The estimated vaccine efficacies from the phase III clinical trials provide good information of how strongly a Covid-19 vaccine reduces the probability of a symptomatic Covid-19 infection in the vaccinated individual, e. g. 95% for the Biontech/Pfizer vaccine. However, the probability that a vaccinated person still becomes infected without symptoms and can infect other non-vaccinated persons seems still largely unknown. That is an important number e.g. for deciding whether it is still legally defensible to impose the same shut-down restrictions on vaccinated persons as on others.

Well, I don't know how to estimate this number with the publicly available data. Yet being curious, I asked myself how one *could* in principle estimate this number if one had data from a mobile phone Corona app that tracks past contacts and would also store whether a contact was vaccinated, and if not, was likely infected with the Corona virus at the time of the meeting (which may be calculated from later arriving test results). I did not check the existing literature, which may have much better approaches, but just rearranged some mathematical expressions and run the resulting estimation approach on simulated data.

Let's first have a glimpse at the simulated data set:

```
set.seed(1)
dat = simulate.meeting.data(init.sick.share=0.3,infect.prob = 0.1,
                            vipe=0.7,n=500000,m=1000000,
                            prob.unobserved = 0.1)
head(dat)
```

subjectid	infected	m	ms	mv
1	0	6	1	3
2	0	3	0	1
3	0	5	1	1
4	0	2	0	1
5	0	3	2	0
6	0	3	0	1

The dummy variable `infected` indicates whether during a considered period the subject got infected with Covid-19. Every row corresponds to an unvaccinated subject. The variable `m` shall be the number of recorded relevant meetings before the infection. The number `ms` indicates how many of those meetings were with an unvaccinated person that had Covid-19. The variable `mv` indicates the number of meetings with a person that was vaccinated. For the met vaccinated individuals we don't observe whether or not they had a Corona infection without symptoms.

The code for the `simulate.meeting.data` function is given at the end of this post. Here is just a short description of the specified parameters (they are not calibrated to the real world):

- `init.sick.share=0.3` means that 30% of the initial population are infected with Corona. For simplicity we assume that the vaccinated population has the same probability to be infected, but they typically have no symptoms. You could also interpret an infected vaccinated person as a person that would be infected if she had not gotten a vaccination.
- `infect.prob` (10%) is the probability that an unvaccinated infected person infects a healthy unvaccinated person given that they meet each other.

- `vipe` (70%) shall stand for the `vaccine's infectiousness prevention efficacy`. This is the number, we want to estimate. I do not know the proper name for it so I just invented a name and like the abbreviation VIPE. We assume that an infected vaccinated person infects in a meeting another person only with probability `infect.prob*(1-vipe)`.
- `n` and `m` are the number of subjects and number of simulated meetings.
- `prob.unobserved` (10%) is the probability that a meeting is not observed in our data set. Unobserved meetings may still lead to infections.

Let us start with some theoretical thoughts that will help us to estimate `vipe`. Using shorter mathematical notation, let

$$\pi_s = \text{infect.prob}$$

be the probability that an unvaccinated infectious person infects another unvaccinated person given that there is a meeting. Let

$$\pi_v = \text{infect.prob*(1-vipe)*init.sick.share}$$

be the probability that a vaccinated person (π_v) who may have or may not have a symptomless corona infection infects an unvaccinated person given that they meet.

The probability that after the recorded number of meetings m^s_i and m^v_i an initially healthy person i gets infected shall be given by:

$$Pr(\text{i infected})=1-(1-\pi_v)^{m^v_i} \cdot (1-\pi_s)^{m^s_i} \cdot U_i$$

That is 1 minus the probability that no meeting leads to an infection. The term U_i shall stand for the probability of infections from meetings that are unobserved in our data set. We also assume that the infection probabilities of all meetings are independent from each other.

The logarithm of the probability that an initially healthy vaccinated person i does not become infected by any meeting therefore is

$$\log Pr(\text{i not infected}) = \log(1-\pi_v) \cdot m^v_i + \log(1-\pi_s) \cdot {m^s_i} + \log U_i$$

Let us rename the terms on the right hand side as following:

$$\beta_1 = \log(1-\pi_v)$$ $$\beta_2 = \log(1-\pi_s)$$

where β_1 and β_2 are coefficients that we will estimate. We also rewrite

$$\log U_i = \beta_0 + u_i$$

where β_0 is the average of the log-probability to get infected from unobserved meetings and u_i is an unobservable deviation for each individual.

We can then write:

$$\log Pr(\text{i not infected}) = \beta_0 + \beta_1 \cdot m^v_i + \beta_2 \cdot {m^s_i} + u_i$$

The right hand side looks like that of a nice linear regression equation. Unfortunately, the left hand side contains a probability that we don't directly observe. One route to proceed could be to assume a distribution for the u_i (or perhaps even just set it equal to 0) derive the log-likelihood function and then compute the Maximum Likelihood estimator.

Yet, I choose another route: I aggregate the individual-level data and then run a weighted least squares regression.

```
agg = dat %>%
  group_by(mv, ms) %>%
  summarize(
    n = n(),
    prob.infected = mean(infected),
    y = log(1-prob.infected)
  ) %>%
  ungroup()
head(agg)
```

mv	ms	n	prob.infected	y
0	0	46300	0.01	-0.01
0	1	42400	0.11	-0.11
0	2	15900	0.19	-0.21
0	3	3960	0.28	-0.33
0	4	747	0.33	-0.4
0	5	117	0.48	-0.65

The data frame `agg` contains one row for each combination of meeting numbers `mv` and `ms` and we computed in `prob.infected` the average share of infected persons in each such cell. We then use the variable `y=log(1-prob.infected)` as an estimator for the left hand side in our regression specification:

\[y = \beta_0 + \beta_1 \cdot m^v_i + \beta_2 \cdot {m^s_i} + u_i\]

We now estimate this regression using the aggregated data frame `agg` with the numbers of subjects `n` in each cell as weights:

```
agg = filter(agg, is.finite(y))
reg = lm(y~mv+ms, data=agg, weights=agg$n)
beta = coef(reg)
beta
```

```
##  (Intercept)           mv           ms
## -0.008729151 -0.009242340 -0.104527982
```

The estimated coefficients themselves are not very informative. We will now convert them to the variables of interest:

```
names(beta) = NULL
pi.v = 1-exp(beta[2])
pi.s = 1-exp(beta[3])
init.sick.share = 0.3
vipe = 1-((pi.v/ init.sick.share) / pi.s)
c(pi.s = pi.s, pi.v = pi.v, vipe = vipe)
```

```
##        pi.s        pi.v        vipe
## 0.099250407 0.009199761 0.691025251
```

We estimate a 9.9% probability to get infected when meeting an infected unvaccinated person. This is close to our assumed parameter `infect.prob` of 10%. When meeting a vaccinated person, we only estimate a 0.9% probability to get infected. Note that our data does not distinguish between vaccinated persons with an unobservable infection and non-infected ones. To estimate the vaccine infection prevention efficacy (VIPE), we must therefore must adjust for the share of initially infected persons. We get a VIPE estimate of 69.1%, which is not too far off the true value of 70%. Correct standard errors could (hopefully) be obtained via bootstrapping. I repeated the simulation a 100 times and find a mean estimate for VIPE of 0.7002 with a standard deviation of 0.02. So, at least with the simulated data, this estimation approach seems to work.

In a real world data set, one would have to think about possible sources of bias, however. For example, the behavior and safety procedures may systematically differ depending on whether a person is vaccinated or not. It could also be the case that the number unobserved meetings is correlated with individual characteristics that are also correlated with the explanatory variables, yielding a classic endogeneity problem in our regression.

Most importantly, I don't know if there exists a country that collects such data, would allow to analyze it and at the same time has sufficiently many Covid-19 cases for a sufficiently precise estimation. But even if the presented regression may never be run on real data, I find it helpful to think about possible statistical approaches to estimate such important measures. For example, it may help to better weigh the costs-and-benefits of stronger data protection vs valuable insights from more data collection.

Overall, I am curious whether we will learn about how infectious vaccinated persons are and which data set and empirical approach corresponding studies will take…

COMMENTS

Appendix: Simulation Code

```
simulate.meeting.data = function(
  init.sick.share = 0.2, # share of initially sick people
  infect.prob = 0.1, # probability
  vipe = 0.7, # vaccine infection prevention efficacy
  vacc.share = 0.3, # share of vaccinated people
  prob.unobserved = 0, # probability that a meeting is unobserved
  n = 100000, # number of persons
  m = 100000 # number of meetings
) {
  # Draw persons that meet
  m1 = sample.int(n,2*m, replace=TRUE)
  m2 = sample.int(n,2*m, replace=TRUE)

  # Only keep first m meetings
  # where different persons meet
  keep = m1 != m2
  m1 = m1[keep][1:m]
  m2 = m2[keep][1:m]

  # Which persons are vaccinated
  vacc = (runif(n) <= vacc.share)
```

```
# Which persons are initially infected
# for vaccinated persons that may be the hypothetical
# infection if they would not be vaccinated
start.sick = (runif(n) <= init.sick.share)

# Simulate which meetings would transfer an infection
infect.dice = runif(m)
infect1.threshold = infect.prob*ifelse(vacc[m2], 1-vipe,1)
infect2.threshold = infect.prob*ifelse(vacc[m1], 1-vipe,1)

infected1 = (infect.dice <= infect1.threshold) & start.sick[m2]
infected2 = (infect.dice <= infect2.threshold) & start.sick[m1]

# Which subjects are sick after all infections
post.sick1 = m1[infected1]
post.sick2 = m2[infected2]
post.sick = start.sick
post.sick[c(post.sick1,post.sick2)] = TRUE

# Is the meeting recorded / observed
record = runif(m) > prob.unobserved

# in every meeting let s be sender
# and r be the receiver of a potential
# infection
# use dtplyr for speed
library(dtplyr)
dat = bind_rows(
  tibble(s = m1, r = m2, record=record),
  tibble(s = m2, r = m1, record=record)
  ) %>%
  filter(record) %>%
  mutate(
    start.sick.s = start.sick[s],
    start.sick.r = start.sick[r],
    post.sick.s = post.sick[s],
    post.sick.r = post.sick[r],
    infected = !start.sick.r & post.sick.r,
    vacc.s = vacc[s],
    vacc.r = vacc[r]
  ) %>%
  # only keep people that are initially sick
  filter(!start.sick.r) %>%
  # aggregate meeting data on person level
  lazy_dt() %>%
  group_by(r) %>%
  summarize(
    infected = 1L*first(infected),
    m = n(),
    ms = sum(start.sick.s & !vacc.s),
    mv = sum(vacc.s)
  ) %>%
```

```
    rename(subjectid = r) %>%
    as_tibble()
  dat
}…
```