Randomization using the randomization function

by Craig W. Slinkman

November 2, 2015

To make your life simpler I provided a function named randomization that will compute the randomization distribution for completely randomized experimental designs.

In this document we will give three difference examples:

* Testing a true null hypothesis
* Testing a false null hypothesis
* Testing a null hypothesis when the response variable is binary.

# Testing a true null hypothesis

In this section we will test a response variable that has the same mean and standard deviation. Both treatment levels will have a mean and a standard deviation .

## Null and alternative hypotheses

The null and alternative hypotheses are

Note that we start with a two=tail test.

## Simulating the data

We will sample from these two populations below.use computer simulation to generate our data. We use the R-function to generate two sample of size 25 with a mean of 75 and a standard deviation of 6. We round then to zero decimal places,

MU <- 75  
SIGMA <- 6  
n1 <- 25  
n2 <- 25  
  
y1 <- round( rnorm( n1, MU, SIGMA ), 0 )  
y2 <- round( rnorm( n2, MU, SIGMA ), 0 )  
y1

## [1] 66 77 67 84 68 70 72 74 67 77 76 66 76 74 77 77 75 75 62 86 81 63 69  
## [24] 81 70

y2

## [1] 72 73 65 68 76 77 73 67 69 76 69 79 79 83 73 79 68 77 67 77 84 68 78  
## [24] 78 70

## Computung the test statistic

The test statistic for the randomization test is the difference in the means of the treatment group. In R we compute the test statistic to two decimal places as follows:

testStatistic <- round( mean(y1) - mean(y2), 2)  
testStatistic

## [1] -0.6

To see if the test statistic we must compute a randomization distribution using the experimental data. We will use the randomization distribution to answer the question how likely are we to observe a test statistic with this a value or a more extreme value if the null hypothesis is true.

## The randomization function

I have written an R-function called . This function computes the randomization distribution from the observed experimental data and returns it as a R-vector. The arguments (inputs) to this function are given in the table below:

|  |  |
| --- | --- |
| Argument | Description |
| y1 | The observed response values for treatment level 1. |
| y2 | The observed response values for treatment level 2. |
| reps | The number of replications to compute. If this argument is left out then 10,000 replications will be used. |

## Loading the randomization function

Before we can use the randomization function we must load it. R uses the statement to load the function. The code below shows you how to do this.

source( 'randomization.r' )

## Computing the randomization distribution

The values of the response variable are stored in the variables and . We will create a randomization distribution of size 10,000 with the R-code givewn below.

distribution <- randomization( y1, y2 )

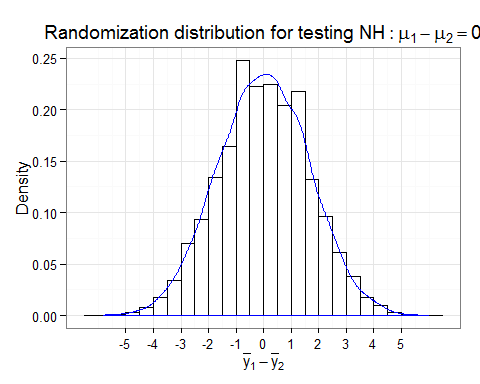
## Plotting the randomization distribution

We should always plot the randomization distribution and look at the shape of the distribution to see if our results make sense in light of our experimental data. We will use ggplot2 histogram function to plot this data.

require( ggplot2 )

## Loading required package: ggplot2

ggplot( data.frame( distribution ),  
 aes( x= distribution )) +  
 geom\_histogram( aes( y=..density..),  
 binwidth=0.50,  
 color="black",  
 fill="white" ) +  
 geom\_density( color="blue" ) +  
 scale\_x\_continuous( breaks=seq( -5, 5, 1 )) +  
 xlab( expression( bar(y)[1] - bar(y)[2] ) ) +  
 ylab( "Density" ) +  
 ggtitle( expression( paste(   
 "Randomization distribution for testing ",  
 NH: mu[1] - mu[2] == 0))) +  
 theme\_bw()



## Computing the p-value

We compute the p-value in the following manner. For a two tail test we count the observation in both tails where the absolute value of randomized distribution values is greater than or equal to absoulute value o p

length( distribution[ abs( distribution ) >=abs(testStatistic) ] )

## [1] 7030

We can now compute the p-value by dividing by the number of replications.

pValue <- length( distribution[ abs( distribution) >=  
 abs(testStatistic) ] ) /   
 length( distribution )   
pValue

## [1] 0.703

pValue <- round( pValue, 4)

## Statistical conclusion

Since I did not specify a level of significance we will use a significance level or 5%. Therefore, we do not have enough sample evidence to reject the null hypotheis.

In this case we did not make a Type I error.

# Testing a false null hypothesis

Suppose that we run an experiment using a new technology that should increase the mean time for lithoum-ion battery failures. We will pretend that we know the population means for each of the two treatment groups. The mean for the control group is 1200 hours. The expermental group is 1300 hours. Our experiment will consist of 25 replications for the control and the treatment groups.

## The data

rm(list=ls())  
  
Control <- round( -1200 \* log( runif( 50, 0, 1 )), 1 )  
New <- round( -2500 \* log( runif( 50, 0, 1 )), 1 )

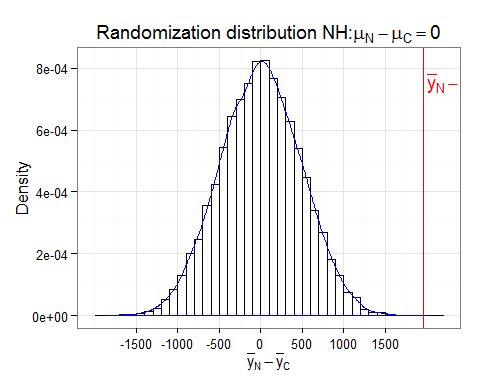
## The null and alternative hypotheses

## Compting the test statistic

testStatistic <- mean(New) - mean( Control )

## Computing the randomization distribution

source('randomization.r')  
  
distribution <- randomization( New, Control )  
  
  
## Ranomization distribution  
  
require( ggplot2 )  
ggplot( data.frame( distribution),  
 aes( x=distribution )) +  
 geom\_histogram( aes(y=..density..),  
 binwidth=100,  
 color="black",  
 fill="White") +  
 geom\_density( color="blue" ) +  
 geom\_vline( x=testStatistic,  
 color="red") +  
 annotate( "text",   
 x=testStatistic+50,  
 y=0.00075,  
 label="bar(y)[N] - bar(y)[C]",  
 parse=TRUE,  
 hjust=0,  
 color="Red" ) +  
 scale\_x\_continuous( breaks=seq( -1500, 1500, 500 )) +  
 xlab( expression( bar(y)[N] - bar(y)[C] ) ) +  
 ylab( "Density" ) +  
 ggtitle( expression( paste(   
 "Randomization distribution NH:",   
 mu[N] - mu[C] == 0))) +  
 theme\_bw()



## P-value

This is a one-tailed test on the upper tail so we compute the p-value as follows.

pValue <-   
 length(distribution[distribution>testStatistic]) /length(distribution)  
pValue

## [1] 0

## Statistical conclusion

Becauw the p-value is greaster than 5% we can not rehect the null hypothesis.