Final Project

Stat 428

12/13/2020

I. Simulation Problem (50 points)

In the lecture, we discussed Nearest Neighbor Tests and Energy Distance Test for two sample testing problem. We consider another two tests: two-sample Hotelling's T-square test statistic and graph-based two sample test. Suppose the data we observe $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_m$, where $X_i, Y_j \in \mathbb{R}^d$ are multivariate random vectors. Here, $X_1, \ldots, X_n$ are drawn from distribution F and $Y_1, \ldots, Y_m$ are drawn from distribution G. The hypothesis of interest in two sample testing problem is

$$H_0 : F = G \qquad \text{and} \qquad H_1 : F \neq G.$$

The Hotelling's T-square test statistic is defined as

$$T^2 = \frac{nm}{n+m}(\bar{X} - \bar{Y})\hat{\Sigma}^{-1}(\bar{X} - \bar{Y})$$

where

$$\hat{\Sigma} = \frac{1}{n+m-2}\left((n-1)\hat{\Sigma}_X + (m-1)\hat{\Sigma}_Y\right).$$

Here, sample mean and sample covariance are defined as

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i, \qquad \bar{Y} = \frac{1}{m}\sum_{i=1}^{m} Y_i$$

and

$$\hat{\Sigma}_X = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})^T, \qquad \hat{\Sigma}_Y = \frac{1}{m-1}\sum_{i=1}^{m}(Y_i - \bar{Y})(Y_i - \bar{Y})^T.$$

Graph-based two sample test is defined in the following way. We pool all data together

$$Z_1, \ldots, Z_{n+m} = X_1, \ldots, X_n | Y_1, \ldots, Y_m$$

Based these $n + m$ observations, we construct a graph $G = (V, E)$ such that the set of vertex is $V = \{1, \ldots, n+m\}$ and there is an edge between i and j if $\|Z_i - Z_j\| \leq Q$, where Q is a positive number. Let E be the collection of edges. The graph-based two sample test statistic is defined as

$$R = \frac{1}{|E|}\sum_{e \in E} I_e,$$

where $|E|$ means the number of edges in the edge set E. Here, $I_e = 1$ if the two vertex connected by e have the same label and $I_e = 0$ otherwise.

Question 1 Report

A pharmaceutical company would like to test whether the effect of two treatments are similar or not. The manager want to choose one two sample testing method from nearest neighbor tests, energy distance test, Hotelling's T-square test and graph-based two sample test and ask your advice for the choice of two sample test. First, could you help the manager to implement these four methods from the scratch: nearest neighbor tests, energy distance test, Hotelling's T-square test and graph-based two sample test? Second, could you prepare a report to provide some suggestions for the manager? In this report, you need to address at least four of the following points:

- Several different parts can be customized in these tests, e.g., the threshold Q in graph-based test, the number of neighbor in nearest neighbor test and the specific form of distance in energy distance test and graph-based test. Could you provide some suggestion on the choice of these customized part? You need to show some numerical experiment as your evidence.
- Are these tests sensitive to the dimension of data d?
- Are these tests sensitive to specific distribution of F or G?
- Which test has larger power under what condition?
- Clearly, the power of the test relies on the sample size n, m and how different F and G are under alternative hypothesis. Could you prepare a plot to show effect of sample size on power? Could you prepare another plot to show effect of the difference bewteen F and G on power?
- Are these methods able to control Type I error?

You need to submit both Rmd and pdf file of your report.

Solution

In this report, four programs are designed to calculate statistics, including nearest neighbor tests, energy distance test, Hotelling's T-square test and graph-based two sample test. The four programs are named as `NNT()`, `edist()`, `HT_sqT()` and `GraphT()`, and a universal program called `permT()` is implemented for these four test to calculate the p-value. Note that the method for calculating the p-value of four test is permutation test.

The structure of the report will be divided into the following points: the first part would report the basic functions related to this question and the second part will address the points mentioned in Question 1 step by step.

- **Nearest Neighbor Tests**

```
NNT <- function(z, ix, size,custom) {
  n1 = size[1]
  n2 = size[2]
  n = n1 + n2
  z = z[ix,]
  o = rep(0, nrow(z))
  z = as.data.frame(cbind(z, o))
  NN = nn2(z, z, custom)
  block1 = NN$nn.idx[1:n1,-1]
  block2 = NN$nn.idx[(n1+1):n,-1]
  i1 = sum(block1 < n1 + 0.5)
  i2 = sum(block2 > n1 + 0.5)
  T_nnt = (i1 + i2) / (custom * n)
  return(T_nnt)
}
```

- **Energy Distance Test**

```
edist <- function(x,ix,size,custom) {
  n1 = size[1]
  n2 = size[2]
  ii = ix[1:n1]
  jj = ix[(n1+1):(n1+n2)]
  w = n1 * n2 / (n1 + n2)
  if(is.null(custom)) {
    custom = 2
  }
  dst = as.matrix(dist(x,method='minkowski',p=custom))
  m11 = sum(dst[ii, ii]) / (n1 * n1)
  m22 = sum(dst[jj, jj]) / (n2 * n2)
  m12 = sum(dst[ii, jj]) / (n1 * n2)
  T_edist = w * ((m12 + m12) - (m11 + m22))
  return (T_edist)
}
```

- **Hotelling's T-square Test**

```
HT_sqT <- function(z,ix,size,custom = NULL) {
  n1 = size[1]
  n2 = size[2]
  n = n1 + n2
  z = z[ix,]
  X = z[1:n1,]
  Y = z[n1+1:n2,]
  X_ = colMeans(X)
  Y_ = colMeans(Y)
  sigma_X = cov(X)
  sigma_Y = cov(Y)
  sigma_hat = ((n1-1)*sigma_X+(n2-1)*sigma_Y)/(n-2)
  T_sq = t(X_-Y_)%*%solve(sigma_hat)%*%(X_-Y_)
  T_sq = T_sq*n1*n2/n
  T_sq = as.double(T_sq)
  return(T_sq)
}
```

- **Graph-based Two Sample Test**

```
GraphT <- function(x,ix,size,custom) {
  n1 = size[1]
  n2 = size[2]
  n = n1 + n2
  x = x[ix,]
  ii = ix[1:n1]
  jj = ix[(n1+1):(n1+n2)]
  if(is.null(custom)) {
    custom = c(2,3)
  }
  if(length(custom)<2) {
    custom = c(custom,3)
  }
```

```
  dst = as.matrix(dist(x,method='minkowski',p=custom[1]))
  Edge = ifelse((dst < custom[2]) == "TRUE", 1, 0)
  num_edge = n+(sum(Edge)-n)/2
  sum1 = n1+(sum(Edge[ii,ii])-n1)/2
  sum2 = n2+(sum(Edge[jj,jj])-n2)/2
  Tr = (sum1+sum2)/num_edge
  return (Tr)
}
```

- **Permutation Test**

```
permT <- function(x,B,ix,size,custom=NULL,Fun) {
  T0 = Fun(x,ix,size,custom)
  Tperm=c(rep(0,B))
  for(b in c(1:B)) {
    per_index = sample(1:nrow(x))
    Tperm[b] = Fun(x,per_index,size,custom)
  }
  P = mean(c(T0,Tperm)>=T0)
  return(P)
}
```

For these four tests, we need to specify the customized option to use these test except the Hotelling's T-square test.Because of the randomness of a single experiment, we always consider comparing the power values repeated 100 times under different conditions in the following numerical experiments.

For the rest of three test,if we use the Nearest Neighbor Test, we need to specify the k nearest neighbor, when we set fixed sample size with different k, then the outcomes are as follows:

we set sample size

$$n_1 = 50, n_2 = 50, d = 5$$

with different

$$k = 3, 7, 10$$

at two conditions: under null hypothesis and alternative hypothesis. When the dimension is low, the selection of k value does not affect the results, so we set $d = 20$.

- **Experiment 1**

```
exp1 <- function() {
  library(RANN)
  times = 100
  Alpha = 0.05
  d = 20
  n1 = 50
  B = 1000
  n2 = 50
  power = rep(0,times)
  for(k in c(3,7,10)){
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d,sd = 2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
```

```
      power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = k, NNT)
      }
    power_ = sum(power < Alpha)/times
    if(k == 3) {
      print("The power with k = 3:")
    }
    if(k == 7) {
      print("The power with k = 7:")
    }
    if(k == 10) {
      print("The power with k = 10:")
    }
    print(power_)
  }
}
#exp1()

# Since it takes such a long time to output the outcomes, I will leave them as comments.
#[1] "The power with k = 3:"
#[1] 0.51
#[1] "The power with k = 7:"
#[1] 0.93
#[1] "The power with k = 10:"
#[1] 1
```

The experiment 1 shows that the selection of k does have effect on the results with k increasing, the power of this test increases when k is set larger with fixed dimension d, but the calculation amount will increase when k is set too large. When k is too small, we can know that less information can be obtained, and the power of test will decrease. A compromise setting of k is better.

The next custom parameters is the selection of specific form of distance in energy distance test and graph-based test. Generally, we consider Minkowski distance. In this numerical experiment setting, we set

$$Lp = L_1, L_2, L_5$$

with small and large dimension $d = 5, 20$. the sample size is $n_1 = n_2 = 50$, and We repeat the simulation 100 times to see the power of these two test.

- **Experiment 2**

```
exp2 <- function(){
  times = 100
  n1 = 50
  n2 = 50
  d1 = 5
  d2 = 20
  B = 1000
  k = 3
  Alpha = 0.05
  for(p in c(1,2,5)){
    power1 = rep(0, times)
    for(i in c(1:times)) {
      x1 = matrix(rnorm(n1*d1,sd = 2), n1, d1)
      y1 = matrix(rnorm(n2*d1), n2, d1)
```

```
    z1 = rbind(x1, y1)
    power1[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = p, edist)
  }
  power1_ = sum(power1<Alpha)/times
  if (p == 1) {
    print("edist power with d = 5, p = 1:")
  }
  if (p == 2) {
    print("edist power with d = 5, p = 2:")
  }
  if (p == 5) {
    print("edist power with d = 5, p = 5:")
  }
  print(power1_)
}
for(p in c(1,2,5)) {
  power1 = rep(0, times)
  for(i in c(1:times)) {
    x1 = matrix(rnorm(n1*d2, sd = 2), n1, d2)
    y1 = matrix(rnorm(n2*d2), n2, d2)
    z1 = rbind(x1, y1)
    power1[i] = permT(z1, B, 1:nrow(z1), c(n1, n2), custom = p, edist)
  }
  power1_ = sum(power1 < Alpha)/times
  if(p == 1) {
    print("edist power with d = 20, p = 1:")
  }
  if(p == 2) {
    print("edist power with d = 20, p = 2:")
  }
  if(p == 5) {
    print("edist power with d = 20, p = 5:")
  }
  print(power1_)
}
Q = 3
for(p in c(1,2,5)){
  power2 = rep(0, times)
  for(i in c(1:times)) {
    x1 = matrix(rnorm(n1*d1, sd = 2), n1, d1)
    y1 = matrix(rnorm(n2*d1), n2, d1)
    z1 = rbind(x1, y1)
    power2[i] = permT(z1, B, 1:nrow(z1), c(n1, n2), custom = c(p,3), GraphT)
  }
  power2_ = sum(power1 < Alpha)/times
  if(p == 1) {
    print("Graph power with d = 5, p = 1:")
  }
  if(p == 2) {
    print("Graph power with d = 5, p = 2:")
  }
  if(p == 5) {
    print("Graph power with d = 5, p = 5:")
```

```
    }
    print(power2_)
  }
  for(p in c(1,2,5)){
    power2 = rep(0, times)
    for(i in c(1:times)) {
      x1 = matrix(rnorm(n1*d2, sd = 2), n1, d2)
      y1 = matrix(rnorm(n2*d2), n2, d2)
      z1 = rbind(x1, y1)
      power2[i] = permT(z1, B, 1:nrow(z1), c(n1, n2), custom=c(p,3), GraphT)
    }
    power2_ = sum(power1 < Alpha)/times
    if(p == 1) {
      print("Graph power with d = 20, p = 1:")
    }
    if(p == 2) {
      print("Graph power with d = 20, p = 2:")
    }
    if(p == 5) {
      print("Graph power with d = 20, p = 5:")
    }
    print(power2_)
  }
}

#exp2()

# Since it takes such a long time to output the outcomes, I will leave them as comments.
# [1] "edist power with d = 5, p = 1:"
# [1] 1
# [1] "edist power with d = 5, p = 2:"
# [1] 1
# [1] "edist power with d = 5, p = 5:"
# [1] 1
# [1] "edist power with d = 20, p = 1:"
# [1] 1
# [1] "edist power with d = 20, p = 2:"
# [1] 1
# [1] "edist power with d = 20, p = 5:"
# [1] 1
# [1] "Graph power with d = 5, p = 1:"
# [1] 1
# [1] "Graph power with d = 5, p = 2:"
# [1] 1
# [1] "Graph power with d = 5, p = 5:"
# [1] 1
# [1] "Graph power with d = 20, p = 1:"
# [1] 1
# [1] "Graph power with d = 20, p = 2:"
# [1] 1
# [1] "Graph power with d = 20, p = 5:"
# [1] 1
```

From this numerical experiment test, we can find out that the powers of these two tests are high whether dimension d is small or high. No matter what value p is, it is equivalent. These two test show a better performance but the calculation is high with p increasing.

The final custom parameters is the selection of Q, where Q is a positive number to control whether there is an edge between Z_i, Z_j if $\|Z_i - Z_j\| \leq Q$. In this numerical experiment, we consider Euclidean distance. In this numerical experiment setting, we set

$$Q = 3, 5, 10$$

with small and large dimension $d = 5, 20$. The sample size is $n_1 = n_2 = 50$, and we repeat the simulation 100 times to see the power of this test.

- **Experiment 3**

```
exp3 <- function(){
  times = 100
  n1 = 50
  n2 = 50
  d1 = 5
  d2 = 20
  B = 1000
  k = 3
  Alpha = 0.05
  power1 = rep(0,times)
  for(Q in c(3,5,10)) {
    for(i in c(1:times)) {
      x1 = matrix(rnorm(n1*d1, sd = 2), n1, d1)
      y1 = matrix(rnorm(n2*d1), n2, d1)
      z1 = rbind(x1, y1)
      power1[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = c(2, Q), GraphT)
    }
    power1_ = sum(power1 < Alpha)/times
    if(Q == 3) {
      print('the power with d = 5, Q = 3:')
    }
    if(Q == 5) {
      print('the power with d = 5, Q = 5:')
    }
    if(Q == 10) {
      print('the power with d = 5, Q = 10:')
    }
    print(power1_)
  }
  for(Q in c(3,5,10)){
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d2, sd = 2), n1, d2)
      y1 = matrix(rnorm(n2*d2), n2, d2)
      z1 = rbind(x1, y1)
      power1[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = c(2,Q), GraphT)
    }
    power1_ = sum(power1 < Alpha)/times
    if(Q == 3) {
      print('the power with d = 20, Q = 3:')
    }
    if(Q == 5) {
```

```
      print('the power with d = 20, Q = 5:')
    }
    if(Q == 10) {
      print('the power with d = 20, Q = 10:')
    }
    print(power1_)
  }
}

#exp3()

# Since it takes such a long time to output the outcomes, I will leave them as comments.
# [1] "the power with d = 5, Q = 3:"
# [1] 1
# [1] "the power with d = 5, Q = 5:"
# [1] 0.98
# [1] "the power with d = 5, Q = 10:"
# [1] 0
# [1] "the power with d = 20, Q = 3:"
# [1] 0
# [1] "the power with d = 20, Q = 5:"
# [1] 1
# [1] "the power with d = 20, Q = 10:"
# [1] 0.36
```

From this numerical experiment test, we can find out that when dimension d is small, the Q with a small one is fine, but if Q is too big, the power of this test is not good. When dimension d is big, the small Q doesn't have a good performance any more. A large Q is fine with a good performance, but if Q is too big, the performance of this test is also not good as same as the condition with a low dimension d. It is necessary that a appropriate threshold is set. I would suggest $Q = 5$.

When we discuss whether those tests are sensitive to dimensions, from some following numerical experiments, these tests show a good fit to experiments with moderate sample sizes if d is not too large.

In those four tests, we set some general settings. Because of the computing power of the computer, we try to test it on a small scale, but it can reflect the changes. We set sample size

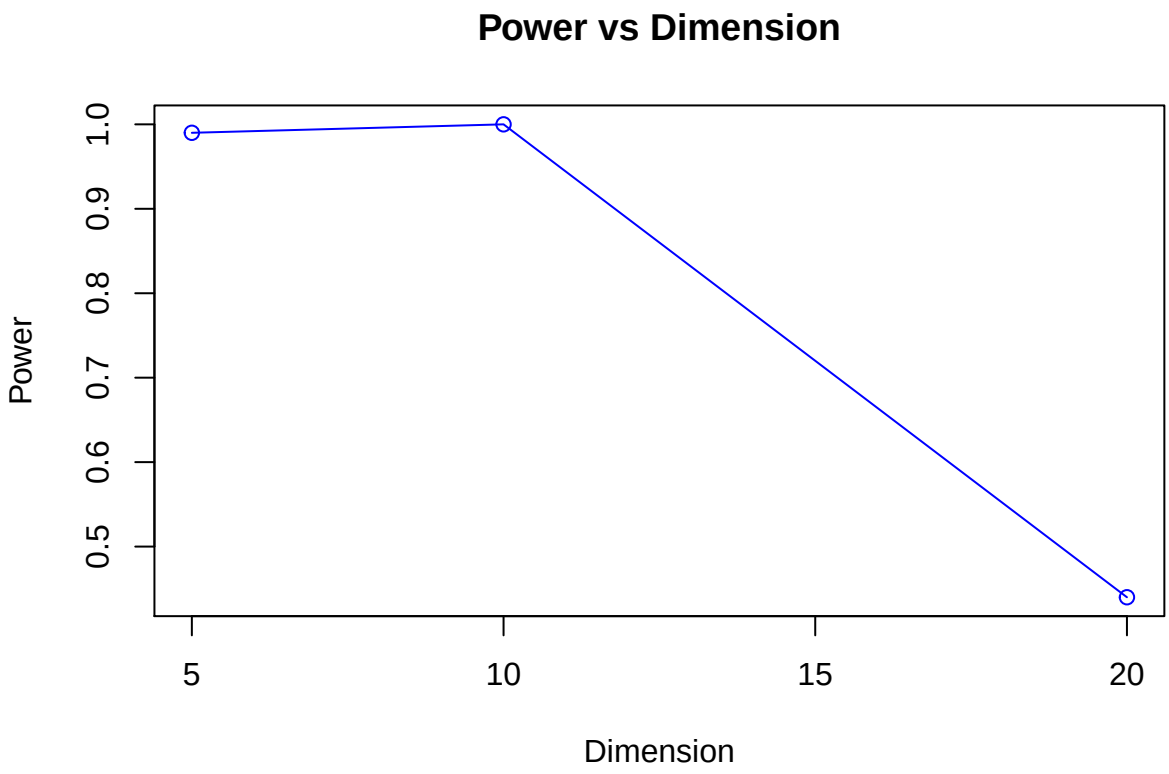
$$d = 5, 10, 20$$

with $n_1 = 50, n_2 = 50$, for some special settings, we set $k = 3$ for the nearest neighbor test, $Q = 5$ for the graph-based two sample test, and $p = 2$ of distance form for energy distance test and the graph-based two sample test. We repeat the simulation for 100 times.

- **Nearest Neighbor Test with Dimension Increasing**

```
NNTdim <- function(){
  library(RANN)
  times = 100
  n1 = 50
  n2 = 50
  B = 1000
  Alpha = 0.05
  power_result = rep(0,3)
  power = rep(0,times)
```

```
  j = 1
  for(d in c(5,10,20)){
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d,sd =2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = 3, NNT)
    }
    power_ = sum(power < Alpha)/times
    if(d == 5) {
      print("The power with d = 5:")
    }
    if(d == 10) {
      print("The power with d = 10:")
    }
    if(d == 20) {
      print("The power with d = 20:")
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result1 = NNTdim()
power_result1 = c(0.99,1,0.44) # The result after running
plot(c(5,10,20), power_result1, type = 'o', col = "blue", xlab = "Dimension", ylab = "Power", main = "P
```

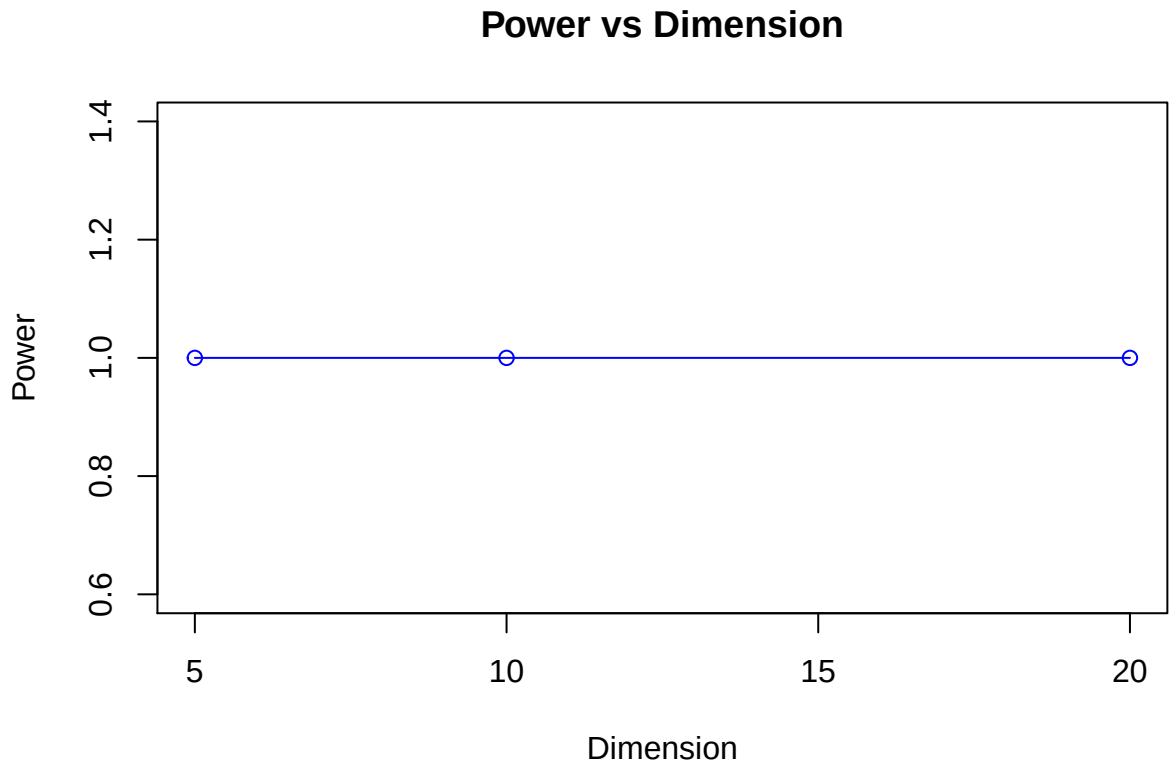


According to this graph, it suggests that the Nearest Neighbor Test is sensitive with dimension d, the power of the test decreases when dimension is high. However, the choice of the number of neighbors also takes into consideration to the sensitivity to the dimension.

- **Energy Distance Test with Dimension Increasing**

```
edistdim <- function() {
  times = 100
  n1 = 50
  n2 = 50
  B = 1000
  Alpha = 0.05
  power_result = rep(0,3)
  power = rep(0,times)
  j = 1
  for(d in c(5,10,20)) {
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d, sd = 2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      power[i] = permT(z1, B, 1:nrow(z1), c(n1, n2), custom = 2, edist)
    }
    power_=sum(power < Alpha)/times
    if(d == 5) {
      print("The power with d = 5:")
```

```
    }
    if(d == 10) {
      print("The power with d = 10:")
    }
    if(d == 20) {
      print("The power with d = 20:")
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result2 = edistdim()
power_result2 = c(1,1,1) # The result after running
plot(c(5,10,20), power_result2, type = 'o', col = "blue", xlab = "Dimension", ylab = "Power", main = "P
```

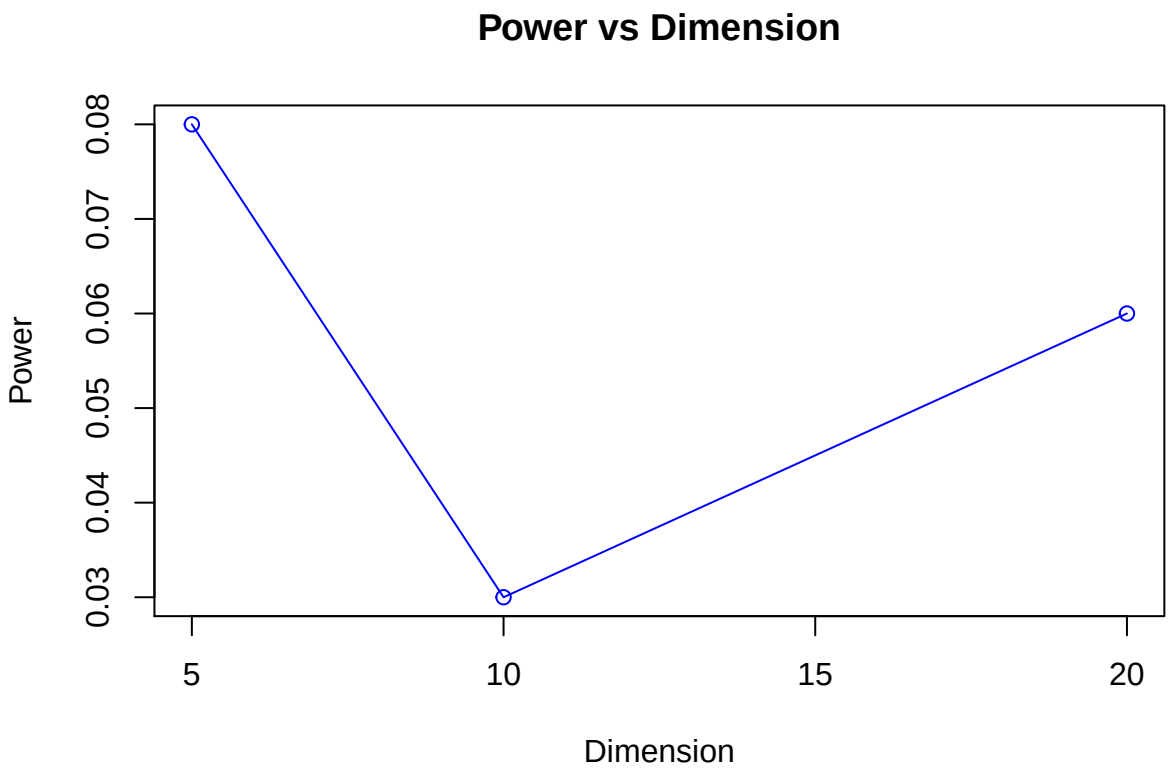


From this experiment, we can find that the Energy Distance Test is not sensitive with dimension d.

- **Hotelling's T-square Test with Dimension Increasing**

```
HT_sqTdim <- function() {
  times = 100
  n1 = 100
  n2 = 100
  B = 1000
```

```
  Alpha = 0.05
  power_result = rep(0,3)
  j = 1
  for(d in c(5,10,20)){
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d, sd = 2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = NULL, HT_sqT)
    }
    power_ = sum(power < Alpha)/times
    if(d == 5) {
      print("the power with d = 5:")
    }
    if(d == 10) {
      print("the power with d = 10:")
    }
    if(d == 20) {
      print("the power with d = 20:")
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result3 = HT_sqTdim()
power_result3 = c(0.08,0.03,0.06) # The result after running
plot(c(5,10,20), power_result3, type = 'o', col = "blue", xlab = "Dimension", ylab = "Power", main = "P
```

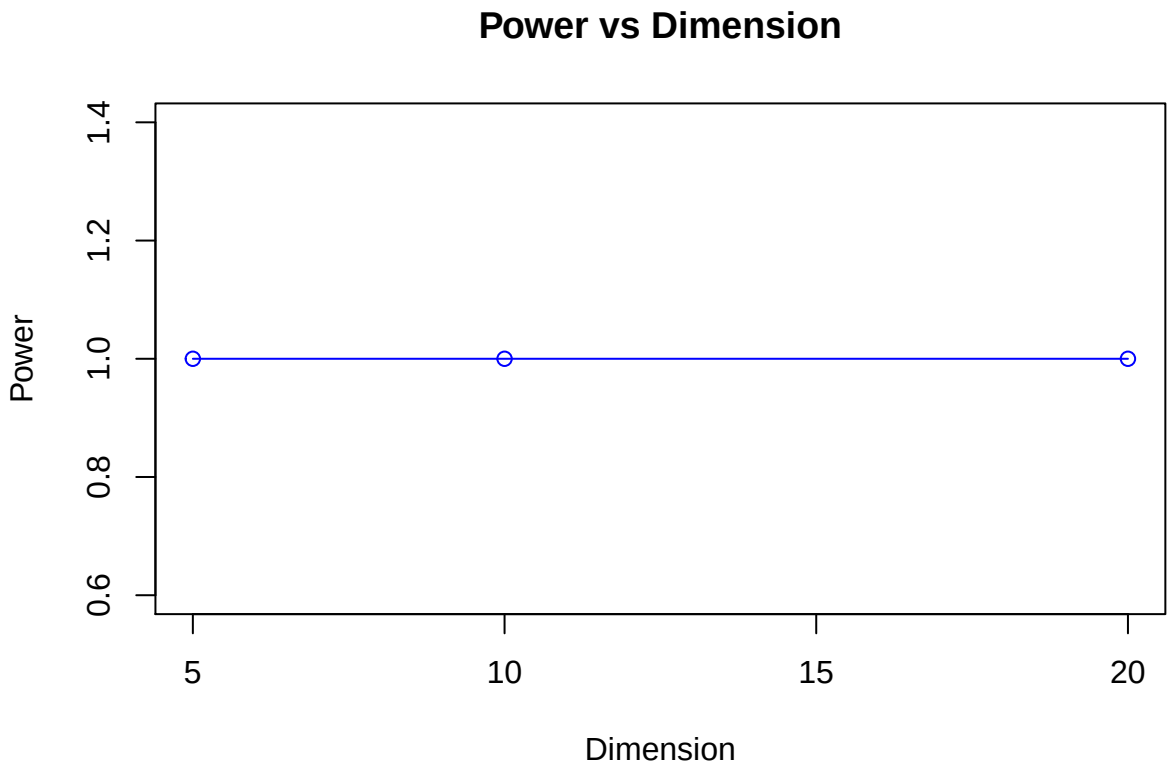


We can see the power is too low in this setting because the difference between two distributions is not significant, and we know the covariance matrix is sensitive to the dimension when the dimension increases. The estimator of covariance is no longer consistent, when we use distributions with significant difference.

- **Hotelling's T-square Test with Dimension Increasing and Significant Different Distributions**

```
HT_sqTdim2 <- function(){
  times = 100
  n1 = 100
  n2 = 100
  B = 1000
  Alpha = 0.05
  power_result = rep(0,3)
  j = 1
  for(d in c(5,10,20)){
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d,mean = 1), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = NULL, HT_sqT)
    }
    power_ = sum(power < Alpha)/times
    if(d == 5) {
      print("The power with d = 5:")
```

```
    }
    if(d == 10) {
      print("The power with d = 10:")
    }
    if(d == 20) {
      print("The power with d = 20:")
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result4 = HT_sqTdim2()
power_result4 = c(1,1,1) # The result after running
plot(c(5,10,20), power_result4, type = 'o', col = "blue", xlab = "Dimension", ylab = "Power", main = "P
```

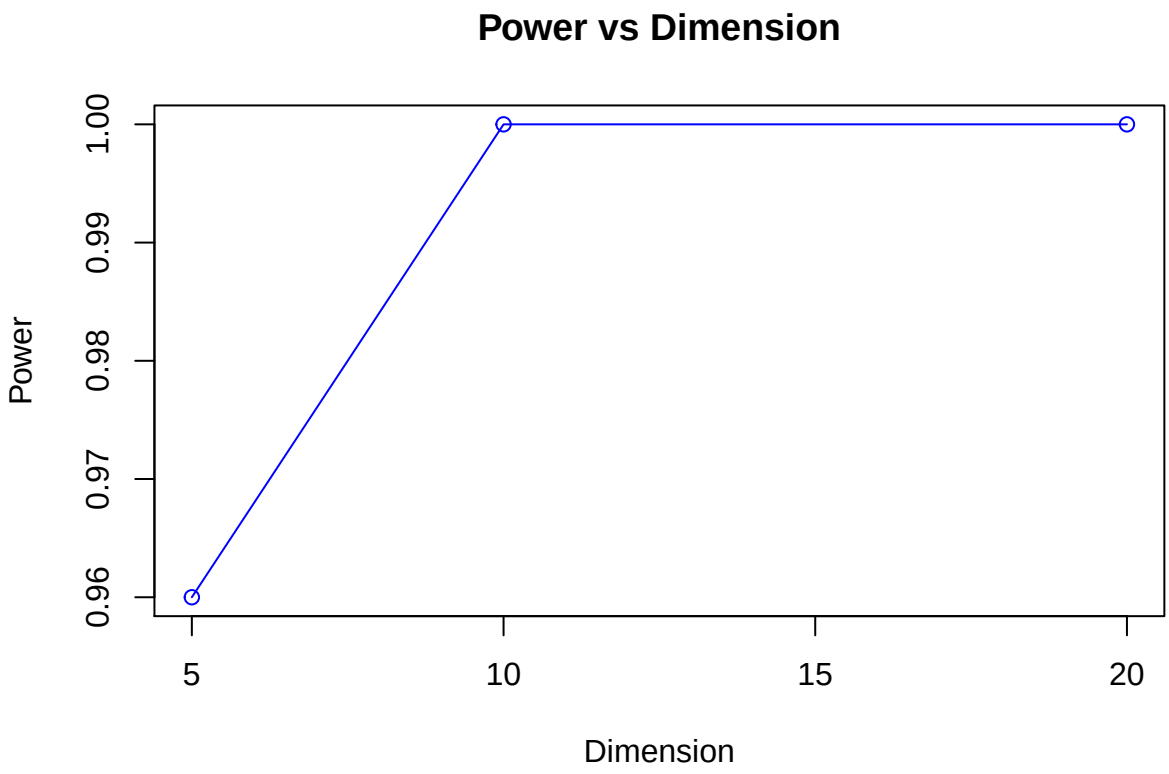


We can see that when the dimension increases, the power stays almost the same and is large. The fluctuation of the values of power under two different conditions shows that the Hotelling's T-square test is sensitive to the dimension.

- **Graph-based Test with Dimension Increasing**

```
GraphTdim <- function(){
  times = 100
```

```
  n1 = 50
  n2 = 50
  B = 1000
  Alpha = 0.05
  power_result = rep(0,3)
  j = 1
  for(d in c(5,10,20)){
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d,sd = 2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = c(2,5), GraphT)
    }
    power_ = sum(power < Alpha)/times
    if(d == 5) {
      print('the power with d=5:')
    }
    if(d == 10) {
      print('the power with d=10:')
    }
    if(d == 20) {
      print('the power with d=20:')
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result5 = GraphTdim()
power_result5 = c(0.96,1,1) # The result after running
plot(c(5,10,20), power_result5, type = 'o', col = "blue", xlab = "Dimension", ylab = "Power", main = "P
```



This graph shows that the graph-based two sample test is not sensitive to dimension d.

To investigate which test has larger power under certain conditions, we will consider the conditions when the dimension is small and large.

- **Small Dimension**

In this numerical experiment, we set some general settings. Because of the computing power of the computer, we try to test it on a small scale, but it can reflect the changes. We set sample size $d = 5$ with $n_1 = 100, n_2 = 100$,for some special settings, we set $k = 3$ for NNT test, $Q = 5$ for GraphT, and $p = 2$ of distance form for energy test and Graph test We repeat the simulation 100 times.

```
smalldim <- function(){
  times = 100
  n1 = 100
  n2 = 100
  B = 1000
  k = 3
  Q = 5
  p = 2
  d = 5
  Alpha = 0.05
  power_result = rep(0,4)
  j = 1
  for(w in c("NNT","edist","HT_sqT","GraphT")){
    if(w == "NNT") {
```

```
      custom = k
    }
    if(w == "edist") {
      custom = p
    }
    if(w == "HT_sqT") {
      custom=NULL
    }
    if(w == "GraphT") {
      custom = c(p,Q)
    }
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d,sd =2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      if(w == "NNT") {
        power[i] = permT(z1,B,1:nrow(z1),c(n1,n2),custom=custom,NNT)
      }
      if(w == "edist") {
        power[i] = permT(z1,B,1:nrow(z1),c(n1,n2),custom=custom,edist)
      }
      if(w == "HT_sqT") {
        power[i] = permT(z1,B,1:nrow(z1),c(n1,n2),custom=custom,HT_sqT)
      }
      if(w == "GraphT") {
        power[i] = permT(z1,B,1:nrow(z1),c(n1,n2),custom=custom,GraphT)
      }
    }
    power_ = sum(power < Alpha)/times
    if(w == "NNT") {
      print("NNT power:")
    }
    if(w == "edist") {
      print("edist power:")
    }
    if(w == "HT_sqT") {
      print("HT_sqT power:")
    }
    if(w == "GraphT") {
      print("GraphT power:")
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result6 = smalldim()
#power_result6

# Since it takes such a long time to output the outcomes, I will leave them as comments.
#[1] "NNT power:"
```

```
#[1] 1
#[1] "edist power:"
#[1] 1
#[1] "HT_sqT power:"
#[1] 0.99
#[1] "GraphT power:"
#[1] 1
```

- **Large Dimension**

In this numerical experiment, we set some general settings. Because of the computing power of the computer, we try to test it on a small scale, but it can reflect the changes. We set sample size $d = 20$ with $n_1 = 100, n_2 = 100$,for some special settings, we set $k = 3$ for NNT test, $Q = 5$ for GraphT, and $p = 2$ of distance form for energy test and Graph test We repeat the simulation 100 times.

```
largedim <- function(){
  times = 100
  n1 = 100
  n2 = 100
  B = 1000
  k = 3
  Q = 5
  p = 2
  d = 20
  Alpha = 0.05
  power_result = rep(0,4)
  j = 1
  for(w in c("NNT","edist","HT_sqT","GraphT")) {
    if(w == "NNT") {
      custom = k
    }
    if(w == "edist") {
      custom = p
    }
    if(w == "HT_sqT") {
      custom = NULL
    }
    if(w == "GraphT") {
      custom=c(p,Q)
    }
    power = rep(0,times)
    for(i in c(1:times)){
      x1 = matrix(rnorm(n1*d,sd = 2), n1, d)
      y1 = matrix(rnorm(n2*d), n2, d)
      z1 = rbind(x1, y1)
      if(w == "NNT") {
        power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = custom, NNT)
      }
      if(w == "edist") {
        power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = custom, edist)
      }
      if(w == "HT_sqT") {
        power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = custom, HT_sqT)
```

```
      }
      if(w == "GraphT") {
        power[i] = permT(z1, B, 1:nrow(z1), c(n1,n2), custom = custom, GraphT)
      }
    }
    power_ = sum(power < Alpha)/times
    if(w == "NNT") {
      print("NNT power:")
    }
    if(w == "edist") {
      print("edist power:")
    }
    if(w == "HT_sqT") {
      print("HT_sqT power:")
    }
    if(w == "GraphT") {
      print("GraphT power:")
    }
    print(power_)
    power_result[j] = power_
    j = j+1
  }
  return(power_result)
}
#power_result7 = largedim()
#power_result7

# Since it takes such a long time to output the outcomes, I will leave them as comments.
#[1] "NNT power:"
#[1] 0.99
#[1] "edist power:"
#[1] 1
#[1] "HT_sqT power:"
#[1] 0.06
#[1] "GraphT power:"
#[1] 1
```

From this report, we can find that Hotelling's T-square test is very sensitive with dimension d because the method it uses, we need to calculate the variance of sample, but this test is fast if we use this test in a condition that dimension d is relativity low to sample size. Also, this test is sensitive with specific distribution. If two distributions are not quite distinguishable, the performance is not good. The other tests, including nearest neighbor test, energy distance test, and graph-based two sample test, are not sensitive to dimension d, but some parameters need to be customized, such as the threshold Q in Graph Test, and the speed of these three tests is relativity slow. All tests can control type I error in specific conditions.

These five functions, including NNT, edist, HT_sqt, GraphT, and permT(), are packaged into an R package called andy.package.

Question 2 Presentation and Slides

Based your report, could you prepare a 3-5 minutes presentation to summarize your findings and suggestions? Assume your audience is the manager from this pharmaceutical company, who has only very limited statistic

background. In this question, you need to submit a video (I need to see you in this video) and your slides (Both Rmd and pdf).

Question 3 R package (Bonus question: extra 10 points for the final project)

Could you prepare an R package to include all your four two sample testing methods and a manual that introduces how these methods can be used? To finish this question, you need to submit a compressed R package.