EDA A3 Q4: Random sampling plans

Consider the study population $\mathcal{P}_{Study}$ of $N = 100$ blocks of uniform thickness and density (all blocks were cut from the same opaque plastic sheet of about 5 mm thickness), but have different shapes.

Data on this population of 100 blocks are available as an R data set 'blocks'. This data set has four variates: block 'id' number, 'weight' in grams, 'perimeter' in centimetres, and the 'group' of the block (being either 'A' or 'B'). It can be loaded from the assignment data directory as follows:

```
load("/Users/rudranibhadra/Downloads/blocks.rda")
head(blocks, n = 3)
```

```
##   id weight perimeter group
## 1  1     55        32     B
## 2  2     35        27     B
## 3  3     35        25     A
```

In this question, you will investigate different sampling plans and estimation procedures.

a. Simple random sampling.

b. *(4 marks)* Collect the sample average block weight from each of 1000 samples, where each sample consists of 10 blocks selected at random (without replacement) from all 100 blocks.

Before sampling, set.seed(314159)

Save the results on the R variable `randomSampleAves`.

Show your code.

```
set.seed(314159)
rs<-t(replicate(1000,sample(blocks$id,10,replace=FALSE)))

randomSampleAves<-data.frame(rs)

s<-randomSampleAves

s[]<-lapply(randomSampleAves,function(x) blocks$weight[match(x,blocks$id)])

randomSampleAves$AverageWeight<-rowMeans(s)

randomSampleAves<-randomSampleAves$AverageWeight
```

ii. *(3 marks)* Using 'randomSampleAves', estimate the sampling bias, the sampling variability, and the sampling mean squared error of this sampling plan.

Show your code.

```
x<-mean(blocks$weight)
SampleError<-randomSampleAves-x

#sampling bias
n<-length(randomSampleAves)
sum(SampleError)/n
```

```
## [1] 0.041
```

```
#sampling variability
var(randomSampleAves)
```

```
## [1] 23.12564
```

```
#sampling mean squared error
var(randomSampleAves) + (sum(SampleError)/n)^2
```

```
## [1] 23.12733
```

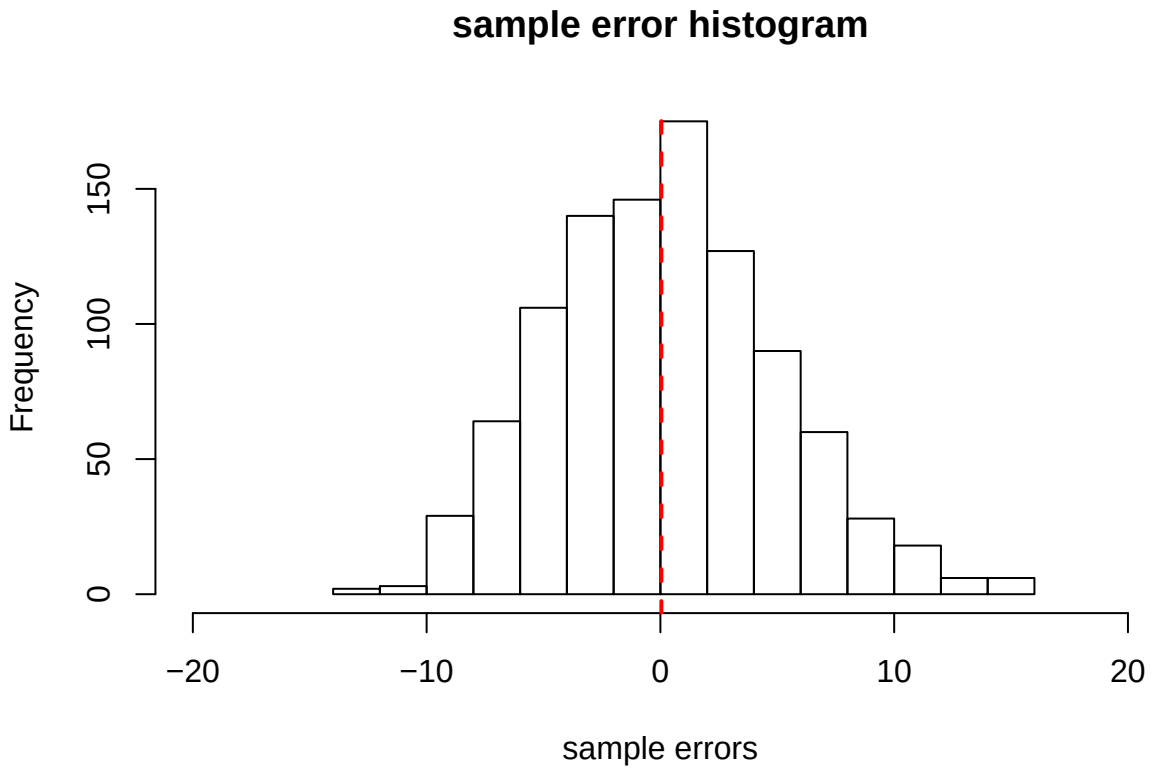
iii. *(3 marks)* Construct a (suitably labelled) histogram of the sample **errors** from this sampling plan.

Use 'xlim = c(-20,20)'.

Add a vertical red dashed line of 'lwd = 2' at the average error.

Show your code.

```
hist(SampleError, main="sample error histogram", xlab="sample errors",xlim = c(-20,20))
abline(v=mean(SampleError), col="red",lwd=2,lty=2)
```



b. Stratified random sampling.

i. *(4 marks)* Collect the sample average block weight from each of 1000 samples, where now each sample consists of 5 blocks selected at random (without replacement) from each of group "A" and group "B".

Before sampling, set.seed(314159)

Save the results on the R variable 'stratifiedSampleAves'.

Show your code.

```
set.seed(314159)

stratifiedSampleAves<-c()
```

```
for(i in 1:1000){
  a<-sample(blocks$weight[blocks$group=="A"],5,replace=FALSE)
  #print(a)
  b<-sample(blocks$weight[blocks$group=="B"],5,replace=FALSE)
  #print(b)
  stratifiedSampleAves[i]<-mean(c(a,b))
}
```

ii. *(3 marks)* Using 'stratifiedSampleAves', estimate the sampling bias, the sampling variability, and the sampling mean squared error of this sampling plan.

Show your code.

```
x<-mean(blocks$weight)
SampleError<-(stratifiedSampleAves-x)

#sampling bias
n<-length(stratifiedSampleAves)
sum(SampleError)/n
```

```
## [1] 0.0515
```

```
#sampling variability
var(stratifiedSampleAves)
```

```
## [1] 11.5029
```

```
#sampling mean squared error
var(stratifiedSampleAves) + (sum(SampleError)/n)^2
```

```
## [1] 11.50555
```

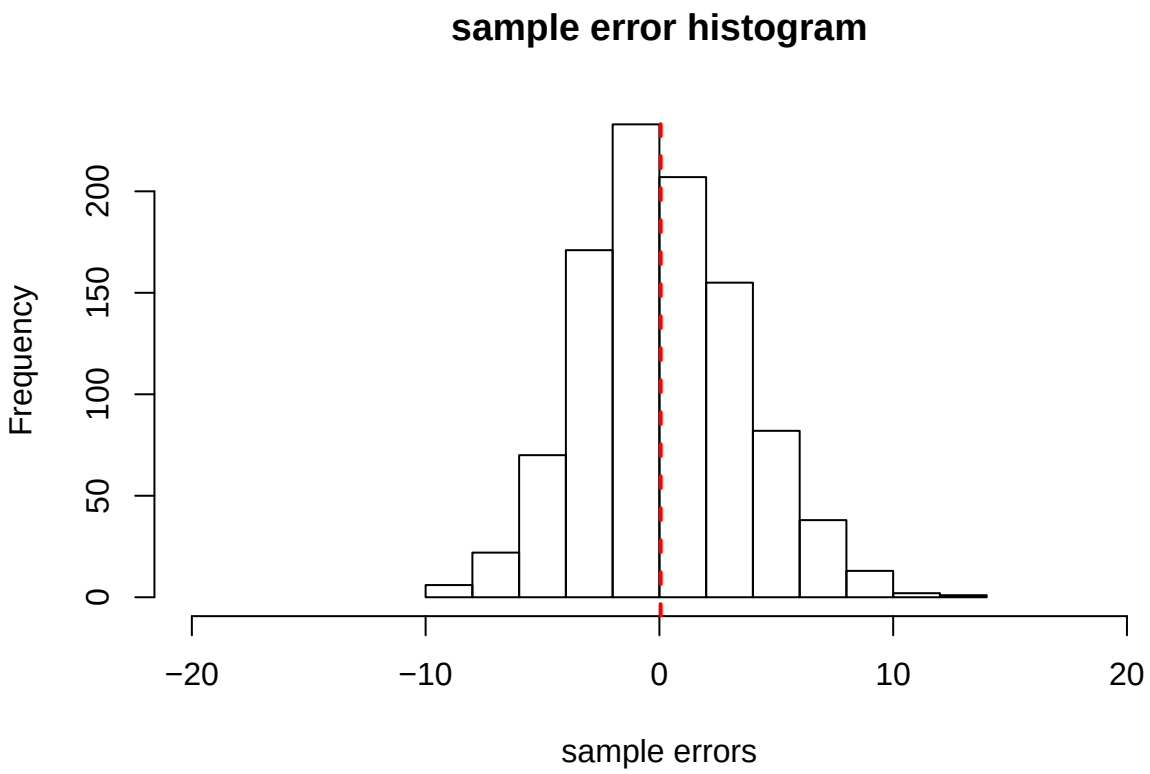
iii. *(3 marks)* Construct a (suitably labelled) histogram of the sample **errors** from this sampling plan.

Use 'xlim = c(-20,20)'.

Add a vertical red dashed line of 'lwd = 2' at the average error.

Show your code.

```
hist(SampleError, main="sample error histogram", xlab="sample errors",xlim = c(-20,20))
abline(v=mean(SampleError), col="red",lwd=2,lty=2)
```



c. Regression estimators. In this question, we suppose that we know something about the population of blocks. In particular, suppose we know that the average perimeter of all 100 blocks is mean(blocks$perimeter$) = $rmean(blocks$perimeter).

We also understand that there is some relationship between `perimeter` and `weight` in this population.

i. *(4 marks)* Here 1,000 samples of 10 blocks are to be selected at random (without replacement) from all 100 blocks. For each sample of 10 blocks, construct a straight line fit of the `weight` on `perimeter`. Then use this fit to predict the mean weight of the population when the 'perimeter' is the actual average perimeter of all 100 blocks. Collect all 1,000 regression estimates.

Before sampling, `set.seed(314159)`

Save the results on the R variable 'regressionEstimates'.

Show your code.

```
set.seed(314159)
#m<-mean(blocks$perimeter)
#m<-data.frame(mean(blocks$perimeter))
#colnames(m)<-c("perimeter")
#m
ra<-t(replicate(1000,sample(blocks$id,10,replace=FALSE)))
regressionEstimates<-data.frame(ra)

s<-regressionEstimates
predicted_mean_weight<-c()
for(r in 1:nrow(regressionEstimates)){
 w<-sapply(s[r,],function(x) blocks$weight[match(x,blocks$id)])
  p<-sapply(s[r,],function(x) blocks$perimeter[match(x,blocks$id)])
```

```
  d<-data.frame(weight=w, perimeter=p)
  fit<-lm(w~p,data=d)
  predicted_mean_weight[r]<-predict(fit,data.frame(p=mean(blocks$perimeter)))
}

regressionEstimates<-cbind(regressionEstimates,predicted_mean_weight)
#head(regressionEstimates)
regressionEstimates<-regressionEstimates$predicted_mean_weight
```

ii. *(3 marks)* Using 'regressionEstimates', estimate the sampling bias, the sampling variability, and the sampling mean squared error of this sampling plan.

Show your code.

```
x<-mean(blocks$weight)
SampleError<-(regressionEstimates-x)

#sampling bias
n<-length(regressionEstimates)
sum(SampleError)/n
```

```
## [1] -0.1381506
```

```
#sampling variability
var(regressionEstimates)
```

```
## [1] 4.931066
```

```
#sampling mean squared error
var(regressionEstimates) + (sum(SampleError)/n)^2
```

```
## [1] 4.950152
```

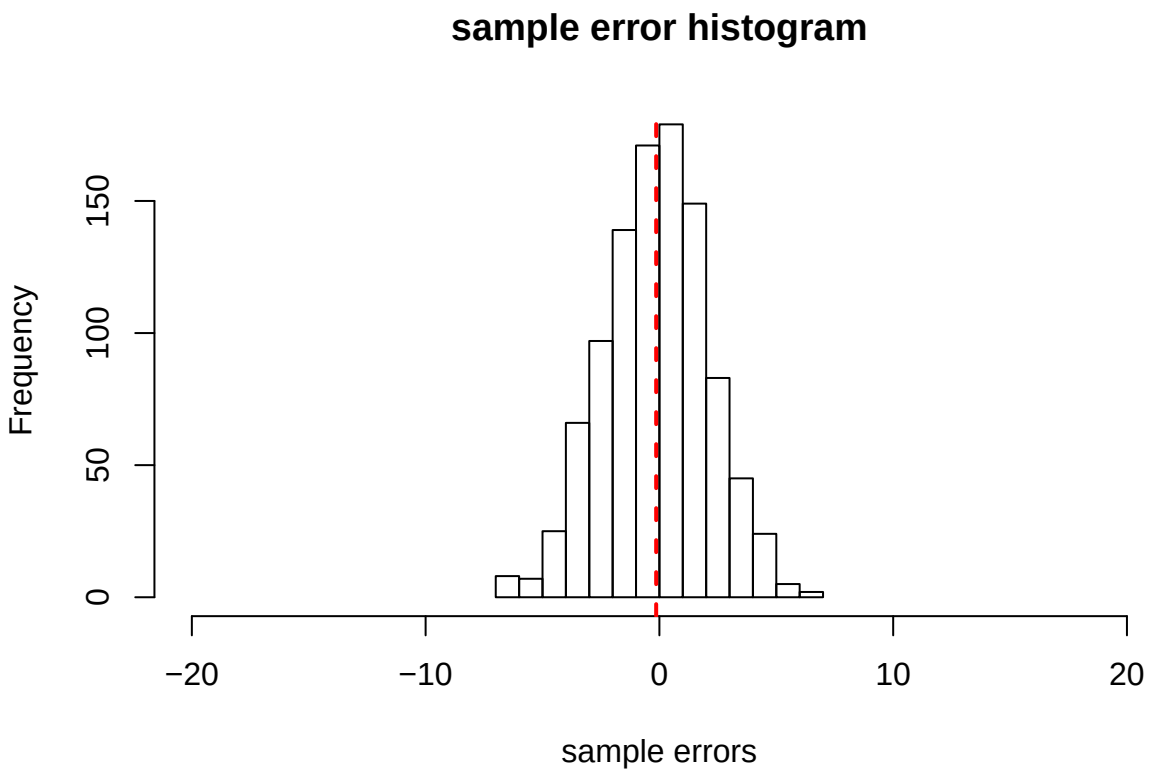
iii. *(3 marks)* Construct a (suitably labelled) histogram of the sample **errors** from this sampling plan.

Use xlim = c(-20,20).

Add a vertical red dashed line of 'lwd = 2' at the average error.

Show your code.

```
hist(SampleError, main="sample error histogram", xlab="sample errors",xlim = c(-20,20))
abline(v=mean(SampleError), col="red",lwd=2,lty=2)
```



iv. *(2 marks)* Is the straight line model used in this question "true"? Is it useful? Explain your answers.

The straight line is true since the predicted average weights are close to the actual weight of all 100 blocks. It is useful as from seen from the histogram, the sample errors are not as spread out and it is condensed.

d. A number of graduate data science students were asked to view the entire collection of 100 blocks and to choose 10 whose average weight they believed came close to matching that of all 100. The sample units selected are recorded in another file, `judgmentSamples.csv`. These can be loaded from the assignment data directory as follows:

```
students <- read.csv("/Users/rudranibhadra/Downloads/judgmentSamples.csv")
head(students, n = 3)
```

```
##   studentID first second third fourth fifth sixth seventh eighth ninth
## 1      5086    12     18    17     11    15    20      14     13    16
## 2      3848    34     35    70     56    32    14       5     88    81
## 3      6656    14     34    41     29    32    55      74     40    16
##   tenth
## 1    18
## 2    73
## 3    70
```

There were a total of r nrow(students) students and hence r nrow(students) samples selected.

In this question, we compare the **judgment** sampling plan of the students with that of the random sampling plans considered above when only 33 samples of size 10 are selected.

i. *(2 marks)* Gather together the average block weights of the student judgment samples. Save the results on the R variable 'judgmentAves'.

Print the average of these averages.

Show your code.

```
judgmentAves<-students
judgmentAves[]<-lapply(students,function(x) blocks$weight[match(x,blocks$id)])
#head(s)
judgmentAves<-cbind(students[,1],judgmentAves[,-1])

judgmentAves$AverageWeight<-rowMeans(judgmentAves[,-1])

judgmentAves<-judgmentAves$AverageWeight
```

ii. *(8 marks)* Using judgmentAves and only the first r nrow(students) entries of each of randomSampleAves, stratifiedSampleAves, and regressionEstimates, construct four histograms one above the other (in the same display, use an appropriate par()) one for each of these sets of results.

Make sure each histogram is labelled appropriately.

Use the same xlim = c(20, 50), ylim = c(0, 15), and breaks = seq(20, 50, 2) for each histogram.

On each histogram add a vertical red dashed line (with lwd = 2) at the true population average weight of all 100 blocks.

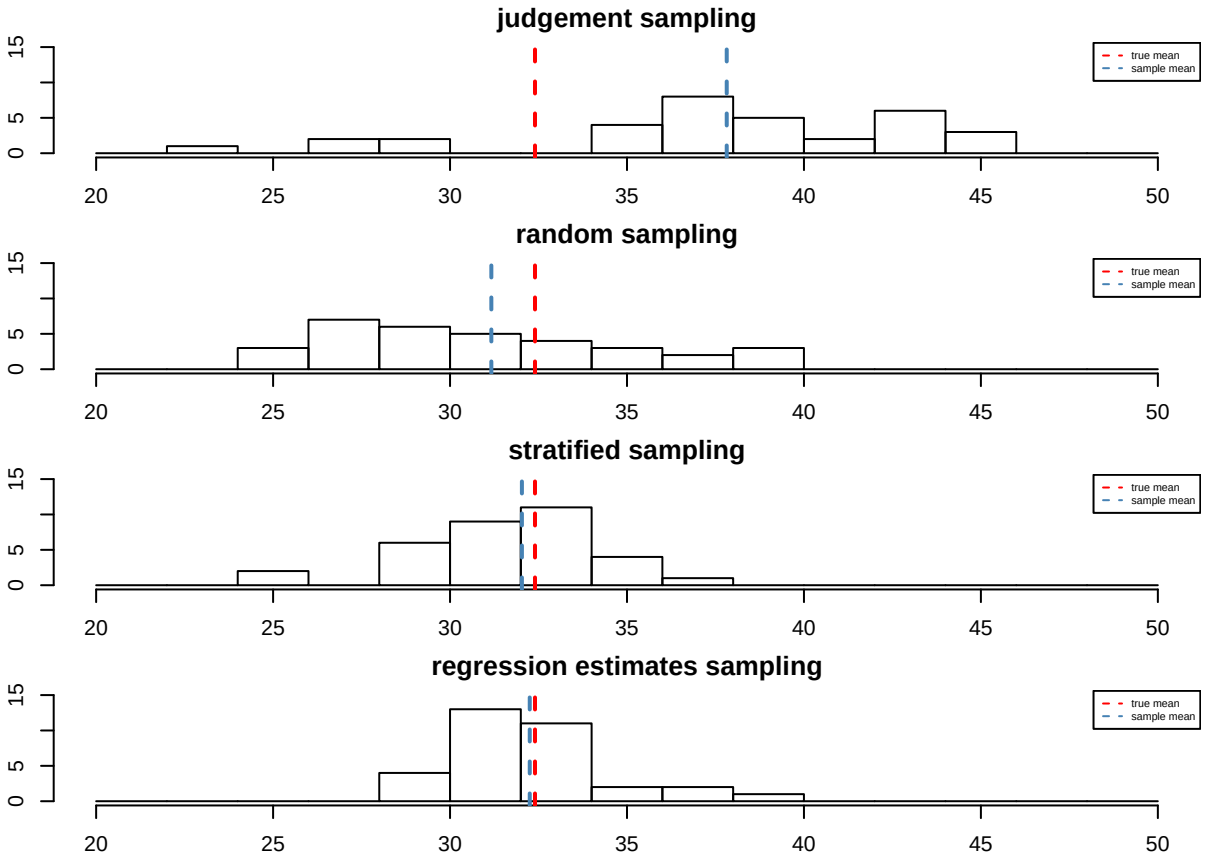
On each histogram add a vertical "steelblue" **solid** line (with lwd = 2) at the average of all r nrow(students) sample estimates.

On each histogram, add a legend indicating which vertical line is which.

Show your code.

```
par(mfrow=c(4,1),mar=c(2,2,2,2))
d1<-randomSampleAves[1:33]
d2<-stratifiedSampleAves[1:33]
d3<-regressionEstimates[1:33]

hist(judgmentAves,xlim = c(20, 50), ylim = c(0, 15),breaks = seq(20, 50, 2),xlab="weights",
main="judgement sampling")
abline(v=mean(blocks$weight),col="red",lwd=2,lty=2)
abline(v=mean(judgmentAves) ,col="steelblue",lwd=2,lty=2)
legend("topright", lty=c(2,2), col=c("red", "steelblue"),
       legend = c("true mean", "sample mean"),cex=0.5)

hist(d1,xlim = c(20, 50), ylim = c(0, 15),breaks = seq(20, 50, 2),xlab="weights",
main="random sampling")
abline(v=mean(blocks$weight),col="red",lwd=2,lty=2)
abline(v=mean(d1) ,col="steelblue",lwd=2,lty=2)
legend("topright", lty=c(2,2), col=c("red", "steelblue"),
      legend = c("true mean", "sample mean"),cex=0.5)

hist(d2,xlim = c(20, 50), ylim = c(0, 15),breaks = seq(20, 50, 2),xlab="weights",
main="stratified sampling")
abline(v=mean(blocks$weight),col="red",lwd=2,lty=2)
abline(v=mean(d2) ,col="steelblue",lwd=2,lty=2)
legend("topright", lty=c(2,2), col=c("red", "steelblue"),
      legend = c("true mean", "sample mean"),cex=0.5)

hist(d3,xlim = c(20, 50), ylim = c(0, 15),breaks = seq(20, 50, 2),xlab="weights",
main="regression estimates sampling")
```

```r
abline(v=mean(blocks$weight),col="red",lwd=2,lty=2)
abline(v=mean(d3) ,col="steelblue",lwd=2,lty=2)
legend("topright", lty=c(2,2), col=c("red", "steelblue"),
       legend = c("true mean", "sample mean"),cex=0.5)
```



e. *(5 marks)* Comment on the relative merits of the four sampling plans. Which would you most recommend? Which least?

The judgement sampling is easy to conduct since the students can choose any 10 weights based on their judgement. Random sampling is the fairest of all since its just picking up samples randomly where each block has an equal chance of being selected. Stratified sampling is selecting weights from all groups to make sure that some samples are picked up from each group so that it highly represents the target population. Regression sample is based on a prior assumption of the relationship between weights and perimeters so if the assumption is correct, it can accurately represent the target population.

I would most recommend regression sampling since its sampling bias, sampling variability and sampling MSE is the lowest out of the all four sampling errors. Also its sample mean weight is closest to the true mean weight out of all sampling plans. Since the sampling bias, sampling variability and sampling MSE is the largest for the judgement sampling, this is least recommended. Its sample mean is also furthest away from the true mean.