(2,0.3,1.7,0.1, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="chartreu
se4")
text(x=2.1,y=0.3, labels = "estimated variance=0.7430663", adj = 0,offset = 2, cex =
0.7, col = "chartreuse4", font = NULL)
```




```
#E(σ^2)
mean(sigest)
```

```
## [1] 0.8909932
```

```
#The bias=E(σ^2)-σ^2
bias = mean(sigest)-1
bias
```

```
## [1] -0.1090068
```

```
#sd of simulated samples
sighat.sd = sd(sigest)
sighat.sd
```

```
## [1] 0.4339328
```

```
#Sampling distribution of the nobias estimator
sighat.nobias = sigest-bias
hist(sighat.nobias,col="deepskyblue4",freq = FALSE)
```

```
# Confidence intervals based on normal theory
mean(sighat.nobias) + c(-1,1)*qnorm(0.975)*sighat.sd
```

```
## [1] 0.1495074 1.8504926
```

```
# Percentile Bootstrap Confidence interval
b0.025 = quantile(sigest,0.025)
b0.975 = quantile(sigest,0.975)
 c(b0.025,b0.975)
```

```
##      2.5%     97.5%
## 0.2805315 1.9282939
```

```
# Basic bootsrap comfidence interval
  2*mean(sighat.nobias) - c(b0.975, b0.025)
```

```
##      97.5%       2.5%
## 0.07170606 1.71946852
```

We simulate a random sample of size n=100 from a standard normal distribution: μ=0, σ^2=1

```
set.seed(26)
nsim = 1000
sigest = rep(0,nsim)

for(i in 1:nsim){
  X = rnorm(100,mean=0,sd=1)
  sigest[i] = mean( (X-mean(X))^2 )
}

hist(sigest, 20, freq = F,col = "deepskyblue4",main = "Distribution of the biased sam
ple estimator",xlab="Sample variance")
points(1,0,pch=16,col="red")
points(0.7430663,0,pch=16,col="chartreuse4")
abline(v=mean(sigest),col="chartreuse4",lwd=3)
abline(v=1,col="red",lwd=3)

#arrows(1.6,0.6,t.obs,0.5, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="red"
)
text(x=1.7,y=0.6, labels = "population variance=1", adj = 0,offset = 2, cex = 0.7, co
l = "red", font = NULL)
#arrows(2,0.3,1.7,0.1, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="chartreu
se4")
text(x=2.1,y=0.3, labels = "estimated variance=0.7430663", adj = 0,offset = 2, cex =
0.7, col = "chartreuse4", font = NULL)
```

```
mean(sigest)
```

```
## [1] 0.9901617
```

```
#The bias=E(σ^2)-σ^2
bias = mean(sigest)-1
bias
```

```
## [1] -0.009838254
```

```
#sd of simulated samples
sighat.sd = sd(sigest)
sighat.sd
```

```
## [1] 0.1440735
```

```
#Sampling distribution of the nobias estimator
sighat.nobias = sigest-bias
hist(sighat.nobias,col="deepskyblue4",freq = FALSE)
```

```
# Confidence intervals based on normal theory
mean(sighat.nobias) + c(-1,1)*qnorm(0.975)*sighat.sd
```

```
## [1] 0.7176211 1.2823789
```

```
# Percentile Bootstrap Confidence interval
b0.025 = quantile(sigest,0.025)
b0.975 = quantile(sigest,0.975)
 c(b0.025,b0.975)
```

```
##     2.5%    97.5%
## 0.736122 1.301755
```

```
# Basic bootsrap comfidence interval
  2*mean(sighat.nobias) - c(b0.975, b0.025)
```

```
##     97.5%      2.5%
## 0.6982449 1.2638780
```

We simulate a random sample of size n=1000 from a standard normal distribution: μ=0, σ^2=1

```
# We simulate a random sample of size n=100 from a standard normal distribution: μ=0, σ^2=1
# simulation time=1000
set.seed(26)
nsim = 1000
sigest = rep(0,nsim)

for(i in 1:nsim){
  X = rnorm(1000,mean=0,sd=1)
  sigest[i] = mean( (X-mean(X))^2 )
}

hist(sigest, 20, freq = F,col = "deepskyblue4",main = "Distribution of the biased sam
ple estimator",xlab="Sample variance")
points(1,0,pch=16,col="red")
points(0.7430663,0,pch=16,col="chartreuse4")
abline(v=mean(sigest),col="chartreuse4",lwd=3)
abline(v=1,col="red",lwd=3)

#arrows(1.6,0.6,t.obs,0.5, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="red"
)
text(x=1.7,y=0.6, labels = "population variance=1", adj = 0,offset = 2, cex = 0.7, co
l = "red", font = NULL)
#arrows(2,0.3,1.7,0.1, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="chartreu
se4")
text(x=2.1,y=0.3, labels = "estimated variance=0.7430663", adj = 0,offset = 2, cex =
0.7, col = "chartreuse4", font = NULL)
```

```
#E(σ^2)
mean(sigest)
```

```
## [1] 0.9982526
```

```
#The bias=E(σ^2)-σ^2
bias = mean(sigest)-1
bias
```

```
## [1] -0.001747432
```

```
#sd of simulated samples
sighat.sd = sd(sigest)
sighat.sd
```

```
## [1] 0.04619866
```

```
#Sampling distribution of the nobias estimator
sighat.nobias = sigest-bias
hist(sighat.nobias,col="deepskyblue4",freq = FALSE)
```

```
# Confidence intervals based on normal theory
mean(sighat.nobias) + c(-1,1)*qnorm(0.975)*sighat.sd
```

```
## [1] 0.9094523 1.0905477
```

```
# Percentile Bootstrap Confidence interval
b0.025 = quantile(sigest,0.025)
b0.975 = quantile(sigest,0.975)
 c(b0.025,b0.975)
```

```
##      2.5%     97.5%
## 0.9109369 1.0917475
```

```
# Basic bootsrap comfidence interval
  2*mean(sighat.nobias) - c(b0.975, b0.025)
```

```
##     97.5%      2.5%
## 0.9082525 1.0890631
```

simulation times=20

```
# We simulate a random sample of size n=100 from a standard normal distribution: μ=0, σ^2=1
# simulation times=10
set.seed(26)
nsim = 20
sigest = rep(0,nsim)

for(i in 1:nsim){
  X = rnorm(100,mean=0,sd=1)
  sigest[i] = mean( (X-mean(X))^2 )
}

hist(sigest, 20, freq = F,col = "deepskyblue4",main = "Distribution of the biased sam
ple estimator",xlab="Sample variance")
points(1,0,pch=16,col="red")
points(0.7430663,0,pch=16,col="chartreuse4")
abline(v=mean(sigest),col="chartreuse4",lwd=3)
abline(v=1,col="red",lwd=3)

#arrows(1.6,0.6,t.obs,0.5, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="red"
)
text(x=1.7,y=0.6, labels = "population variance=1", adj = 0,offset = 2, cex = 0.7, co
l = "red", font = NULL)
#arrows(2,0.3,1.7,0.1, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="chartreu
se4")
text(x=2.1,y=0.3, labels = "estimated variance=0.7430663", adj = 0,offset = 2, cex =
0.7, col = "chartreuse4", font = NULL)
```

```
#E(σ^2)
mean(sigest)
```

```
## [1] 0.9845199
```

```
#The bias=E(σ^2)-σ^2
bias = mean(sigest)-1
bias
```

```
## [1] -0.01548014
```

```
#sd of simulated samples
sighat.sd = sd(sigest)
sighat.sd
```

```
## [1] 0.1521975
```

```
#Sampling distribution of the nobias estimator
sighat.nobias = sigest-bias
hist(sighat.nobias,col="deepskyblue4",freq = FALSE)
```

```
# Confidence intervals based on normal theory
mean(sighat.nobias) + c(-1,1)*qnorm(0.975)*sighat.sd
```

```
## [1] 0.7016985 1.2983015
```

```
# Percentile Bootstrap Confidence interval
b0.025 = quantile(sigest,0.025)
b0.975 = quantile(sigest,0.975)
 c(b0.025,b0.975)
```

```
##      2.5%     97.5%
## 0.7131467 1.2349168
```

```
# Basic bootsrap comfidence interval
  2*mean(sighat.nobias) - c(b0.975, b0.025)
```

```
##     97.5%      2.5%
## 0.7650832 1.2868533
```

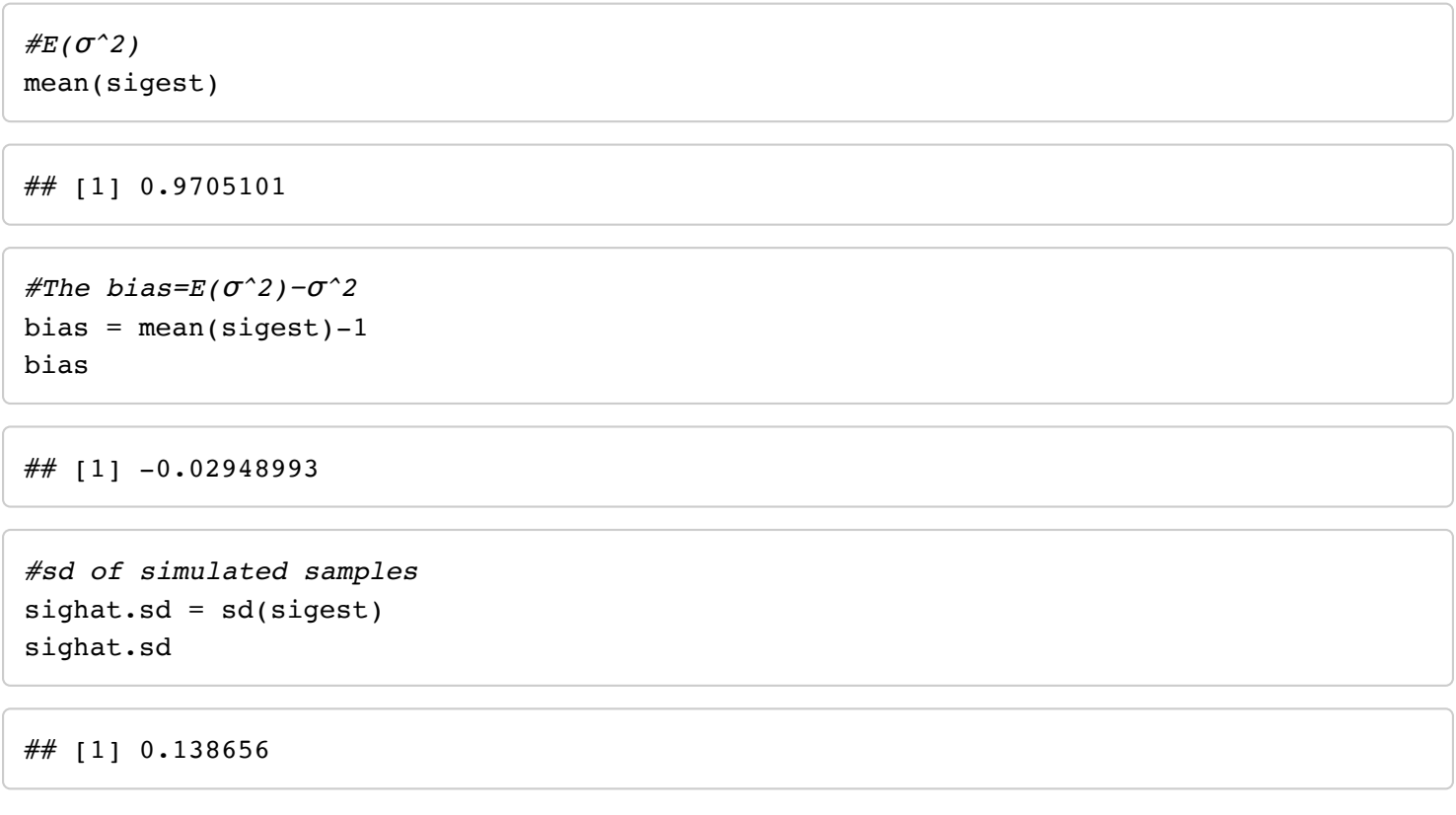
simulation times=200

```
# We simulate a random sample of size n=100 from a standard normal distribution: μ=0, σ^2=1
# simulation times=20
set.seed(26)
nsim = 200
sigest = rep(0,nsim)

for(i in 1:nsim){
  X = rnorm(100,mean=0,sd=1)
  sigest[i] = mean( (X-mean(X))^2 )
}

hist(sigest, 20, freq = F,col = "deepskyblue4",main = "Distribution of the biased sam
ple estimator",xlab="Sample variance")
points(1,0,pch=16,col="red")
points(0.7430663,0,pch=16,col="chartreuse4")
abline(v=mean(sigest),col="chartreuse4",lwd=3)
abline(v=1,col="red",lwd=3)

#arrows(1.6,0.6,t.obs,0.5, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="red"
)
text(x=1.7,y=0.6, labels = "population variance=1", adj = 0,offset = 2, cex = 0.7, co
l = "red", font = NULL)
#arrows(2,0.3,1.7,0.1, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="chartreu
se4")
text(x=2.1,y=0.3, labels = "estimated variance=0.7430663", adj = 0,offset = 2, cex =
0.7, col = "chartreuse4", font = NULL)
```




```
#E(σ^2)
mean(sigest)
```

```
## [1] 0.9705101
```

```
#The bias=E(σ^2)-σ^2
bias = mean(sigest)-1
bias
```

```
## [1] -0.02948993
```

```
#sd of simulated samples
sighat.sd = sd(sigest)
sighat.sd
```

```
## [1] 0.138656
```

```
#Sampling distribution of the nobias estimator
sighat.nobias = sigest-bias
hist(sighat.nobias,col="deepskyblue4",freq = FALSE)
```

```
# Confidence intervals based on normal theory
mean(sighat.nobias) + c(-1,1)*qnorm(0.975)*sighat.sd
```

```
## [1] 0.7282393 1.2717607
```

```
# Percentile Bootstrap Confidence interval
b0.025 = quantile(sigest,0.025)
b0.975 = quantile(sigest,0.975)
 c(b0.025,b0.975)
```

```
##      2.5%     97.5%
## 0.7337055 1.2521940
```

```
# Basic bootsrap comfidence interval
  2*mean(sighat.nobias) - c(b0.975, b0.025)
```

```
##    97.5%     2.5%
## 0.747806 1.266295
```

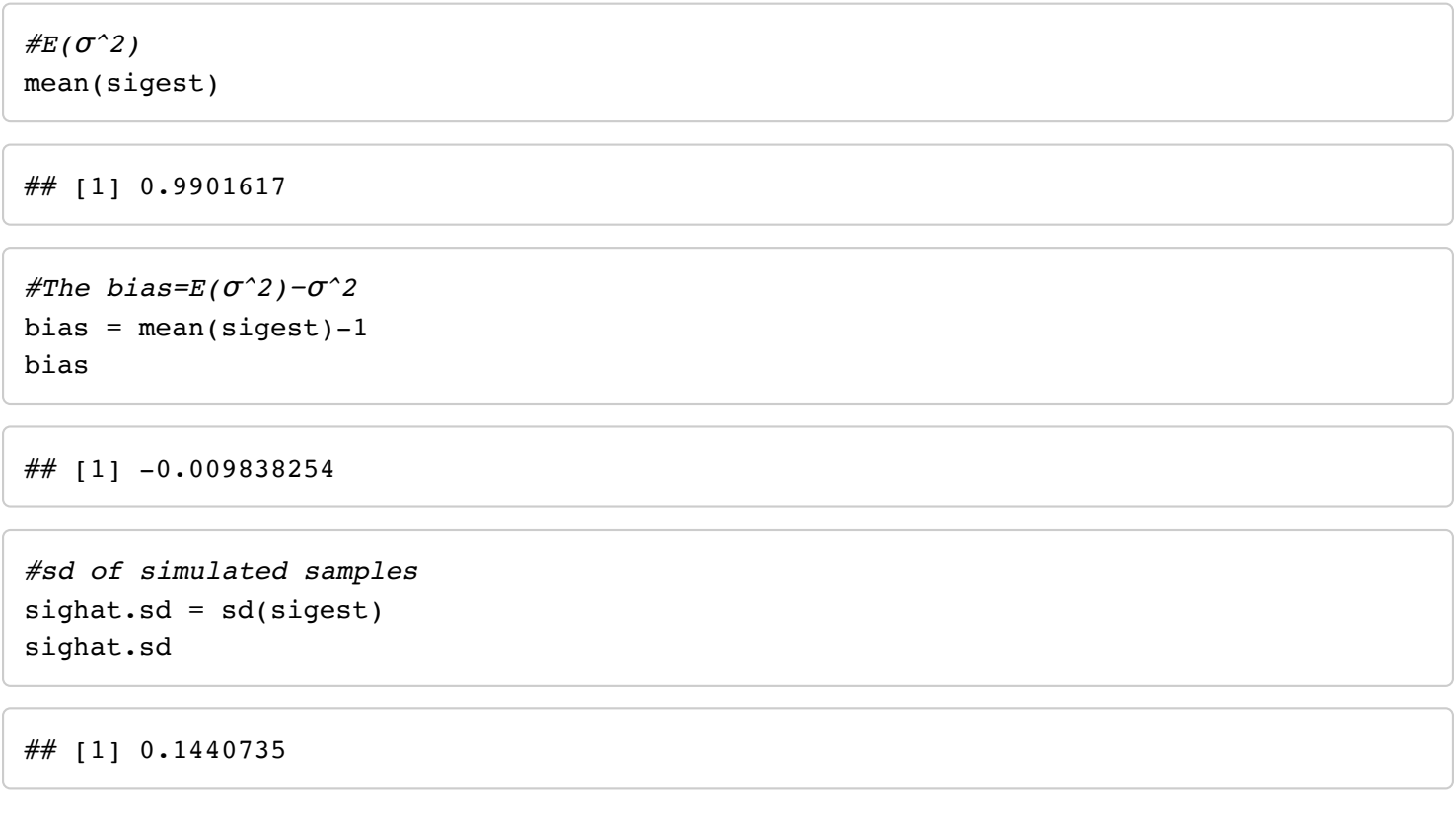
simulation times=1000

```
# We simulate a random sample of size n=100 from a standard normal distribution: μ=0, σ^2=1
# simulation times=50
set.seed(26)
nsim = 1000
sigest = rep(0,nsim)

for(i in 1:nsim){
  X = rnorm(100,mean=0,sd=1)
  sigest[i] = mean( (X-mean(X))^2 )
}

hist(sigest, 20, freq = F,col = "deepskyblue4",main = "Distribution of the biased sam
ple estimator",xlab="Sample variance")
points(1,0,pch=16,col="red")
points(0.7430663,0,pch=16,col="chartreuse4")
abline(v=mean(sigest),col="chartreuse4",lwd=3)
abline(v=1,col="red",lwd=3)

#arrows(1.6,0.6,t.obs,0.5, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="red"
)
text(x=1.7,y=0.6, labels = "population variance=1", adj = 0,offset = 2, cex = 0.7, co
l = "red", font = NULL)
#arrows(2,0.3,1.7,0.1, length = 0.07, code = , angle = 45, lty=1, lwd=2,col="chartreu
se4")
text(x=2.1,y=0.3, labels = "estimated variance=0.7430663", adj = 0,offset = 2, cex =
0.7, col = "chartreuse4", font = NULL)
```




```
#E(σ^2)
mean(sigest)
```

```
## [1] 0.9901617
```

```
#The bias=E(σ^2)-σ^2
bias = mean(sigest)-1
bias
```

```
## [1] -0.009838254
```

```
#sd of simulated samples
sighat.sd = sd(sigest)
sighat.sd
```

```
## [1] 0.1440735
```

```
#Sampling distribution of the nobias estimator
sighat.nobias = sigest-bias
hist(sighat.nobias,col="deepskyblue4",freq = FALSE)
```

```
# Confidence intervals based on normal theory
mean(sighat.nobias) + c(-1,1)*qnorm(0.975)*sighat.sd
```

```
## [1] 0.7176211 1.2823789
```

```
# Percentile Bootstrap Confidence interval
b0.025 = quantile(sigest,0.025)
b0.975 = quantile(sigest,0.975)
 c(b0.025,b0.975)
```

```
##     2.5%    97.5%
## 0.736122 1.301755
```

```
# Basic bootsrap comfidence interval
  2*mean(sighat.nobias) - c(b0.975, b0.025)
```

```
##     97.5%      2.5%
## 0.6982449 1.2638780
```