SRS sampling

Peter Lugtig

16 July 2021

This file is the product of an R Markdown file. If you are reading the pdf-file , open the .Rmd file as well in R just to see what R Markdown does. There is no need yet to understand exactly what R Markdown does. For now, the only thing that is worth knowing is that when you press "knit" in R studio (and you have the right dataset in the same folder as the .Rmd file) you will get the .html file. You can also type text in the .Rmd file, and add R code snippets (or comment). Play around! More on R Markdown later in this course, and in the "Programming with R" course.

Today we are going to learn: i) how to sample from data using simple random sampling (exercise 1), ii) work with the {survey package} and iii) do some exercises (can be done by hand or in R) on sample size calculations

---

Exercise 1 - Sampling

Open the `boys.RDS` (orginal .sav file also available). Inspect the variable labels and make yourself familiar with the dataset. The key variables in the dataset are the height and weight for *all* 738 boys under the age of 21 living in an imaginary county (gemeente) in the Netherlands in 2014. In other words, we have the whole population, and the idea is we explore different hypothetical ways to sample from this population.The dataset stems from a surveillance system, in which every child is seen every few years. The following measures are available:

|age | Decimal age (0-21 years) | |hgt | Height (cm) | |wgt | Weight (kg) | |bmi | Body mass index | |hc | Head circumference (cm) | |gen | Genital Tanner stage (G1-G5) | |phb | Pubic hair (Tanner P1-P6) | |tv | Testicular volume (ml) | |Town | town1 to town5 |

```r
library(tidyverse)
library(survey)
library(sampling)
boys <- read_rds("boys.RDS")
```

Getting to know the population

1. The variable town consists of five towns in the county, numbered town1 to town5. Investigate how age is distributed over the towns and write down the respective numbers of cases in the following crosstable (HINT: create a new variable `agecat` that indicates the age categories. It is also useful to ask for row, column and total percentages for each cell in the crosstable; you'll need them later).

```r
# create agecat
boys$agecat[boys$age<1] <- "under 1"
boys$agecat[boys$age>=1] <- "1-5"
```

```r
boys$agecat[boys$age>=5] <- "5-10"
boys$agecat[boys$age>=10] <- "over 10"
# I am also adding an ID
boys$id <- 1:748
```

and now create the crosstable:

```r
table(boys$agecat,boys$town)
prop.table(table(boys$agecat,boys$town))
```

	town1	town2	town3	town4	town5	total
age < 1	34	7	52	22	19	134
age 1 - <5	50	14	44	34	12	154
age 5 - <10	18	5	22	18	5	68
age >=10	89	55	121	87	37	389
total	191	81	239	161	79	745

2. Based on the above table, if we would draw a single person at random, this person most likely would live in town 3 and most likely belongs to age category `age >= 10`.

3. Based on the above table, what would be the least likely combination of characteristics if we would draw a single person at random ?

4. What is the mean age in the population?

```r
mean(boys$age) # specify what you want to compute
```

5. Randomly sample 200 cases (without replacement) from the dataset and recompute the mean for the sample only. How large is the difference in your population mean? What is (for your single sample) the estimate of the mean age of the child?

R code

```r
library(sampling) # see THE week 37
set.seed=123 #for replication purposes
boys$sample <-srswor(200,nrow(boys))
# explanation: we add an indicator (0,1), that will be 1 for sample 200 cases
# we sample here WithOut Replacement (WOR).
# Explore the sampling package for advanced settings that we will use later
table(boys$sample) # check, does it work? Yes
```

```
##
##   0   1
## 548 200
```

You can also View the dataset with the {View()} command.

```r
#View(boys)
```

Now, lets compute the mean age of the child!

```r
srssample1 <- filter(boys,sample== 1)
# or write srssample1 <- boys[boys$sample==1]
mean(srssample1$age)
mean(boys$age)-  mean(srssample1$age) # bias!
```

Please note that the small bias that we observe here is due to random sampling error. Every time we sample we will get a slightly different estimate for the mean, but averaged over an infinite amount of samples we can draw, the bias will be 0. We can estimate the size of the random error (and empirically establish that our estimator is indeed unbiased) by running this code many times (see lecture). However, the Central Limit Theorem posits that we can simply predict the average error, and summarize this in the *standard error* statistic (s.e) .

6. We would normally always like to get an estimate of the standard error with our estimate of interest (in this case of the mean). The standard error is equal to: sample standard deviation (square root of variance / sqrt(n). Compute the standard error of the mean by hand (or in R).
7. Can you also compute the 95% CI of the mean for the variable age in our srs?

Exercise 2 - The survey package

Most statistical packages compute the s.e. for us assuming we always use a simple random sample. You have probably done this before.However, we will soon start using more complicated sampling designs, where computing the se. accurately is complicated, so I will show here how we can compute the *correct* standard error for any design using the {survey} package in R. The survey package can also deal with more complicated sampling designs quite easily. Although not needed yet now, I will also show how we can compute means across categories, compute a General Linear Model (ANOVA, regression), or plot the data using the {survey} package. You can in later weeks look back to this exercise to get inspiration for

1. First, explore the Survey package by using the {help} function.

```
help(package="sampling")
```

2. The idea of the {survey} package is that you "tell" R what sampling design was used to generate a sample. The survey package will then compute accurate statistics for you, no matter how complicated your design is. There are there always two steps in analyses you do with the survey package:

i. You tell R the structure of your sample using the {svydesign} function
ii. You do some analysis

This week we will stick with a simple random sample. You can tell this to R using the function:

```
library(survey)
SomeName <- svydesign(ids=~1,fpc=NULL,data=srssample1)
```

```
## Warning in svydesign.default(ids = ~1, fpc = NULL, data = srssample1): No
## weights or probabilities supplied, assuming equal probability
```

The warning you get is normal and can be ignored. R "warns" us that our sample looks overly simple.

The elements in the function mean the following: ids=~1. Here, you tell R there is no id variable. If you have an id-variable for people in your dataset, you can define this here. Later, the {ids=~} important when we use cluster sampling

fpc=... Here you need to specify the size of the population that your sample was drawn from, so that R knows how to apply the Finite Population Correction. We left this out for now (NULL), so we have assumed that the population is of infinte size (or, we do not apply fpc.)

data=... The original sample dataset.

We would however like to use fpc correction. We have to add the population size as a new variable to your dataset, and add that to the svydesign function.

3. In step 2 of the analysis you compute the thing you are interested in. In this exercise we are interested in the mean. The code below shows how to ask for a mean. Run the code, and compare your answer to the mean and standard error you computed in exercise 1. They should be the same (note the survey package rounds to 4 decimals).

```
svymean(~age,srs1)
```

4. In the code below, I will show you some other useful functions of the {survey} package that we will use in later package. Feel free to explore these functions

```r
?svyby
svyby(~age,~town, srs1,svymean) # as a bonus, this shows the age distribution across towns
```

```
##   town       age        se
## 1    1  8.283660 0.8484737
## 2    2 14.429222 1.2639431
## 3    3  9.563339 0.8094334
## 4    4  9.910255 0.7052673
## 5    5  5.838800 1.3069396
```

```r
svyglm(hgt ~ age,srs1)
```

```
## Independent Sampling design
## svydesign(ids = ~1, fpc = srssample1$popsize, data = srssample1)
##
## Call:  svyglm(formula = hgt ~ age, design = srs1)
##
## Coefficients:
## (Intercept)          age
##      69.384        6.713
##
## Degrees of Freedom: 192 Total (i.e. Null);  191 Residual
##   (7 observations deleted due to missingness)
## Null Deviance:       449000
## Residual Deviance: 20540     AIC: 1455
```

```r
svyglm(wgt ~ town+age+town:age,srs1) # formula can be more complicated, e.g. with interaction effects
```

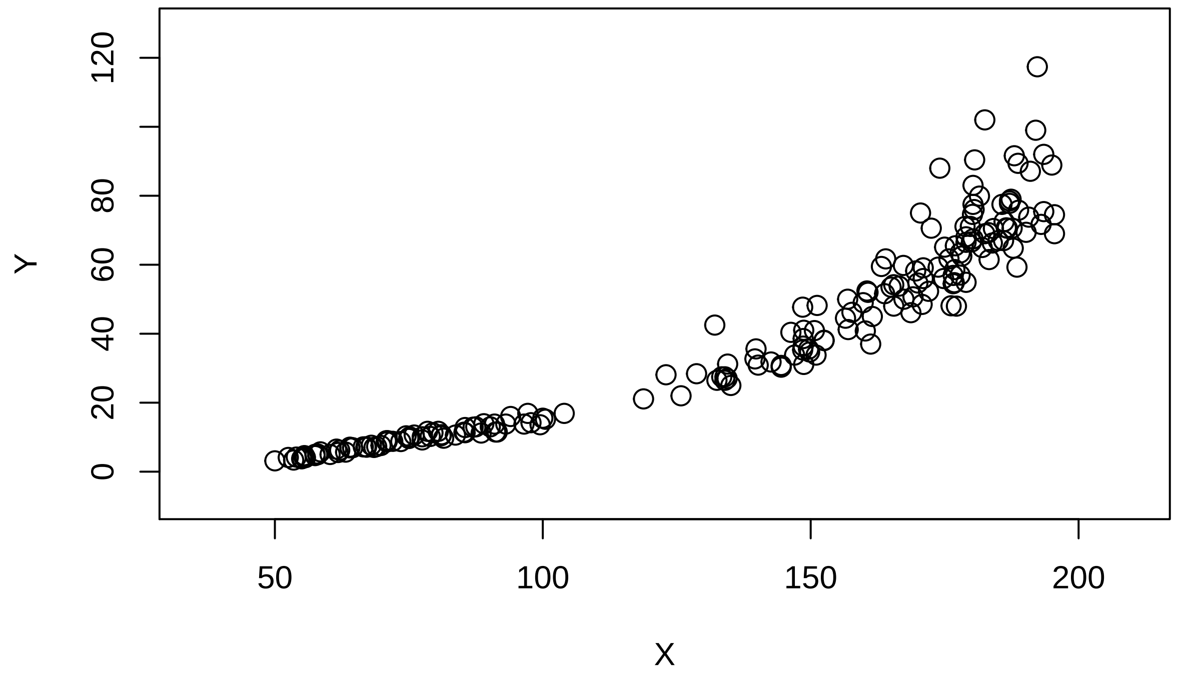
```
## Independent Sampling design
## svydesign(ids = ~1, fpc = srssample1$popsize, data = srssample1)
##
## Call:  svyglm(formula = wgt ~ town + age + town:age, design = srs1)
##
## Coefficients:
## (Intercept)         town          age     town:age
##     3.92437     -0.05530      3.67803      0.02029
##
## Degrees of Freedom: 197 Total (i.e. Null);  194 Residual
##   (2 observations deleted due to missingness)
## Null Deviance:       152000
## Residual Deviance: 16000     AIC: 1442
```

```r
glm1 <- svyglm(wgt ~ town+age+town:age,srs1)
summary(glm1) # and we can get standard errors
```

```
##
## Call:
## svyglm(formula = wgt ~ town + age + town:age, design = srs1)
##
## Survey design:
## svydesign(ids = ~1, fpc = srssample1$popsize, data = srssample1)
```

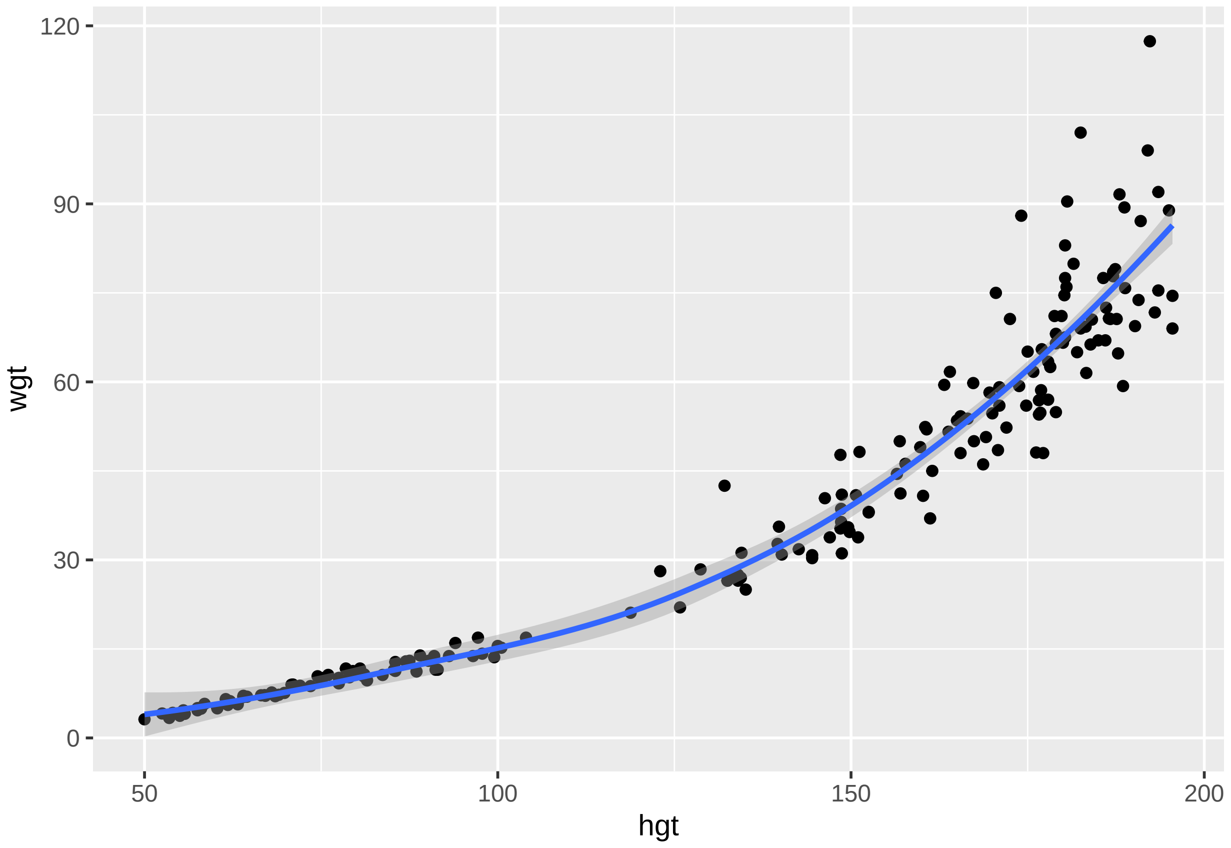
```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.92437    0.73892   5.311 2.97e-07 ***
## town        -0.05530    0.24558  -0.225    0.822
## age          3.67803    0.17298  21.263  < 2e-16 ***
## town:age     0.02029    0.05807   0.349    0.727
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 81.22454)
##
## Number of Fisher Scoring iterations: 2
```

```r
# and plot the data using the survey package
# svyplot(formula, design, style = c("bubble", "hex", "grayhex","subsample","transparent"),
svyplot(wgt~hgt, design=srs1, style=c("bubble"))
```



```r
# try to play with the plot styles!

# the plotting capabilities of the {survey} package are a bit limited. Therefore, you can also resort t
# ggplot2. However, functions like geom_smooth (below) will not be entirely correct.

# compare with non-survey plot GGplot (note that s.e. may be wrong if you use something like geom_smoot
ggplot(data=subset(boys,sample==1),aes(y=wgt,x=hgt))+
    geom_point()+
    geom_smooth(method="loess")
```

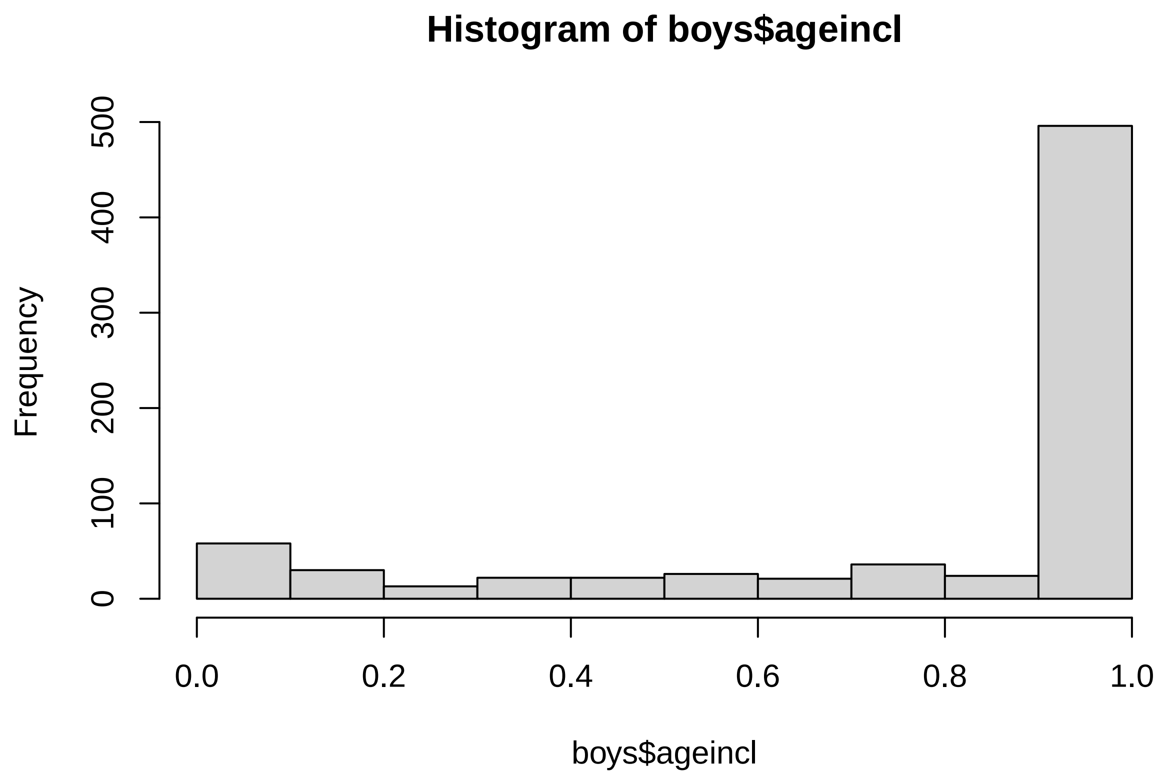
```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 7 rows containing non-finite values (stat_smooth).

## Warning: Removed 7 rows containing missing values (geom_point).
```



bonus only do this if you feel like it

5. There are many cases where you in practice want to sample with unequal probabilities. This is now no longer a Simple Random Sample! Consider a case where we want to give older kids a higher probability to be included into the sample. In the case of our dataset, there will perhaps by more variation in how boys grow during puberty, and for that reason, we may want to 'oversample' older boys. We can do this formally using stratification, which we will discuss next week. Here we will for now use the cumulative probability-distribution of the variable age (pdf) to different inclusion probabilities.

```
# add the pdf to the population using a logistic function
boys$ageincl <- (1 / (1 + exp(-boys$age)))*2-1
hist(boys$ageincl) # plot it
```



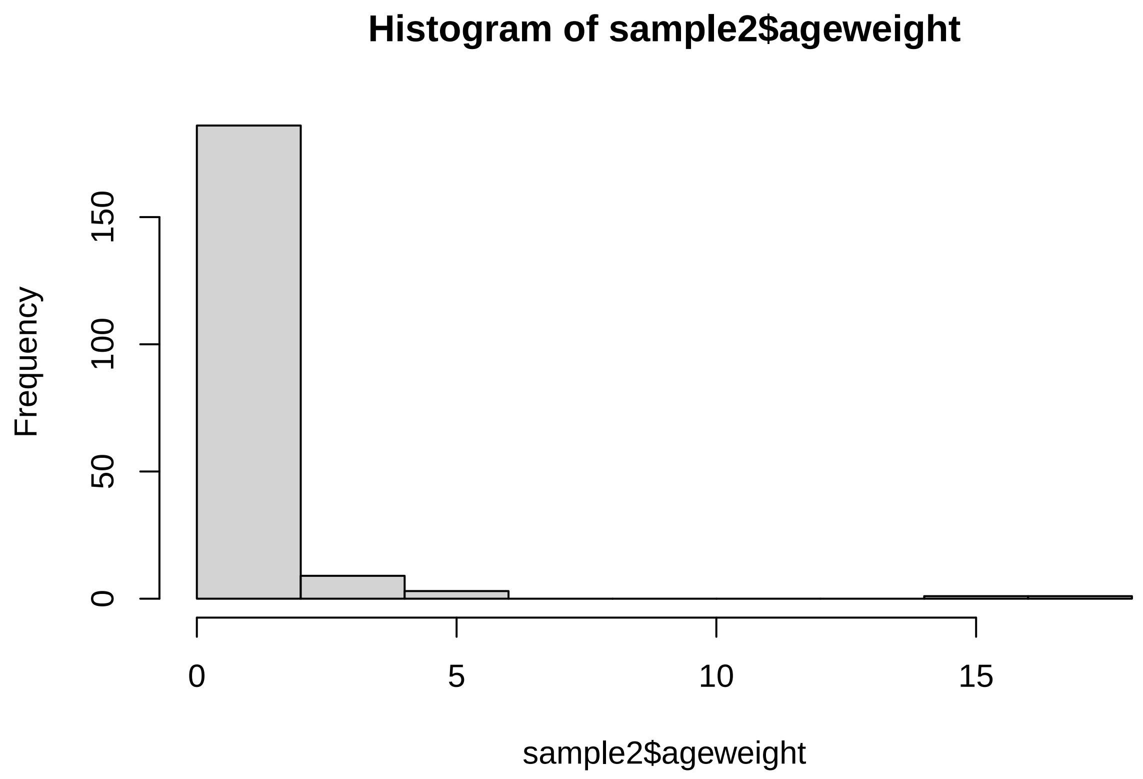
```
# now sample using the sample function from base r
set.seed=123 #for replication purposes
# adding an Id varianle (not strickly needed)
boys$id <- 1:nrow(boys)
# the code below samples proportional to the probability distribution
# using the {sample} function from base R
sample2 <- boys[sample(nrow(boys),200, prob=boys$ageincl),]

# The mean in our sample 2 is now biased! (why?)
mean(sample2$age) # way too high as we want
```

```
## [1] 11.29861
```

```
# we now need to correct for the differences in inclusion probabilities!
# we can do this in the survey package by including a {weight} variable that is the inverse
# of the inclusuon probabilities

# compute the inverse of ageweight to use as weight
sample2$ageweight <- 1/sample2$ageincl
# specify the Svydesign object
rsdesign2 <- svydesign(ids=~1, probs=NULL,strata=NULL,variables=NULL, fpc=sample2$popsize
                          ,weights=~ageweight,data=sample2) # notice the large error here
svymean(~age,rsdesign2)
```

```
##        mean     SE
## age 8.6517 0.8884
```

```
# Notice the increase in the se! We oversample a lot!
hist(sample2$ageweight)
```



```
# we may use trimming to reduce variance inflation here.
sample2$ageweighttrim <- sample2$ageweight
sample2$ageweighttrim[sample2$ageweight>4] <- 4
rsdesign3 <- svydesign(ids=~1, probs=NULL,strata=NULL,variables=NULL, fpc=sample2$popsize
                           ,weights=~ageweighttrim,data=sample2) # notice the large error here
svymean(~age,rsdesign3)
```

```
##       mean     SE
## age 9.5627 0.6037
```

Exercise 3 - sample size calculations, working with se

Imagine you are doing a survey to predict voting behavior in the upcoming Dutch elections (in March 2021). The Dutch electoral system is fully proportional, meaning that there is no threshold to acquire a seat in parliament. There are 150 seats, so whenever you reach 0.66% or a multiple thereof, you get a seat in parliament. For more info on the Dutch electoral system, see https://en.wikipedia.org/wiki/Elections_in_the_Netherlands

In this exercise, we will assume that you can do a Simple Random Sample of all Dutch people who are 18 years and older (the eligible voting population). In practice, it is hard to do an SRS because both telephone and e-mail lists are of poor quality (or don't exist), but we will return to this issue in a later week.

The data are a bit artificial, but resemble how the different Dutch parties polled in August 2020.

The polling data show the following in terms of the % voting for each party: | |VVD | 21% | liberal-conservatives | |PVV | 15% | anti-establishment right | |D66 | 9% | liberals | |CDA | 9% | christian democrats | |PvdA | 10% | social democrats | |SP | 8% | socialists | |GL | 8% | greens | |FvD | 7 % | anti-establishment right | |PvdD | 4% | animal rights party | |others | 9% | small christian parties, elderly, immigrant parties | |

1. The Table just shows the point estimates. What would be the Confidence Interval for the percentage of people voting for the VVD (liberal-right) in the population if this sample was based on 1500 respondents? Use the formula for the standard error for a proportion (see also Stuart p. 32) SEp = sqrt [p(1 - p) / n] and use = .05
2. And what about if the sample would be larger (n=2500) or smaller (n=700)? Compute the se again
3. The Netherlands has many smaller parties which usually only get a few seats. Please compute the confidence interval for the percentage of people voting for PvdD (animal rights party) who are polling at 4%. Please compare the width of the confidence interval to that of the VVD. What do you notice?
4. Imagine that the PvdA would ask you to conduct a study for them. They don't like the idea that estimates from samples are uncertain, and ask you to determine how large a sample should be so that the confidence interval is at most 4 seats (2.6%). Can you work out what the sample size should for a sample?
5. After hearing your estimate, there is someone within the PvdA who has taken a statistics course, and has picked up that if you would use a different value for alpha the sample size would be different.

A) What would the sample size be if you would use an alpha of .10?

B) And an alpha of .01?

6. Another way of expressing uncertainty is by using the margin of error, or the coefficient of variation. What would the required sample size need to be if the PvdA would stick to using an of .05, but would ask you to instead of a confidence interval, use a margin of error of +- 1%,
7. Forming a government is a complicated matter that usually takes a lot of time and negotiations. Imagine the VVD would prefer to govern with the following three parties: CDA (14 seats), FvD (10 seats) and PVV (23 seats). Together the point estimate for the coalition would be 78, enough to form a government. Can you calculate the probability that such a coalition would indeed be able to achieve at least 76 seats? For now, assume that the votes for this bloc are independent of votes for another party.