1. A group of researchers is running an experiment over the course of 30 months, with a single observation collected at the end of each month. Let $X_1, ..., X_{30}$ denote the observations for each month. From prior studies, the researchers know that

$$X_i \sim f_X(x),$$

but the mean μ_X is unknown, and they wish to conduct the following test

$$H_0 : \mu_X = 0$$
$$H_a : \mu_X > 0.$$

At month k, they have accumulated data $X_1, ..., X_k$ and they have the t-statistic

$$T_k = \frac{\bar{X} - 0}{S_k/\sqrt{n}}.$$

The initial plan was to test the hypotheses after all data was collected (at the end of month 30), at level $\alpha = 0.05$. However, conducting the experiment is expensive, so the researchers want to "peek" at the data at the end of month 20 to see if they can stop it early. That is, the researchers propose to check whether t_{20} provides statistically discernible support for the alternative. If it does, they will stop the experiment early and report support for the researcher's alternative hypothesis. If it does not, they will continue to month 30 and test whether t_{30} provides statistically discernible support for the alternative.

(a) What values of t_{20} provide statistically discernible support for the alternative hypothesis?

```r
mu0 <- 0
alpha <- 0.05
df_20 <- 20 -1
#get the critical value for statistically discernible support
critical_20 <- qt(p = 1 - alpha, df = df_20)
critical_20

## [1] 1.729133
```

The value $t_{20} = 1.729$ provides statisticaly discernible support for the alterative hypothesis.

(b) What values of t_{30} provide statistically discernible support for the alternative hypothesis?

```r
df_30 <- 30 -1
#get the critical value for statistically discernible support
critical_30 <- qt(p = 1 - alpha, df = df_30)
critical_30

## [1] 1.699127
```

The value $t_{30} = 1.699$ provides statisticaly discernible support for the alterative hypothesis.

(c) Suppose $f_X(x)$ is a Laplace distribution with $a = 0$ and $b = 4.0$. Conduct a simulation study to assess the Type I error rate of this approach.
Note: You can use the `rlaplace()` function from the `VGAM` package for `R` (Yee, 2010).

```r
#Conduct a simulation study to assess the Type I error
a <- 0
b <- 4.0
n.simulations <- 1000
type1.count <- 0 #count the number of times we got Type I error

#conduct the simulation
for (i in 1:n.simulations){
sim <- rlaplace(n = 30, location = a, scale = b)
#calculate the critical point
t_sim <- mean(sim)/ (sd(sim)/sqrt(30))

#check if t is larger than the critical point
if (t_sim > critical_30){
  type1.count = type1.count + 1
}
}

#calculate the rate of receiving Type I error
rate.type1 <- type1.count/n.simulations
rate.type1

## [1] 0.065
```

The Type I error rate for a Laplace distribution with $a = 0$ and $b = 4.0$ by using a simulation is $0.065 = 6.5\%$.

2. Perform a simulation study to assess the robustness of the T test. Specifically, generate samples of size $n = 15$ from the Beta(10,2), Beta(2,10), and Beta(10,10) distributions and conduct the following hypothesis tests against the actual mean for each case (e.g., $\frac{10}{10+2}$, $\frac{2}{10+2}$, and $\frac{10}{10+10}$).

   (a) What proportion of the time do we make an error of Type I for a left-tailed test?

```r
n <-15
#calculate the true means for each of beta distributions
mean.beta.10.2 <- 10/(10+2)
mean.beta.2.10 <- 2/(2+10)
mean.beta.10.10 <- 10/(10+10)

#left-tailed test for beta(10,2)
count.error.left.10.2 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 10, shape2 = 2)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "less", mu = mean.beta.10.2)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.left.10.2 <- count.error.left.10.2 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.left.10.2 <- count.error.left.10.2/n.simulations


#left-tailed test for beta(2,10)
count.error.left.2.10 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 2, shape2 = 10)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "less", mu = mean.beta.2.10)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.left.2.10 <- count.error.left.2.10 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.left.2.10 <- count.error.left.2.10/n.simulations


#left-tailed test for beta(10,10)
count.error.left.10.10 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 10, shape2 = 10)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "less", mu = mean.beta.10.10)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.left.10.10 <- count.error.left.10.10 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.left.10.10 <- count.error.left.10.10/n.simulations

rate.left.10.2

## [1] 0.035

rate.left.2.10

## [1] 0.066

rate.left.10.10

## [1] 0.058
```

We make an error of Type I for a left-tailed test 0.035 = 3.5% for Beta(10,2), 0.066 = 6.6% for Beta(2,10), and 0.058 = 5.8% for Beta(10,10).

(b) What proportion of the time do we make an error of Type I for a right-tailed test?

```
#right-tailed test for beta(10,2)
count.error.right.10.2 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 10, shape2 = 2)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "greater", mu = mean.beta.10.2)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.right.10.2 <- count.error.right.10.2 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.right.10.2 <- count.error.right.10.2/n.simulations


#right-tailed test for beta(2,10)
count.error.right.2.10 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 2, shape2 = 10)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "greater", mu = mean.beta.2.10)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.right.2.10 <- count.error.right.2.10 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.right.2.10 <- count.error.right.2.10/n.simulations


#right-tailed test for beta(10,10)
count.error.right.10.10 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 10, shape2 = 10)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "greater", mu = mean.beta.10.10)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.right.10.10 <- count.error.right.10.10 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.right.10.10 <- count.error.right.10.10/n.simulations

rate.right.10.2

## [1] 0.08

rate.right.2.10

## [1] 0.034

rate.right.10.10

## [1] 0.044
```

We make an error of Type I for a right-tailed test 0.08 = 8% for Beta(10,2), 0.034 = 3.4% for Beta(2,10), and 0.044 = 4.4% for Beta(10,10).

(c) What proportion of the time do we make an error of Type I for a two-tailed test?

```
#two-tailed test for beta(10,2)
count.error.two.10.2 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 10, shape2 = 2)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "two.sided", mu = mean.beta.10.2)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.two.10.2 <- count.error.two.10.2 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.two.10.2 <- count.error.two.10.2/n.simulations


#two-tailed test for beta(2,10)
count.error.two.2.10 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 2, shape2 = 10)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "two.sided", mu = mean.beta.2.10)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.two.2.10 <- count.error.two.2.10 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.two.2.10 <- count.error.two.2.10/n.simulations


#two-tailed test for beta(10,10)
count.error.two.10.10 <- 0
#perform the simulation
for (i in 1:n.simulations){
  #get a sample from beta distribution
  sample <- rbeta(n, shape1 = 10, shape2 = 10)
  #perform a t-test on the sample
  t_test <- t.test(sample, alternative = "two.sided", mu = mean.beta.10.10)
  #check if we got a type 1 error
  if (t_test$p.value < alpha){
count.error.two.10.10 <- count.error.two.10.10 +1
  }
}
#calculate the proportion of time we make a Type 1 error
rate.two.10.10 <- count.error.two.10.10/n.simulations

rate.two.10.2

## [1] 0.062

rate.two.2.10

## [1] 0.067

rate.two.10.10

## [1] 0.049
```

We make an error of Type I for a two-tailed test 0.062 = 6.2% for Beta(10,2), 0.067 = 6.7% for Beta(2,10), and 0.049 = 4.9% for Beta(10,10).

(d) How does skewness of the underlying population distribution effect Type I error across the test types?

For a right-skewed distribution, Beta(2,10), the Type I error is inflated for the right-tailed test. For a left-skewed distribution, Beta(10,2), the Type I error is inflated for the left-tailed test. For a symmetrical distribution, Beta(10,10), the Type I error is approximately 0.05 for all test types. Overall, for the left-tailed test, the right-skewed distribution has the smallest Type I error. For the right-tailed test, the left-skewed distribution has the smallest Type I error. For the two-tailed test, the symmetrical distribution has the smallest Type I error.

References

Yee, T. W. (2010). The VGAM package for categorical data analysis. *Journal of Statistical Software*, 32(10):1–34.