Point Estimation & Confidence Intervals

POINT ESTIMATION

Parameter: Characteristics that are used to describe the population.

Statistic: a function of the observable random variables in a sample which does not include any unknown quantities.

Estimator: A statistic that is used to estimate an unknown parameter.

	Sample Statistics		Population Parameters
Mean	$\bar{X}$	$\longrightarrow$	μ
Standard Deviation	s	$\longrightarrow$	σ
Proportion	$\hat{p}$	$\longrightarrow$	p

MAXIMUM LIKELIHOOD ESTIMATORS

- The point in the parameter space that maximizes the likelihood function
- Likelihood function is given by;

$$L(\theta, x) = \prod_{i=1}^{n} f(x_i, \theta)$$
$$= f(x_1, \theta) \cdots f(x_n, \theta)$$

- The idea of maximum likelihood estimation is to first assume our data come from a known family of distributions that contain parameters.
- Then the maximum likelihood estimates (MLEs) of the parameters will be the parameter values that are most likely to have generated our data.

NORMAL DISTRIBUTION - MAXIMUM LIKELIHOOD ESTIMATION

- The MLE of μ is defined as
- $\mu\hat{}_{MLE}=\text{argmax}(x1,...,xn|\mu,\sigma^2)$; where $\mu\hat{}_{MLE}$ is the value of that maximizes the likelihood function.

$$f(x_1,\ldots,x_n|\mu,\sigma^2)=\prod_{i=1}^{n} f(x_i|\mu,\sigma^2)=\prod_{i=1}^{n}\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2\sigma^2}(x_i-\mu)^2\right)$$

- If we maximize the above likelihood function, we get $\mu\hat{}_{MLE}=\bar{x}$.
- Since the MLE of μ is the sample mean, computing the MLE in R becomes straightforward.

INTERVAL ESTIMATION

- Point estimators are often use as sample measures for population parameters.
- It is also helpful to know how reliable this estimate is, that is, how much sampling uncertainty is associated with it.
- A useful way to express this uncertainty is to calculate an interval estimate or confidence interval for the population parameter
- In other words, the confidence interval is of the form

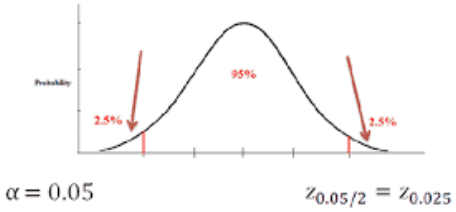
  "point estimate ± uncertainty"

CONFIDENCE INTERVAL FOR MEAN

- **Case 1**: When data is normal/ large sample and σ

  is known.

$$\bar{x} \pm z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$$



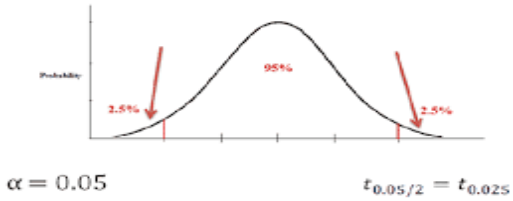
$\alpha = 0.05$ $z_{0.05/2} = z_{0.025}$

```
set.seed(20)
sample_size<-500
pop_stdev<-10
weight<-sample(45:80,size=sample_size,replace = T)
sample_mean<-mean(weight)
z_critical<-qnorm(0.975)
margin_error<-z_critical*(pop_stdev/sqrt(sample_size))
c_i<-c(sample_mean -margin_error,sample_mean+margin_error)
c_i
```

```
## [1] 62.28748 64.04052
```

- **Case 2:** When data is normal/ large samples and σ is unknown.

$$\bar{x} \pm t_{\frac{\alpha}{2}}\left(\frac{s}{\sqrt{n}}\right)$$



$\alpha = 0.05$ $\qquad$ $t_{0.05/2} = t_{0.025}$

```
set.seed(20)
large_sam_weight=sample(weight,150)
large_sam_t_critical=qt(0.975,df=149)
large_sam_mean=mean(large_sam_weight)
large_sam_stdev<-sd(large_sam_weight)
large_sam_margin_of_error=large_sam_t_critical*(large_sam_stdev/sqrt(15
0))
large_sam_confi_interval=c(large_sam_mean - large_sam_margin_of_error,
large_sam_mean + large_sam_margin_of_error)
large_sam_confi_interval
```

```
## [1] 61.32493 64.87507
```

- **Case 3**: When data is non-normal/ small samples

  For this, bootstrap approach is used as follows.

```
> library(boot)
> blood_pressure <- c( 72, 66, 64, 66, 40, 74, 50, 70, 96, 92,
74, 80, 60, 72, 84, 74, 80, 70, 88, 94)
> mean_fn <- function(x, indices) mean(x[indices])
> level.boot <- boot(blood_pressure, mean_fn, R=999)
> boot.ci(level.boot, conf= 0.95)
```

CONFIDENCE INTERVALS FOR DIFFERENCE OF MEANS

- **Case 1**: Sampling from two independent normal distributions with known variances.

Let $X_i \sim \mathcal{N}(\mu_1, \sigma_1^2)$, $i = 1, 2, \ldots, n_1$ and $Y_i \sim \mathcal{N}(\mu_2, \sigma_2^2)$, $i = 1, 2, \ldots, n_2$.

Then $\overline{X} \sim \mathcal{N}(\mu_1, \sigma_1^2/n_1)$, $\quad \overline{Y} \sim \mathcal{N}(\mu_2, \sigma_2^2/n_2)$

$(1-\alpha)100\%$ confidence interval for $(\mu_1 - \mu_2)$ is

$$(\overline{X} - \overline{Y}) \pm Z_{\alpha/2}\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

```
library("BSDA")

## Warning: package 'BSDA' was built under R version 4.1.3

## Loading required package: lattice

##
## Attaching package: 'BSDA'

## The following object is masked from 'package:datasets':
##
##     Orange


z.test(x,y = NULL,alternative = "two.sided",
sigma.x = NULL, sigma.y = NULL, conf.level = 0.95)
```

- **Case 2**: Sampling from two independent normal distributions with unknown variances (small samples).

- when population variances are equal

$(1-\alpha)100\%$ confidence interval for $(\mu_1 - \mu_2)$

$$(\overline{X} - \overline{Y}) \pm t_{n_1+n_2-2}(\alpha/2) * S_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

```
t.test(x,y,alternative = "two.sided",
    var.equal=TRUE, conf.level = 0.95)
```

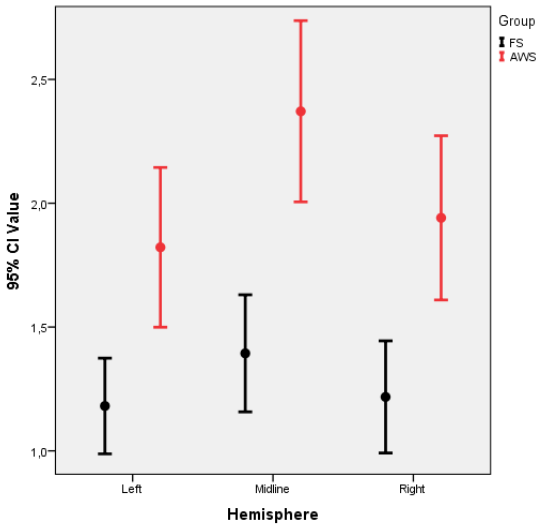
- when population variances are unequal

$$(\overline{X} - \overline{Y}) \pm t_k\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}$$

$$\text{where } k = \frac{\left(\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}\right)^2}{\frac{1}{(n_1-1)}\left(\frac{S_1^2}{n_1}\right)^2 + \frac{1}{(n_2-1)}\left(\frac{S_2^2}{n_2}\right)^2}$$

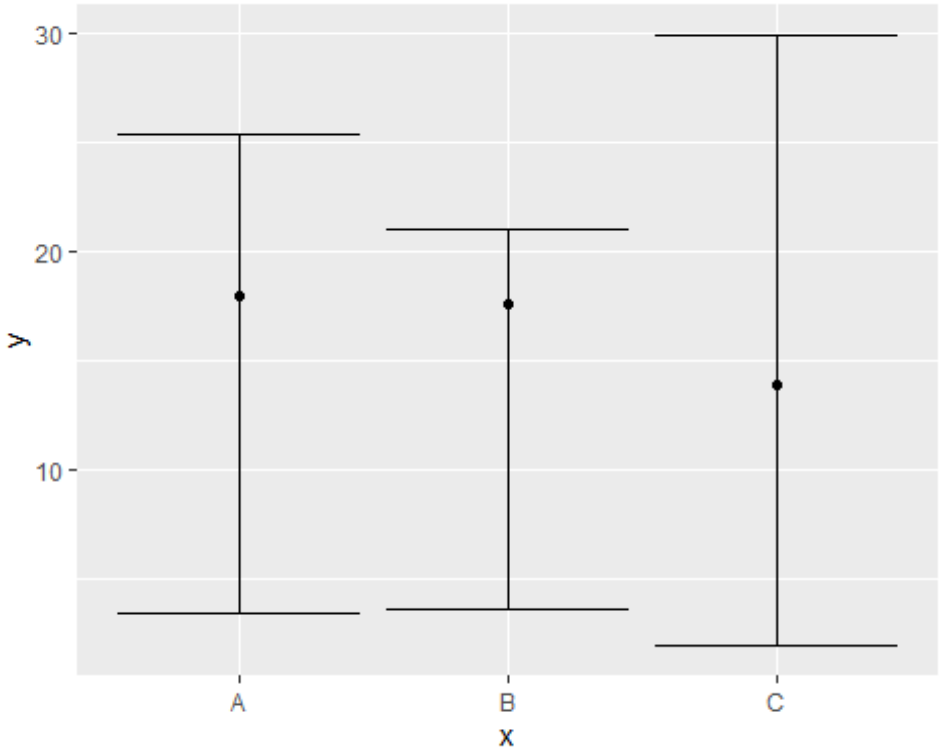
```
t.test(x,y,alternative = "two.sided",
    var.equal=FALSE, conf.level = 0.95)
```

CONFIDENCE INTERVAL CHART IN R (INDEPENDENT MEANS & CIS)



Example

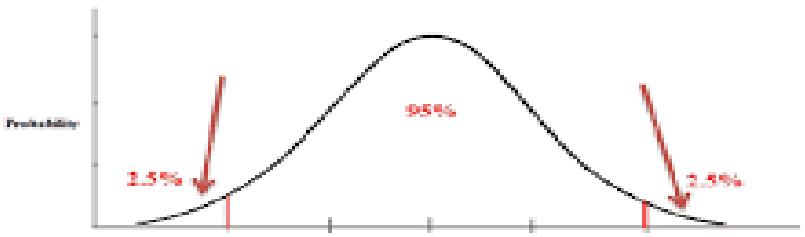
```
set.seed(123456)                    # Create example data
data <- data.frame(x = c("A","B","C"),
                   y = round(runif(3, 10, 20),2),
                   lower = round(runif(3, 0, 10),2),
                   upper = round(runif(3, 20, 30),2))

library(ggplot2)

ggplot(data, aes(x, y)) +           # ggplot2 plot with confidence intervals
  geom_point() +
  geom_errorbar(aes(ymin = lower, ymax = upper))
```



CONFIDENCE INTERVALS FOR PROPORTION

- **Case 1**: For large sample (Using Normal approximation)

$$\hat{p} \pm z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$



$$\alpha = 0.05 \qquad z_{0.05/2} = z_{0.025}$$

```
set.seed(10)
Hair_col=c(rep('black',1500),rep('brown',1000),rep('blonde',500))
sampleP=sample(Hair_col,1000)
Htable=table(sampleP) #calculate each occurances for each factor
Htable
```

```
## sampleP
##  black blonde  brown
##    498    176    326

prop.table(Htable) #caluculate sample proportions for each factor

## sampleP
##  black blonde  brown
##  0.498  0.176  0.326

p_bar=0.498
n=1000
z_critical=qnorm(0.975) #calculate critical value
margin_error=z_critical*sqrt((p_bar*(1-p_bar))/n)
c_i=c(p_bar-margin_error , p_bar+margin_error)
```

Case 1: For large sample (Using Binomial Distribution)

- we can use the following functions from R package epitools for this case.

```
library(epitools)
binom.exact(x=498,n=500,conf.level=0.95) #calculates exact confidence
intervals for binomial counts or proportions

##     x   n proportion     lower     upper conf.level
## 1 498 500      0.996 0.9856259 0.9995152       0.95

binom.wilson(x=498,n=500,conf.level=0.95)#calculates CI for binomial counts
or proportions using Wilson's formula which approximate the exact method.

##     x   n proportion     lower     upper conf.level
## 1 498 500      0.996 0.9855343 0.9989024       0.95

binom.approx(x=498,n=500,conf.level=0.95) #calculates CI for binomial counts
or proportions using normal approximate to the binomial distribution.

##     x   n proportion     lower    upper conf.level
## 1 498 500      0.996 0.9904675 1.001533       0.95
```

Case 2: For small sample (Using Binomial Distribution)

- When sample size is small, confidence interval for population can be calculated using binom.test() function.

```
gender=c('f','f','f','m','m','f','f','m','m','f')
table(gender)
```

```
## gender
## f m
## 6 4

binom.test(6,10,conf.level=0.95)

##
##  Exact binomial test
##
## data:  6 and 10
## number of successes = 6, number of trials = 10, p-value = 0.7539
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.2623781 0.8784477
## sample estimates:
## probability of success
##                    0.6
```

CONFIDENCE INTERVALS FOR VARIANCE

Case 1: Under normality assumption

- User defined function to obtain confidence interval for variance.

```
var.interval = function(data, conf.level = 0.95) {
 df = length(data) - 1
 chilower = qchisq((1 - conf.level)/2, df)
 chiupper = qchisq((1 - conf.level)/2, df, lower.tail = FALSE)
 v = var(data)
 c(df * v/chiupper, df * v/chilower)
 }

 lizard = c(6.2, 6.6, 7.1, 7.4, 7.6, 7.9, 8, 8.3, 8.4, 8.5, 8.6, 8.8, 8.8,
9.1, 9.2, 9.4, 9.4, 9.7, 9.9, 10.2, 10.4, 10.8, 11.3, 11.9)

 var.interval(lizard)

## [1] 1.235162 4.023559
```

Case 2: Under non-normality assumption

When no assumption is made about data, a bootstrap method is used to obtain confidence intervals for the population variance.

```
library(boot)

blood_presure=c(72,66,64,66,40,74,50,70,96,92,74,80,60,72,84,74,80,70,88,94)
variance=function(x,indices) var(x[indices])
level.boot=boot(blood_presure,variance,R=999)
boot.ci(level.boot,conf=0.95)

## Warning in boot.ci(level.boot, conf = 0.95): bootstrap variances needed
for
## studentized intervals

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 999 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = level.boot, conf = 0.95)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 84.0, 326.6 )   ( 71.4, 307.6 )
##
## Level     Percentile            BCa
## 95%   ( 88.4, 324.6 )   (109.5, 402.6 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

References

https://youtu.be/28aluI4wsMM

https://youtu.be/DT-fPG0Hff8