UNIVERSITY OF COPENHAGEN

Københavns Universitet

Design og analyse af eksperimenter i R

Dahl, Malte

Publication date:
2017

Citation for published version (APA):
Dahl, M. R. (2017). Design og analyse af eksperimenter i R.

Download date: 26. Mar. 2019

Design og analyse af eksperimenter i R

Malte Dahl

8 dec 2017

Introduktion

Hvordan designer vi transparente og reproducerbare eksperimenter med tilstrækkelig power? Hvordan vrider vi mest mulig efficiens ud af dem? Og hvordan kan vi anvende selve randomiseringen som afsæt for inferens? Denne note introducerer et udsnit af metoder til at designe og analysere eksperimentelle studier med afsæt i statistikprogrammet R. Det er håbet, at notatet vil være en hjælp til at reflektere over, designe, og analysere eksperimenter i forbindelse med bachelorprojekter eller specialer – og forhåbentligt også i arbejde med eksperimenter uden for instituttets mure. Notatet berører selvsagt kun en lille del af den omfattende litteratur, men for så vidt mulig henvises til videre læsning, særligt til relevante kapitler i Gerber & Green (2012).

Notatets indhold kan inddeles i tre. I første del introduceres metoder til at maksimere efficiens og transparens i designfasen, herunder behandles simulering af data, poweranalyse, komplet randomisering og blokrandomisering. I notatets anden del introduceres designbaseret inferens – såkaldt randomiseringsinferens – som framework til at analysere eksperimentelle data. I tredje del uddybes eksempler på analyser af eksperimentelle data, herunder balancetest, hypotesetest med og uden kovariatjustering, interaktioner samt udledning af konfidensintervaller. Det antages, at læseren har et grundlæggende kendskab til R samt til *potentiel outcomes*-frameworket. De indledende kapitler i Gerber & Green (2012) eller Imbens & Athey (2017) er gode steder at starte for en introduktion eller genopfriskning.

Pakker

For at køre eksemplerne er det nødvendigt at installere følgende pakker:

```
install.packages("randomizr")
install.packages("lmtest")
install.packages("sandwich")
```

1. Design

1.1 Simulér data inden dataindsamlingen

Hvor mange enheder skal samples? Det er et fundamentalt valg i designfasen som skal kvalificeres og træffes inden dataindsamlingen påbegyndes. Effekter i samfundsvidenskaben er ofte små og det stiller krav til samplestørrelsen. Konsekvenserne af et utilstrækkelig sample illustreres i følgende simple eksempel:

Vi tildeler 300 eksperimentelle enheder et potentielt outcome under kontrol, Y(0), med et gennemsnit på 6 og en standardafvigelse på 0,5. Samme respondenter tildeles et potentielt outcome under treatment, Y(1), givet ved Y(0) + 0,15. Med andre ord er der for alle enheder i samplet en konstant treatmenteffekt på 0,15.

```
#Sætter et seed så resultaterne er identiske hver gang koden eksekveres
set.seed(123)

#Tildeler outcomes under kontrol (Y0) og treatment (Y1)
Y0 <-  rnorm(n=300, mean=6, sd=.5) # Potentialle outcomes for 300 individer i kontrol
effekt <- 0.15                     # Et bud på treatment effekt
Y1 <- Y0 + effekt                  # Potentialle outcomes for 300 individer som treates
```

På den baggrund kan vi simulere et eksperiment. Vi randomiserer en treatmentindikator, Z, så vi for alle enheder observerer ét outcome (outcome i treatment eller outcome i kontrol).

```
# Randomisering
Z <- rbinom(n=300, size=1, prob=.5)

# Fordeler outcomes ift. tildelingen af treatment
Y.sim <- Y1*Z + Y0*(1-Z)

#Kører analysen
fit <- lm(Y.sim ~ Z)

#Forskellen i gennemsnit ml. treat og kontrol:
summary(fit)

  ## Effekt: 0.088
  ## p: 0.107
```

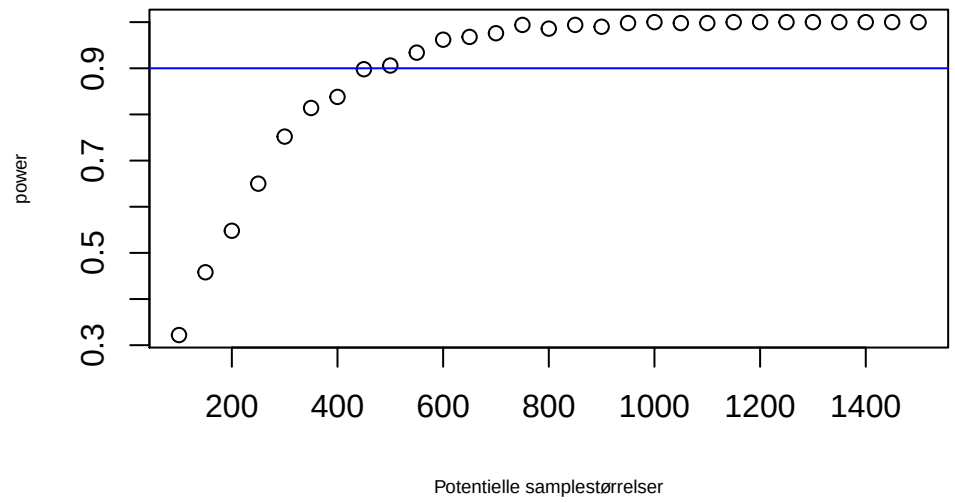
Effekten af treatment i dette eksperiment er 0.088 og borderline signifikant på et 10 pct. signifikansniveau. Det er et "uheldigt" træk i den forstand, at effekten falder ganske langt fra hele samplets treatmenteffekt på 0,15. Rydder vi hukommelsen og kører eksemplet flere gange uden *seed* vil effektestimaterne variere og i de fleste tilfælde falde tættere på 0,15. Men i dette tilfælde afviser vi altså fejlagtigt at treatment har en statistisk signifikant effekt. Eksemplet illustrerer, hvordan der med små samples følger stor usikkerhed - og dermed risiko for Type II-fejl.

Vi ønsker i udgangspunktet at sample så mange enheder som muligt, men omvendt kan det være dyrt, besværligt og tidskrævende. I visse tilfælde pålægges de eksperimentelle enheder samtidig en byrde og surveyrespondenter er i øvrigt ikke en uendelig ressource. Vi bør derfor kvalificere beslutningen om samplets størrelse ved at tænke over eksperimentets power.[1] Beslutninger vedrørende samplestørrelse handler også om troværdighed. Som vi vil se senere, er det problematisk at lade resultaterne guide hvornår dataindsamlingen er tilstrækkelig. Med afsæt i et kvalificeret gæt på effektstørrelse og varians, eksempelvis på baggrund af et pilotstudie eller tidligere studier, kan vi simulere et datasæt, som vi forventer det vil komme til at se ud.

[1] Power er sandsynligheden for at vi kan afvise nul-hypotesen. Konventionen tilsiger et powerniveau på 80%, men det er en arbitrær grænse. At 20% af vores eksperimenter skal falde ud som insignifikante kan forekomme unødigt højt. I Figur 1 angives en grænse på 90%.

Hvis vi, som i ovenstående eksempel, har en antaget effekt på 0,15 og en standardafvigelse på 0,5, kan vi estimere den ideelle samplestørrelse ved at undersøge hvordan p-værdierne fra et stort antal eksperimenter afhænger af samplestørrelse. Ved at bruge loop-funktionen øges samplestørrelsen i dette tilfælde gradvist i intervaller af 50 fra et udgangspunkt på 100 til og med 1500 respondenter. For hver samplestørrelse trækker vi 500 nye eksperimenter. Samples med en p-værdi under 0.05 tæller som et positivt outcome. Figur 1 illustrerer sammenhængen mellem samplestørrelse og power.

Figur 1. Powerberegning



Som Figur 1 viser, skal vi – givet den forudsagte effektstørrelse og varians – sample omkring 450 enheder for at vi i 90% af eksperimenterne vil afvise H0. Hvis vi derimod nøjes med at sample 100 enheder, vil kun 30% af eksperimenterne resultere i p-værdier under 0.05. Vi kan samtidig se, at gevinsten ved at sample mere end 650 enheder er begrænset: vi vil være nærmest fuldsændig sikre på at kunne afvise H0 med et sample > 650.

Eksempel på powerberegning

```
#Angiver samplestørrelser, antal samples samt aplhaniveau:
potentielle_samples <- seq(from=100, to=1500, by=50)
power <- rep(NA, length(potentielle_samples))  #Tomt objekt til at samle estimater
alpha <- 0.05                                  #Konventionelt signifikansniveau på 5%
sims <- 500                                    #500 simulationer for hvert sample

# Opstiller et ydre loop
for (j in 1:length(potentielle_samples)){
  N <- potentielle_samples[j]                  #Definerer varierende samplestørrelser

  signifikante_eksperimenter <- rep(NA, sims)  #Tomt objekt tæller signifikante samples

#Indre loop som angiver, at vi gennemfører eksperimentet 500 gange for hvert sample (N)
for (i in 1:sims){
    Y0 <-  rnorm(n=N, mean=6, sd=0.5)          #Potentialle outcomes under kontrol
    effekt <- 0.15                             #Et bud på treatment effekt
    Y1 <- Y0 + effekt                          #Potentialle outcomes under treat
    Z <- rbinom(n=N, size=1, prob=.5)          #Simpel randomisering
```

```
    Y.sim <- Y1*Z + Y0*(1-Z)                    #Fordel outcomes
    fit <- lm(Y.sim ~ Z)                        #Kør regression
    p.value <- summary(fit)$coefficients[2,4]   #træk p-værdierne
    signifikante_eksperimenter[i] <- (p.value <= alpha)  #Vælg p-værdier =< 0.05
  }

  power[j] <- mean(signifikante_eksperimenter)  #Gem succesraten (p<0.05)
}

##obs. koden tager lidt tid at processe

#plot forholdet mellem samplestørrelse og andel signifikante resultater
plot(potentielle_samples, power)
abline(h=0.9, col="blue")
```

Power er relateret til samplestørrelse, effekt, efficiens og en ideel fordeling af treatment og kontrol.[2] Som vi vil se i de følgende afsnit, har vi en række muligheder for at øge efficiensen og dermed øge power i både designet og analysen, hvilket naturligvis bør inkorporeres i powerbegningen.

1.2 Randomisering

Vi ønsker en randomisering som er fejlfri, reproducerbar og transparent. Dette eksempel anvender pakken randomizr til at gennemføre en komplet randomisering i vores sample (N) som sikrer at et defineret antal enheder (her N/2) modtager treatment. Komplet randomisering er i udgangspunktet altid at foretrække over simpel randomisering.

Eksempler på komplet randomisering

```
set.seed(123) # Vi anvender et seed så koden kan reproduceres.
library(randomizr)
N <-450
 #Komplet randomisering
Z <- complete_ra(N, N/2)

#tjekker fordeling
table(Z)

## Z
##   0   1
## 225 225

#Tilføjer treatment-indikatoren, Z, til en dataframe
df <- data.frame(Z)
```

Vi kan tilføje flere treatments, navngive grupperne samt specificere fordelingen af enheder:

```
set.seed(123)
library(randomizr)
N<-450
Z <- complete_ra(N = N, m_each = c(100, 200, 150),
                 condition_names=c("kontrol", "treat_1", "treat_2"))
```

[2] Hvis variansen er forskellig på tværs af de eksperimentelle grupper, er det en fordel at tildele et større N til gruppen med størst varians.

```r
#Tjekker fordeling
table(Z)

## Z
##  kontrol  treat_1  treat_2
##      100      200      150
```

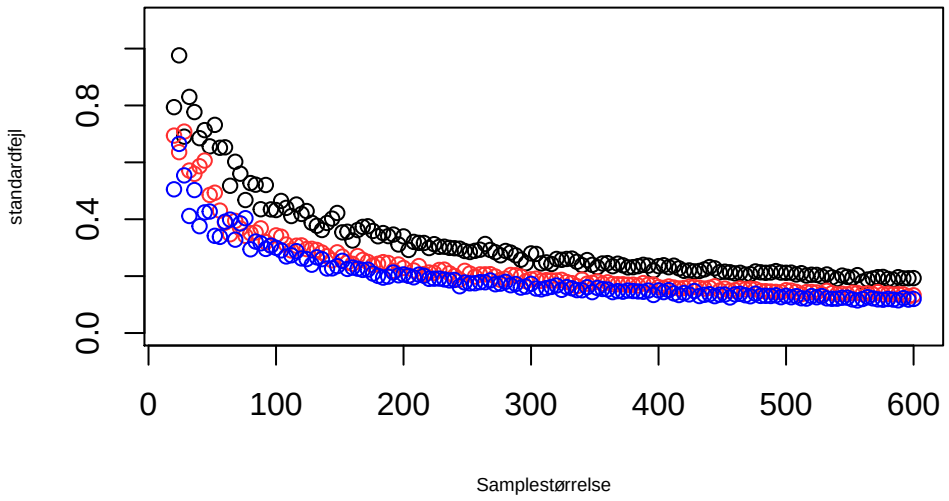
1.3 Blokrandomisering

Vi kan øge efficiensen ved at stratificere randomiseringen inden for blokke som korrelerer med outcome og gennemføre en komplet randomisering i hver blok.[3]

Blokrandomisering har flere fordele. Hvis mænd eksempelvis gennemsnitligt scorer højere på den afhængige variabel sammenlignet med kvinder, vil vi for det første gerne undgå, at treatment og køn tilfældigvis korrelerer. Ved at blokke på køn undgår vi en overvægt at mænd (eller kvinder) i treatmentgruppen i forhold til kontrolgruppen. Det øger efficiensen, særligt i små samples. For det andet kan vi minimere uforklaret varians ved at justere for køn i analysen - og det kan vi gøre med god samvittighed idet vi på forhånd har angivet, at netop den kovariat er vigtig. Udover øget efficiens bidrager blokrandomisering altså potentielt til øget gennemsigtighed.

Figur 2 illustrerer denne pointe ved at gentage et eksperiment ved forskellige samplestørrelser hhv. uden og med blocking samt med blocking og justering. X-aksen angiver samplestørrelser, mens Y-aksen angiver størrelsen på standardfejlene for effektestimaterne. De sorte datapunkter indikerer eksperimenter uden blocking mens de røde illustrerer eksperimenter med blocking på fire kovariater som korrelerer med outcome. Alene ved at fjerne risikoen for "skæve" fordelinger af kovariaterne på tværs af treatment- og kontrolgruppen opnår vi større efficiens. Endeligt er de blå datapunkter blockede eksperimenter som samtidig justerer for blokvariablen, hvilket bidrager yderligere til at øge efficiensen. [4]

Figur 2. Efficiens afhængigt af design og samplestørrelse



[3]Simpel randomisering inden for blokke er meningsløst - blokrandomisering betyder nødvendigvis en komplet randomisering i hver blok

[4]Den ekstra efficiens vi kan opnå med blocking kan i større samples vindes ved kovariatjustering, men blocking giver gennemsigtighed idet valget om blokke er truffet på forhånd.

Det er værd at bemærke, at gevinsterne ved kovariatjusteringen forudsætter at kovariaterne korrelerer med outcome. Hvis det ikke er tilfældet - eller de blot korrelerer svagt - kan inklusion af kovariater potentielt reducere power ved at mindske antallet af frihedsgrader.

Eksemplet herunder viser, hvordan vi kan gennemføre en blockrandomisering på to kovariater

Eksempel på blockrandomisering

```
library(randomizr)
set.seed(123)

#Simulerer eksperiment
N<-350
Y0 <-  rnorm(n=N, mean=6, sd=0.5)
effekt <- 0.15
Y1 <- Y0 + effekt

#Tilføjer kovariater (Køn og sektor)
Kon <- sample(c("Kvinde", "Mand"), N, replace = TRUE)
Sektor <- sample(c("Offentlig","Privat"), N, replace = TRUE)

#samler i data frame
df_block <- data.frame(Y0, Y1, Kon, Sektor)

#Definerer blokvariabel (kombinationer af køn+sektor)
block_var <- with(df_block, paste(Kon, Sektor, sep = "_"))

#Udfører blokrandomisering
 df_block$Treat <- block_ra(block_var = block_var,
                            condition_names = c("Kontrol", "treat1","treat2"))
```

Vær opmærksom på, at blokrandomiserede eksperimenter hvor sandsynligheden for tildelingen af treatment varierer skal analyseres med højde for dette (Gerber & Green, 2012 s. 77). Andre former for randomisering, eksempelvis clusterrandomisering eller designs med varierende sandsynlighed for tildeling af treatment kan ligeledes appliceres i randomizr.

1.4 Analyseplan: Styrk eksperimentets troværdighed

I eksperimentelle (såvel som i observationelle) studier har forskeren en række frihedsgrader som udspænder et mulighedsrum for at fiske efter bemærkelsesværdige effekter.[5] Den videnskabelige proces fra design til analyse er præget af mange valg, og mere eller mindre bevidst, kan man ende med et resultat som i bedste fald er tvivlsomt. En måde at fryse disse valg og dermed øge transparens og troværdighed er ved at nedfælde dem på forhånd i en præ-analyseplan. En præ-analyseplanen angiver eksempelvis samplestørrelsen og studiets hypoteser samt en plan for hvordan data analyseres, herunder hvordan man forholder sig til frafald, manglende data mv. Planen uploades med et datostempel på en hjemmeside og tjener dermed som et bevis på at forskeren følger de retningslinjer som hele tiden har været udstukket. Samtidig bør information om det indsamlede data samt kode til at gennemføre analyserne være tilgængelige og akkompagneres med kommentarer, så andre kan replikere eksperimentet eller reproducere analyserne. Bemærk at analyseplanen ikke er uafravigelig - vi skal blot gøre opmærksom på uoverensstemmelser i afrapporteringen så andre kan vurdere rimeligheden bag eventuelle afvigelser.

[5] Dataindsamlingen kan stoppes tidligere eller senere end planlagt, data kan analyseres på måder der begunstiger muligheden for at finde et interessant resultat, eksempelvis ved at behandle outliers eller manglende data på en bestemt måde, hypoteser kan justeres eller opfindes post-hoc, effekter for diverse subgrupper kan undersøges og en teoretisk interessant vinkel opfindes post hoc etc.

2. Randomiseringsinferens

Givet at de eksperimentelle antagelser er opfyldt, udgør den gennemsnitlige forskel i outcomes mellem treatment- og kontrolgruppen et kausalt estimat af effekten af treatment. Med dette estimat følger en usikkerhed: kan effekten være et resultat af tilfældige forskelle de to grupper imellem? I de indledende metodekurser på statskundskab lærer vi at estimere usikkerhed med afsæt i sample-baseret inferens. Vi antager med andre ord, at vores samples er trukket fra en population og vurderer effektestimatet i forhold til en hypotetisk stikprøvemålsfordeling (eksempelvis en t-fordeling).

Men den typiske nulhypotese i eksperimenter (at treatment ikke påvirker outcome) behøver vi ikke teste pba. en fiktiv stikprøvemålsfordeling. I stedet kan vi bygge en referencedistribution med afsæt i data. Da eksperimenter i samfundsvidenskaben ofte tager afsæt i samples, der ikke er repræsentative for eller tilfældigt udtrukne fra en population kan design-baserede inferensmetoder – såkaldt randomiseringsinferens – være en mere intuitiv tilgang til at anlysere eksperimenter. Ved at undersøge potentielle effekter under en alternativ fordeling af treatments til de eksperimentelle enheder i samplet, anvendes selve randomiseringen som det statistiske afsæt for at vurdere usikkerhed. Som vi vil se, er forskellene mellem randomiseringsinferens og konventionelle metoder begrænsede når samples har en vis størrelse. Ikke desto mindre kan det være en nyttig, og i visse tilfælde mere korrekt, måde at inferere på.

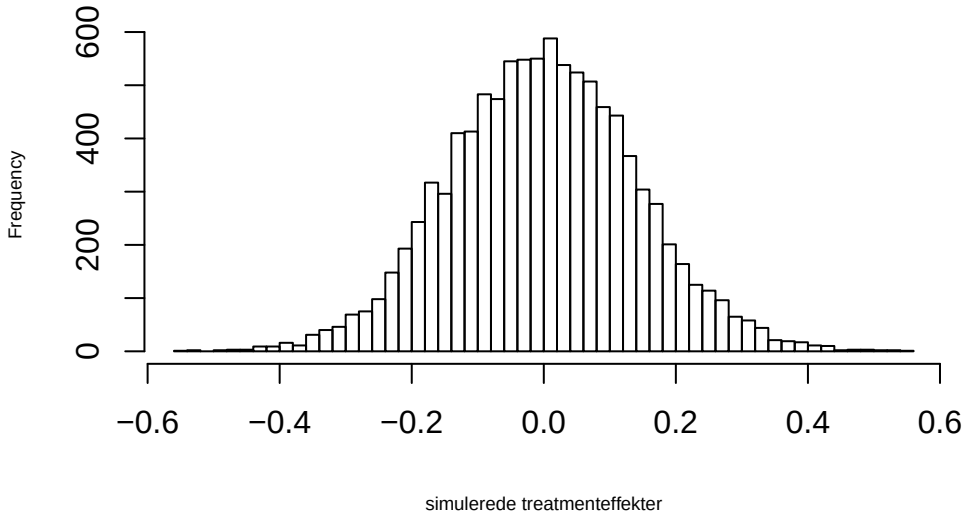
2.1 Intuition

Lad os antage et eksperiment, hvor den afhængige variabel er selvrapporteret vh-placering og treatment er en oversigt over næste uges vejrudsigt. Det er et fjollet eksperiment, hvor treatment har *nul effekt* på respondenternes svar (det antager vi i hvert fald). Fordi treatment ikke har en effekt, vil respondenternes potentielle outcomes være ens: outcomes er de samme under treatment og kontrol. Effekten er således nul for hver eksperimentel enhed. En situation som denne fungerer selvfølgelig kun som et hypotetisk eksempel - i virkelighedens verden kan vi aldrig observere outcomes for både treatent og kontrol samtidig.

Lad os dernæst antage, at en gruppe studerende sætter sig for at gennemføre eksperimentet med afsæt i en hypotese om, at vejrudsigter faktisk influerer vh-placering. De sampler 300 medstuderende og randomiserer tildelingen af treatment (informationen) og kontrol (ingen information). De indsamler svar og sammenligner outcomes for de to grupper og finder, at treatment-gruppen gennemsnitligt scorer 0.06 højere end kontrol-gruppen. Vi *ved*, at der reelt ikke er en effekt af treatment, men deres eksperiment indikerer alligevel en lille effekt. Det rejser tre spørgsmål: 1) Hvorfor finder de en effekt, når der ikke er nogen? 2) Tyder det på, at deres estimat er biased? 3) Hvordan afgør de, om effekten er statistisk signifikant?

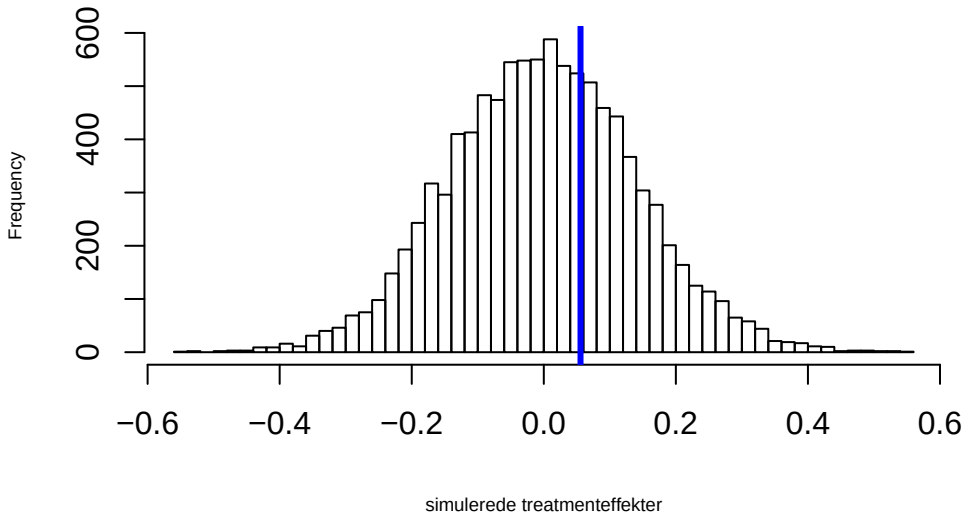
For det første er der stor sandsynlighed for, at deres eksperiment viser en lille effekt i positiv eller negativ retning. Som vi så tidligere, vil effekterne typisk variere en smule fra eksperiment til eksperiment. Det betyder, for det andet, at deres resultat ikke er biased. Hvis de kunne gentage deres eksperiment, men med en anden tildeling af treatment, ville de sandsynligvis få et lidt andet effekestimat. Ved gentagende eksperimenter vil resultaterne centrere sig omkring den sande værdi (i dette tilfælde 0). Det illustreres i figur 3a. som bygger en distribution af effekter fra 10.000 eksperimenter trukket fra det samme data. I hvert eksperiment tildeles treatment og kontrol på ny ved komplet randomisering.

Figur 3a. Distribution af effekter fra 10.000 eksperimenter



Det leder dem direkte til svaret på det tredje spørgsmål: de kan teste hypotesen ved at undersøge om en effekt på 0.06 er usandsynlig ved at sammenholde estimatet med distributionen af effektestimater. Den blå streg i Figur 3b angiver estimatet fra deres eksperiment. Ved at sammenholde deres eksperiment med distributionen af effektestimater *under antagelse af nul-effekt for alle enheder* fremgår det klart, at en effekt på 0.06 ikke er usædvanlig. Dermed kan hypotesen afvises. Givet nul-effekt for alle respondenter* kan effekten sammenholdes med en referencedistribution, der tager afsæt i data. I dette tilfælde vil 6.922 af de 10.000 eksperimenter vise en numerisk effekt på 0.06 eller større. Med andre ord er p-værdien for en tosidet test 0.69 og de kan (korrekt) afvise deres hypotese.

Figur 3b. Sammenligning af estimat og effektdistribution



Fremgangsmåden følger de samme trin som en konventionel hypotesetest. Men i stedet for at *antage* en

normalfordelt stikprøvemålsfordeling, bygges en faktisk fordeling med afsæt i data.[6] Modsat den klassiske hypotesetest som sammneligner en test-statistic mod en antaget stikprøvemålsfordeling, bygges en observeret effektfordeling dannet af et stort antal permuteringer. Bemærk at vi arbejder med en såkaldt *sharp-null* hypotese hvor effekten er 0 for alle enheder. Uden antagelser om en underliggende fordeling af data kan hypotesen testes. Som vi vil se er forskellen på randomiseringsinferens og konventionelle metoder begrænset når N har en vis størrelse, men hvis samples er små kan randomiseringsinferens være mere korrekt (Keele et al., 2012).

[6] I dette tilfælde har vi anvendt forskellen i gennemsnit som *test statistic*, men det kan praktisk talt være alle mulige funktioner af data.

3. Analyse af eksperimentelle data

Dette afsnit giver eksempler på balancetest, hypotesetest, estimerering af konfidensintervaller samt analyse af blockede designs og interaktioner. I alle eksempler analyseres eksemplerne både vha. randomiseringsinferens og OLS.

3.1 Balancetest

Hvis distributionen af en kovariat er ens på tværs af treatment- og kontrolgrupper, er eksperimentet balanceret med hensyn til den givne kovariat. Randomiseringen sikrer forventeligt en sådan balancering – men der er naturligvis altid en risiko for tilfældige ubalancer på observerede (eller uobserverede) kovariater. En anden årsag til ubalancer er fejl i selve randomiseringsprocessen, hvilket selvsagt underminerer eksperimentets validitet. Afhængigt af eksperimentets type kan der være en række udfordringer i randomiseringen, og det kan derfor være fornuftigt at teste balanceringen af treatments på kovariater. Desuden kan det være informativt i forhold til en sammenligning af ens sample og den population man eventuelt ønsker at referere til. Det gør vi med en balancetest, som grundlæggende er en vurdering af, hvorvidt fordelingen af treatment og kontrol er balanceret på tværs af kovariater. Vi ser på kovariaterne samlet i en omnibus-test, eksempelvis en F-test.

Vi tester nul-hypotesen om ingen signifikante ubalancer ved at genererer 10.000 permuteringer og hver gang regressere treatment-indikatoren, Z, på kovariaterne (i dette tilfælde køn, sektor og alder). F-statistikken fra hver af de 10.000 regressioner tjener som en referencedistribution mod hvilken vi kan vurdere eksperimentet. I dette tilfælde er F-statistikken i vores sample større end knap 30% af F-statistikkerne under 0-hypotesen, og der er intet som indikerer at randomiseringen ikke er forløbet korrekt.

Eksempel på balancetest

```
set.seed(123)
library(randomizr)

#Simulerer data med kovariater
N<-500
kon <- sample(c("F","M"), N, replace = TRUE)
offentlig <- sample(c("O","P"), N, replace = TRUE)
alder <- sample(x=18:65, size=N, replace=TRUE)

# Tilføjer effekter
effectofkon <- 0.3
effectofalder <- 0.15
effectofoffentlig <- 0.4
Z_simul <- complete_ra(N, N/2)

#1.Udled f-statistikken fra en regression af treatment på kovariater
fit_sim <- lm(Z_simul ~ kon + offentlig + alder)
F_stat <- summary(fit_sim)$fstatistic[1]
##F-stat = 1,264

#2.Simuler distribution af f-statistikker ved at tildele treatment et stort antal gange
sims <- 10000
Distribution_F <- numeric(sims)

for(i in 1:sims){
  Z_sim <- complete_ra(N, N/2)
  fit_sim <- lm(Z_sim ~ kon + offentlig + alder)
```

```
  f<-summary(fit_sim)$fstatistic[1]

  Distribution_F[i] <- f
  #    as.numeric(Rbeta.hat %*% solve(RVR, Rbeta.hat))
}

#3.Se hvor ofte - under antagelsen om nul-effekt vi får en f-stat. så stor eller større
hist(Distribution_F)
abline(v = F_stat, lwd=3, col="blue")

#3 udled p-værdi
p <- mean((Distribution_F) >= (F_stat))
p
##p: 0.292

#Læg mærke til, at de matcher p-værdien fra OLS-modellen temmelig præcist:
fit_sim <- lm(Z_simul ~ kon + offentlig + alder)
summary(fit_sim)
##F-statistic: 1.265 on 3 and 496 DF,  p-value: 0.2858
```

Hvorvidt en *randomisering har virket* er et misvisende spørgsmål – enten er tildelingen af treatment randomiseret eller også er det ikke. En ubalance kan skyldes et uheldigt træk, hvilket kan håndteres ved tilføje analyser som inkluderer de(n) ubalancerede kovariat(er) – se afsnittet om kovariatjustering. Men en balancetest som viser en p-værdi < 0.01 bør give anledning til en meget grundig gennemgang af randomiseringsprocessen. Hvis der kan stilles spørgsmålsten ved randomiseringen, kan der stilles spørgsmålstegn ved validiteten af resultatet.

3.2 Hypotesetest

Dette eksempel illustrerer hvordan vi kan beregne effekter og estimere usikkerhed. Vi starter med at simulere data:

```
#Simulerer data
set.seed(123)
N<-450                          #Definerer N
Y0 <-  rnorm(n=N, mean=6, sd=0.5) #Potentialle outcomes for N individer i kontrol
effekt <- 0.15                  #Et bud på treatment effect
Y1 <- Y0 + effekt               #Potentialle outcomes for N individer som treates
Z <- complete_ra(N, N/2)        #Komplet randomisering
Y <- Y1*Z + Y0*(1-Z)            #Fordel outcomes ift. tildelingen af treatment og kontrol

#Tilføjer disse variabler til en dataframe
df <- data.frame(Y, Z)          #Dataframe som afspejler et eksperiment
```

Vi er først og fremmest på jagt efter en gennemsnitlig treatmenteffekt, som udtrykker forskellen mellem kontrol og treatmentgruppen. Vi kan beregne forskellen i gennemsnit og estimere standardfejlen "i hånden" med afsæt i kontrol og treatmentgruppernes varians:

```
#Beregner treatmenteffekten som forskel i gennemsnit
ate_dim <- with(df, mean(Y[Z == 1]) - mean(Y[Z == 0]))
ate_dim
##0.128

# Estimerer standardfejl
```

```
with(df, sqrt(var(Y[Z == 1]) / sum(Z == 1) +
                var(Y[Z == 0]) / sum(Z == 0)))
##se: 0.045992
```

Præcis samme resultater opnår vi med OLS-regression af Y på Z og HC2-robuste standardfejl. Robuste standardfejl bør som udgangspunkt altid anvendes (med undtagelse af clusterrandomiserede eksperimenter, se Gerber & Green, 2012 s. 103). Fremgangsmåden er i øvrigt fuldstændig den samme hvis outcome er dikotomt. Der er altså i udgangspunktet ingen grund til at bruge hverken probit eller logit-modeller – OLS klarer ærterne.

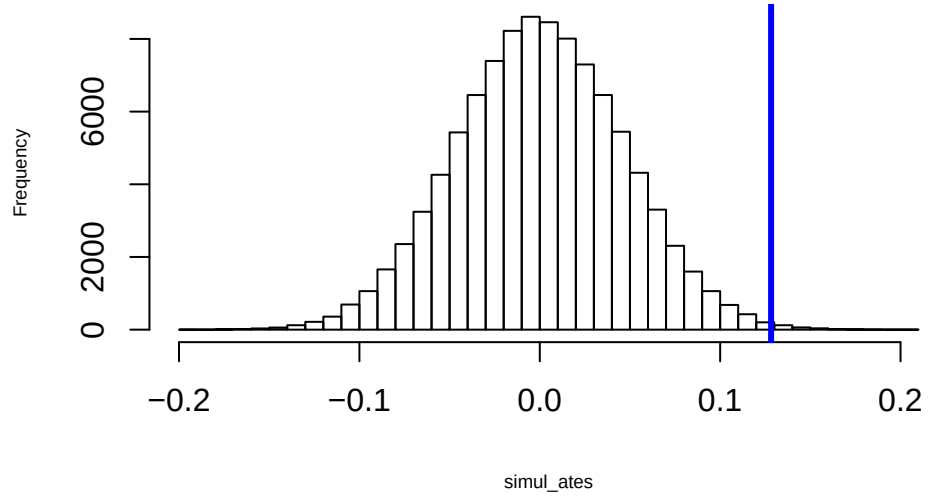
```
library(sandwich)
library(randomizr)
library(lmtest)

# Præcis det samme som OLS med HC2-robuste standardfejl:
fit <- lm(Y~Z, data=df)
robust_vcov <- vcovHC(fit, type = "HC2")
coeftest(fit, vcov. = robust_vcov)
##         ate: 0.128241   se: 0.045992   p: 0.0055

#Bemærk i øvrigt hvordan de klassiske standardfejl er lidt anderledes.
```

Tilsvarende analyse kan gennemføres ved randomiserigsinferens. I plottet herunder ses ATE (blå linje) og en fordeling af 100.000 effektestimater fra samme data under antagelse om nul-effekt for alle eksperimentelle enheder

Figur 4. Hypotesetest ved RI



Koden til ovenstående test følger tre overordnede trin:

```
##obs. vi bygger videre på data fra foregående eksempel

# (1): Dan to nye variable på baggrund af din outcome-variabel:
Y0_sharp <- df$Y
Y1_sharp <- df$Y
```

```r
#Yo_sharp og Y1_sharp er identiske outcomevariable
#De simulerer outcomes under antagelse af nul-effekt for hver respondent.

# Der er præcis nul effekt for hver enhed (Y1_sharp og Y0_sharp er ens)
Y1_sharp - Y0_sharp

# (2): sæt et loop op med 100.000 gentagelser
simul <- 100000             #sims er en scalar til hvilken vi allokerer tomme slots
simul_ates <- rep(NA, simul)#gemmer dem i simulated_ate

#genererer en række nye eksperimenter. Hver gang er der 225 enheder i treatment
for(i in 1:simul){

  # Gennemfør randomisering
  Z_simul <- complete_ra(450, 225)

  # Tildel treatment (gemmer Y0 eller Y1 afhængigt af Z)
  Y_simul <- Y0_sharp * (1 - Z_simul) + Y1_sharp * Z_simul

  # Beregn forskelle i gennemsnitsnit i hvert eksperiment
  simul_ates[i] <-
    mean(Y_simul[Z_simul == 1]) - mean(Y_simul[Z_simul == 0])

}

# (3): Sammenlign distributionen med ate i dit eksperiment
hist(simul_ates)

#Tilføj ate_dim
abline(v = ate_dim, col = "blue", lwd = 3)

#Beregn p-værdi
p_værdi <- mean(abs(simul_ates) >= abs(ate_dim))
p_værdi
##P: 0.00541
```

Endelig er standardfejlen givet ved effektfordelingens standardafvigelse:

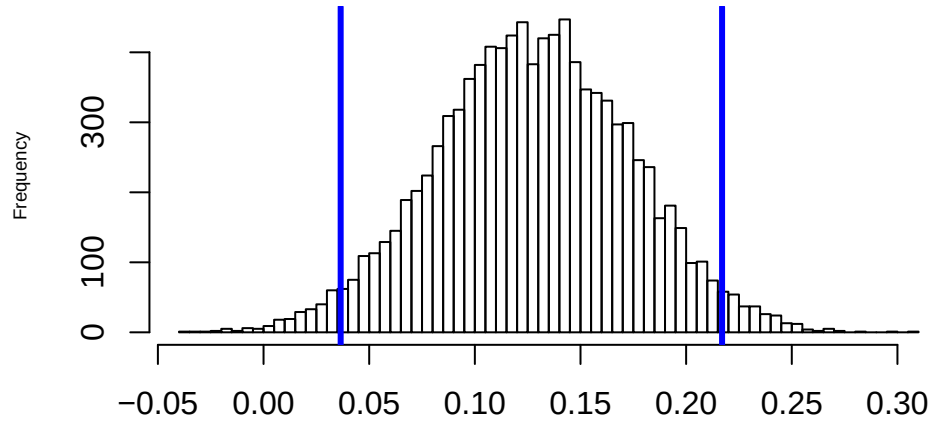
```r
#Beregner standardfejl
sd(simul_ates)
## se: 0.0464
```

Bemærk hvordan standardfejlen er nærmest identisk med HC2-standardfejlen.

3.3 Inverterede konfidensintervaller

Konfidensintervaller estimeres typisk ved ± standardfejlen*1,96, men valide konfidensintervaller kan ligeledes estimeres med afsæt i designbaseret inferens. Dette gøres ved at antage konstante treatmenteffekter og gennemføre hypotesetest med forskellige test statistics der gradvist øges (mindskes) til vi rammer en p-værdi på 0.025 og 0.975 (ved et 95% konfidensniveau).

Figur 5. Konfidensintervaller



Eksemplet bygger videre på data fra ovenstående eksempler.

```
##Konventionelle konfidensintervaller
#Øvre
0.128 + (1.96*0.045992)
#0.218

#Nedre
0.128 - (1.96*0.045992)
#0.037

##Konfidensintervaller med randomiseringsinferens
#1. Potentielle outcomes under kontrol
Y0 <- ifelse(Z==1,Y-.128,Y)      #Trækker effekten af Z fra når Z=1

#allokerer 10.000 tomme slots
simul <- 10000
simul_ates <- rep(NA, simul)

#genererer en række nye eksperimenter under antagelse af konstante effekter.
for(i in 1:simul){

  # Komplet randomisering
  Z_simul <- complete_ra(450, 225)   #Hver gang er der 225 enheder i treatment

  #definerer model
  Y_simul <- Y0 + (Z_simul*0.128)

  #samler effektestimater
  simul_ates[i] <-
    mean(Y_simul[Z_simul == 1]) - mean(Y_simul[Z_simul == 0])
  }
```

```r
#Vi kan nu prøve os frem ved at gætte på forskellige test statistics
#Vi ved, at konfidensintervallerne sandsynligvis er omkring omkring .037 og .218.
CI_nedre <- 0.0365
p_værdi <- mean((simul_ates) >= (CI_nedre))
p_værdi
##.975

CI_øvre <- 0.217
p_værdi <- mean((simul_ates) >= (CI_øvre))
p_værdi
##0.025

# -> Ganske rigtigt!
#Vi har nu hhv. nedre (0.0365) og øvre (0.217) ci omkring effektestimatet
```

3.4 Analyse af blockrandomiserede designs

Vi kan justere for de blockede variable og dermed øge efficiensen. I en OLS-model gøres det ved at indkludere de kovariater, som der er blocket på som "kontrolvariable" (regresser Y på Z + kovariater). I randomiseringsinferens (vist i eksemplet herunder) gøres det ved at anvende samme randomiseringsskema i alle permuteringer. Bemærk, at hvis blockvariablene er meget svagt korrelerede med outcomes, kan inklusion af kovariaterne i analysen faktisk reducere efficiensen i det antallet af frihedsgrader reduceres. Det skal imidlertid ikke afholde en fra at blockrandomisere: i fald blockene viser sig ikke at prediktere outcomes, kan det forsvares at analysere data *uden* inklusion af blok-dummies. Se en uddybning af dette argument hos Imbens & Athey (2017).

```r
library(randomizr)
rm(list=ls())
set.seed(123)

#Simulerer data med kovariater
N<-500
kon <- sample(c("F","M"), N, replace = TRUE)
offentlig <- sample(c("O","P"), N, replace = TRUE)
alder <- sample(x=18:65, size=N, replace=TRUE)

# Tilføjer effekter
effectofkon <- 0.3
effectofalder <- 0.15
effectofoffentlig <- 0.4

# Kontrolgruppens outcomes er en funktion af køn, sektor og alder
Y0 <- effectofkon*(kon=="M") + effectofalder*alder  +
  effectofoffentlig*(offentlig=="P")+ rnorm(n=N, mean=6, sd=0.7)

#tildeler outcomes til Y1 (Treatment):
effekt <- .3          # Et bud på treatment effekt
Y1 <- Y0 + effekt     # Potentialle outcomes for individer som treates

#Udfører blokrandomisering
df_block <- data.frame(Y0, Y1, kon, offentlig)   #samler i data frame
block_var <- with(df_block, paste(kon, offentlig, sep = "_"))
```

```r
#Tilføjer block-variablen til dataframe for overskuelighedens skyld
df_block$block_var <-block_var

#Derefter tilføjes treatment på tværs af kovariater.
df_block$Treat <- block_ra(block_var = block_var, condition_names = c(0,1))

Z <- df_block$Treat

#fordeler outcomes ift. Z. (Z er vores treatment-indikator)
Y.sim <- Y1*Z + Y0*(1-Z)

# beregner effekten
ate_dim <- mean(Y.sim[Z == 1]) - mean(Y.sim[Z == 0])
ate_dim
##ate = 0.058

  #Randomiseringsinferens
# Første skridt: antag sharp null - dvs. ens outcome for både y0 og y1
Y0_sharp <- Y.sim
Y1_sharp <- Y.sim

# Der er præcis nul effekt for hver enhed (de er ens)
Y1_sharp - Y0_sharp

# Andet skridt: sætter et loop up med 100.000 gentagelser
#Vi allokerer 100000 empty slots og gemmer i simul_ate
simul <- 100000
simul_ates <- rep(NA, simul)

#vi genererer en række nye eksperimenter under præcis samme randomisering:
for(i in 1:simul){

df_block <- data.frame(Y0, Y1, kon, offentlig)
block_var <- with(df_block, paste(kon, offentlig, sep = "_"))
df_block$Treat <- block_ra(block_var = block_var, condition_names = c(0,1))

Z_simul <- df_block$Treat

# Fordeler outcomes
Y_simul <- Y0_sharp * (1 - Z_simul) + Y1_sharp * Z_simul

# Genererer simulerede forskelle i snit for alle 100.000 eksperimenter
simul_ates[i] <-
mean(Y_simul[Z_simul == 1]) - mean(Y_simul[Z_simul == 0])
}

#obs: denne bid tager lidt tid at køre

# Nu har vi en distribution af effektestimater ved 100.000 eksperimenter
hist(simul_ates)

#Tilføjer det faktiske (oprindelige) effektestimat som blå linje
abline(v = ate_dim, col = "blue", lwd = 3)
```

```r
#Beregner p-værdi ved at vurdere hvor mange ate der er => end vores estimat
p_værdi <- mean(abs(simul_ates) >= abs(ate_dim))
p_værdi
##P = 0.0039
```

3.5 Justering for kovariater

Randomiserede treatments er statistisk uafhængige af både observerede og uobserverede variable og hele idéen om "kontroller" i eksperimenter kan derfor umiddelbart virke mærkværdig. Alligevel kan det give mening at inkludere pre-treatment kovariater i analysen for at reducere støj (såkaldt kovariatjustering). Kovariater er karakteristika ved de eksperimentelle enheder der (ideelt) predikterer outcome, og ved at justere for dem reduceres uforklaret varians, hvilket mindsker standardfejlen for vores effektestimat. Kovariatjustering er selvsagt en ex post-justering og hvorvidt det er god praksis er omdiskuteret - bl.a. fordi det gør det muligt at fiske efter interessante resutater (se eksempelvis Imbens and Rubin (2015) eller Deaton (2010)). En central styrke ved eksperimenter er netop simpliciteten og gennemsigtigheden i og det kan diskuteres om kovariatjustering kompromitterer dette. Hvilke kovariater der skal justeres før er ideelt set specificeret i en analyseplan forud for analysen for at maksimere transparens.

Vi kan justere for kovariater i en multipel regressionsanalayse ved at inkludere kovariaterne som uafhængige variable. Hvis treatment og kontrolgrupperne ikke er lige store, kan der være yderligere gevinst at hente ved at interagerer treatmentindikatoren og kovariaterne som i eksemplet herunder.

Eksempel på kovariatjustering i OLS

```r
set.seed(123)

library(sandwich)
library(randomizr)
library(lmtest)

#Simulerer data med kovariater
N<-1000
kon <- sample(c(0,1), N, replace = TRUE)
alder <- sample(x=18:65, size=N, replace=TRUE)
# Tilføjer effektstørrelser
effectofkon <- 0.3
effectofalder <- 0.15
Z <- complete_ra(N, N/2)

# Kontrolgruppens outcomes er en funktion af køn og alder
Y0 <- effectofkon*(kon=="M") + effectofalder*alder  + rnorm(n=N, mean=6, sd=0.7)

##Vi tildeler outcomes til Y1 (Treatment):
effekt <- .6          # Et bud på treatment effekt
Y1 <- Y0 + effekt     # Potentialle outcomes for individer som treates

##Vi fordeler outcomes ift. Z.
Y.sim <- Y1*Z + Y0*(1-Z)

# Centrerer kovariaterne
kon_c <- kon - mean(kon)
alder_c <- alder - mean(alder)
```

```r
# fitter en model med justering
fit_just <- lm(Y.sim ~ Z + Z*(kon_c + alder_c))
fit <- lm(Y.sim ~ Z + kon + alder)
summary(fit)
# Robuste se
coeftest(fit_just, vcov = vcovHC(fit_just, type = "HC2"))

##              Estimate Std. Error
##Z            0.57  0.043

# Sammenligner med en model uden justering
fit <- lm(Y.sim ~ Z)

# Robuste se
coeftest(fit, vcov = vcovHC(fit, type = "HC2"))
##Z            0.70   0.138
```

Det er helt afgørende at kovariaterne falder før treatment for at undgå post-treatment bias, ligesom de ikke skal have påvirket tildelingen af treatment. Bemærk ligeledes som tommelfingerregel, at der bør være mindst 20 observationer mere end antallet af kovariater (Se Gerber & Green (2012), s. 104).

I resultatafsnittet bør de ujusterede resultater rapporteres først. Dernæst kan de justerede estimater rapporteres. I den forbindelse bør estimater for kovariaterne *ikke* tolkes kausalt idet de jo ikke er randomiseret. Det kan anbefales at undlade afrappoteringen af estimater for kovariaterne direkte i analysen, idet det giver læseren anledning til at tolke på dem. Angiv i stedet, at estimatet er kovariatjusteret og uddyb kovariaterne i en fodnote.

3.6 Interaktionseffekter

Vi kan anvende randomiseringsinferens til at estimere usikkerhed omkring interaktionseffekter.[7]

Eksemplet herunder tager afsæt i et eksperiment med to dikotome treatments, imellem hvilke der er en interaktionseffekt. P-værdien for interaktionsleddet beregnes ved at antage konstante, additive effekter for Z og W og tildele treatments på ny i et stort antal eksperimenter. På den baggrund vurderes sandsynligheden for at observere interaktionsled lige så store eller større end interaktionen fra vores eksperiment.

Hypotesetest af interaktionseffekt

```r
#Simulerer data med en interaktionseffekt
rm(list=ls())
set.seed(123456)
N<-500

# Tilføjer to treatments (Z og W)
Z <- complete_ra(N, N/2)
W <- complete_ra(N, N/2)

#tilføjer effekter
konst <- 6
beta <- .45
beta2 <- .1
```

[7] Det er centralt at skelne mellem to fundamentalt forskellige interaktioner: interaktioner mellem to treatments og interaktioner mellem en treatment og en kovariat (se kapitel 9 i Gerber & Green, 2012).

```r
beta3 <- 0.28
e = rnorm(500, 0, sd=.56)

# Fitter interaktionsmodel
Y <- konst + Z*beta  + W*beta2 + beta3*Z*W + e

# samler i dataframe
df = data.frame(cbind(Z, W, Y))

#1a. Estimerer interaktionsleddet i eksperimentet
fit<-lm(Y~Z+W+Z*W, data=df)
summary(fit)
##Bemærk p-værdien for interaktionsleddet er 0.0152

#gemmer interaktionen
ate_dim <- summary(fit)$coef[4]
ate_dim

#1b.
##Udleder potentielle outcomes for enheder i kontrol
##Det gøres ved at subtrahere effekterne når Z=1 og/eller W=1

Y0 <- ifelse(Z==1,Y-.423,Y)    #Trækker effekten af Z fra v. Z=1
Y0_2 <- ifelse(W==1,Y-.176,Y0) #Trækker effekten af W fra v. W=1

#Y0_2 svarer dermed til kontrolgruppens potentielle outcomes


#2. simulerer distributionen
#allokerer tomme slots
simul <- 10000
simul_ates <- rep(NA, simul)

##Kører nu loopet under antagelse af konstante effekter
#vi genererer en række nye eksperimenter.
for(i in 1:simul){

  # Komplet randomisering
  Z_simul <- complete_ra(N, N/2)
  W_simul <- complete_ra(N, N/2)

#Simulerer konstante additive effekter
  Y_simul <- Y0_2 + (Z_simul*0.21) + (W_simul*0.48)

  df = data.frame(cbind(Z_simul, W_simul, Y_simul))
  fit<-lm(Y_simul~Z_simul+W_simul+(Z_simul*W_simul))
  #samler
  simul_ates[i] <-
    summary(fit)$coef[4]
}

hist(simul_ates, breaks=50, main= "Figur 6. Distribution af interaktionseffekter", xlab = "simulerede i
abline(v = ate_dim, col = "blue", lwd = 3)
```

```r
#Standardfejl:
sd(simul_ates)
##0.08

#Beregner p-værdi
p <- mean((simul_ates) >= abs(ate_dim))
p
#p: 0.015
```

Alternativt kan vi anvende en F-test til at vurdere to modeller hhv. med og uden interaktion mod hinanden:

```r
#samme data som eksemplet ovenfor

lm1  <- lm(Y~Z*W)  # alternative model
lm2  <- lm(Y~Z+W)  # null model

Ftest <- (sum(lm2$residuals^2) - sum(lm1$residuals^2)) / (sum(lm1$residuals^2) / (N - 4))


#allokerer tomme slots
simul <- 1000
Fdist <- rep(NA,simul)


for (i in 1:simul) {
    Z_loop <- complete_ra(N, N/2)
    W_loop <- complete_ra(N, N/2)
    Y0 <- ifelse(Z==1,Y-.423,Y)      #Trækker effekten af Z fra v. Z=1
    Y0_2 <- ifelse(W==1,Y-.176,Y0) #Trækker effekten af W fra v. W=1

    lm1_loop  <- lm(Y0_2~Z_loop*W_loop)
    lm2_loop  <- lm(Y0_2~Z_loop+W_loop)

    Fdist[i] <- (sum(lm2_loop$residuals^2) - sum(lm1_loop$residuals^2)) / (sum(lm1_loop$residuals^2) /

    }

#p-value
mean(Fdist >= Ftest)

##0.016
```

Litteratur

Athey, Susan, and Guido W. Imbens (2017). "The Econometrics of Randomized Experiments." In Handbook of Economic Field Experiments, vol. 1 (E. Duflo and A. Banerjee, eds.).

Athey, Susan, and Guido W. Imbens (2016), "Recursive Partitioning for Heterogeneous Causal Effects," Proceedings of the National Academy of Sciences 113.

Gerber, Alan S., and Donald P. Green (2012). Field Experiments: Design, Analysis, and Interpretation, chapter 4.

Keele, Luke, and McConnaughy, Corrine, and White, Ismail (2012). "Strengthening the Experimenter's Toolbox: Statistical Estimation of Internal Validity" American Journal of Political Science

Lin, Winston (2012). "Regression Adjustment in Randomized Experiments: Is the Cure Really Worse than the Disease?" Development Impact blog post, part I and part II.