Red Giants: Chemical Abundance & Temperature

Haoyuan Chen, Anand Singh, Xiyan Chen, Shaojie Zhang

This pdf contains the codes of how to reproduce the results we gained and 6 visualizations. Only brief descriptions are included. For detailed analysis and conclusion, please refer to the final report.

First, we want to install the required packages in order to see the data in .h5 file that contains the information of 99705 red giants. "tidyverse" allows us to use required functions and the other packages allow us to convet .h5 file to tibbles.

```
#install.packages("tidyverse")
#install.packages("BiocManager")
#BiocManager::install("rhdf5")
#install.packages("devtools")
#devtools::install_github("r-lib/conflicted")
#tinytex::install_tinytex()
```

We need to activate these required packages.

```
library(tidyverse)
library(rhdf5)
#library(conflicted)
library(dplyr)
#filter(mtcars, cyl == 8)
```

We can now gain data from .h5 file using h5read() function. The column "name" represents the possible variables that can be used for the analysis.

```
header = h5ls("C:/Capstone/STA130_APOGEE.h5", all=TRUE)
glimpse(header)
```

```
## Rows: 22
## Columns: 12
## $ group     <chr> "/", "/", "/", "/", "/", "/", "/", "/", "/", "/", "/", "/", ~
## $ name      <chr> "al_h", "al_h_err", "c_h", "c_h_err", "ca_h", "ca_h_err", "f~
## $ ltype     <chr> "H5L_TYPE_HARD", "H5L_TYPE_HARD", "H5L_TYPE_HARD", "H5L_TYPE~
## $ cset      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ otype     <chr> "H5I_DATASET", "H5I_DATASET", "H5I_DATASET", "H5I_DATASET", ~
## $ num_attrs <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ dclass    <chr> "FLOAT", "FLOAT", "FLOAT", "FLOAT", "FLOAT", "FLOAT", "FLOAT~
## $ dtype     <chr> "H5T_IEEE_F32BE", "H5T_IEEE_F32BE", "H5T_IEEE_F32BE", "H5T_I~
## $ stype     <chr> "SIMPLE", "SIMPLE", "SIMPLE", "SIMPLE", "SIMPLE", "SIMPLE", ~
## $ rank      <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, ~
## $ dim       <chr> "99705", "99705", "99705", "99705", "99705", "99705", "99705~
## $ maxdim    <chr> "99705", "99705", "99705", "99705", "99705", "99705", "99705~
```

Now we extract the specific variables. These three variables are required for our analysis. We can now turn them into three tibbles.

```
teff = "C:/Capstone/STA130_APOGEE.h5" %>% h5read("teff") %>% as_tibble()
MG_H = "C:/Capstone/STA130_APOGEE.h5" %>% h5read("mg_h") %>% as_tibble()
N_H = "C:/Capstone/STA130_APOGEE.h5" %>% h5read("n_h") %>% as_tibble()
```

Question 1: Is the mean effective temperature for stars with negative abundance of magnesium the same as the mean temperature for stars with positive abundance of magnesium?

To answer this question, we want to conduct a two sample hypothesis testing.

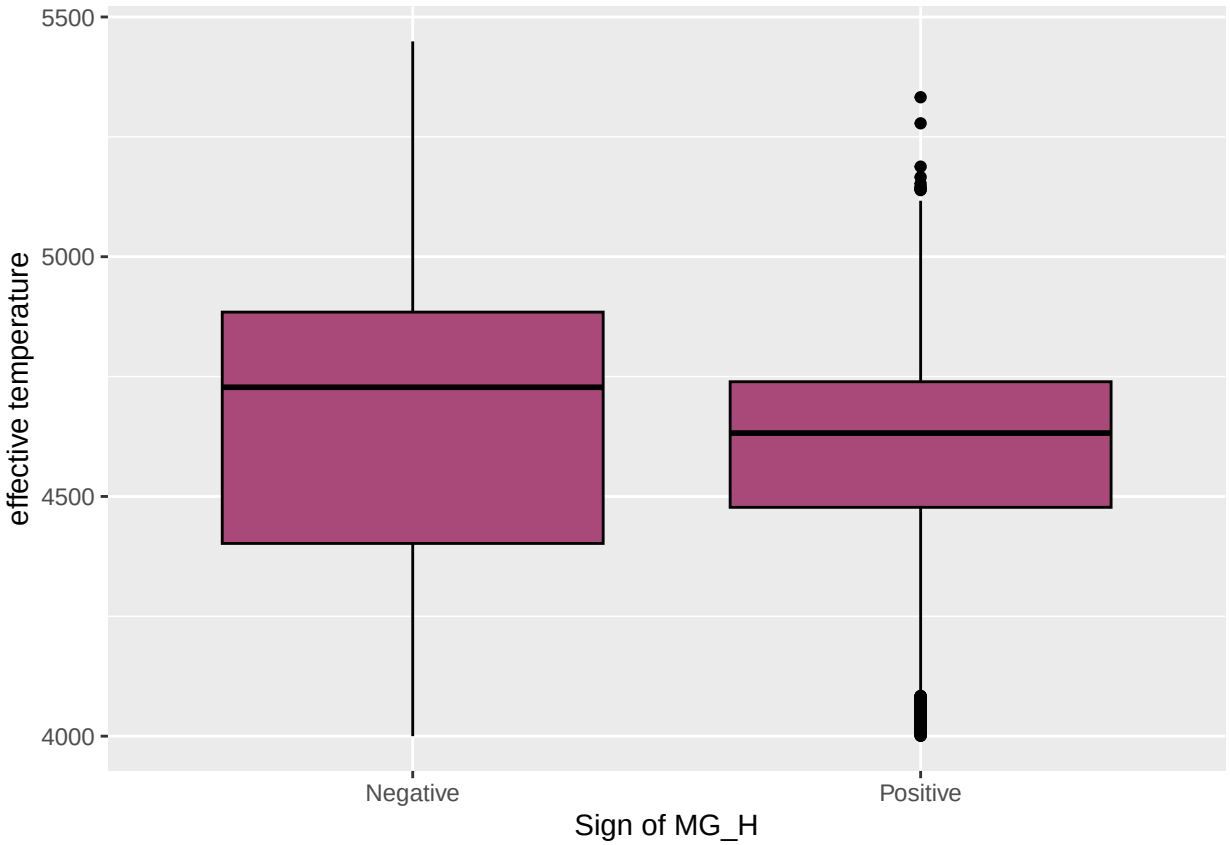
```
#First combine the two variables MG_H and N_H into a tibble and rename the columns.
MG_teff <- cbind(MG_H, teff)
colnames(MG_teff) = c("MG_H", "teff")
#Create a new column name, Negative_or_Positive_MG,
MGTE = MG_teff %>% mutate(Negative_or_Positive_MGH = case_when(MG_H < 0 ~ "Negative",
MG_H > 0 ~ "Positive")) %>% select(teff, Negative_or_Positive_MGH)
head(MGTE)
```

```
##       teff Negative_or_Positive_MGH
## 1 5031.264                 Negative
## 2 4975.689                 Negative
## 3 4981.525                 Negative
## 4 4073.770                 Negative
## 5 4757.323                 Negative
## 6 4669.081                 Negative
```

Now we have two groups. Notice that each group can be characterized by the mean of their effective temperature, We would assume M_{pos}, the mean teff of stars with positive MG_H is the same as M_{neg}, the mean teff of stars with negative MG_H. Therefore, our null hypothesis and alternative hypothesis are:

H_0: M_{pos} - $M_{\text{neg}} = 0$ H_1: M_{pos} - M_{neg} != 0

```
set.seed(12021)
#Now we want to take a sample from the population
sample_MGTE = MGTE %>% filter(!is.na(Negative_or_Positive_MGH)) %>% slice_sample(n=28000)
#Let's use a side by side box plot to do comparison
sample_MGTE %>% ggplot(aes(x=Negative_or_Positive_MGH,
y=teff)) + geom_boxplot(color="black", fill="#A8497A") + labs(x="Sign of MG_H",
y="effective temperature")
```




```
glimpse(sample_MGTE)
```

```
## Rows: 28,000
## Columns: 2
## $ teff                      <dbl> 4739.082, 4635.750, 4443.461, 4992.466, 4383.~
## $ Negative_or_Positive_MGH <chr> "Positive", "Positive", "Positive", "Negative~
```

We then want to find the difference between two means from this sample. The difference would be our observed test statistic.

```
observed <- sample_MGTE %>% group_by(Negative_or_Positive_MGH) %>%
  summarize(means = mean(teff), .groups="drop") %>%
summarize(value = diff(means))
observed
```

```
## # A tibble: 1 x 1
##   value
##   <dbl>
## 1 -61.9
```

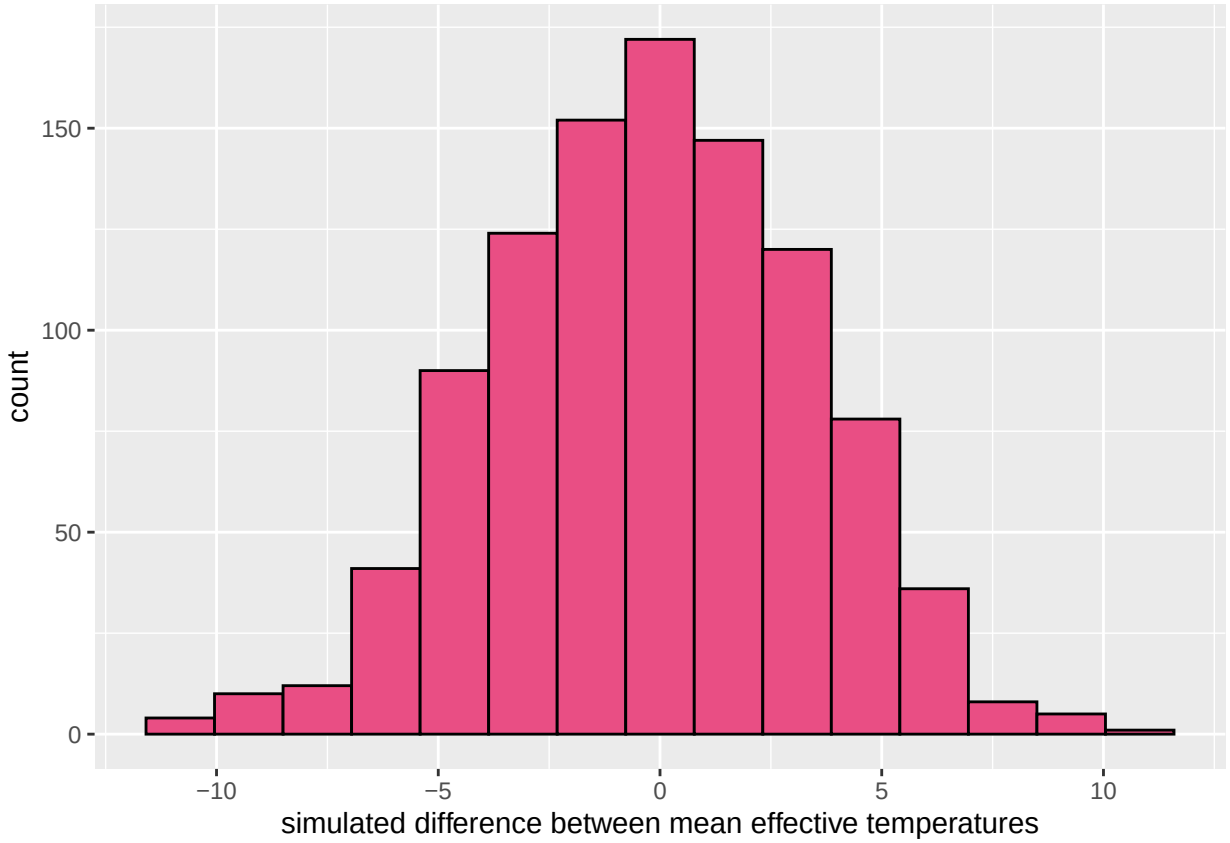
Since this is a two-sample hypothesis test about finding the difference between means, we need to apply a permutation test to simulate under the assumption that the null hypothesis is True.

```r
set.seed(101)
R = 1000
simulate_result1K = rep(NA, R)
#Using a loop allows us to collect the data gained from each simulation
for(i in 1:R){
simulate_data = sample_MGTE %>%
mutate(Negative_or_Positive_MGH = sample(Negative_or_Positive_MGH, replace=FALSE))

sim_dif <- simulate_data %>% group_by(Negative_or_Positive_MGH) %>%
  summarize(means = mean(teff), .groups="drop") %>%
summarize(value = diff(means))

simulate_result1K[i] = as.numeric(sim_dif)}
#Let's create a summary table to see the results
result_tibble = tibble(diff_of_mean = simulate_result1K)
result_tibble %>% summarize(n=n(),mean_diff=mean(diff_of_mean),median_diff=median(diff_of_mean),
min_diff=min(diff_of_mean), max_diff=max(diff_of_mean))
```

```
## # A tibble: 1 x 5
##       n mean_diff median_diff min_diff max_diff
##   <int>     <dbl>       <dbl>    <dbl>    <dbl>
## 1  1000    -0.211      -0.208    -11.1     10.6
```

We could create a histogram to visualize the result of the permutation test.

```r
result_tibble %>% ggplot(aes(x=diff_of_mean))+ geom_histogram(color="black",fill="#e94e84",
bins = 15) + labs(x="simulated difference between mean effective temperatures")
```



Above is the histogram that shows the distribution of all simulated test statistics. Since this is a two-sided test, we need to calculate the two-sided p-value.

```
n = 0
for (i in simulate_result1K) {
if (abs(i) >= abs(observed)) {
n = n + 1}}
p_value_mgte = n/R
p_value_mgte
```

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

Assume a significant level of 0.05.

```
ifelse(p_value_mgte < 0.05, "Reject", "Fail to reject")
```

```
## [1] "Reject"
```

The p value is equal to 0 and we reject the null hypothesis.

Question 2: Is the mean effective temperature of the stars equal to 4286 Kelvin or not?

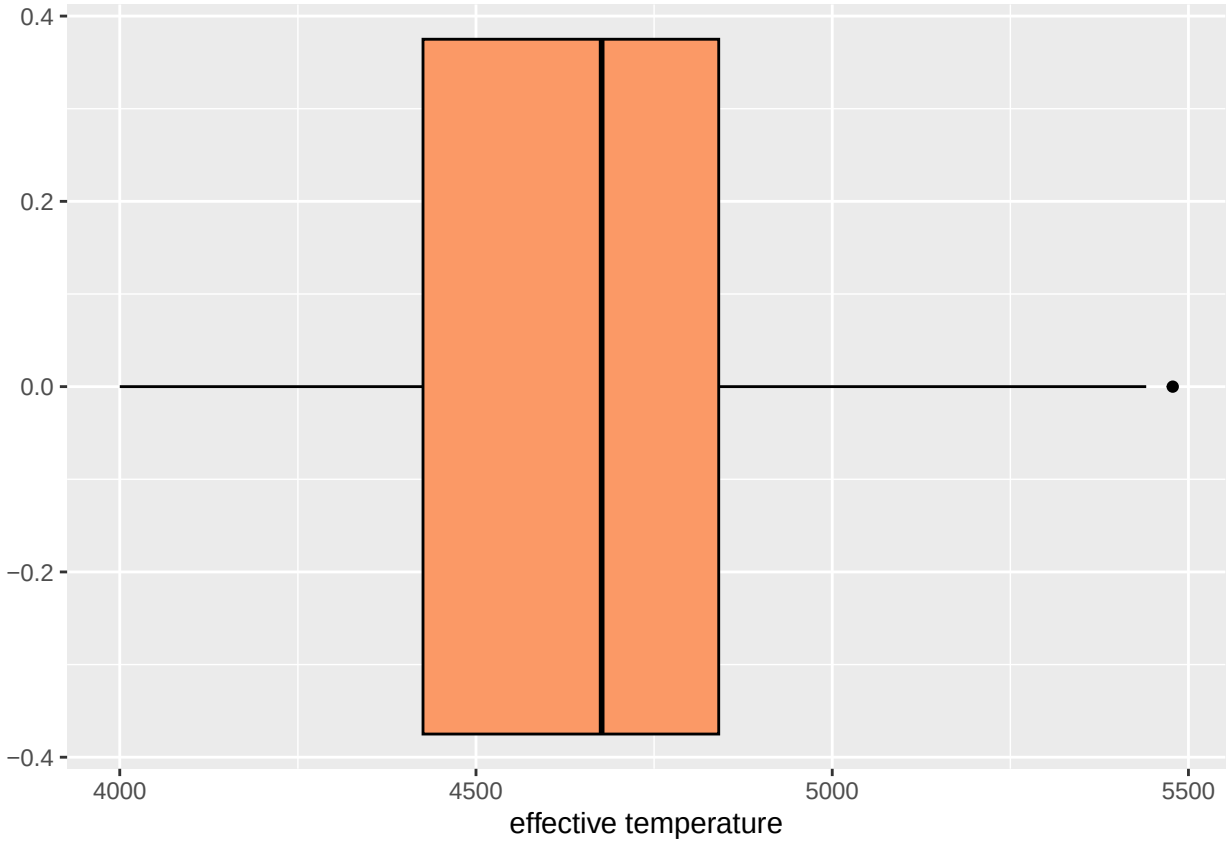
In this question, we would estimate the true mean effective temperature of the stars by applying bootstrapping. The first step is to select a random sample of size 30000 from the 99705 red giants. We will then

calculate the mean and plot a box plot.

```
set.seed(111)
sample_s30000 = teff %>% sample_n(size = 30000)
sample_s30000 %>% summarize(mean_effective_temperature =mean(value),n=n())
```

```
## # A tibble: 1 x 2
##   mean_effective_temperature     n
##                        <dbl> <int>
## 1                      4628. 30000
```

```
sample_s30000 %>% ggplot(aes(x=value))+geom_boxplot(color="black",
fill="#FB9966")+labs(x="effective temperature")
```



Now it's time to do bootstrapping. We would resample with the same size as the randomly chosen sample.

```
set.seed(333)
ssize = 30000
bootstap_sample = sample_s30000 %>% sample_n(size= ssize, replace = TRUE)
bootstap_sample %>% summarize(mean_effective_temperature =mean(value),n=n())
```

```
## # A tibble: 1 x 2
##   mean_effective_temperature     n
##                        <dbl> <int>
## 1                      4629. 30000
```

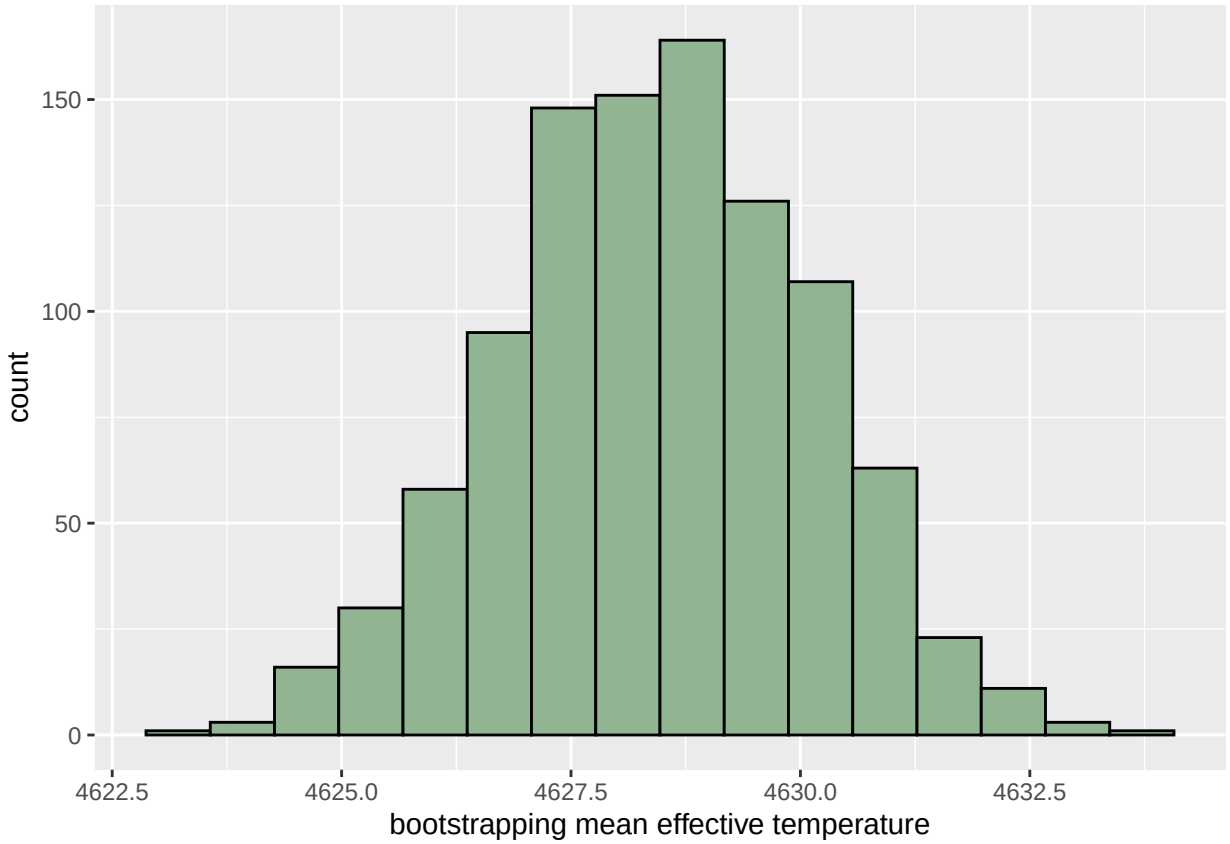
Above shows how do we get one sample statistic from one bootstrap sample. The statistics we gained is different from the original sample's mean. We now want to repeat this resampling process for 1000 times.

```
set.seed(1117)
R1 = 1000
storage_mean = rep(NA, R1)
for(i in 1:R1){test_r = sample_s30000 %>% sample_n(size= ssize, replace = TRUE)
storage_mean[i] = as.numeric(test_r %>% summarize(mean(value)))}
result_t = tibble(teff_mean = storage_mean)
glimpse(result_t)
```

```
## Rows: 1,000
## Columns: 1
## $ teff_mean <dbl> 4629.348, 4627.620, 4628.629, 4628.553, 4627.296, 4629.155, ~
```

Now we have obtained 1000 mean simulated test statistics. We will construct a confidence interval. First, let's make a histogram.

```
result_t %>% ggplot(aes(x=teff_mean)) + geom_histogram(color="black", fill="#91B493",
bins = 16)+ labs(x="bootstrapping mean effective temperature")
```



Now, let's find the 90% confidence interval.

```r
quantile(result_t$teff_mean, c(0.1, 0.9))
```

```
##      10%      90%
## 4626.264 4630.573
```

How about taking a different confidence level? Let's try to find the 99% confidence interval.

```r
quantile(result_t$teff_mean, c(0.01, 0.99))
```

```
##       1%      99%
## 4624.625 4632.352
```

Notice that the confidence intervals do not contain 4286K.

Question 3: Is the proportion of stars that have a positive abundance of Nitrogen greater than 50%?

We would conduct a one-sided hypothesis testing for this question. The null hypothesis is that the proportion of stars that have a positive abundance of Nitrogen is equal to 50%. The alternative hypothesis is that the the proportion of stars that have a positive abundance of Nitrogen is greater than 50%

H_0: $P_{\text{pos}} = 0.5$ H_1: $P_{\text{pos}} > 0.5$

To begin with, let's take a random sample from the population and find out how many stars have positive abundance of Nitrogen.

```r
#Take a sample of size 35000
set.seed(231)
obs_test_stat = N_H$value %>% sample(size = 35000)
sample = tibble(N_H = obs_test_stat)
count_positive = sum(sample$N_H > 0)
Observed_P_n = count_positive/35000
Observed_P_n
```

```
## [1] 0.4986
```

```r
glimpse(sample)
```

```
## Rows: 35,000
## Columns: 1
## $ N_H <dbl> 0.3353400, -0.1715200, 0.0833060, -0.5326090, -0.3161800, 0.442490~
```

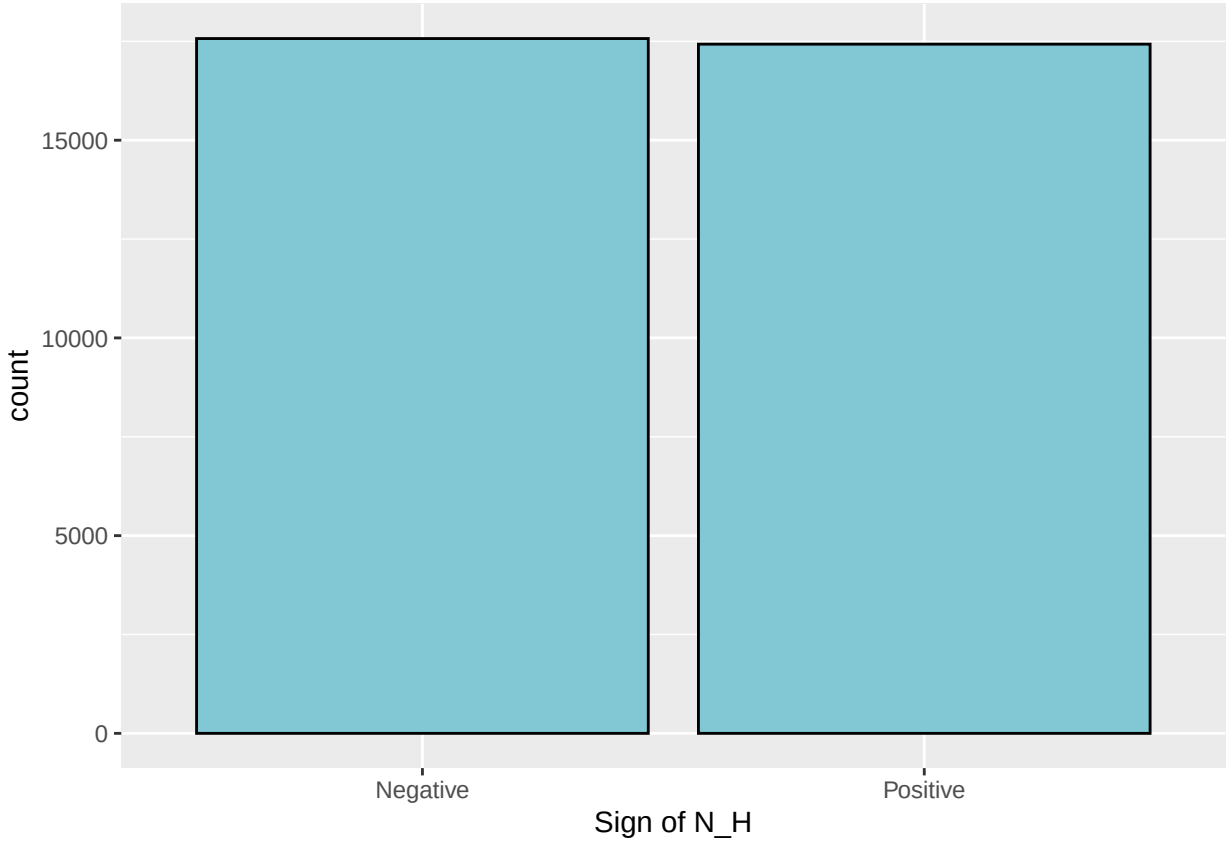
Then we will start to simulate. We would assume the probability of getting a negative N_H is the same as getting a positive N_H.

```r
set.seed(20120)
n_sim_sample = 35000
options = c("Negative", "Positive")
#Set the probability of getting a "Positive" response as 0.5
```

```
p = 0.5
p_options = c(p, 1 - p)
responses = sample(options, size=n_sim_sample, prob=p_options, replace=TRUE)
Positive_NH = sum(responses == "Positive")
```

By creating a bar plot, we could see the simulated result more clearly.

```
tibble_sim = tibble(result = responses)
tibble_sim %>% ggplot(aes(x=result))+geom_bar(color="black",
fill="#81C7D4")+labs(x="Sign of N_H")
```



Let's find the simulated test statistic, the proportion of positive N_H.

```
positive_fraction_sim = sum(responses == "Positive") / n_sim_sample
positive_fraction_sim
```

```
## [1] 0.4979714
```

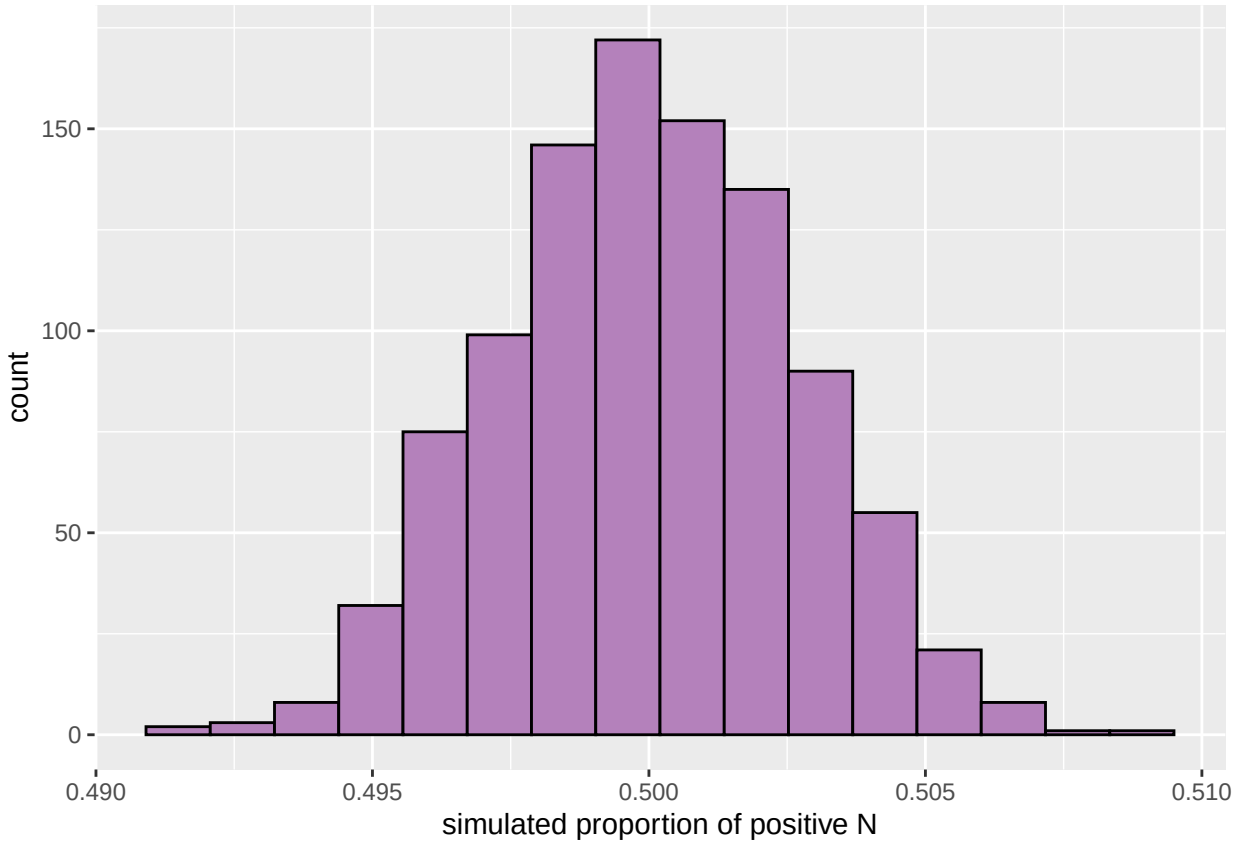
We get 0.4979, and we know that the observed test statistic is 0.4986. But still, we can't make any conclusion about the null hypothesis, so we have to repeat the same simulation process for many times. We decided to repeat for 1000 times.

```
set.seed(11111)
N = 1000
store_sim = rep(NA, N)
for(i in 1:N){simulated_x = sample(c("Negative", "Positive"),
size=n_sim_sample, prob=(c(0.5,0.5)), replace=TRUE)
sim_positive = sum(simulated_x == "Positive")/n_sim_sample
store_sim[i] = sim_positive}
tibble2 = tibble(P_N = store_sim)
glimpse(tibble2)
```

```
## Rows: 1,000
## Columns: 1
## $ P_N <dbl> 0.5043143, 0.5024857, 0.4986571, 0.5004286, 0.5021143, 0.4993143, ~
```

Now we have 1000 test statistics, by generating a histogram, we can see the distribution of the simulated test statistics and find the p-value.

```
tibble2 %>% ggplot(aes(x=P_N))+geom_histogram(color="Black",fill="#B481BB",
bins=16)+labs(x="simulated proportion of positive N")
```



We will calculate the one sided p value:

```
p_value = tibble2 %>% filter(P_N >= Observed_P_n) %>% summarize(p_value = n()/N)
p_value
```

```
## # A tibble: 1 x 1
##   p_value
##     <dbl>
## 1   0.697
```

Assume a significant level of 0.05,

```
ifelse(p_value < 0.05, "Reject", "Fail to reject")
```

```
##      p_value
## [1,] "Fail to reject"
```

We failed to reject the null hypothesis.