Lab 10

Zeren Li

11/13/2019

Roadmap

- Review HW7&8
- Power Calculator
- Sampling

Power Calculator

https://egap.shinyapps.io/Power_Calculator/

```
# https://egap.shinyapps.io/Power_Calculator/
# example
# we have a sample as follows, 1000 obs, 10 groups, 2 blocks
set.seed(2019)

# set a population
pop =data.frame(y =  rnorm(1000,50,30),
                group = rep( seq(1:10), 100 ),
                block = as.factor(c(rep(1,600), rep(2,400))))

mean(pop$y)  # rnorm output may be different using different machines
```

```
## [1] 47.65453
```

- we expect a 20% increase in y if the treatment is assigned
- we expect the standard deviation of the treatment: 20
- mean of the treatment: 49.14805+ 9.82961 = 58.97766
- mean of the control: 49.14805
- delta (treatment effect size): 9.82961
- level of power: 0.8

```
# We will sample at 67 obs for treatment and control groups.
# In total, we have to sample at least 134 obs.
power.t.test( delta = 9.688, sd = 20, power = .8, sig.level = .05)
```

```
##
##      Two-sample t test power calculation
##
##               n = 67.87436
##           delta = 9.688
##              sd = 20
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

```r
# the algorithm from power calculator app
# the function
power_calculator <- function(mu_t, mu_c, sigma, alpha=0.05, N){
  lowertail <- (abs(mu_t - mu_c)*sqrt(N))/(2*sigma)
  uppertail <- -1*lowertail
  beta <- pnorm(lowertail- qnorm(1-alpha/2), lower.tail=TRUE) +
    1 - pnorm(uppertail- qnorm(1-alpha/2), lower.tail=FALSE)
  return(beta)
}
Ns_small <- as.matrix(1:100000)

# specify mu_t, mu_c, sigma, alpha in the power

betas_small <- apply(X=Ns_small,MARGIN = 1,FUN = power_calculator, mu_t= 58.128, mu_c=48.44, sigma=20 ,a

Ns_small[which.max(betas_small>=.8)]
```

```
## [1] 134
```

Clustered Design

```r
# example
# mean of the treatment: 48.44+ 9.688 = 58.128
# mean of the control: 48.44
# delta(treatment effect size): 9.688
# level of power: 0.8
# standard deviation of treatment: 20
# level of power: 0.8
# number of clusters: 2
# size of each cluster: 5
# ICC (intra cluster correlation): ICC is from the population

# compute ICC
library(fishmethods)
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
## The following object is masked from 'package:dplyr':
##
##     select
## The following objects are masked from 'package:openintro':
##
##     housing, mammals
## Loading required package: boot
## Loading required package: bootstrap
##
## Attaching package: 'bootstrap'
## The following object is masked from 'package:broom':
```

```
##
##     bootstrap

## Loading required package: lme4

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':
##
##     expand

## Loading required package: numDeriv
```

```
library(haven)

# ICC = 0.00789
clus.rho(popchar=pop$y,cluster=pop$group)
```

```
## $icc
##                          value
## Lohr rho           0.009328989
## Adjusted r-square 0.010319660
## ANOVA rho          0.011441834
```

```
power_calculator_cluster <- function(mu_t, mu_c, sigma, ICC, alpha=0.05, n_clus_per_arm, N){
  n_per_clus <- N/(n_clus_per_arm*2)
  if(n_per_clus < 1){return(NA)}
  lowertail <- (abs(mu_t - mu_c) * sqrt((n_clus_per_arm - 1)*n_per_clus))/
                      sqrt((2*(sigma^2) * (1 + (n_per_clus-1)*ICC)))
  uppertail <- -1*lowertail
  beta <- pnorm(lowertail - qnorm(1-alpha/2), lower.tail=TRUE) +
    1 - pnorm(uppertail - qnorm(1-alpha/2), lower.tail=FALSE)
  return(beta)
}

Ns_small <- as.matrix(1:100000)

# specify mu_t,mu_c,sigma,alpha,
# ICC (intra cluster correlation), n_clus_per_arm(size of each cluster)

betas_small <- apply(X=Ns_small,MARGIN = 1, FUN = power_calculator_cluster, mu_t= 58.128, mu_c=48.44, si
# specify the level of power: .8
Ns_small[which.max(betas_small>=.8)]
```

```
## [1] 192
```

Sampling

```
# randomly sample 134 obs from the population
sample1 = pop %>%
  # add id
  mutate(id = row_number()) %>%
  sample_n(134)
```

```r
## by proportion  134/1000
sample2 = pop %>%
  # add id
  mutate(id = row_number()) %>%
  sample_frac(.134)

# blocking: randomly sample (136/2 = 68) obs from block 1 and 2
sample3 <- pop %>%
  # add id
  mutate(id = row_number())  %>%
  group_by(block) %>%
  sample_n(68)

table(pop$block)
```

```
##
##   1   2
## 600 400
```

```r
table(sample3$block)
```

```
##
##  1  2
## 68 68
```

```r
# blocking: randomly sample 34 treated and 34 control in each block

sample4 <- pop %>%
  # add id
  mutate(id = row_number())  %>%
  group_by(block) %>%
  sample_n(68)

# package for random assignment
library(randomizr)

z = block_ra(sample4$block )

table(z, sample4$block)
```

```
##
## z    1  2
##   0 34 34
##   1 34 34
```

```r
sample4$treat <- z

sample4 %>% arrange(block ,id, treat)
```

```
## # A tibble: 136 x 5
## # Groups:   block [2]
##        y group block    id treat
##    <dbl> <int> <fct> <int> <int>
##  1  1.52    10 1        30     1
##  2 68.9      2 1        32     0
##  3 72.8      3 1        33     1
```

```
##  4 34.7      4 1        34     0
##  5 36.9     10 1        40     0
##  6 39.9      3 1        53     0
##  7 84.8     10 1        90     1
##  8 30.0      2 1        92     1
##  9 29.5      3 1       103     1
## 10 63.9      8 1       108     1
## # ... with 126 more rows
```

```
head(sample4)
```

```
## # A tibble: 6 x 5
## # Groups:   block [1]
##       y group block    id treat
##   <dbl> <int> <fct> <int> <int>
## 1  17.6     2 1       552     1
## 2  49.6     8 1       298     0
## 3  31.3    10 1       510     0
## 4  37.7     7 1       387     0
## 5  68.9     6 1       486     0
## 6  63.9     8 1       108     1
```

Alternative Sampling Strategy

https://cran.r-project.org/web/packages/randomizr/vignettes/randomizr_vignette.html

```
# complete random assignment
Z <- complete_ra(N = 100, m = 50)
table(Z)
```

```
## Z
##  0  1
## 50 50
```

```
# This makes a cluster variable: one unit in cluster "a", two in "b"...
clust_var <- rep(letters[1:15], times = 1:15)

Z <- cluster_ra(
  clusters = clust_var,
  m_each = c(4, 4, 7),
  conditions = c("control", "placebo", "treatment")
  )
table(Z, clust_var)
```

```
##            clust_var
## Z            a  b  c  d  e  f  g  h  i  j  k  l  m  n  o
##   control    0  0  3  0  0  0  7  0  0  0  0  0  0 14 15
##   placebo    0  0  0  4  0  6  0  0  0  0 11 12  0  0  0
##   treatment  1  2  0  0  5  0  0  8  9 10  0  0 13  0  0
```

```
blocks <- rep(c("A", "B", "C"), c(50, 100, 200))

block_ra(blocks = blocks)
```

```
##   [1] 1 0 0 1 0 1 0 1 1 0 1 1 0 1 1 0 0 0 0 1 0 1 1 0 0 0 1 1 0 0 0 0 1 0 1
##  [36] 0 1 1 1 0 0 0 1 1 1 1 1 0 1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 0 1 1
```

```
##  [71] 1 0 1 1 1 1 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 1 0 0
## [106] 0 1 0 0 1 1 0 1 0 1 0 0 0 0 1 1 0 1 0 1 1 1 1 0 0 1 1 1 1 0 1 1 1 0
## [141] 1 0 0 0 1 0 0 1 1 1 1 1 0 1 0 1 0 1 1 1 0 0 1 0 1 1 1 1 0 1 1 1 1 1 0
## [176] 0 0 1 1 0 0 0 0 0 1 0 0 1 0 1 0 1 1 0 1 1 0 0 0 1 1 0 0 1 1 1 0 1 0 1
## [211] 0 1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 1 0 0 0
## [246] 0 1 1 1 0 1 0 0 1 0 0 1 0 0 1 0 1 1 1 1 0 1 0 1 0 0 0 0 1 0 1 0 1 1 0
## [281] 0 1 0 1 1 1 0 1 0 0 1 1 1 0 1 0 1 0 1 0 0 0 0 1 0 1 1 1 0 1 1 1 0 0 0
## [316] 1 0 1 0 1 1 1 1 0 1 1 0 1 0 0 1 0 1 0 0 0 0 1 1 1 0 1 1 0 0 1 1 0 0 1
```

```
# defaults to half of each block block_ra(blocks = blocks)
# can change with block_m block_ra(blocks = blocks,

block_ra(blocks = blocks,
block_m = c(20, 30, 40))
```

```
##   [1] 1 1 0 0 1 1 0 0 0 0 0 1 0 0 1 0 1 1 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0
##  [36] 0 1 1 1 1 0 1 0 1 0 1 0 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 1 1 0 1 0 0 0 0
##  [71] 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0
## [106] 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 1 1 1 0 1 1 1 0 0 1 0 1 0 1 0 1 0 1 1
## [141] 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0
## [176] 0 1 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
## [211] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0
## [246] 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0
## [281] 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0
## [316] 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0
```