# BUDS Training: Error rates, coverage rates, and power Lab

Raven Ico

## Problem 1

Each time we make a statistical decision, say based on a confidence interval or p-value, we are making a decision in the face of uncertainty, and therefore there is an associated risk of making an incorrect decision. In hypothesis testing we refer to two types of errors: type I errors, when we reject a true $H_0$; and type II errors, when we fail to reject a false $H_0$. The type I error rate can be controlled directly by the construction of our test - in particular, our $\alpha$ significance level tells us the maximum $P(\text{Type I Error})$. The likelihood of a type II error depends on several things: the sample size, the significance level, the variability in the data, and the true effect size.

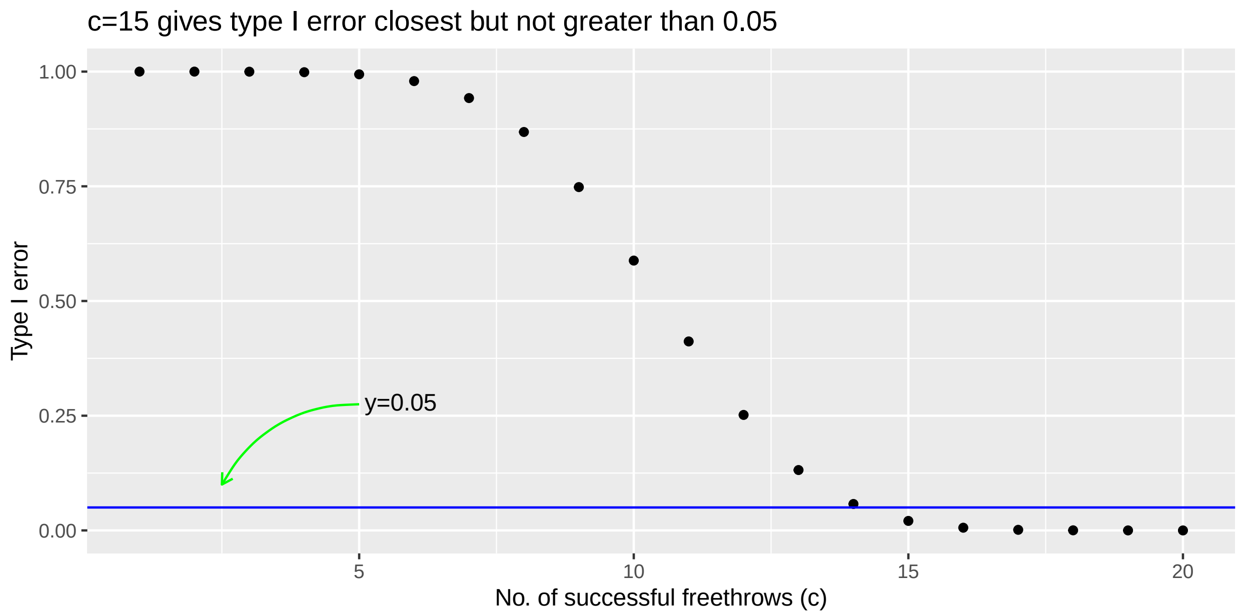
### Part (a)

Suppose we observe Shaq shoot 20 freethrows, and we want to use our sample to decide if he shoots freethrows with better than 50% accuracy. Let $p$ be his true freethrow percentage, and consider a test that rejects $H_0 : p \leq 0.5$ when he make $c$ or more shots out of 20. $c/20$ is the critical value for this hypothesis test (i.e. will reject the null hypothesis if the sample freethrow percentage is greater than or equal to $c/20$).

Suppose his freethrow percentage is exactly 50% (so rejecting $H_0$ is an error). Make a plot of $P(\text{Type I Error})$ versus $c$, for $c = 0, 1, \ldots, 20$. Which value of $c$ will give us a test with a type I error rate closest to but not greater than 0.05?

(hint: notice that $P(\text{Type I Error}) = P(X \geq c)$, where $X \sim binomial(20, 0.5)$. Be careful to include the $Pr(X = c)$ in your calculations).

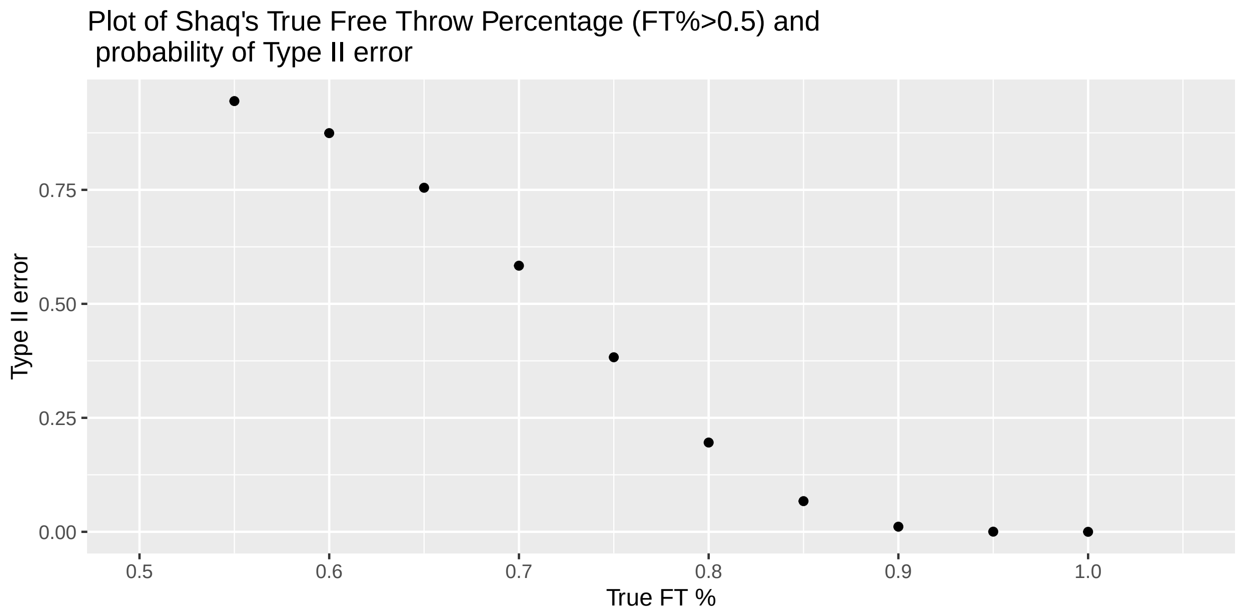
```
c = seq(20)
typeIerror = pbinom(c-1,20,0.5,lower.tail = FALSE)
shaq = tibble(c,typeIerror)
ggplot(shaq,aes(c,typeIerror))+
  geom_point()+
  geom_hline(yintercept = 0.05,color="blue")+
  labs(x = "No. of successful freethrows (c)", y="Type I error",
       title="c=15 gives type I error closest but not greater than 0.05")+
  annotate(
    geom = "curve", x = 5, y = 0.275, xend = 2.5, yend = 0.1,
    curvature = .3, arrow = arrow(length = unit(2, "mm")), color="green"
  ) +
  annotate(geom = "text", x = 5.1, y = 0.28, label = "y=0.05", hjust = "left")
```



## Part (b)

Assume now, we don't know the true value of Shaq's free throw percentage. Use your value of $c$ from the previous problem and calculate the value of a type II error, $Pr(X < c)$, for all values of freethrow percentage $(p = X/20)$ where it is possible to make a type II error (it's not possible to make this error when $H_0 : p \leq 0.5$ is true). Create a plot of type II error vs. Shaq's true FT%,

```r
c = seq(11,20)
tibble(FT=c/20,typeIIerror=pbinom(14,20,c/20))%>%
ggplot(aes(FT,typeIIerror))+
  geom_point()+
  labs(x = "True FT %", y="Type II error",
       title= "Plot of Shaq's True Free Throw Percentage (FT%>0.5) and  \n probability of Type II error
  coord_cartesian(xlim=c(0.5,1.05))
```

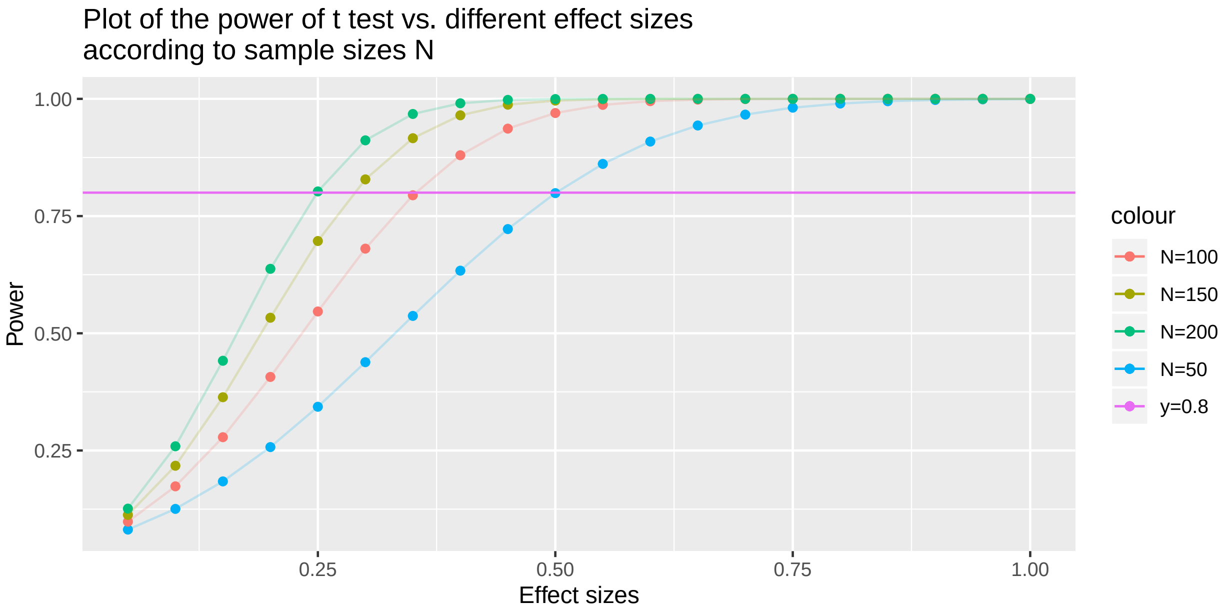


# Problem 2

## Part (a)

We would like to conduct a clinical trial to compare the mean levels of expression of a specific protein in patients given drug A versus patients given drug B. The null hypothesis is that the expression levels for drug B are less than or equal to those of drug A. Assuming the protein expression levels in each of the groups is approximately normally distributed with a pooled standard deviation of 1. The effect size is the true difference in means between the two groups. Plot the power of a t-test with significance level 0.05 by the effect size for sizes $\{0.05, .1, .15, ...0.95, 1\}$. Plot different curves for sample sizes $n = 50, 100, 150, 200$.

```r
alpha = 0.05
sizes = seq(0.05,1,by=0.05) #effect sizes

#creating different curves for sample sizes n= 50, 100, 150, 200
power50 = power.t.test(n=50,delta=sizes,sd=1,sig.level=alpha,type="two.sample",
                       alternative="one.sided")$power
power100 = power.t.test(n=100,delta=sizes,sd=1,sig.level=alpha,type="two.sample",
                        alternative="one.sided")$power
power150 = power.t.test(n=150,delta=sizes,sd=1,sig.level=alpha,type="two.sample",
                        alternative="one.sided")$power
power200 = power.t.test(n=200,delta=sizes,sd=1,sig.level=alpha,type="two.sample",
                        alternative="one.sided")$power

tibble(sizes,power50,power100,power150,power200)%>%
  ggplot(aes(sizes,power50))+
    geom_point(mapping=aes(colour="N=50"))+
    geom_point(mapping=aes(sizes,power100,colour="N=100"))+
    geom_point(mapping=aes(sizes,power150,colour="N=150"))+
    geom_point(mapping=aes(sizes,power200,colour="N=200"))+
    geom_line(mapping=aes(colour="N=50"),alpha=0.2)+
    geom_line(mapping=aes(sizes,power100,colour="N=100"),alpha=0.2)+
    geom_line(mapping=aes(sizes,power150,colour="N=150"),alpha=0.2)+
    geom_line(mapping=aes(sizes,power200,colour="N=200"),alpha=0.2)+
  labs(x = "Effect sizes", y="Power", title = "Plot of the power of t test vs. different effect sizes \n
    geom_hline(mapping=aes(colour="y=0.8",yintercept = 0.8))
```



## Part (b)

For each sample size, what effect size can you detect with a power of 0.8? Explain this result in words.

**From the plot above, a power of 0.8 can detect an effect size of 0.25 for N=200, about 0.2625 for N=150, 0.27 for N=100, and 0.50 for N=50. Recall that effect size is the difference in the current mean vs. the actual mean. Thus, the effect size that can be detected increases as the number of samples decreases.**

# Problem 3

```
set.seed(2019)
n <- 9 # Sample size
alpha <- 1 # Shape parameter
beta <- 10 # Scale parameter

x <- rgamma(n, alpha, beta) #A random sample to play with
ci <- t.test(x)$conf.int[1:2] #Get the confidence interval bounds
```

## Part (a)

The above piece of code simulates a sample from a gamma distribution and calcultes the confidence interval for the mean. Simulate 100 random samples and store the confidence interval and mean for each sample, arrange the dataframe by the value of the means.

## Part (b)

Create a plot of the 100 simulated confidence intervals with the true mean indicated by a line on the graph.

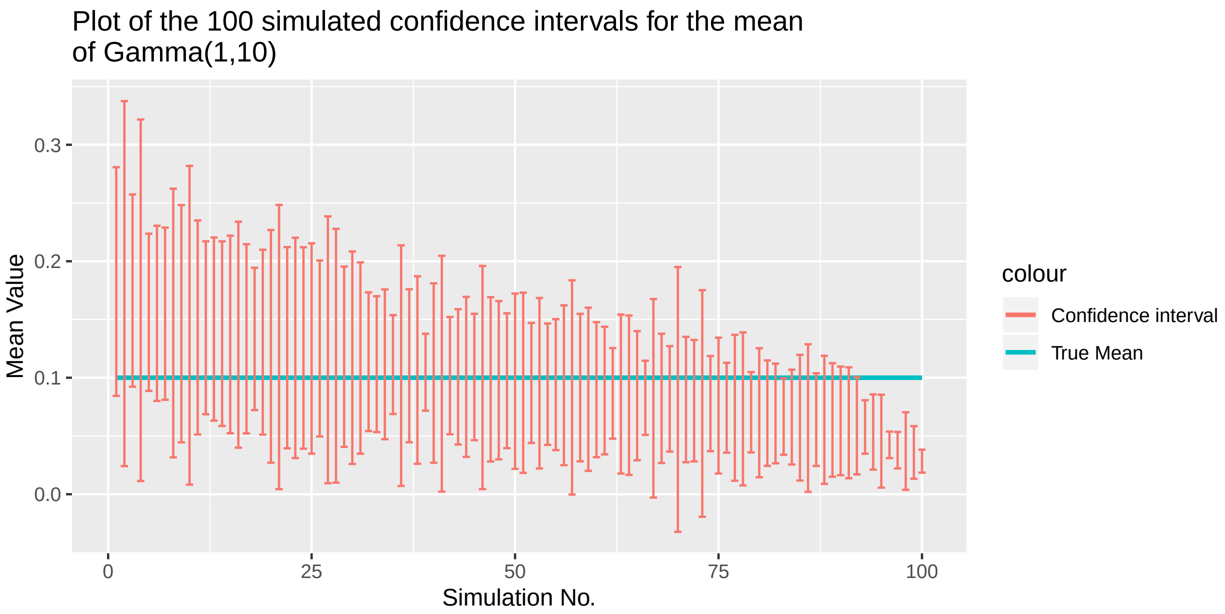
### Part (c)

Report the percentage of confidence intervals that successfully capture the true expected value (the coverage rate), and conversely, the percentage of confidence intervals that don't contain the true value (the error rate).

```r
set.seed(2019)
reps = 100 #No. of repetitions
n <- 9 # Sample size

alpha <- 1 # Shape parameter
beta <- 10 # Scale parameter

#empty matrices to store the confidence interval and mean
confint = matrix(0,nrow=reps,ncol=2)
mean = matrix(0,nrow=reps,ncol=1)

#For loop to compute the confidence interval and true mean for each simulated sample, repeated 1000 tim
for(i in seq(reps)){
  x <- rgamma(n, alpha, beta) #A random sample to play with
  confint[i,] <- t.test(x)$conf.int[1:2] #Get the confidence interval bounds
  mean[i] <- mean(x)
}

simul = tibble(means=mean[,1],lower_int=confint[,1],upper_int=confint[,2])%>%
  arrange(desc(means))
head(simul)
```

```
## # A tibble: 6 x 3
##   means lower_int upper_int
##   <dbl>     <dbl>     <dbl>
## 1 0.183    0.0844     0.281
## 2 0.181    0.0241     0.337
## 3 0.175    0.0923     0.257
## 4 0.167    0.0114     0.322
## 5 0.156    0.0887     0.224
## 6 0.155    0.0801     0.230
```

```r
ggplot(simul, aes(x = 1:reps, y = rep(alpha/beta,reps), color="True Mean")) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymax = upper_int, ymin = lower_int,color="Confidence interval"))+
  labs(x="Simulation No.", y="Mean Value", title="Plot of the 100 simulated confidence intervals for th
```



```
perc_with <- sum(apply(matrix(1:reps,ncol=1),1,function(x) between(alpha/beta,simul$lower_int[x],simul$
perc_without <- 1-perc_with
tibble(coverage_rate=perc_with, error_rate=perc_without)
```

```
## # A tibble: 1 x 2
##   coverage_rate error_rate
##           <dbl>      <dbl>
## 1          0.91     0.0900
```

## Part (d)

Is the error rate close to what it should be?

**The error rate is 0.09 which is slightly close to the expected value of 0.05.**

## Part (e)

Now run the simulation 1,000 times. Calculate the coverage rate and error rate and comment. Can you think of any reasons these intervals might have a higher error rate than they should?

```
set.seed(2019)
reps = 1000 #No. of repetitions
n <- 9 # Sample size

alpha <- 1 # Shape parameter
beta <- 10 # Scale parameter

#empty matrices to store the confidence interval and mean
confint = matrix(0,nrow=reps,ncol=2)
mean = matrix(0,nrow=reps,ncol=1)

#For loop to compute the confidence interval and true mean for each simulated sample, repeated 1000 tim
for(i in seq(reps)){
  x <- rgamma(n, alpha, beta) #A random sample to play with
  confint[i,] <- t.test(x)$conf.int[1:2] #Get the confidence interval bounds
  mean[i] <- mean(x)
}

simul = tibble(means=mean[,1],lower_int=confint[,1],upper_int=confint[,2])%>%
  arrange(desc(means))

perc_with <- sum(apply(matrix(1:reps,ncol=1),1,function(x) between(alpha/beta,simul$lower_int[x],simul$u
perc_without <- 1-perc_with
tibble(coverage_rate=perc_with, error_rate=perc_without)
```

```
## # A tibble: 1 x 2
##   coverage_rate error_rate
##           <dbl>      <dbl>
## 1         0.894      0.106
```