Power study on real data (B=10000) TEMPORARY

03-05-2023

Digits dataset

The aim is to compare on Digits dataset (available at http://odds.cs.stonybrook.edu/pendigits-dataset) the performance of three closed testing procedures, which respectively use Simes local test with and without Storey estimator for the proportion of true null hypotheses and Wilcoxon-Mann-Whitney local test.

Digits dataset consists of 6870 observations, among which $n_{inliers} = 6714$ items are inliers and the remaining $n_{outliers} = 156$ are outliers. We will denote by n, l, m respectively the train set, the calibration set and the test set size. And reproducing the same setting as in [1], we have that $n + l = n_{inliers}/2$, $l = min\{2000, n/2\}$ and $m = min\{2000, n/3\}$. In the case of Digits dataset we obtain

$$n + l = 6714/2, \;\; l = n/3, \;\; m = n/3$$

from which $n = 2517.75, \;\; l = 839.25, \;\; m = 839.25$. Arbitrarily, we choose to set

$$n = 1258, \;\; l = 2099, \;\; m = 420$$

in order to have exact control of type I errors at the significance level $\alpha = m/(l + 1) = 0.2$.

R functions and libraries

```
library(nout)
library(R.matlab)
library(isotree)
library(tictoc)


critWMW2 = function(m, n, alpha=0.1, m.exact=5, exact=F){

  # exact=F
  if(exact==F){
    if(n>=10^3){
      crit = sapply(1:m, function(k) stats::qnorm(alpha, mean=((m-k+1)*n/2-0.5),
                                                  sd = sqrt((m-k+1)*n*((m-k+1)+n+1)/12),
                                                  lower.tail = F))
    }
    else{
      if(m>m.exact){
        mm=m-m.exact
        crit2 = sapply(1:mm, function(k) stats::qnorm(alpha, mean=((m-k+1)*n/2-0.5),
                                                      sd = sqrt((m-k+1)*n*((m-k+1)+n+1)/12),
                                                      lower.tail = F))

        crit1 = sapply((mm+1):m, function(k) stats::qwilcox(p=alpha, m=m-k+1, n=n,
                                                            lower.tail = FALSE))
        crit=c(crit2, crit1)
```

```r
      }
      else{
        crit = sapply(1:m, function(k) stats::qwilcox(p=alpha, m=m-k+1, n=n,
                                                       lower.tail = FALSE))
      }
    }
    res = list("m" = m, "n" = n, "crit.vals" = crit, "alpha" = alpha)
    class(res) = "crit.vals.info"
  }

  # exact=T
  else{
    if(n>20){
      if(m>m.exact){
        mm=m-m.exact
        crit2 = sapply(1:mm, function(k) stats::qnorm(alpha, mean=((m-k+1)*n/2-0.5),
                                                       sd = sqrt((m-k+1)*n*((m-k+1)+n+1)/12),
                                                       lower.tail = F))

        crit1 = sapply((mm+1):m, function(k) stats::qwilcox(p=alpha, m=m-k+1, n=n,
                                                             lower.tail = FALSE))
        crit=c(crit2, crit1)
      }
      else{
        crit = sapply(1:m, function(k) stats::qwilcox(p=alpha, m=m-k+1, n=n,
                                                       lower.tail = FALSE))
      }
    }

    if(n<=20){
      crit = sapply(1:m, function(k) stats::qwilcox(p=alpha, m=m-k+1, n=n,
                                                     lower.tail = FALSE))
    }
  }

  # sign_level = sapply(crit, function(k) stats::dwilcox(x=k, m=m-k+1, n=n))
  #
  # res = list("m" = m, "n" = n, "crit.vals" = crit,
  #            "sign_level" = sign_level,"alpha" = alpha)

  res = list("m" = m, "n" = n, "crit.vals" = crit,"alpha" = alpha)
  class(res) = "crit.vals.info"

  return(res)
}




sim_realdata = function(B, dataset, m1, m, n, l, in_index,
                        out_index=NULL, alpha=m/(l+1), lambda = 0.5){

  m0=m-m1
```

```r
if(m1!=0 & is.null(out_index)){
  stop("Error: arg out_index must be initialized.")
}

# if(m!=(m1+m0)){
#   stop("Error: equation m=m1+m0 must be verified.")
# }

if(m1!=0){
  tr_ind = sample(in_index, size = n)
  tr = dataset[tr_ind,]
  iso.fo = isolation.forest(tr, ndim=ncol(dataset), ntrees=10, nthreads=1,
                            scoring_metric = "depth", output_score = TRUE)
  in_index2 = setdiff(in_index, tr_ind)

  crit=critWMW2(m=m, n=l, alpha=alpha, exact = T)

  d_WMW = rep(0,B)
  d_Simes = rep(0,B)
  d_StoSimes = rep(0,B)
  d_BH = rep(0,B)
  d_StoBH = rep(0,B)
  uniques_cal = rep(0,B)
  uniques_te = rep(0,B)

  for(b in 1:B){
    cal_ind = sample(in_index2, size = l)
    in_index3 = setdiff(in_index2, cal_ind)
    tein_ind = sample(in_index3, size = m0)
    teout_ind = sample(out_index, size = m1)

    cal = dataset[cal_ind,]
    te = dataset[c(tein_ind, teout_ind),]

    S_cal = predict.isolation_forest(iso.fo$model, cal, type = "score")
    S_te = predict.isolation_forest(iso.fo$model, te, type = "score")

    uniques_cal[b] = unique(S_cal) == length(S_cal)
    uniques_te[b] = unique(S_te) == length(S_te)
    d_WMW[b] = d_mannwhitney(S_X=S_cal, S_Y=S_te, crit=crit)
    d_Simes[b] = d_Simes(S_X=S_cal, S_Y=S_te, alpha=alpha)
    d_StoSimes[b] = d_StoreySimes(S_X=S_cal, S_Y=S_te, alpha=alpha)
    d_BH[b] = d_benjhoch(S_X=S_cal, S_Y=S_te, alpha=alpha)
    d_StoBH[b] = d_StoreyBH(S_X=S_cal, S_Y=S_te, alpha=alpha)
  }
}

else{
  tr_ind = sample(in_index, size = n)
  tr = dataset[tr_ind,]
  iso.fo = isolation.forest(tr, ndim=ncol(dataset), ntrees=10, nthreads=1,
                            scoring_metric = "depth", output_score = TRUE)
  in_index2 = setdiff(in_index, tr_ind)
```

```r
    crit=critWMW2(m=m, n=l, alpha=alpha, exact = T)

    d_WMW = rep(0,B)
    d_Simes = rep(0,B)
    d_StoSimes = rep(0,B)
    d_BH = rep(0,B)
    d_StoBH = rep(0,B)
    uniques_cal = rep(0,B)
    uniques_te = rep(0,B)

    for(b in 1:B){
      cal_ind = sample(in_index2, size = l)
      in_index3 = setdiff(in_index2, cal_ind)
      te_ind = sample(in_index3, size = m0)

      cal = dataset[cal_ind,]
      te = dataset[te_ind,]

      S_cal = predict.isolation_forest(iso.fo$model, cal, type = "score")
      S_te = predict.isolation_forest(iso.fo$model, te, type = "score")

      uniques_cal[b] = unique(S_cal) == length(S_cal)
      uniques_te[b] = unique(S_te) == length(S_te)
      d_WMW[b] = d_mannwhitney(S_X=S_cal, S_Y=S_te, crit=crit)
      d_Simes[b] = d_Simes(S_X=S_cal, S_Y=S_te, alpha=alpha)
      d_StoSimes[b] = d_StoreySimes(S_X=S_cal, S_Y=S_te, alpha=alpha)
      d_BH[b] = d_benjhoch(S_X=S_cal, S_Y=S_te, alpha=alpha)
      d_StoBH[b] = d_StoreyBH(S_X=S_cal, S_Y=S_te, alpha=alpha)
    }
  }

  discov = as.data.frame(cbind("d_BH"=d_BH, "d_StoBH"=d_StoBH, "d_Simes"=d_Simes,
                               "d_StoSimes"=d_StoSimes, "d_WMW"=d_WMW))
  colnames(discov) = c("BH", "BHSto", "CTSim", "CTSimSto", "CTWMW")
  mean.discov = apply(discov, MARGIN = 2, FUN = mean)

  powerGlobalNull = as.data.frame(cbind("d_BH"=d_BH>0, "d_StoBH"=d_StoBH>0, "d_Simes"=d_Simes>0,
                                        "d_StoSimes"=d_StoSimes>0, "d_WMW"=d_WMW>0))
  colnames(powerGlobalNull) = c("BH", "BHSto", "CTSim", "CTSimSto", "CTWMW")
  mean.powerGlobalNull = apply(powerGlobalNull, MARGIN = 2, FUN = mean)

  return(list("discoveries"=discov, "mean.discoveries" = mean.discov,
              "powerGlobalNull"=powerGlobalNull, "mean.powerGlobalNull"=mean.powerGlobalNull,
              "uniques_te"=uniques_te, "uniques_cal"=uniques_cal, "m1"=m1, "alpha"=alpha))
}
```

Load the data and set the parameters ad described above.

```r
set.seed(321)

# Initializing parameters
n = 2518
l = 2099
m = 420
```

```r
alpha = m/(l+1)

data = readMat("G:\\Il mio Drive\\PHD\\Progetto di ricerca\\Conformal Inference Project\\Simulazioni\\7
dataset = cbind(data$X, data$y); colnames(dataset)[ncol(dataset)] = "y"
in_ind = which(dataset[,ncol(dataset)]==0)
out_ind = which(dataset[,ncol(dataset)]==1)
```

We fixed the train set on which we train the isolation forest algorithm and we generate $B = 1000$ calibration and test sets. For each $b = 1, \ldots, B$ we compute the number of discoveries obtained by Benjamini-Hochberg procedure with and without Storey's estimator for the proportion of true null hypotheses, by closed testing using Simes local test with and without Storey estimator and by closed testing using Wilcoxon-Mann-Whitney local test.

All inliers

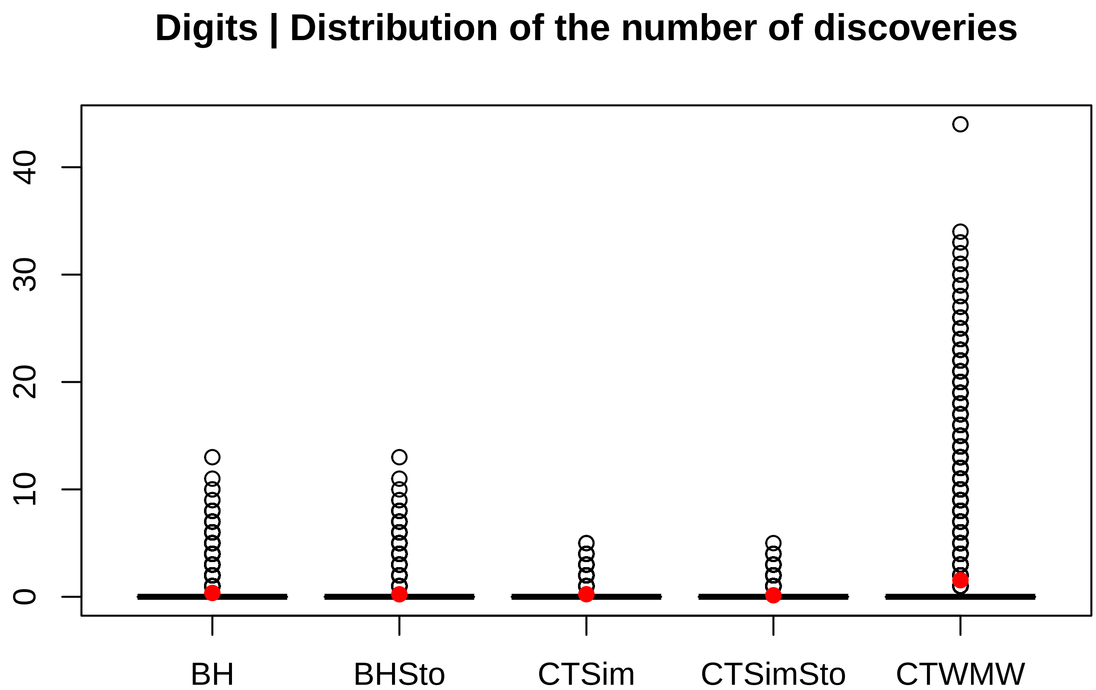
We now set the proportion of inliers equal to 1, so that the number of outliers $m_1 = 0$.

```r
B=10^4
m1=0

tic()
res = sim_realdata(B=B, in_index=in_ind, out_index=out_ind, dataset=dataset,
                   alpha=alpha,l=l, n=n, m=m, m1=m1)
toc()
```

```
## 623.54 sec elapsed
```

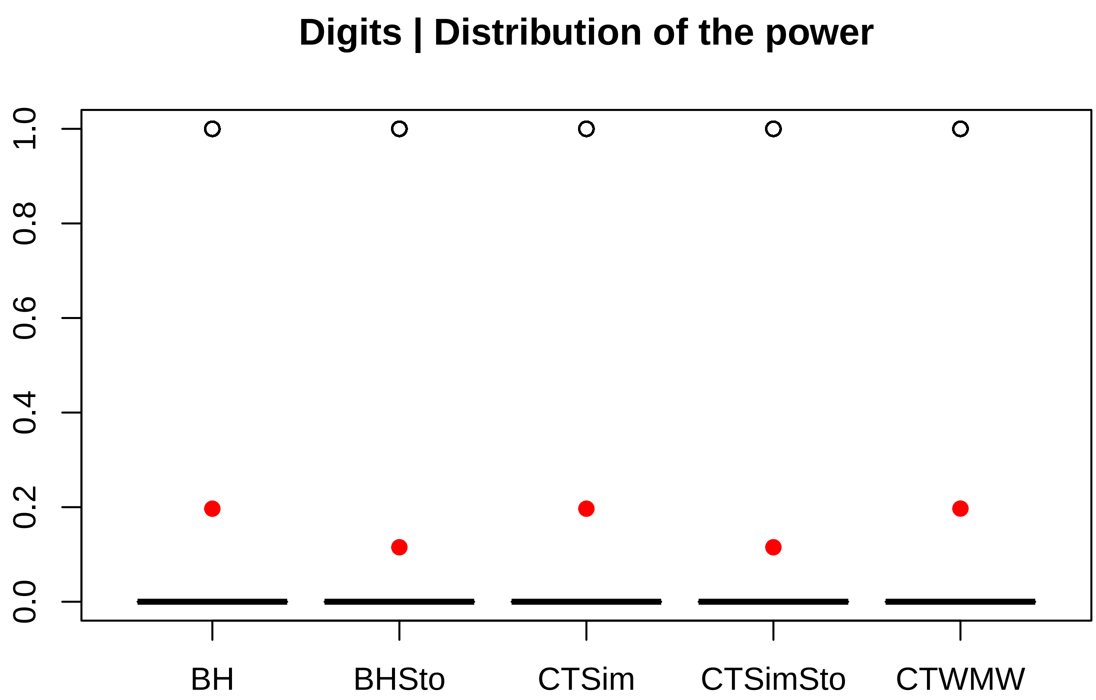
```r
boxplot(res$discoveries, main="Digits | Distribution of the number of discoveries")
points(x=1:5, y=res$mean.discoveries, pch=19, col="red")
```




```
res$mean.discoveries
```

```
##        BH     BHSto     CTSim CTSimSto     CTWMW
##    0.3562    0.2341    0.2396    0.1420    1.5452
```

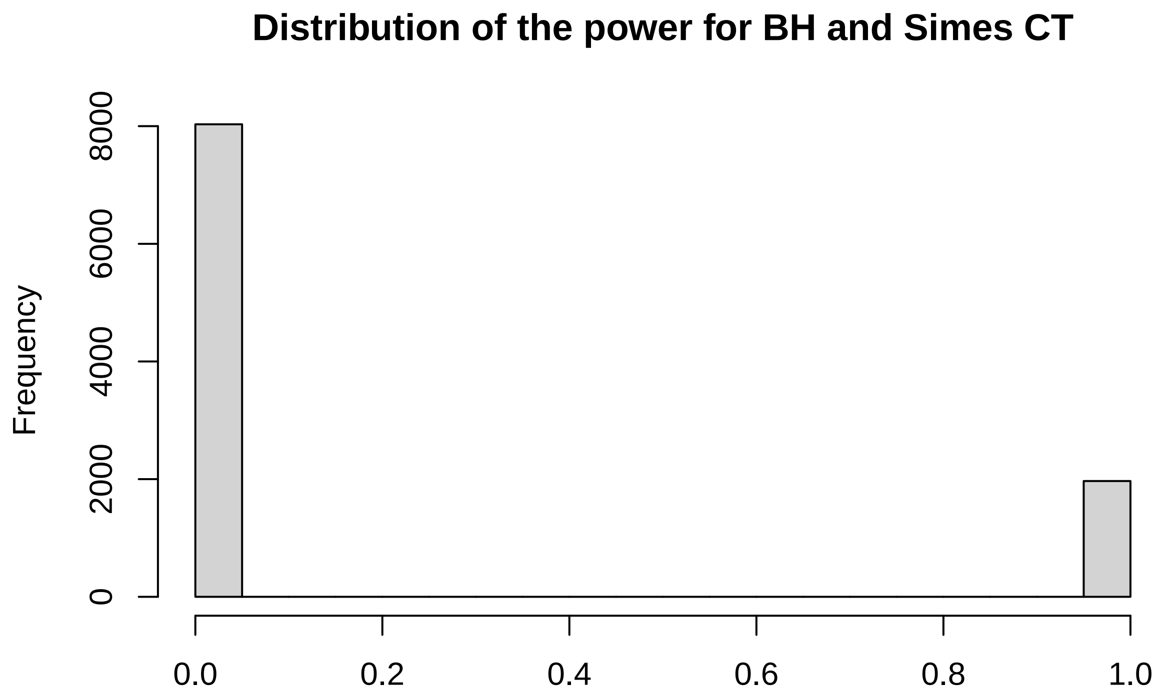
```
boxplot(res$powerGlobalNull, main="Digits | Distribution of the power")
points(x=1:5, y=res$mean.powerGlobalNull, pch=19, col="red")
```



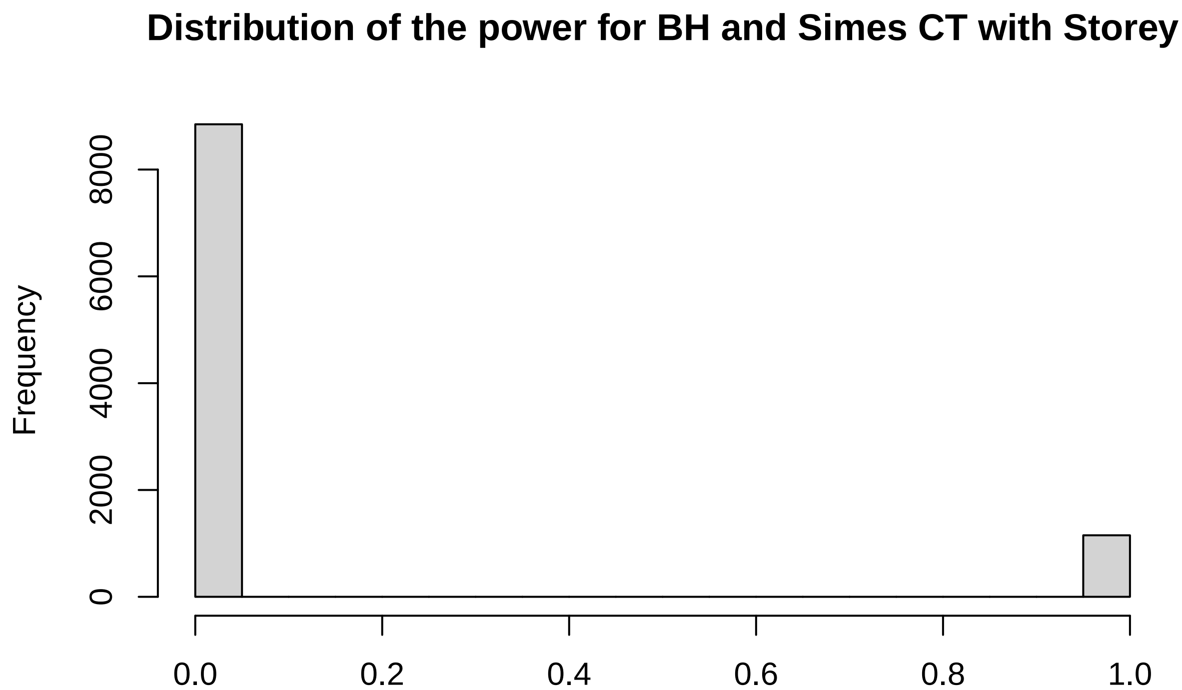

```
res$mean.powerGlobalNull
```

```
##        BH    BHSto    CTSim CTSimSto    CTWMW
##    0.1968   0.1152   0.1968   0.1152   0.1970
```

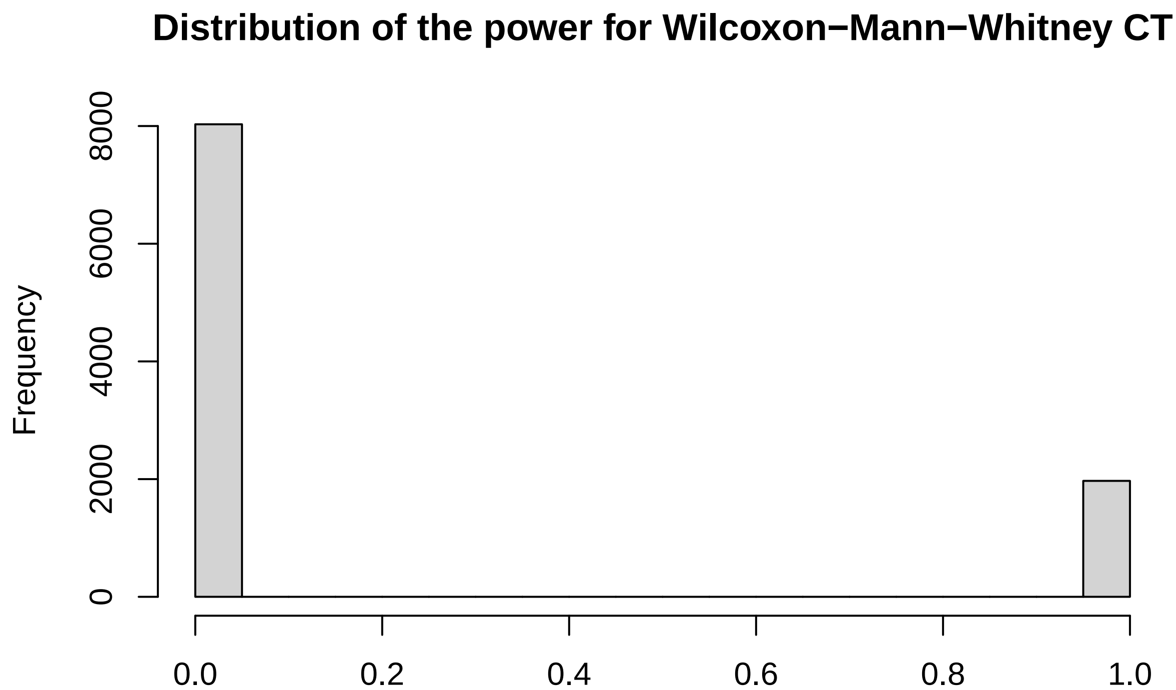
```
hist(as.integer(res$powerGlobalNull[[1]]), xlab="",
     main="Distribution of the power for BH and Simes CT")
```




```
hist(as.integer(res$powerGlobalNull[[2]]), xlab="",
     main="Distribution of the power for BH and Simes CT with Storey")
```




```
hist(as.integer(res$powerGlobalNull[[5]]), xlab="",
     main="Distribution of the power for Wilcoxon-Mann-Whitney CT")
```




```
resDigits_exact0 = res
save(resDigits_exact0,
     file="C:/Users/c.magnani9/Documents/nout/trials/RealData/PowerStudy/Boxplot&ExactCrit/resDigits_exa
```

10% outliers

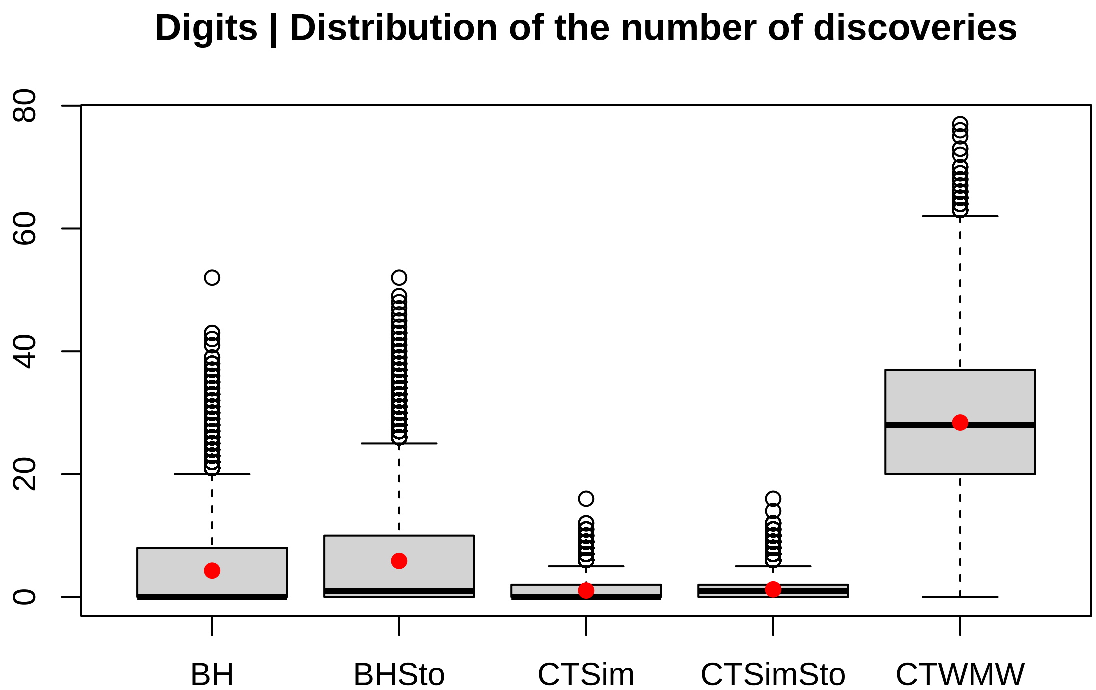
We now set the proportion of inliers equal to 0.9. Referring to Digits dataset we have that the number of inliers is $m_0 = 378$ and the number of outliers is $m_1 = 42$.

```
B=10^4
m1=round(0.1*m)

tic()
res = sim_realdata(B=B, in_index=in_ind, out_index=out_ind, dataset=dataset,
                   alpha=alpha,l=l, n=n, m=m, m1=m1)
toc()
```

```
## 626.21 sec elapsed
```

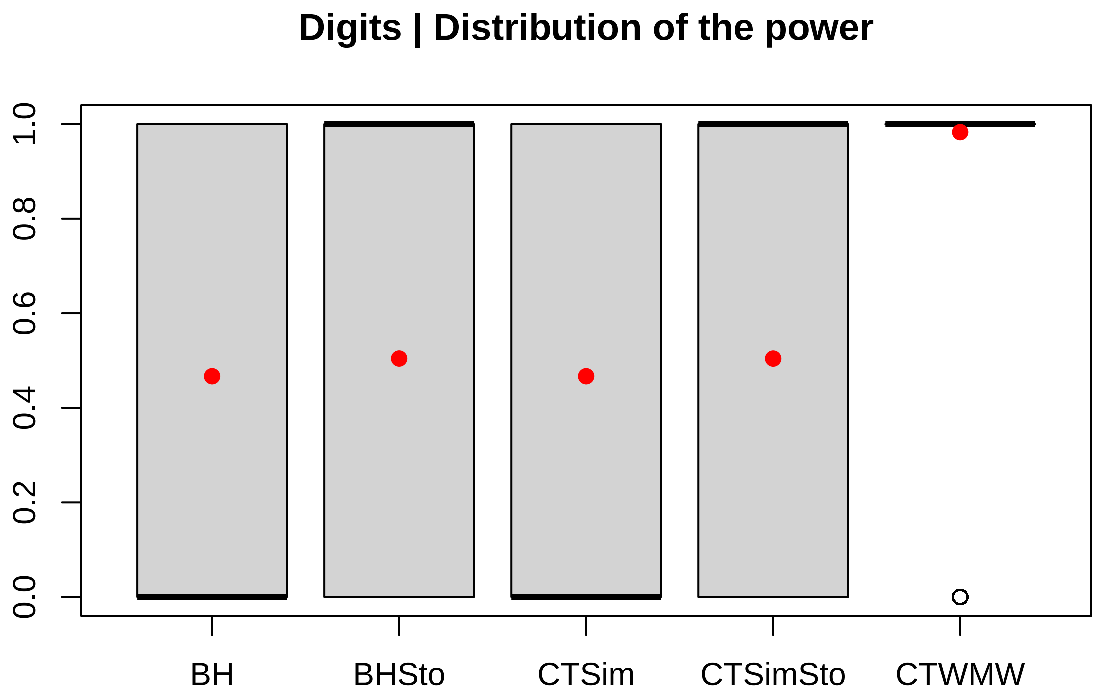
```
boxplot(res$discoveries, main="Digits | Distribution of the number of discoveries")
points(x=1:5, y=res$mean.discoveries, pch=19, col="red")
```




```r
res$mean.discoveries
```

```
##        BH     BHSto     CTSim  CTSimSto     CTWMW
##    4.2935    5.8736    1.0500    1.2383   28.4074
```

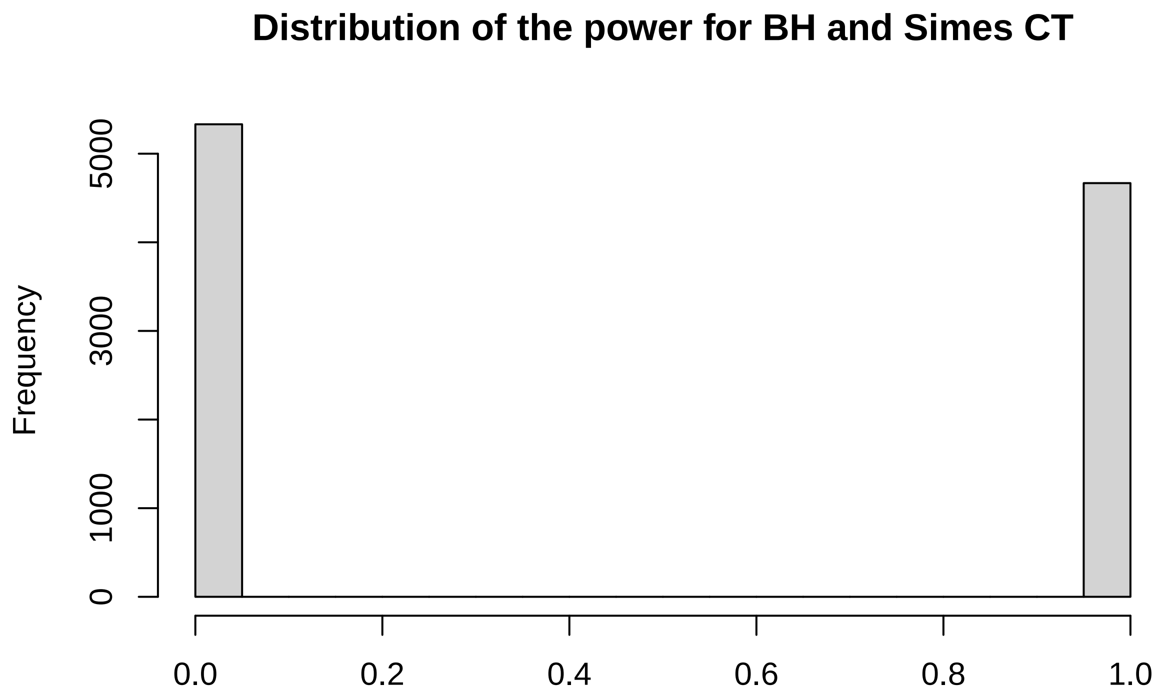
```r
boxplot(res$powerGlobalNull, main="Digits | Distribution of the power")
points(x=1:5, y=res$mean.powerGlobalNull, pch=19, col="red")
```



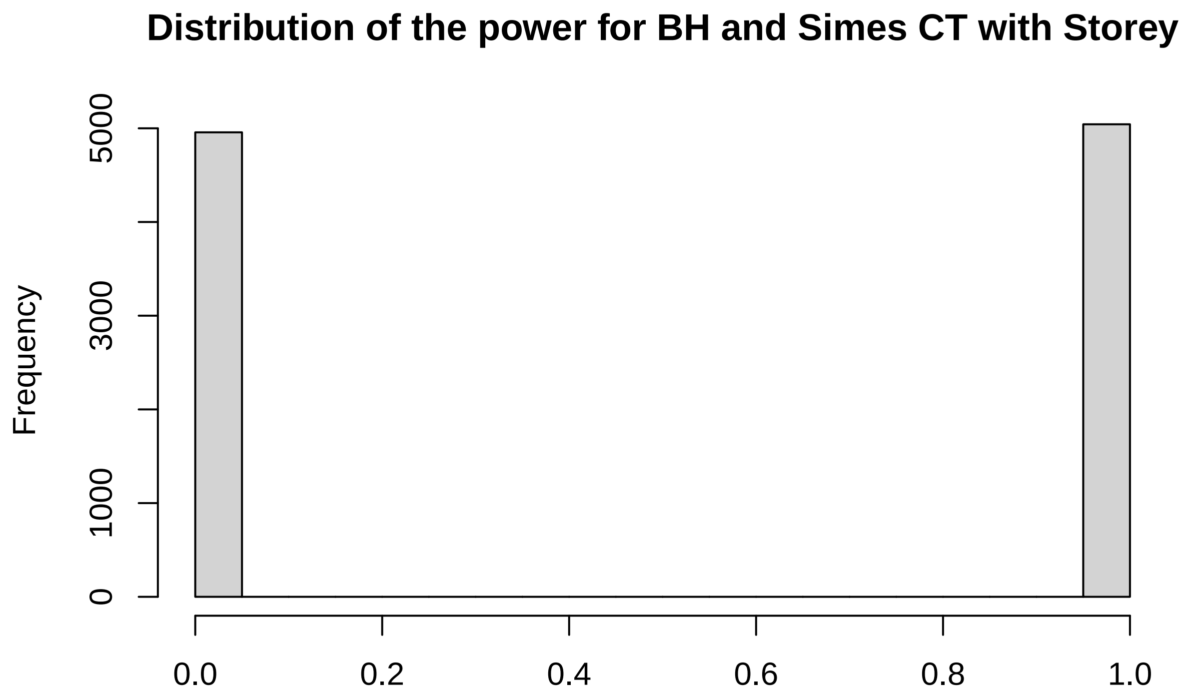

```
res$mean.powerGlobalNull
```

```
##        BH    BHSto    CTSim CTSimSto    CTWMW
##    0.4668   0.5043   0.4668   0.5042   0.9830
```

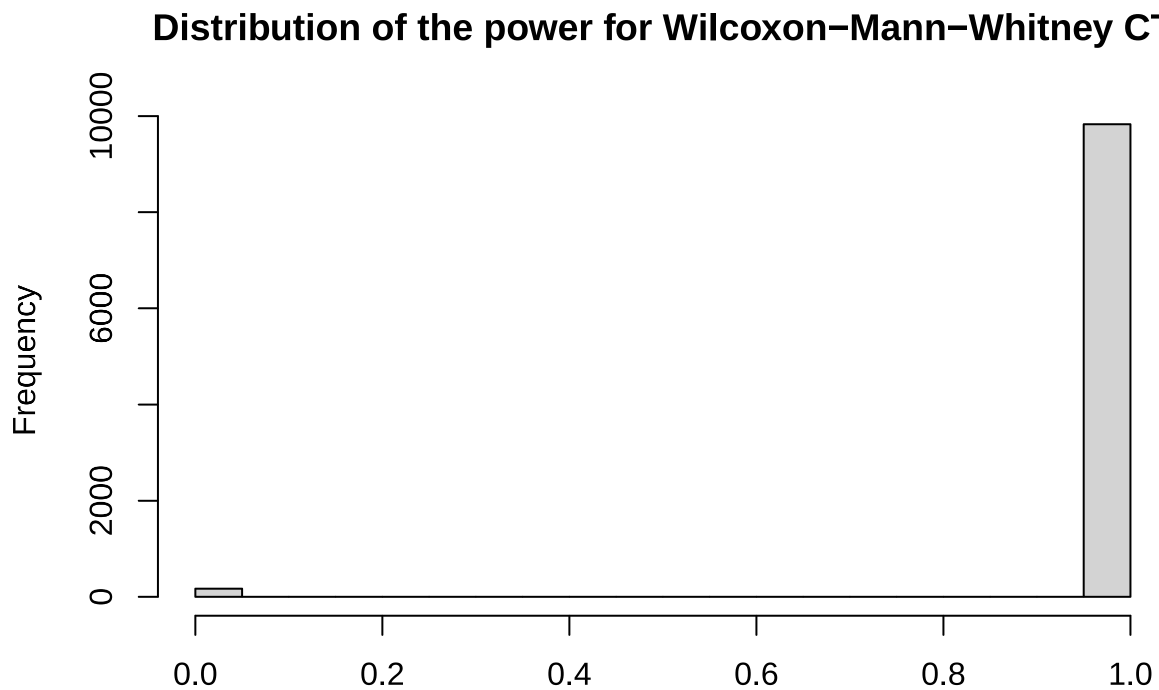
```
hist(as.integer(res$powerGlobalNull[[1]]), xlab="",
     main="Distribution of the power for BH and Simes CT")
```




```
hist(as.integer(res$powerGlobalNull[[2]]), xlab="",
     main="Distribution of the power for BH and Simes CT with Storey")
```




```
hist(as.integer(res$powerGlobalNull[[5]]), xlab="",
     main="Distribution of the power for Wilcoxon-Mann-Whitney CT")
```




```
resDigits_exact10 = res
save(resDigits_exact10,
     file="C:/Users/c.magnani9/Documents/nout/trials/RealData/PowerStudy/Boxplot&ExactCrit/resDigits_exa
```

Credit Card Fraud Detection dataset

The aim is to compare on Digits dataset (available at https://www.kaggle.com/mlg-ulb/creditcardfraud) the performance of three closed testing procedures, which respectively use Simes local test with and without Storey estimator for the proportion of true null hypotheses and Wilcoxon-Mann-Whitney local test.

Credit card dataset consists of 284807 observations, among which $n_{inliers} = 284315$ items are inliers and the remaining $n_{outliers} = 492$ are outliers. We will denote by n, l, m respectively the train set, the calibration set and the test set size. And reproducing the same setting as in [1], we have that $n + l = n_{inliers}/2$, $l = min\{2000, n/2\}$ and $m = min\{2000, n/3\}$. In the case of Digits dataset we obtain

$$n + l = 284315/2, \quad l = 2000, \quad m = 2000$$

from which $n = 140157.5, \quad l = 2000, \quad m = 2000$. Arbitrarily, we choose to set

$$n = 132158, \quad l = 9999, \quad m = 2000$$

in order to have exact control of type I errors at the significance level $\alpha = m/(l + 1) = 0.2$.

Load the data and set the parameters ad described above.

```
set.seed(321)

# Initializing parameters
n = 132158
```

```r
l = 9999
m = 2000
alpha = m/(l+1)

dataset = read.csv("G:\\Il mio Drive\\PHD\\Progetto di ricerca\\Conformal Inference Project\\Simulazioni
out_ind = which(dataset$Class==1)
in_ind = which(dataset$Class==0)
```

We fixed the train set on which we train the isolation forest algorithm and we generate $B = 1000$ calibration and test sets. For each $b = 1, \dots, B$ we compute the number of discoveries obtained by Benjamini-Hochberg procedure with and without Storey's estimator for the proportion of true null hypotheses, by closed testing using Simes local test with and without Storey estimator and by closed testing using Wilcoxon-Mann-Whitney local test.

All inliers

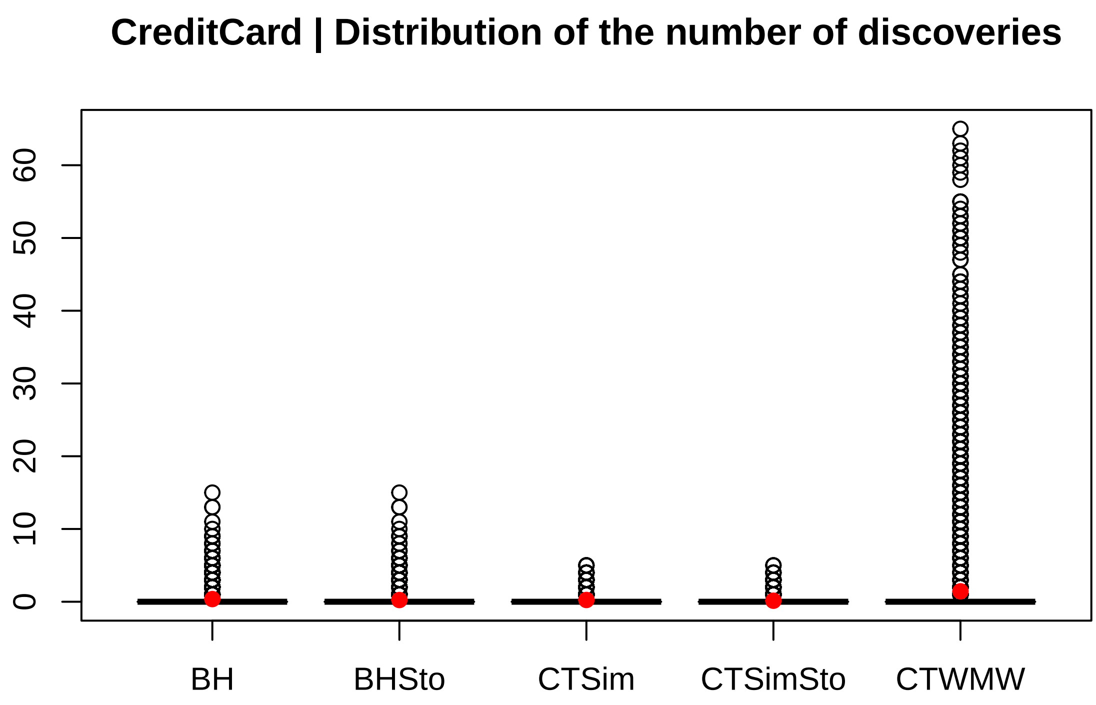
We now set the proportion of inliers equal to 1, so that the number of outliers $m_1 = 0$.

```r
B=10^4
m1=0

tic()
res = sim_realdata(B=B, in_index=in_ind, out_index=out_ind, dataset=dataset,
                   alpha=alpha,l=l, n=n, m=m, m1=m1)
toc()
```

```
## 8181.78 sec elapsed
```

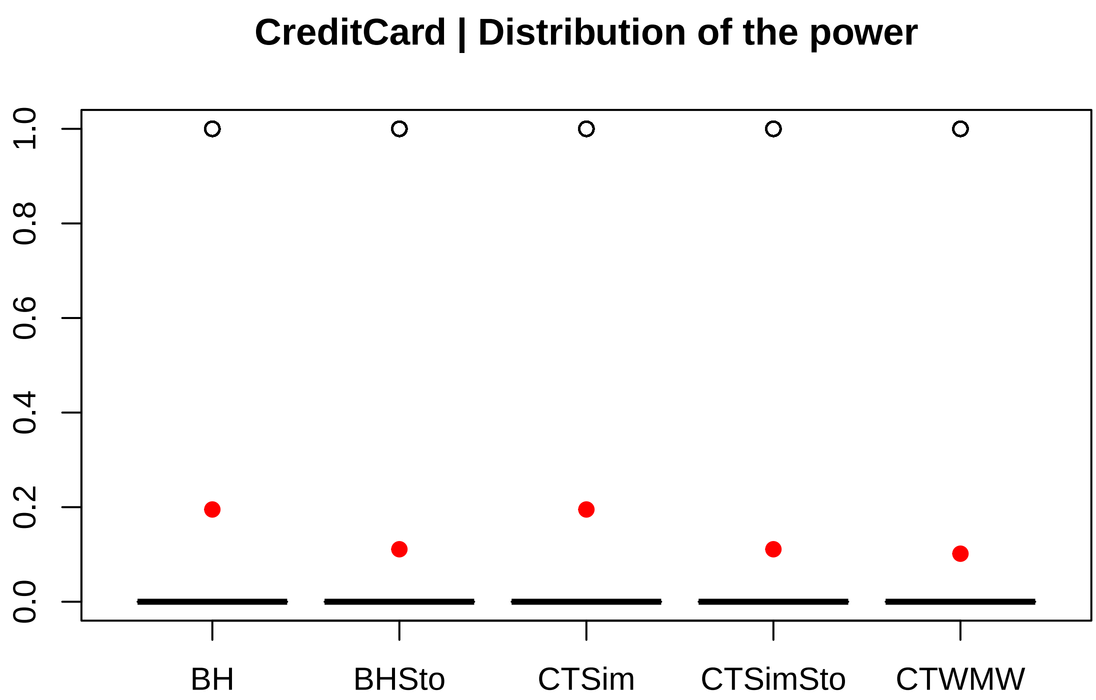
```r
boxplot(res$discoveries, main="CreditCard | Distribution of the number of discoveries")
points(x=1:5, y=res$mean.discoveries, pch=19, col="red")
```




```
res$mean.discoveries
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
##   0.3654   0.2329   0.2414   0.1372   1.4266
```

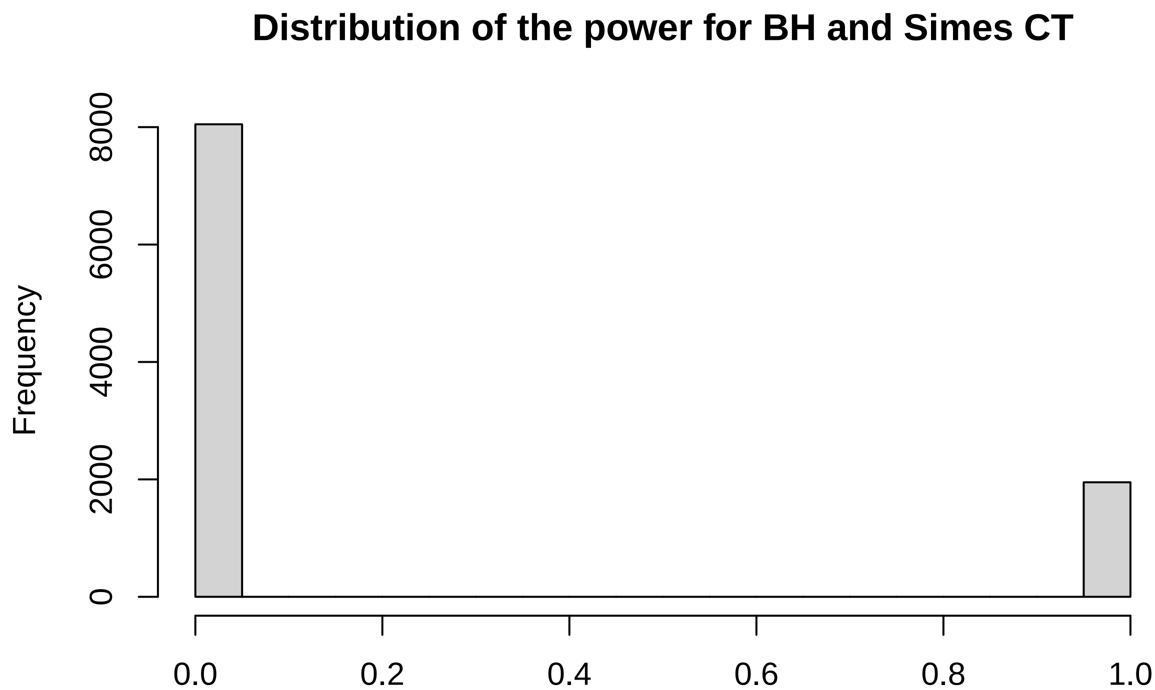
```
boxplot(res$powerGlobalNull, main="CreditCard | Distribution of the power")
points(x=1:5, y=res$mean.powerGlobalNull, pch=19, col="red")
```



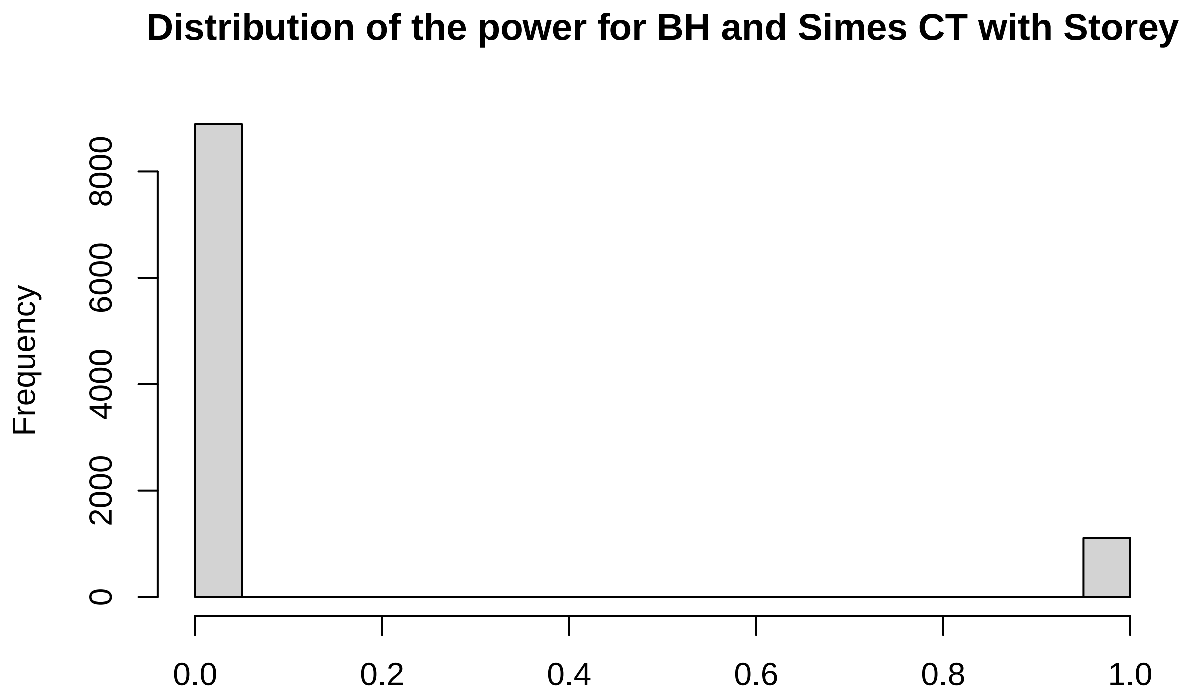

```r
res$mean.powerGlobalNull
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
##   0.1951   0.1110   0.1951   0.1110   0.1016
```

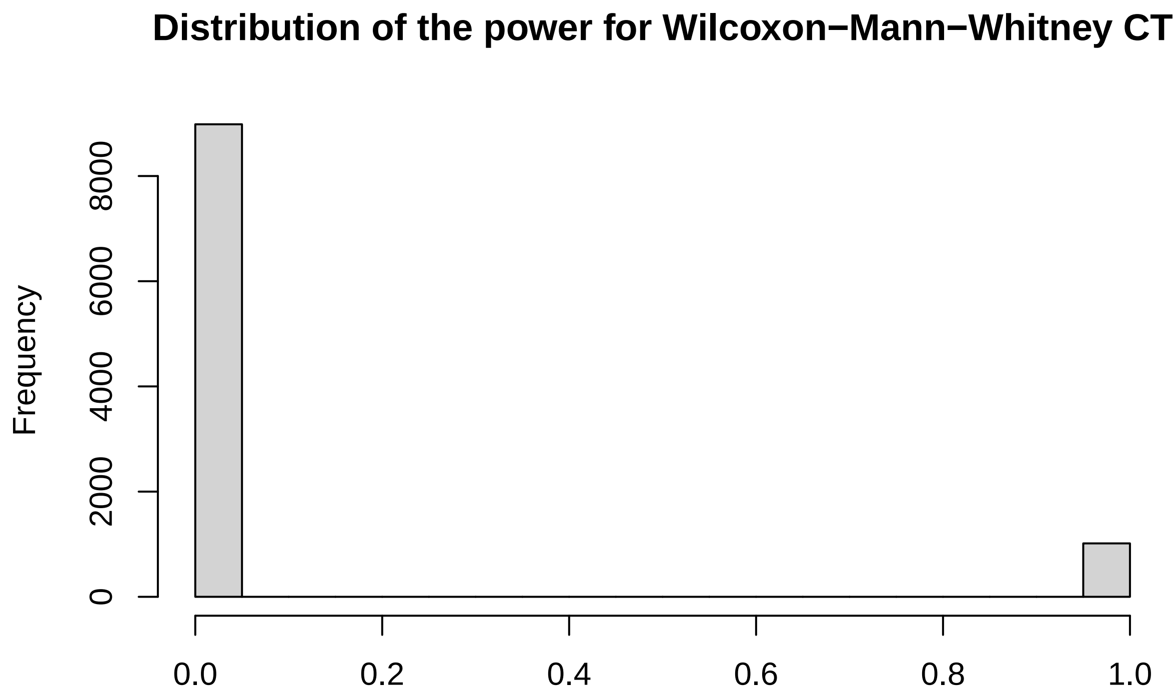
```r
hist(as.integer(res$powerGlobalNull[[1]]), xlab="",
     main="Distribution of the power for BH and Simes CT")
```




```
hist(as.integer(res$powerGlobalNull[[2]]), xlab="",
     main="Distribution of the power for BH and Simes CT with Storey")
```




```r
hist(as.integer(res$powerGlobalNull[[5]]), xlab="",
     main="Distribution of the power for Wilcoxon-Mann-Whitney CT")
```




```
resCredit_exact0 = res
save(resCredit_exact0,
     file="C:/Users/c.magnani9/Documents/nout/trials/RealData/PowerStudy/Boxplot&ExactCrit/resCredit_exa
```

10% outliers

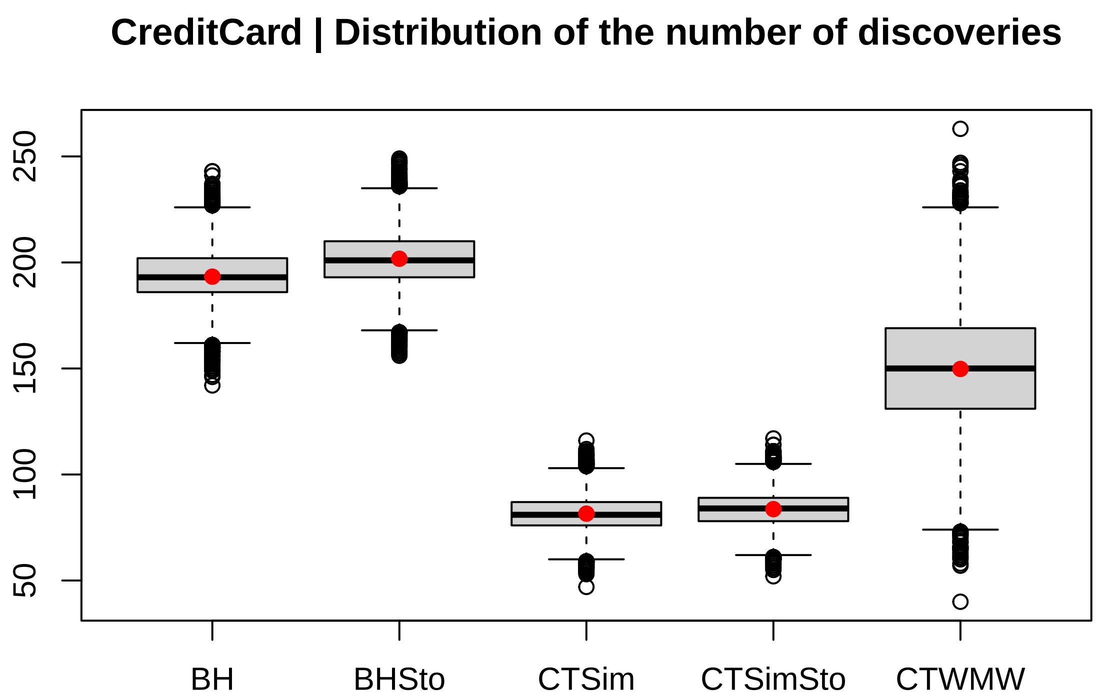
We now set the proportion of inliers equal to 0.9. Referring to Digits dataset we have that the number of inliers is $m_0 = 1800$ and the number of outliers is $m_1 = 200$.

```
B=10^4
m1=round(0.1*m)

tic()
res = sim_realdata(B=B, in_index=in_ind, out_index=out_ind, dataset=dataset,
                   alpha=alpha,l=l, n=n, m=m, m1=m1)
toc()
```

```
## 8291.72 sec elapsed
```

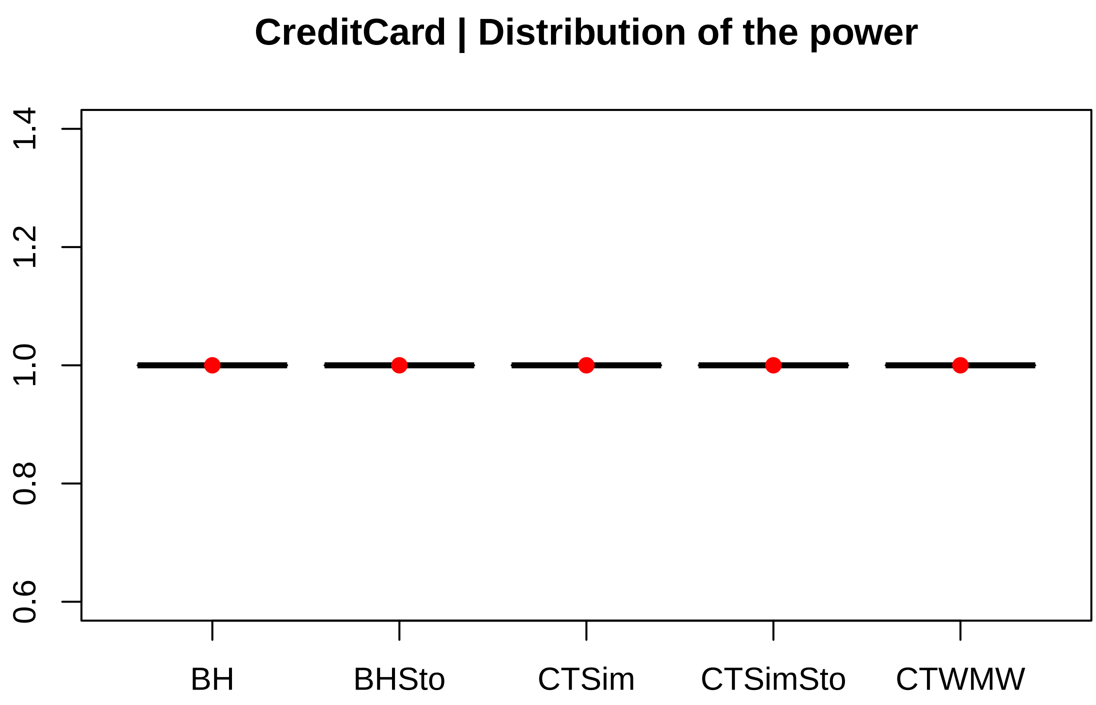
```
boxplot(res$discoveries, main="CreditCard | Distribution of the number of discoveries")
points(x=1:5, y=res$mean.discoveries, pch=19, col="red")
```




```
res$mean.discoveries
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
## 193.3018 201.7384  81.5388  83.6672 149.7832
```

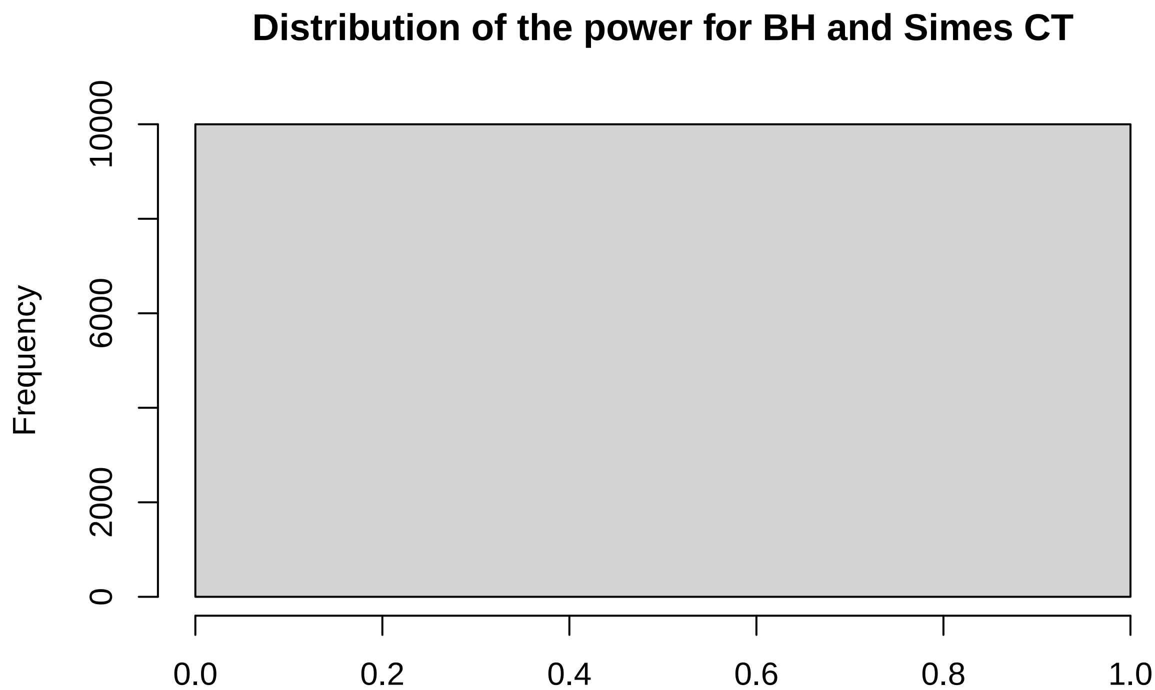
```
boxplot(res$powerGlobalNull, main="CreditCard | Distribution of the power")
points(x=1:5, y=res$mean.powerGlobalNull, pch=19, col="red")
```



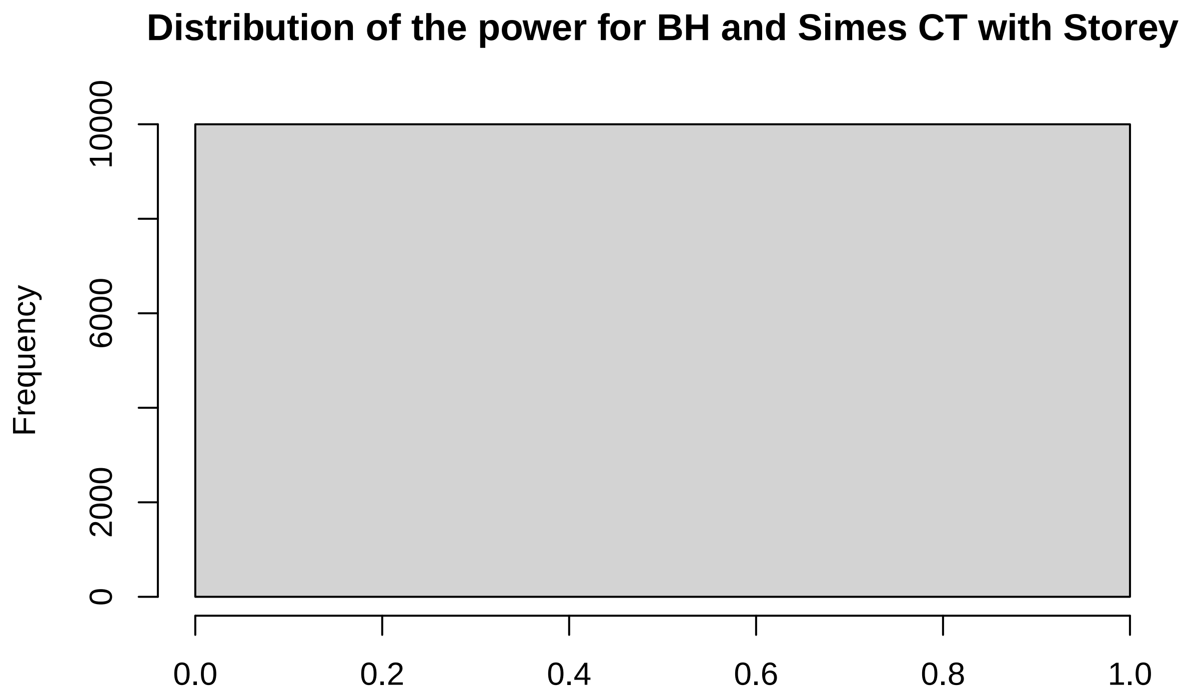

```
res$mean.powerGlobalNull
```

```
##        BH     BHSto     CTSim  CTSimSto     CTWMW
##         1         1         1         1         1
```

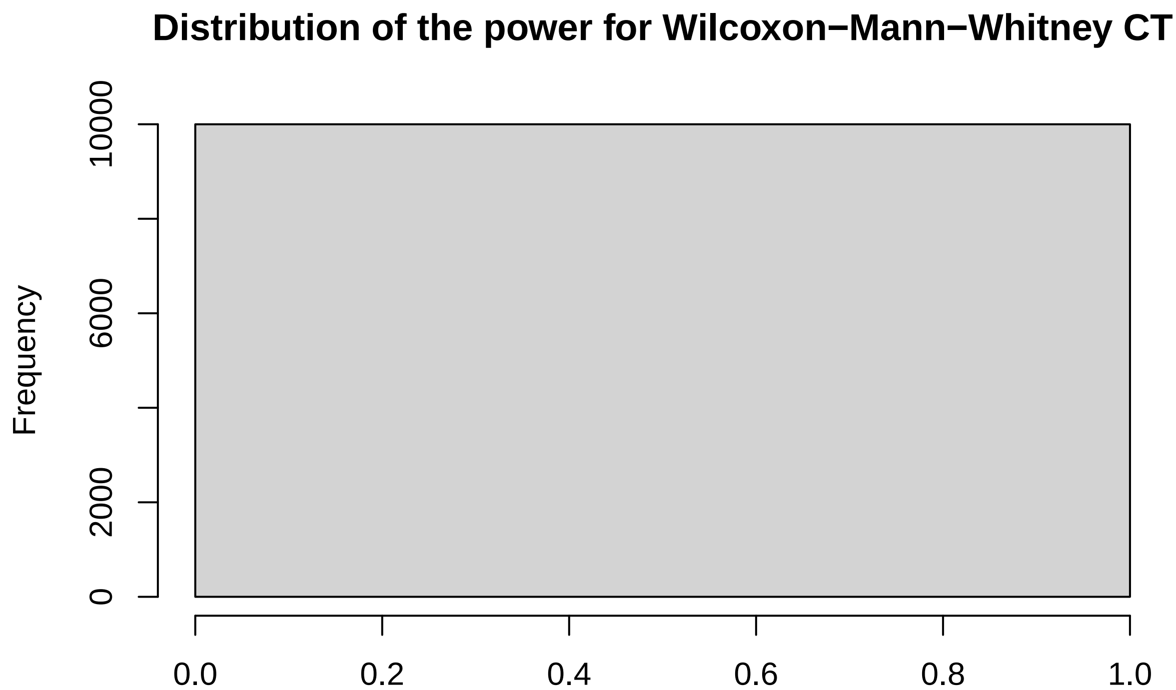
```
hist(as.integer(res$powerGlobalNull[[1]]), xlab="",
     main="Distribution of the power for BH and Simes CT")
```




```
hist(as.integer(res$powerGlobalNull[[2]]), xlab="",
     main="Distribution of the power for BH and Simes CT with Storey")
```




```
hist(as.integer(res$powerGlobalNull[[5]]), xlab="",
     main="Distribution of the power for Wilcoxon-Mann-Whitney CT")
```




```
resCredit_exact10 = res
save(resCredit_exact10,
     file="C:/Users/c.magnani9/Documents/nout/trials/RealData/PowerStudy/Boxplot&ExactCrit/resCredit_exa
```

Statlog (Shuttle) dataset

The aim is to compare on Digits dataset (available at http://odds.cs.stonybrook.edu/shuttle-dataset) the performance of three closed testing procedures, which respectively use Simes local test with and without Storey estimator for the proportion of true null hypotheses and Wilcoxon-Mann-Whitney local test.

Credit card dataset consists of 49097 observations, among which $n_{inliers} = 45586$ items are inliers and the remaining $n_{outliers} = 3511$ are outliers. We will denote by n, l, m respectively the train set, the calibration set and the test set size. And reproducing the same setting as in [1], we have that $n + l = n_{inliers}/2$, $l = min\{2000, n/2\}$ and $m = min\{2000, n/3\}$. In the case of Digits dataset we obtain

$$n + l = 45586/2, \quad l = 2000, \quad m = 2000$$

from which $n = 20793, \quad l = 2000, \quad m = 2000$. Arbitrarily, we choose to set

$$n = 12794, \quad l = 9999, \quad m = 2000$$

in order to have exact control of type I errors at the significance level $\alpha = m/(l + 1) = 0.2$.

Load the data and set the parameters ad described above.

```
set.seed(321)

# Initializing parameters
n = 12794
```

```r
l = 9999
m = 2000
alpha = m/(l+1)

data = readMat("G:\\Il mio Drive\\PHD\\Progetto di ricerca\\Conformal Inference Project\\Simulazioni\\7
dataset = cbind(data$X, data$y); colnames(dataset)[ncol(dataset)] = "y"
out_ind = which(dataset[,ncol(dataset)]==1)
in_ind = which(dataset[,ncol(dataset)]==0)
```

We fixed the train set on which we train the isolation forest algorithm and we generate $B = 1000$ calibration and test sets. For each $b = 1, \ldots, B$ we compute the number of discoveries obtained by Benjamini-Hochberg procedure with and without Storey's estimator for the proportion of true null hypotheses, by closed testing using Simes local test with and without Storey estimator and by closed testing using Wilcoxon-Mann-Whitney local test.

All inliers

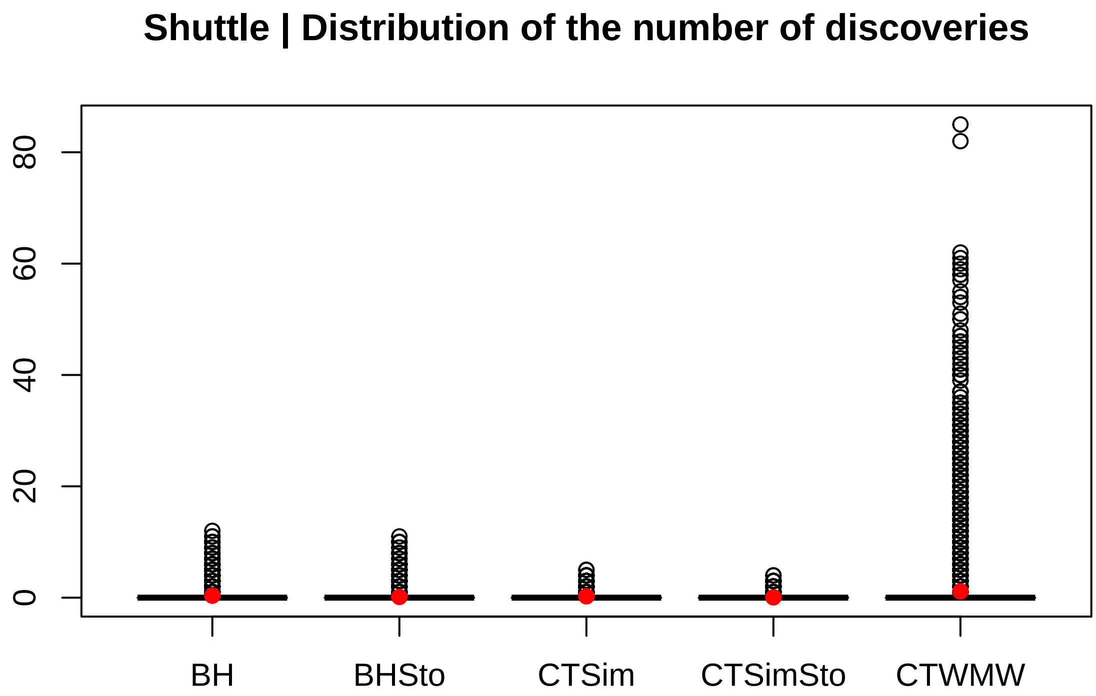
We now set the proportion of inliers equal to 1, so that the number of outliers $m_1 = 0$.

```r
B=10^4
m1=0

tic()
res = sim_realdata(B=B, in_index=in_ind, out_index=out_ind, dataset=dataset,
                   alpha=alpha,l=l, n=n, m=m, m1=m1)
toc()
```

```
## 6191.9 sec elapsed
```

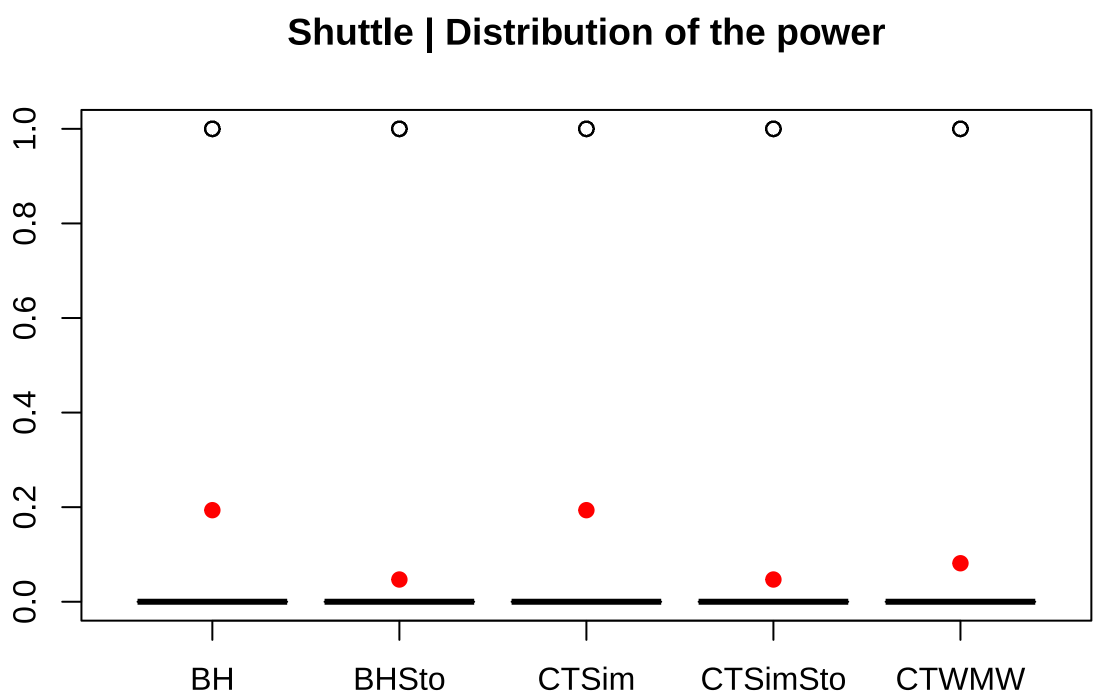
```r
boxplot(res$discoveries, main="Shuttle | Distribution of the number of discoveries")
points(x=1:5, y=res$mean.discoveries, pch=19, col="red")
```




```
res$mean.discoveries
```

```
##        BH     BHSto     CTSim  CTSimSto     CTWMW
##    0.3656    0.1325    0.2393    0.0570    1.1259
```

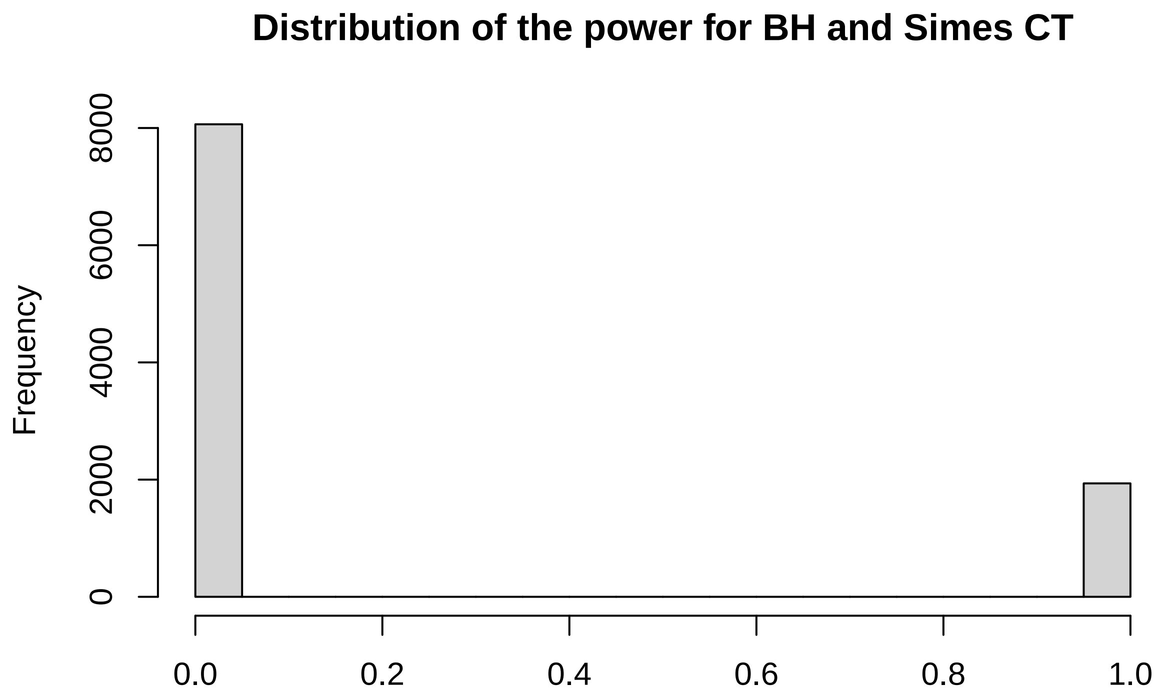
```
boxplot(res$powerGlobalNull, main="Shuttle | Distribution of the power")
points(x=1:5, y=res$mean.powerGlobalNull, pch=19, col="red")
```



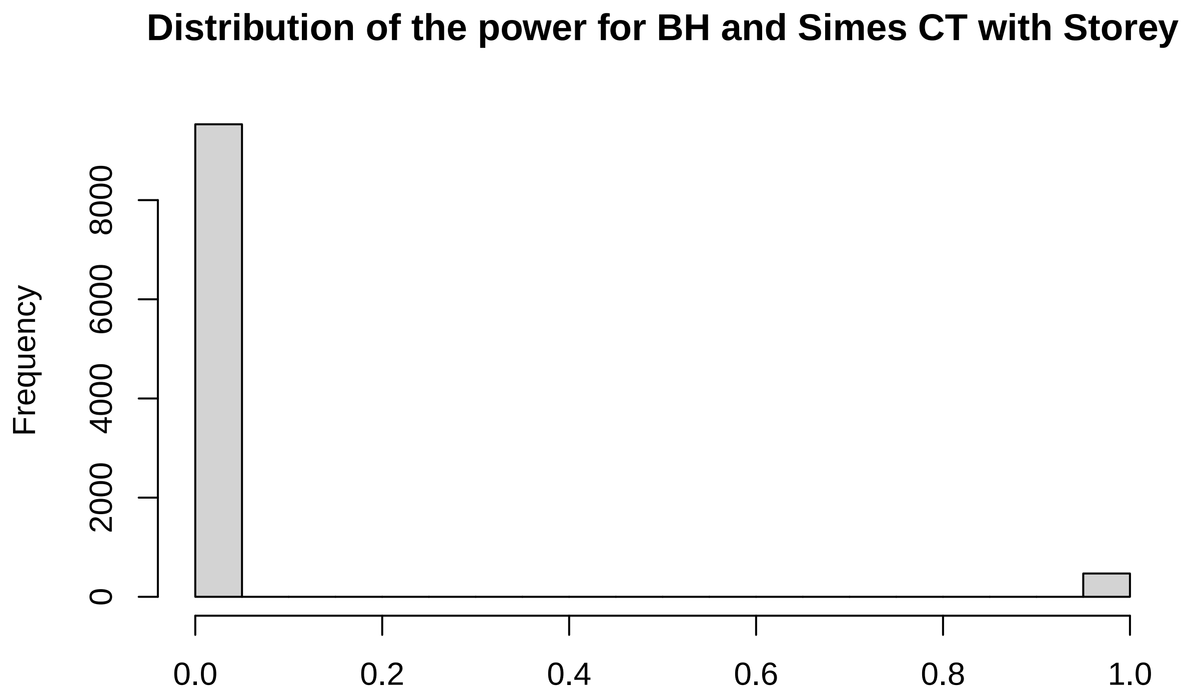

```
res$mean.powerGlobalNull
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
##   0.1936   0.0470   0.1936   0.0470   0.0814
```

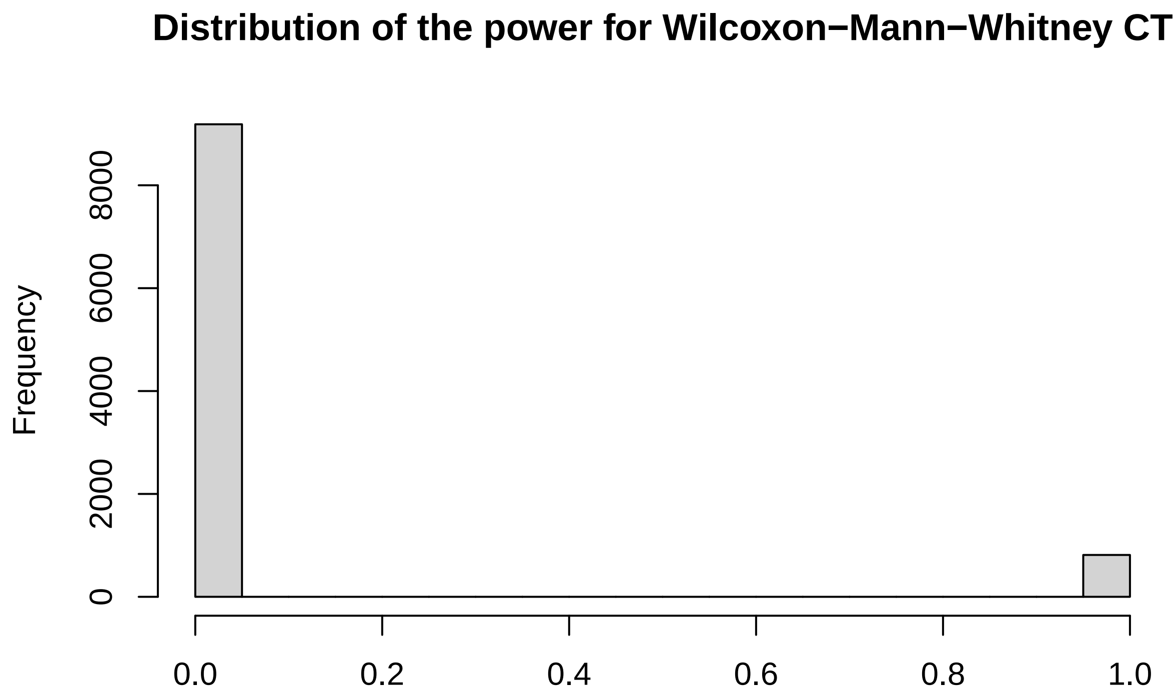
```
hist(as.integer(res$powerGlobalNull[[1]]), xlab="",
     main="Distribution of the power for BH and Simes CT")
```




```
hist(as.integer(res$powerGlobalNull[[2]]), xlab="",
     main="Distribution of the power for BH and Simes CT with Storey")
```




```
hist(as.integer(res$powerGlobalNull[[5]]), xlab="",
     main="Distribution of the power for Wilcoxon-Mann-Whitney CT")
```




```
resShuttle_exact0 = res
save(resShuttle_exact0,
     file="C:/Users/c.magnani9/Documents/nout/trials/RealData/PowerStudy/Boxplot&ExactCrit/resShuttle_e
```

10% outliers

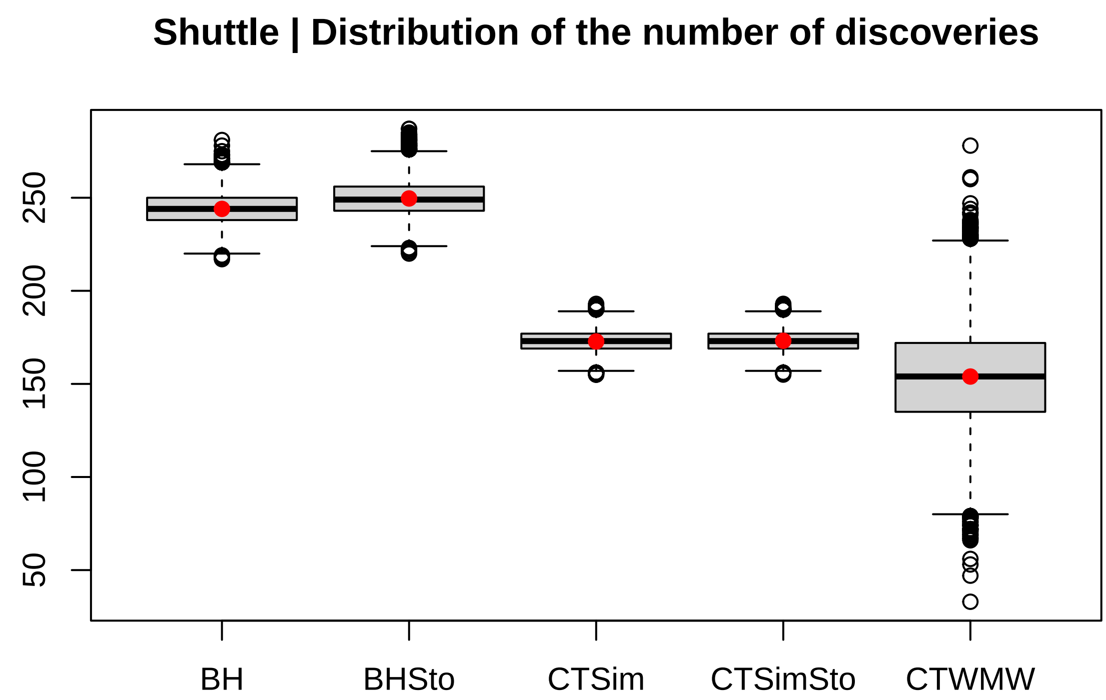
We now set the proportion of inliers equal to 0.9. Referring to Digits dataset we have that the number of inliers is $m_0 = 1800$ and the number of outliers is $m_1 = 200$.

```
B=10^4
m1=round(0.1*m)

tic()
res = sim_realdata(B=B, in_index=in_ind, out_index=out_ind, dataset=dataset,
                   alpha=alpha,l=l, n=n, m=m, m1=m1)
toc()
```

```
## 6106.36 sec elapsed
```

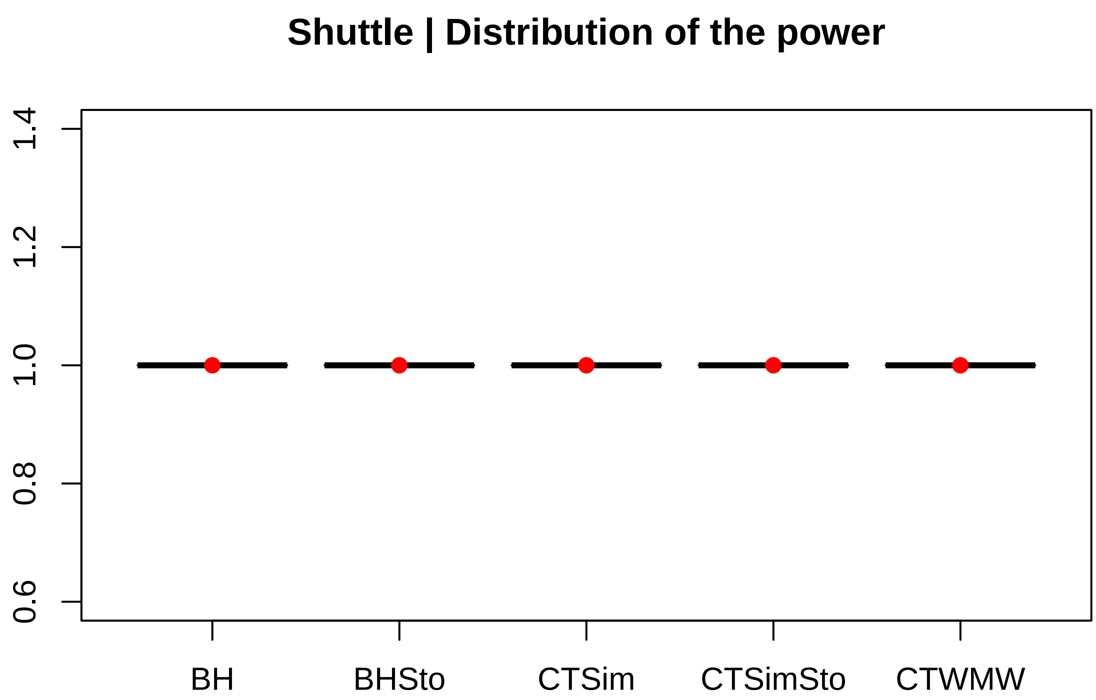
```
boxplot(res$discoveries, main="Shuttle | Distribution of the number of discoveries")
points(x=1:5, y=res$mean.discoveries, pch=19, col="red")
```




```
res$mean.discoveries
```

```
##        BH    BHSto    CTSim CTSimSto    CTWMW
## 244.0126 249.6119 172.7924 173.1773 153.9530
```

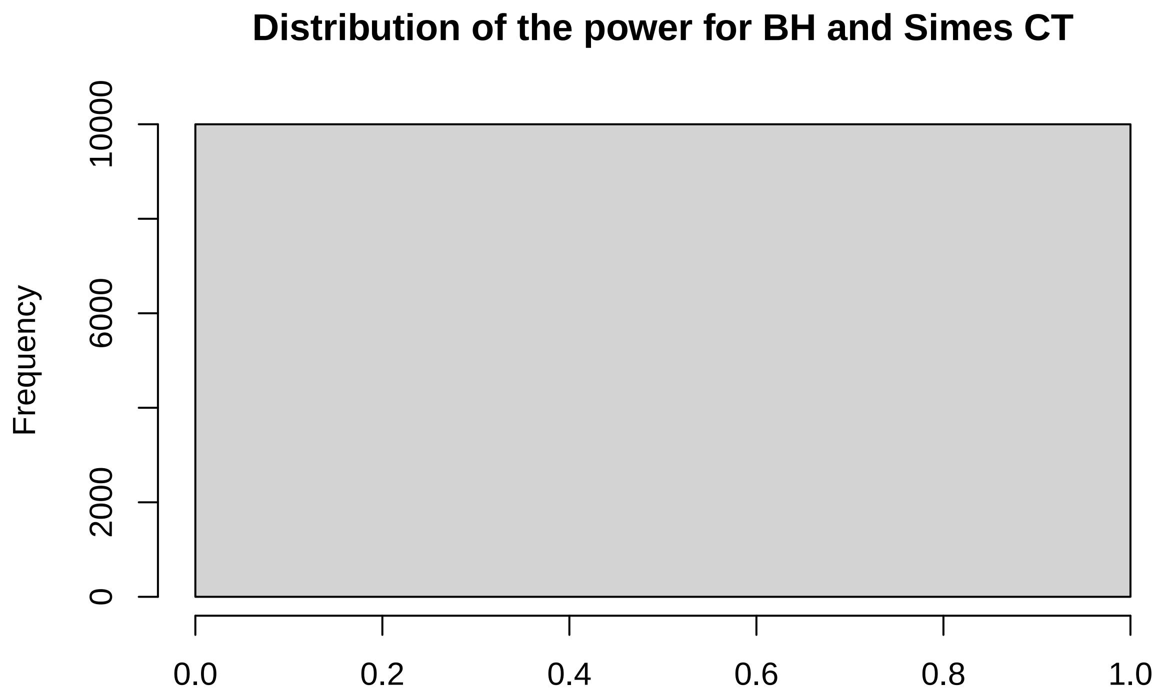
```
boxplot(res$powerGlobalNull, main="Shuttle | Distribution of the power")
points(x=1:5, y=res$mean.powerGlobalNull, pch=19, col="red")
```



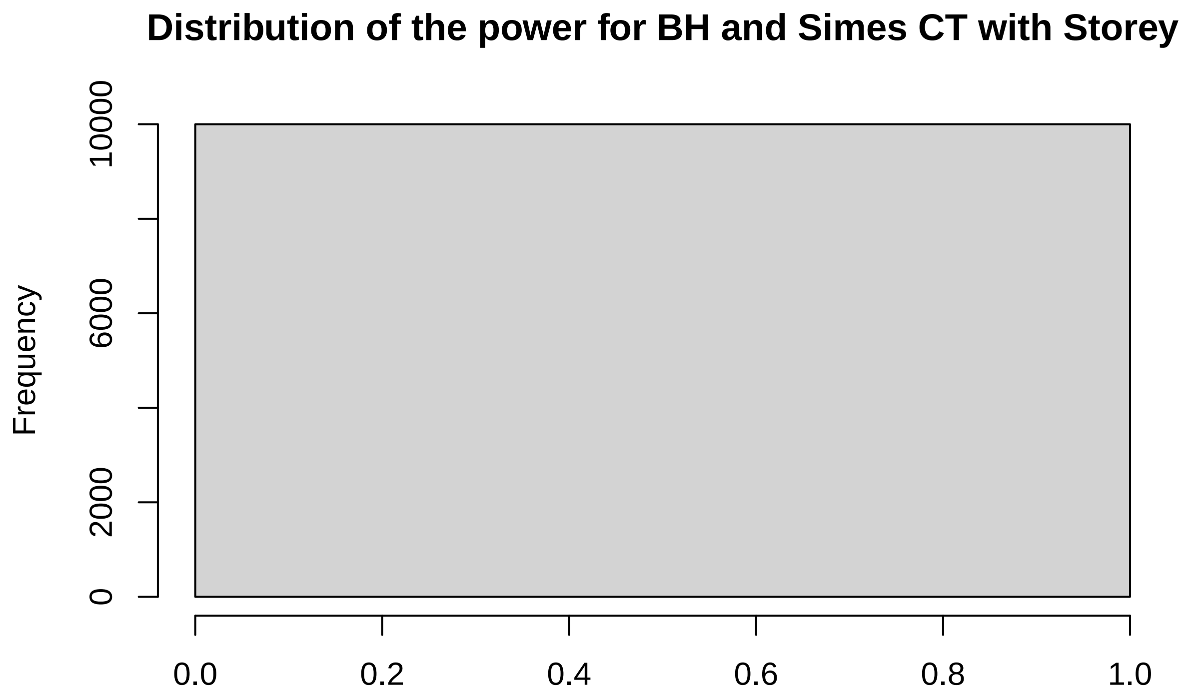

```
res$mean.powerGlobalNull
```

```
##        BH     BHSto     CTSim  CTSimSto     CTWMW
##         1         1         1         1         1
```

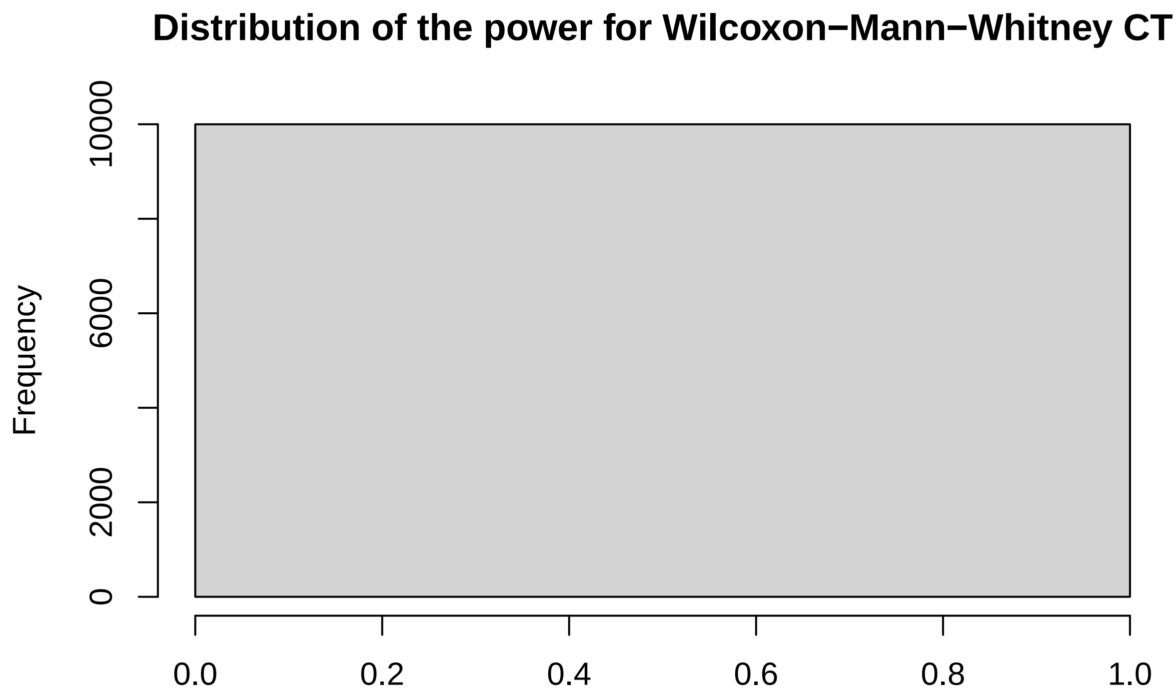
```
hist(as.integer(res$powerGlobalNull[[1]]), xlab="",
     main="Distribution of the power for BH and Simes CT")
```




```r
hist(as.integer(res$powerGlobalNull[[2]]), xlab="",
     main="Distribution of the power for BH and Simes CT with Storey")
```




```
hist(as.integer(res$powerGlobalNull[[5]]), xlab="",
     main="Distribution of the power for Wilcoxon-Mann-Whitney CT")
```




```
resShuttle_exact10 = res
save(resShuttle_exact10,
     file="C:/Users/c.magnani9/Documents/nout/trials/RealData/PowerStudy/Boxplot&ExactCrit/resShuttle_e
```

References

[1] Bates, S., E. Candes, L. Lei, Y. Romano, and M. Sesia (2023). Testing for outliers with conformal p-values. *Annals of Statistics*, {**51}, 149–178.**