week 6 - Ratio and Regression estmation

Peter Lugtig

09 oktober, 2023

Class/take home exercises week 6

Two exercises this week. In case you do not finish in class, finish them at home: i) an exercise on ratio estimation (relevant in cluster samples) (model-assisted estimation)
ii) an exercise on regression estimation (model-based estimation)

Exercise 1: Ratio estimation

For the next exercise, you will need the following libraries:

```r
require(sampling)
require(survey)
require(dplyr)
require(ggplot2)
```

In this short exercise, you will practice with specifying a ratio estimator in R, and hopefully will learn more about why ratio estimators are so useful in specific circumstances; especially in cluster samples

We will use a new dataset for this exercise that centers on my favourite drink in the world: coffee. Imagine a situation where Utrecht University would like to keep track of how many coffees are being drunk on the buildings on campus in a month. Perhaps they want to know how much coffee is being taken from the coffee machines, perhaps they simply need to know how many coffees are being drunk to prepare a new coffee contract with a supplier. We will first concentrate on estimating the mean number of coffees per machine, but can also estimate the total directly.

There are about 1000 machines on campus, some of them used more often than others. Lets imagine for now the university is not willing to use any information on the type of building or other auxiliary information to inform a potentially stratified design.

Also, they opt to equip a SRS of 100 machines with a device that counts the number of coffees drunk.

Let's sample some data. First the population data (pretend for now you don't know about these data, like in real life.)

```r
set.seed(11)
coffees <- round(rpois(1000,350)+rnorm(100,0,sd=150))
coffees[coffees<0] <- 0
# and energy use
energy <- 0.072*coffees+rnorm(n=1000,mean=0,sd=1)
energy[energy<0] <- 0
coffeedata <- as.data.frame(cbind(coffees,energy))
names(coffeedata) <- c("n","energy")
```

Question 1.

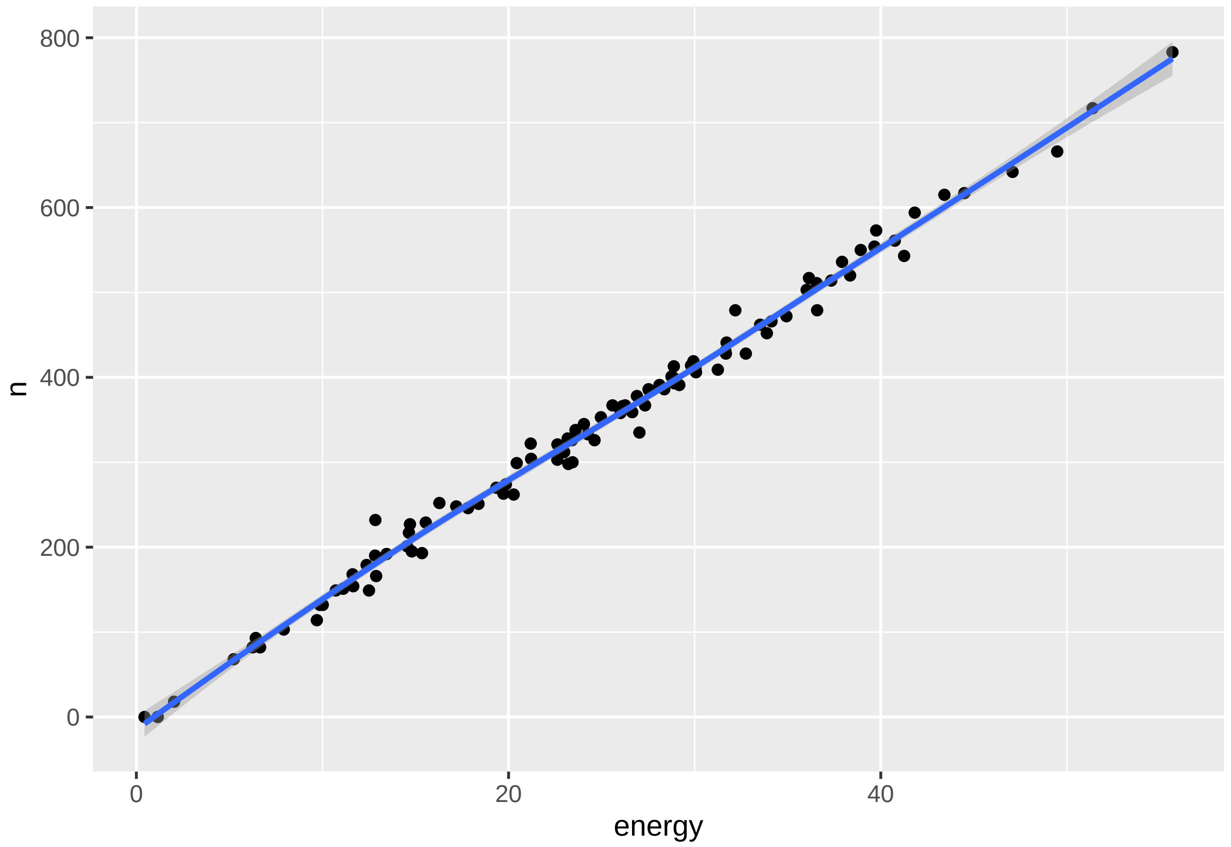
Sample an SRS of size=100 from the just-created population, and estimate the mean number of coffees using normal SRS estimation (as covered in week 3), including the se.:

You will find that the confidence interval is quite wide. The university hires a survey researcher to design a more efficient sample and estimate the total number of coffees drunk more efficiently.
The researcher finds out that the although the university does not keep track of the number of coffees made per machine, it does track the energy consumption per machine. Can this information somehow be used?

Question 2:

Now what? Can we do better? Perhaps! But we have to model the relation between energy use and the number of coffees a mcahine makes. Lets make a plot (using just the survey data), and decide whether we can use a ratio estimator. Do you think a ratio estimator is here a good idea judging the plot?

```
ggplot(coffee,aes(y=n, x=energy))+
         geom_point()+
         geom_smooth()
```



Question 3:

Run the code below to first estimate the ratio between coffees/energy. This is done in mmultiple steps. Inspect the code to see how this is done. What happens to the precision when we use a ratio estimator?

1. We estimate the ratio of energy/n

```
# estimate the ratio betwen energy/n
ratio <- svyratio(~n,~energy, design=srsdesign)
```

```r
# what is this ratio?
print(ratio) # with 1 KwH of energy, 13.81 coffees are made on average
```

We can now use the ratio, to predict for every value of energy we have in our population, the predicted value on n (coffees). For that we first:

2. Need to create a new predicted variable for the number of coffees (predictions per cluster)

```r
# With the ratio0-estimator we will predict for every case in our population,
# based on a covariate x (energy) what the value of the dependent variable y (n) is.

# I am creating a new predicted population for this,
# where I delete the values for all cases that I did not sample
predictedpop <-coffeedata
predictedpop$n[predictedpop$srs!=1] <- NA
```

3. In a third step we can use the ratio as calculated in step 1. to predict the number of coffees based on the mean energy consumption of the different machines (see below). We will now predict the number of coffeees for all machines (clusters) in the population

```r
# We here now use the predict value, which takes the output of a regression model (below: "ratio")
# and then we multiply these coefficients with the values of our covariate (energy)
# to estimate the mean on y (n coffees)
# the "total" command is used in the predict() function to predict a new variable
predict(ratio, total=mean(predictedpop$energy))
# we can also manually get to this mean by taking the ratio * mean(x), but the predict
# function is made for this, so we will use that from here
ratio[[1]] * mean(coffeedata$energy)
```

Queston 4:

What happened to the standard error when you compare the ratio estimation method to the SRS method?

```r
# answer: and the standard error = only 1.35,
# see below, compared to # se =15.4 with an SRS!
```

Question 5:

In our dataset, the standard error becomes much smaller because of ratio estimation. Why then not always use a ratio-estimator? First, we need to have a dependent variable that is of ratio measurement level. The ratio estimator will not work when we for example try to estimate weight or height in the "boys" dataset, as boys never have a height or weight of zero. Secondly, we need to have a covariate on our sampling frame (like "energy use") that is ideally a perfectly linear predictor of our dependent variable. The relationship between height and weight is not perfectly linear for example. The relationship between energy use and the number of coffees from a machine is however. Or is it not?

Investigate whether the ratio estimator you used in the previous question is actually unbiased by comparing the mean to the population mean in the "coffeedata". What is the prediction of the total number of coffees drunk, and how much bias is there in this estimate?

Question 6:

Can you think of a reason why there is some bias in the ratio estimator for the total number of coffees drunk at the UU? Think about an answer first, and then run the code below. What do you find?

```r
 # why?
summary(glm(n~energy, data=coffee))
```

A possible solution to deal with a biased ratio estimator, is to extend the model with more covariates. In such a case, we move from using a “model-assisted” estimator like ratio estimation to a truly “model-based” estimator. In order to understand how model-based estimators work, look at exercise 2. “Regression estimation”

Question 7 (bonus):

The ratio estimator is in practice used quite often in cluster samples. Can you think of why?
Rather than estimating a mean, an outcome is often the total, which can be estimated with the svytotal() command. Can you estimate the total number of coffees produced, rather than the mean (see .Rmd file for answers)

Exercise 2 - Regression estimation

For the purpose of this exercise, we are using the example we also used for ratio estimation. At Utrecht University, there are about 1000 machines on campus, some of them used more often than others. We want to estimate the total coffee consumption at the university. Instead of using a stratied and/or cluster design, the previous exercise showed that ratio estimation can really improve on those designs in terms of improving precision.

A disadvantage in the coffee example was however that the ratio of energy use/number of coffees produced per machine was not completely constant: when a machine produced close to 0 coffees, there was still some energy use, leading to a small bias in the ratio estimator. In this exercise, we will try to improve on the ratio estimator, and introduce model-based estimation as a completely new way to do inference. Lets first generate the cofffee population dataset again (same as in previous exercise), draw an SRS and produce a ratio estimator. The code below is copied from exercise 1.

```
set.seed(11)
coffees <- round(rpois(1000,350)+rnorm(100,0,sd=150))
coffees[coffees<0] <- 0
# and energy use
energy <- 0.072*coffees+rnorm(n=1000,mean=0,sd=1)
energy[energy<0] <- 0
coffeedata <- as.data.frame(cbind(coffees,energy))
names(coffeedata) <- c("n","energy")
```

And we will draw a SRS sample of size=100 for the purpose of later comparing our results to. We will again focus on estimating the total number of coffees drunk at the UU.

```
set.seed(11)
coffeedata$srs <- srswor(100,nrow(coffeedata))
coffee <- subset(coffeedata, srs==1)

# specify population size and survey design object
coffee$fpc <- 1000
srsdesign <-svydesign(ids=~1,fpc=~fpc,data=coffee)

svymean(~n, design=srsdesign) # se =15415
# or the confidence interval
confint(svymean(~n, design=srsdesign)) #[312237;372663]
```

We will also use the ratio estimator we designed in the previous exercise using the srs sample.

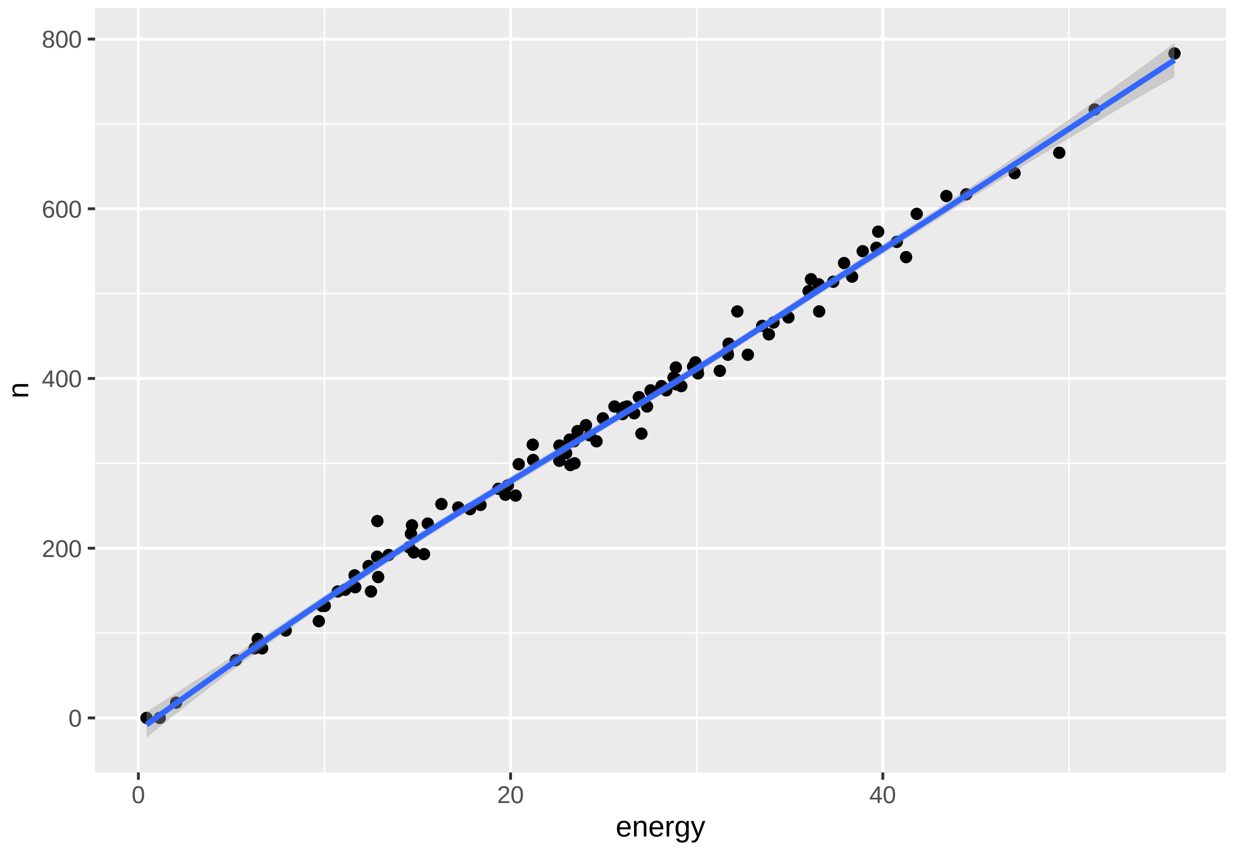
```
#1. estimate the ratio
ratio <- svyratio(~n,~energy, design=srsdesign)
# what is this ratio?
print(ratio) # with 1 KwH 13.81 coffees are made on average

#2. create a new dataset, without n as dependent variable
predicted <-coffeedata
# I am deleting the true coffee here for the machines not in the survey
predicted$n[predicted$srs!=1] <- NA

#3. below, we take the sum(energy-values * ratio estimator)
predict(ratio, total=mean(predicted$energy))

# and store the results as an object
meancoffeeratio <- predict(ratio, total=mean(predicted$energy))[[1]]
meancoffeeratio - mean(coffeedata$n) # -2.103, there seems to be a bit of bias. Again, machines that do
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Question 1:

The plot you see just above this question shows the relation between energy use and the number of coffees (n). The ratio estimator is doing the same thing as fitting a regression model with just one predictor, and with the intercept being 0 (the origin) Lets fit a regression model that does just that. What do you find?

Question 2:

In regression estimation, we cannot estimate the mean directly (in contrast to ratio estimation). We first need to predict the individual values in the population, using our regression coefficients (use $coefficients), and the auxiliary variable we use, and then use the predicted values to estimate the quantity we want to estimate. So rather than relying on the assumptions of design based inference (the Central Limit Theorem), we are using the predicted values to estimate our statistic of interest. Run the code below to estimate the mean number of coffees using the regression model we just fitted. Do we find a difference as compared to the ratio estimator?

```r
# we will add a second column of predictions (coffee) to the new dataset
# that multiplies the values on ' energy'  with the regression coefficent
predicted$predregr <- predict(out,newdata=predicted, total=mean(coffeedata$n))
mean(predicted$predregr)
# or by hand
mean(coffeedata$energy)*out$coefficients

# difference with the ratio estimator:
mean(predicted$predregr) - meancoffeeratio #last object here was the mean using ratio-estimation
# there is a very small difference
```

The means are roughly the same! As they should be, because we fit basicly the same model.

#What about the standard error from our regression estimator?

If we want to get the SE for the regression-based estimator, this is a bit more tricky. First, we have to fit the regression model with a normal lm() function, not the 'svyglm()' function. Now, you are probably slightly puzzled about NOT using svyglm, because we are now ignoring the whole sampling design! Indeed, I will show you in a minute why. Lets first show how to get the standard error. Run the code below:

```
# first we fit a normal lm function, all the rest is copied from the earlier regression model.
# note, we don't estimate an intercept!
out2 <- lm(n~ 0 + energy,data=coffee)
print(out2$coefficients) # same regression as with svyglm
# and we are again adding a new column to our data with the newly predicted values
predicted$predregr2 <- predict(out2,newdata=predicted, total=mean(coffeedata$n))
mean(predicted$predregr2) #341.49 = same

# the predict() function includes a handy way to calculate
# standard errors following the "Delta"  method:

# here: Taylor series expansion (see also Fundamentals course),
# that you can get by adding "se=T" and then ask for the residual
# the reason we need Taylor series is (like the Hurvitz Thompson estimator)
# we are using *the individual* predicted values
predict(out2,newdata=predicted, total=mean(coffeedata$energy),se=T)$residual.scale
# se= 14.33, which is much larger than the ratio estimate
```

Question 3:

Compare the standard errors from the ratio estimator, and regression estimator. Write these down below:

Standard error ratio-estimator:
Standard error regression-estimator:

Question 4:

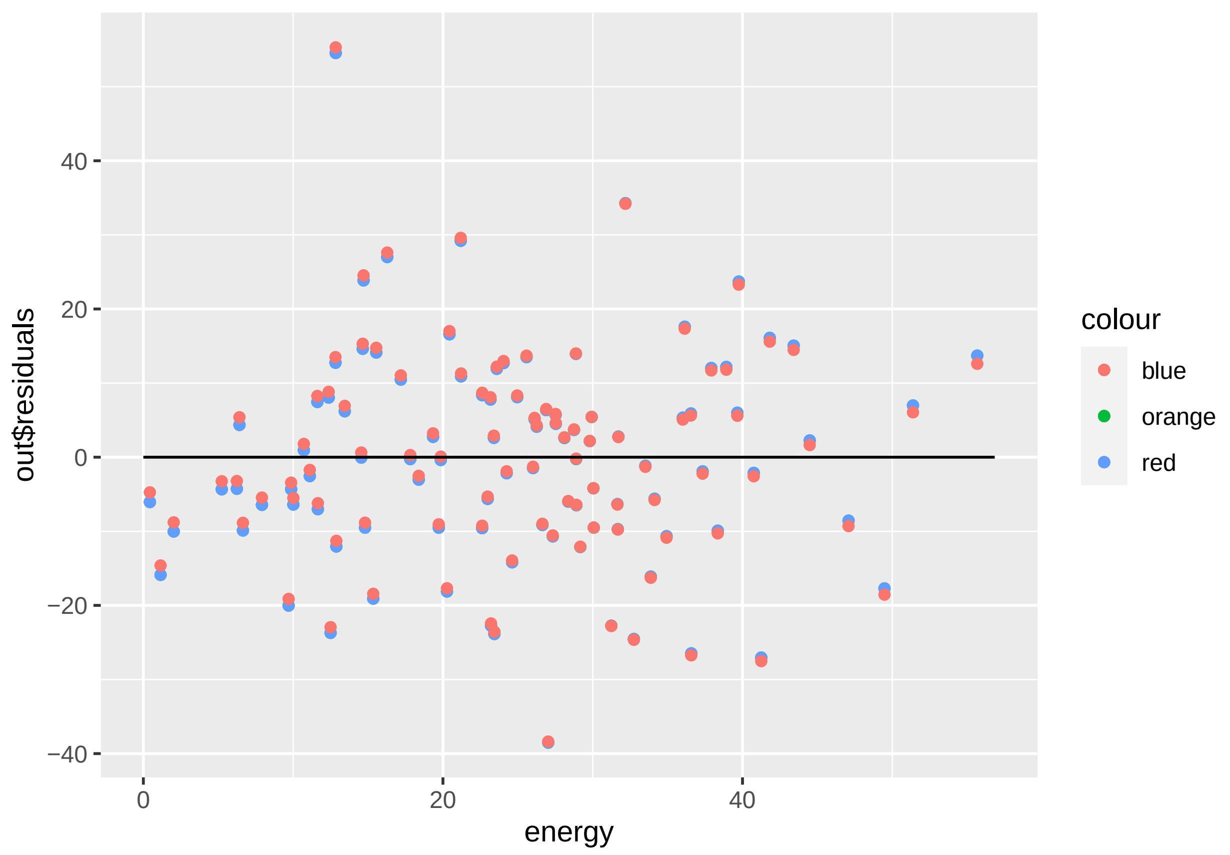
We saw earlier that the ratio estimator was a bit biased, because we assume the regression line goes through the origin. Or in other words, that the ratio between the two variables in the ratio-estimator is constant. When we switch to a regression model, we can however also add an intercept to the model.

Run the regression model from question 1 again, but now add an intercept to the model. What do you find when you compare the two models?

Question 5:

The standard error in regression-based models do not represent *sampling* uncertainty anymore, but rather represent *uncertainty about the regression line*. Although the standard errors of the SRS estimate and the regression-based estimate seem similar in size, they have completely different meanings now! We can plot the residuals from the different models we have estimated so far, to understand better what the standard error means. Run the code below. Can you explain from looking at the plot what the size of the standard error is?

```
# plot the residuals
ggplot(predicted, aes(x=energy))+
  geom_point(data=out, aes(y=out$residuals, colour="orange"))+ # ratio model
  geom_point(data=out2, aes(y=out2$residuals, colour="red"))+ # regression model with 0 intercept
  geom_point(data=out3, aes(y=out3$residuals, colour="blue"))+ # with an intercept
  geom_line(y=0) # this is the regression line
```



```
# the square root of the squared deviations from the regression line is about 14
```

Question 6:

Imagine there is more information on the sampling frame. Apart from energy consumption, there are now two additional predictors:
- age of the machine (in years) - brand (A or B)

I will also now create a new sample, no longer consisting of a random sample out of the population, but a selective sample of the 100 machines with the highest energy consumption only. This is one of the major advantages of model-based estimation. We do not strictly need a random sample for inference. This is also why we run just a "lm()" function, rather than specifying the "svydesign" object first.

We will after this try to improve the model-based inference by adding more covariates and working from a biased sample.

```
coffeedata$age <- out3$residual/4  + 10
coffeedata$brand <- round((out$residual+rnorm(1000,mean=45,sd=8))/28)
coffeedata$brand[coffeedata$brand==4] <- 0
coffeedata$brand <- as.factor(coffeedata$brand)
levels(coffeedata$brand) <- c("maas","bravilor","nescafe","douwe egberts")

# selecting the sample: arrange by energy (lo:hi), take the last 100 observations (slice_tail) and dele
biasedsample <- coffeedata %>%
  arrange(energy)%>%
  slice_tail(n=100)%>%
  mutate(srs=NULL)
```

Question 7:

In regression-based inference we need to minimize the standard error of the model to get the best estimator. We can then take the mean values of X, and use the regression line to predict the mean in Y. In other words, inference will work under the condition that:

- we have a a good regression model (low se)
- we need to have good information on the level of the population on the covariates. In order to understand this last point: it was nice that we had the variable "energy use" for every machine in our dataset, but it would have sufficed had we only had the mean level of energy for all the machines combined as long as we have the regression-model coefficients.

To summarize, we need to have some sample data (no longer a randomly sampled one!), where we can estimate a good regression model, for which we then only need the assumption that the coefficients that we find in our sample also hold in our population. Then, we take the mean values in the population for the covariates used in the regression model, and then multiply:

"our regression model coefficients" * "population means on covariates in model"

Following this method of inference, first compute the mean values for "brand", "age" and "energy in the population

Mean(brand): Mean(age): Mean(energy):

In a second step, try to estimate the mean number of coffees from the "biasedsample". Use a similar syntax as in question 4, but now use the new sample, and find out whether the new covariates "brand" and "age" improve the model. Do they?

Question 8

Using your model estimated above again, did the fact that we now use a very biased sample (only including the machines that consume a lot of energy), lead to a very different result? Why not?

– End of document –