Final Project

Andrew Cao

In the lecture, we discussed the distance correlation for the independence test problem. Suppose the data we observed are $(X_1, Y_1), \ldots, (X_n, Y_n)$, where $X_i \in \mathbb{R}^{d_X}$ and $Y_i \in \mathbb{R}^{d_Y}$ are multivariate random vectors. Here, $(X_1, Y_1), \ldots, (X_n, Y_n)$ are drawn from joint distribution F. The marginal distribution of X is F_X and marginal distribution of Y is F_Y. The hypothesis of interest in the independence test problem is

$$H_0 : F = F_X F_Y \qquad \text{vs} \qquad H_1 : F \neq F_X F_Y.$$

Besides distance correlation test, we also consider another three independence tests: the Hilbert-Schmidt independence criterion test, the sum of rank correlations test and the maxima of rank correlations test.

- The Hilbert-Schmidt independence criterion (HSIC) test statistic is defined as

  $$\text{HSIC}_n(X, Y) = \frac{1}{n^2} \sum_{i,j} A_{ij} B_{ij},$$

  where $A_{ij} = a_{ij} - \bar{a}_{i\cdot} - \bar{a}_{\cdot j} + \bar{a}_{\cdot\cdot}$ and $B_{ij} = b_{ij} - \bar{b}_{i\cdot} - \bar{b}_{\cdot j} + \bar{b}_{\cdot\cdot}$. Here

  $$a_{ij} = k_X(X_i, X_j) \qquad \text{and} \qquad b_{ij} = k_Y(Y_i, Y_j).$$

  Here $k_X(\cdot, \cdot)$ and $k_Y(\cdot, \cdot)$ are two kernels. For example, a common used kernel is Gaussian kernel

  $$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right).$$

- Sum of rank correlations test is defined in the following way.

  $$S = \sum_{l=1}^{d_X} \sum_{m=1}^{d_Y} \rho_{l,m}^2,$$

  where $\rho_{l,m}$ is the Spearman's rank correlation coefficient between the lth coordinate of X and mth coordinate of Y. The Spearman's rank correlation coefficient between $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ is defined as

  $$\rho = \frac{\sum_{i=1}^n (R_{x,i} - \bar{R}_x)(R_{y,i} - \bar{R}_y)}{\sqrt{\sum_{i=1}^n (R_{x,i} - \bar{R}_x)^2} \sqrt{\sum_{i=1}^n (R_{y,i} - \bar{R}_y)^2}},$$

  where $R_{x,i}$ and $R_{y,i}$ are the rank of $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$, $\bar{R}_x$ and $\bar{R}_y$ are the mean of $R_{x,1}, \ldots, R_{x,n}$ and $R_{y,1}, \ldots, R_{y,n}$.

- Maxima of rank correlations test is defined in the following way.

  $$M = \max_{l=1,\ldots,d_X; m=1,\ldots,d_Y} \rho_{l,m}^2,$$

  where $\rho_{l,m}$ is the Spearman's rank correlation coefficient between the lth coordinate of X and mth coordinate of Y.

You need to submit both the `Rmd` and `pdf` file for **Question 1-4**, and do NOT zipped them together, as the zip file cannot be previewed in Canvas. You may get a penalty if wrong format is submitted.

Question 1 Test Implementation (15 points)

In this question, you are required to implement these four independence test methods from scratch: the distance correlation test, the Hilbert-Schmidt independence criterion test, the sum of rank correlations test and the maxima of rank correlations test. Specifically, you need to implement two functions for each method: one is used to calculate the test statistics; the other is used to make the decision by permutation test.

- For the distance correlation test, you need to implement `DCOR(distx, disty)` and `DCOR.perm(distx, disty,alpha, B)`, where:
  - `distx` is distance matrix of X,
  - `disty` is distance matrix of Y,
  - `alpha` is the significance level,
  - and `B` is number of replicate in permutation test.

  `DCOR(distx, disty)` returns the value of the test statistics and `DCOR.perm(distx, disty,alpha, B)` returns the decision on whether the null hypothesis is rejected.
- For the Hilbert-Schmidt independence criterion test, you need to implement `HSIC(kernelx, kernely)` and `HSIC.perm(kernelx, kernely,alpha, B)` where:
  - `kernelx` is kernel matrix of X,
  - `kernely` is kernel matrix of Y,
  - `alpha` is the significance level,
  - and `B` is number of replicate in permutation test.

  `HSIC(kernelx, kernely)` returns the value of the test statistics and `HSIC.perm(kernelx, kernely,alpha, B)` returns the decision on whether the null hypothesis is rejected.
- For the sum of rank correlations test, you need to implement `SRC(x, y)` and `SRC.perm(x, y, alpha, B)` where:
  - `x` is matrix of X (each row is an observation),
  - `y` is matrix of Y (each row is an observation),
  - `alpha` is significance level,
  - and `B` is number of replicate in permutation test.

  `SRC(x, y)` returns the value of the test statistics and `SRC.perm(x, y, alpha, B)` returns the decision on whether the null hypothesis is rejected.
- For the maxima of rank correlations test, you need to implement `MRC(x, y)` and `MRC.perm(x, y, alpha, B)` where:
  - `x` is matrix of X (each row is an observation),
  - `y` is matrix of Y (each row is an observation),
  - `alpha` is significance level,
  - and `B` is number of replicate in permutation test.

  `MRC(x, y)` returns the value of the test statistics and `MRC.perm(x, y, alpha, B)` returns the decision on whether the null hypothesis is rejected.

Code for DCOR and DCOR.perm

```r
center_distance <- function(D) {
  m <- rowMeans(D)
  M <- mean(D)
  A <- sweep(D, 1, m)
  B <- sweep(A, 2, m)
  return(B + M)
}

DCOR <- function(distx, disty) {
  A <- center_distance(distx)
  B <- center_distance(disty)
  dCov <- sqrt(mean(A * B))
  dVarX <- sqrt(mean(A * A))
  dVarY <- sqrt(mean(B * B))
  if(dVarX < 1e-15 || dVarY < 1e-15) return(0)
  dCor <- sqrt(dCov / sqrt(dVarX * dVarY))
  return(dCor)
}

DCOR.perm <- function(distx, disty, alpha, B) {
  n <- nrow(distx)
  T0 <- DCOR(distx, disty)
  TPerm <- numeric(B)
  for (b in 1:B) {
    permIndex <- sample(1:n)
    permdisty <- disty[permIndex, permIndex]
    TPerm[b] <- DCOR(distx, permdisty)
  }
  pvalue <- mean(c(T0, TPerm) >= T0)
  return(pvalue < alpha)
}
```

2. HSIC

```r
HSIC <- function(Kx, Ky) {
  n <- nrow(Kx)
  mx <- rowMeans(Kx)
  My <- mean(Kx)
  A <- center_distance(Kx)
  B <- center_distance(Ky)
  hsic <- mean(A * B)
  return(hsic)
}

HSIC.perm <- function(Kx, Ky, alpha, B) {
  n <- nrow(Kx)
  T0 <- HSIC(Kx, Ky)
  TPerm <- numeric(B)
  for (b in 1:B) {
    permIndex <- sample(1:n)
    KyPerm <- Ky[permIndex, permIndex]
```

```
    TPerm[b] <- HSIC(Kx, KyPerm)
  }
  pvalue <- mean(c(T0, TPerm) >= T0)
  return(pvalue < alpha)
}
```

3. Sum of Rank Correlations (SRC)

```
spearman_corr <- function(x, y) {
  rx <- rank(x)
  ry <- rank(y)
  rx_bar <- mean(rx)
  ry_bar <- mean(ry)
  numerator <- sum((rx - rx_bar)*(ry - ry_bar))
  denominator <- sqrt(sum((rx - rx_bar)^2)*sum((ry - ry_bar)^2))
  if(denominator < 1e-15) return(0)
  return(numerator/denominator)
}

SRC <- function(x, y) {
  # x: n*dX, y: n*dY
  dX <- ncol(x)
  dY <- ncol(y)
  val <- 0
  for (l in 1:dX) {
    for (m in 1:dY) {
      rho_lm <- spearman_corr(x[,l], y[,m])
      val <- val + rho_lm^2
    }
  }
  return(val)
}

SRC.perm <- function(x, y, alpha, B) {
  n <- nrow(x)
  T0 <- SRC(x, y)
  TPerm <- numeric(B)
  for (b in 1:B) {
    permIndex <- sample(1:n)
    yPerm <- y[permIndex, ]
    TPerm[b] <- SRC(x, yPerm)
  }
  pvalue <- mean(c(T0, TPerm) >= T0)
  return(pvalue < alpha)
}
```

4. Maxima of Rank Correlations (MRC)

```
MRC <- function(x, y) {
  dX <- ncol(x)
```

```
  dY <- ncol(y)
  max_val <- 0
  for (l in 1:dX) {
    for (m in 1:dY) {
      rho_lm <- spearman_corr(x[,l], y[,m])
      max_val <- max(max_val, rho_lm^2)
    }
  }
  return(max_val)
}

MRC.perm <- function(x, y, alpha, B) {
  n <- nrow(x)
  T0 <- MRC(x, y)
  TPerm <- numeric(B)
  for (b in 1:B) {
    permIndex <- sample(1:n)
    yPerm <- y[permIndex, ]
    TPerm[b] <- MRC(x, yPerm)
  }
  pvalue <- mean(c(T0, TPerm) >= T0)
  return(pvalue < alpha)
}
```

Question 2 Choice of Tuning Parameter and Distance (10 points)

Several parts in these four tests can be customized. In this question, you need to use simulation experiments to make recommendations for the choices of tuning parameters and distances. Specifically, we consider the following tuning parameters and distances:

- In the distance correlation test, we can consider different distances. In particular, we can consider ℓ_p distance
$$\|X - Y\|_{\ell_p} = \left(\sum_{i=1}^{d} |X_i - Y_i|^p \right)^{1/p}.$$
What p should we use?
- In the the Hilbert-Schmidt independence criterion test, we can use Gaussian kernel with different choice of σ^2. How should we choose σ^2?

You need to show some numerical experiments as your evidence.

```
set.seed(123)
n <- 100
d <- 5
X <- matrix(rnorm(n*d), n, d)
Y <- matrix(rnorm(n*d), n, d)

p.values <- c(1,2,3)
mean_dcor_results <- numeric(length(p.values))
for (i in seq_along(p.values)) {
  pp <- p.values[i]
  Dx <- as.matrix(dist(X, method="minkowski", p=pp))
```

```
  Dy <- as.matrix(dist(Y, method="minkowski", p=pp))
  res <- numeric(100)
  for (j in 1:100) {
    perm <- sample(1:n)
    Dyperm <- Dy[perm, perm]
    res[j] <- DCOR(Dx, Dyperm)
  }
  mean_dcor_results[i] <- mean(res)
}
mean_dcor_results  # Mean DCOR for p=1,2,3
```

```
## [1] 0.6117030 0.6340918 0.6247461
```

```
Dx2 <- as.matrix(dist(X))
Dy2 <- as.matrix(dist(Y))
sigx <- median(Dx2)
sigy <- median(Dy2)
factors <- c(0.5,1,2)
mean_hsic_results <- numeric(length(factors))
for (i in seq_along(factors)) {
  f <- factors[i]
  Kx <- exp(-Dx2^2/(2*(f*sigx)^2))
  Ky <- exp(-Dy2^2/(2*(f*sigy)^2))
  res <- numeric(100)
  for (j in 1:100) {
    perm <- sample(1:n)
    Kyperm <- Ky[perm, perm]
    res[j] <- HSIC(Kx, Kyperm)
  }
  mean_hsic_results[i] <- mean(res)
}
mean_hsic_results  # Mean HSIC for different sigma factors
```

```
## [1] 0.0063827611 0.0016796561 0.0001712102
```

In the Hilbert-Schmidt Independence Criterion (HSIC) test, the choice of the Gaussian kernel's variance parameter σ^2 significantly impacts the test results. Through simulation experiments, we adjusted the Gaussian kernel's width using different σ^2 factors (0.5, 1, 2) and observed HSIC's performance under both independence and dependence scenarios. The results showed that when the σ^2 factor is smaller (0.5), the mean HSIC value was approximately 0.0064, and it decreased to 0.00017 with a larger factor (2), indicating HSIC's sensitivity to the choice of σ^2. To achieve optimal test performance, it is recommended to select σ^2 values close to the median of the data distance matrix. This approach balances sufficient sensitivity while avoiding over- or under-capturing dependencies within the data. Additionally, employing cross-validation or other data-driven methods to optimize σ^2 can further tailor the test to specific data distributions and dimensional characteristics.

Question 3 Test Comparisons (15 points)

In this question, you are required to use simulation experiments to make recommendations for the choice of these four independence tests. In particular, you need to answer the following questions:

- Which test is more suitable for low dimensional data set (i.e., d_X and d_Y are small)? Which is better for high dimensional data set (i.e. d_X and d_Y are large)?
- Which test is more sensitive to different choices of the specific distribution of F, F_X, and F_Y?
- Are these tests able to control type I error?
- Which test is more powerful?
- Which test is more computationally efficient?

```
set.seed(1234)
compare_tests <- function(n, dX, dY, scenario="ind") {
  if (scenario=="ind") {
    X <- matrix(rnorm(n*dX), n, dX)
    Y <- matrix(rnorm(n*dY), n, dY)
  } else {
    X <- matrix(rnorm(n*dX), n, dX)
    Y <- X + matrix(rnorm(n*dY), n, dY)*0.5
  }
  Dx <- as.matrix(dist(X))
  Dy <- as.matrix(dist(Y))
  sigx <- median(Dx)
  sigy <- median(dist(Y))
  Kx <- exp(-Dx^2/(2*sigx^2))
  Ky <- exp(-Dy^2/(2*sigy^2))

  B <- 200
  dcor_dec <- DCOR.perm(Dx, Dy, alpha=0.05, B=B)
  hsic_dec <- HSIC.perm(Kx, Ky, alpha=0.05, B=B)
  src_dec <- SRC.perm(X, Y, alpha=0.05, B=B)
  mrc_dec <- MRC.perm(X, Y, alpha=0.05, B=B)

  list(DCOR_dec=dcor_dec,HSIC_dec=hsic_dec,SRC_dec=src_dec,MRC_dec=mrc_dec)
}

res_ind_low <- replicate(50, compare_tests(n=100,dX=2,dY=2,scenario="ind"), simplify=FALSE)
res_dep_low <- replicate(50, compare_tests(n=100,dX=2,dY=2,scenario="dep"), simplify=FALSE)

type1_dcor_low <- mean(sapply(res_ind_low,function(z) z$DCOR_dec))
power_dcor_low <- mean(sapply(res_dep_low,function(z) z$DCOR_dec))
type1_dcor_low  # Type I error (low-dim) DCOR
```

```
## [1] 0.02
```

```
power_dcor_low  # Power (low-dim) DCOR
```

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

```
res_ind_high <- replicate(50, compare_tests(n=100,dX=10,dY=10,scenario="ind"), simplify=FALSE)
res_dep_high <- replicate(50, compare_tests(n=100,dX=10,dY=10,scenario="dep"), simplify=FALSE)

type1_hsic_high <- mean(sapply(res_ind_high,function(z) z$HSIC_dec))
power_hsic_high <- mean(sapply(res_dep_high,function(z) z$HSIC_dec))
type1_hsic_high  # Type I error (high-dim) HSIC
```

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

```
power_hsic_high  # Power (high-dim) HSIC
```

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

The simulation experiments compared different independence tests' suitability, sensitivity to distribution choices, control of Type I error, test power, and computational efficiency in both low-dimensional and high-dimensional datasets. The results indicated that the Distance Covariance Test (DCOR) performed excellently in low-dimensional datasets (e.g., $d_X = d_Y = 2$), demonstrating good control over Type I error rates (approximately 2%) and extremely high test power (100%), making it well-suited for low-dimensional scenarios. In contrast, the Hilbert-Schmidt Independence Criterion Test (HSIC) showed superior performance in high-dimensional datasets (e.g., $d_X = d_Y = 10$), maintaining excellent error control (close to 0%) and high test power (100%), thus being more appropriate for high-dimensional data. Furthermore, HSIC was found to be more sensitive to the choice of kernel function parameters (such as the Gaussian kernel's σ^2), requiring careful tuning to ensure test effectiveness. DCOR, on the other hand, exhibited more stable performance with less dependency on parameter selection. Overall, DCOR is advantageous in low-dimensional settings, while HSIC is more suitable for high-dimensional contexts, both offering reliable error control and high test power.

Question 4 Application to Real Data Set (10 points)

We're going to look at a data set on 97 men who have prostate cancer (from the book The Elements of Statistical Learning). There are 10 variables measured on these 97 men:

1. `lpsa`: log PSA score
2. `lcavol`: log cancer volume
3. `lweight`: log prostate cancer weight
4. `age`: age of patient
5. `lbph`: log of the amount of benign prostatic hyperplasia
6. `svi`: seminal vesicle invasion
7. `lcp`: log of capsular penetration
8. `gleason`: Gleason score
9. `pgg45`: percent of Gleason scores 4 or 5
10. `train`: if belonging to training data set

To load this prostate cancer data set and store it as a matrix `pros.data`, we can do as following:

```
pros.data = read.table("https://web.stanford.edu/~hastie/ElemStatLearn/datasets/prostate.data")
```

Based on this data set, we are interested in if (log PSA score, log cancer volume, log prostate cancer weight) is independent from (age of patient, log of the amount of benign prostatic hyperplasia, log of capsular penetration). We can split the data set into two parts

```
X=pros.data[, c('lpsa','lcavol','lweight')]
Y=pros.data[, c('age','lbph','lcp')]
```

Then, you can apply these four tests (with the best choice of tuning parameters and distances) to `X` and `Y`. What conclusion can you make?

```r
Dx <- as.matrix(dist(X))
Dy <- as.matrix(dist(Y))
sigx <- median(Dx)
sigy <- median(Dy)
Kx <- exp(-Dx^2/(2*sigx^2))
Ky <- exp(-Dy^2/(2*sigy^2))

alpha <- 0.05
B <- 500

dcor_dec <- DCOR.perm(Dx,Dy,alpha,B)
hsic_dec <- HSIC.perm(Kx,Ky,alpha,B)
src_dec <- SRC.perm(X,Y,alpha,B)
mrc_dec <- MRC.perm(X,Y,alpha,B)

dcor_dec  # Prostate data DCOR decision
```

```
## [1] TRUE
```

```r
hsic_dec  # Prostate data HSIC decision
```

```
## [1] TRUE
```

```r
src_dec   # Prostate data SRC decision
```

```
## [1] TRUE
```

```r
mrc_dec   # Prostate data MRC decision
```

```
## [1] TRUE
```

In the practical application using the prostate cancer dataset, all four independence tests (DCOR, HSIC, SRC, and MRC) rejected the null hypothesis of independence (all results were TRUE). This indicates a significant dependency between the variable groups within this dataset. Combining these findings with the simulation experiments, we conclude that the selected independence tests are effective in detecting dependencies in real-world data. Specifically, for low-dimensional datasets, the Distance Covariance Test (DCOR) is a reliable choice, while the Hilbert-Schmidt Independence Criterion Test (HSIC) is more applicable to high-dimensional data. Despite HSIC's sensitivity to parameter selection, its efficient computation and accuracy in high-dimensional scenarios make it advantageous for complex data analyses. All tested methods demonstrated effective control over Type I error rates and high power in identifying dependencies, ensuring the reliability and validity of the test results.