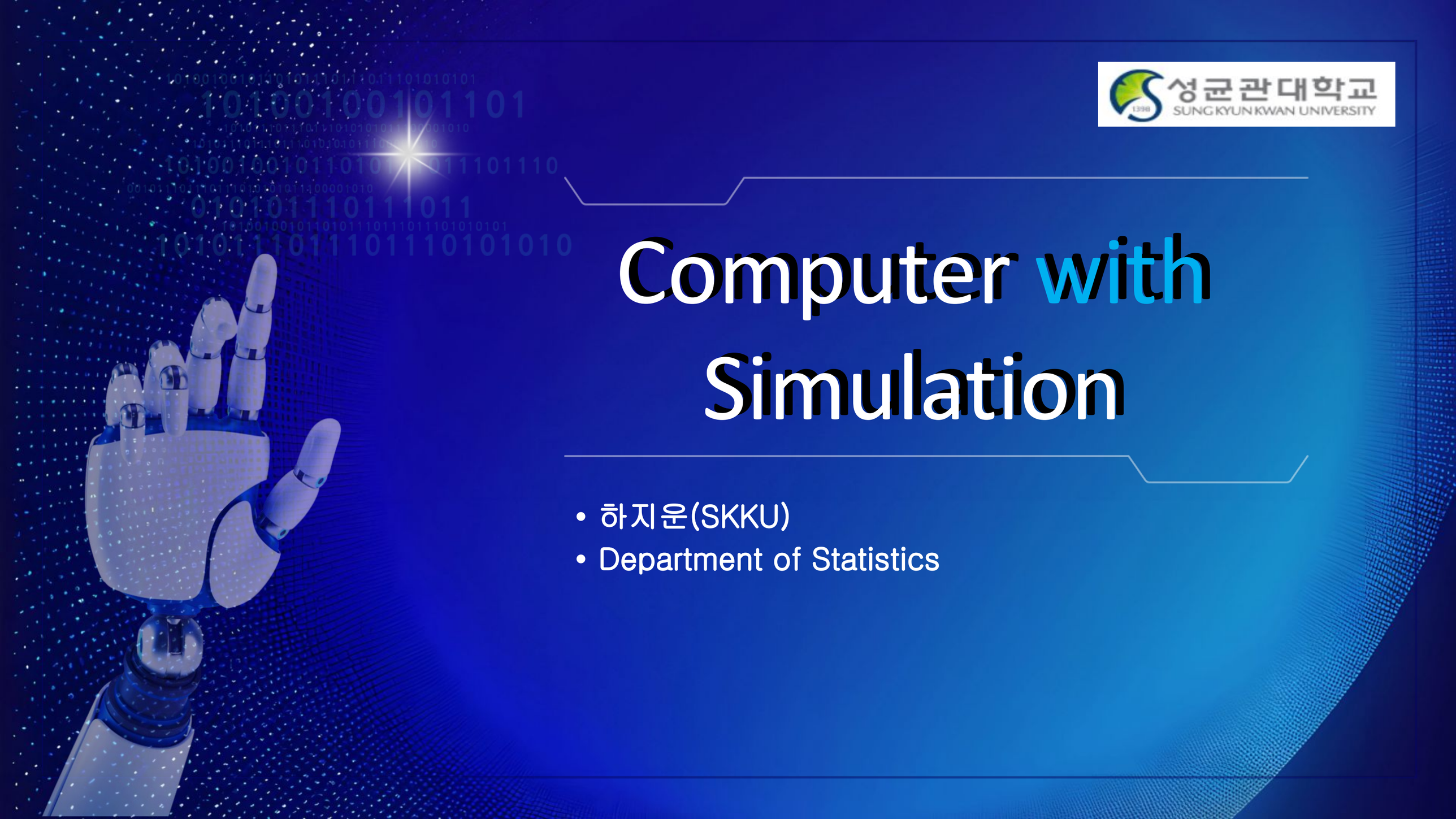

Computer with Simulation

- 하지운(SKKU)
- Department of Statistics

Index

1 Motivation

In AI period and big data period, we need the ability to analyze the data efficiently and to interpret the result of estimations.

But we have faced several problems: efficiency of running code and using memory, estimation for population, and evaluation for estimators.

How to deal with those problems?

Time Complexity

Definition

Let f and g be functions with domain $\{1, 2, 3, \dots\}$. We write

$$f(n) = O(g(n))$$

And say that $f(n)$ is of order at most $g(n)$ or $f(n)$ is *big oh* of $g(n)$ if there exists a positive constant C_1 such that

$$|f(n)| \leq C_1 |g(n)|$$

for all but finitely many positive integer n

Note that this inequality holds:

$$O(1) < O(\log\ n) < O(n) < O(n \log\ n) < O(n^2) < O(2^n) < O(n!) < O(n^n)$$

3 Time Complexity

Definition

Let f and g be functions with domain $\{1, 2, 3, \dots\}$. We write

$$f(n) = \Omega(g(n))$$

And say that $f(n)$ is of order at least $g(n)$ or $f(n)$ is *omega* of $g(n)$ if there exists a positive constant C_2 such that

$$|f(n)| \geq C_2|g(n)|$$

for all but finitely many positive integer n

4 Time Complexity

Definition

Let f and g be functions with domain $\{1, 2, 3, \dots\}$. We write

$$f(n) = \Theta(g(n))$$

And say that $f(n)$ is of order at $g(n)$ or $f(n)$ is *theta* of $g(n)$ if

$$f(n) = O(g(n)) \text{ and } f(n) = \Omega(g(n))$$

We will focus on $O(\cdot)$, called Big O notation.

5 Example

python3

```
print("Hello World!₩n")
```

C

```c
#include <stdio.h>

int main(){
    printf("Hello World!\n");
}

```

These two codes are exactly identical to print “Hello World!” from python3 and C respectively.
Each code is simple by executing the command only once!
Thus, these two codes’ time complexity is $O(1)$.

6 Example

```
n = int(input("Type positive integer: "))
x = 0
for i in range(n):
    for j in range(0,i):
        print(i, j)
        x = x+1
print(x)
```

```
Type positive integer: 5
1 0
2 0
2 1
3 0
3 1
3 2
4 0
4 1
4 2
4 3
10
```

```
Type positive integer: 6
1 0
2 0
2 1
3 0
3 1
3 2
4 0
4 1
4 2
4 3
5 0
5 1
5 2
5 3
5 4
15
```

```
Type positive integer: 7
1 0
2 0
2 1
3 0
3 1
3 2
4 0
4 1
4 2
4 3
5 0
5 1
5 2
5 3
5 4
6 0
6 1
6 2
6 3
6 4
6 5
21
```

Since the code executes the loop of for, it executes $1 + 2 + \cdots + (n-1) = \frac{n(n-1)}{2}$ times.

Thus, this code's time complexity is $O(n^2)$.

7 Space Complexity

One primitive variable → 1~8 bytes(char, int, double, …) → $O(1)$

An array of n variables → $O(n)$ (remark)
Note that char : 1 byte, short : 2 byte, int : 4 byte,
Double : 8 byte, long : 8 byte

We will focus on integer data with array(or list in python).

8 Example

1

```c
#include <stdio.h>

int main(){
    int N;
    scanf("%d", &N);
    int array[N];
    for (int i = 0 ; i < N; i++){
        array[i] = i*(i+1);
    }
    for (int j = 0; j < N; j++){
        printf("%d ", array[j]);
    }
    printf("\n");
    return 0;
}

```

2

```
20
0 2 6 12 20 30 42 56 72 90 110 132 156 182 210 240 272 306 342 380
```

We want to find the space complexity of this code.
Note that this code has an integer array with the size of N given number by user.
The result above says that $N = 20$.
Since integer is 4 byte in memory, this code uses $4 \times 20 = 80$ bytes. In general, $4 \times N$ bytes.
Thus, this code's space complexity is $O(N)$.

Q.What is the time complexity of this code?

A. $O(N)$

Why?

- Considering the time complexity of code is highly related to measure the efficiency of code and the execution time of it.
- Computer has a limit of execution, about $10^6 \sim 15 \times 10^9$ per a sec.
- Considering the space complexity of code is highly related to the memory system.
- To avoid any inefficient usage of memory and reduce the time of execution, both time complexity and space complexity count.

Basic Statistics

- $E(X)$: mean, $Var(X)/\sigma^2$: Variance, S^2: Sample Variance
- $\frac{n-1}{n}S^2 = \sigma^2, n$: Sample Size
- 복원추출(sampling with replacement): ex) 주머니에서 숫자 1부터 5 까지 있을 때, 복원 추출하여 5번 뽑는 경우: $3, 4, 3, 1, 5$ / $1, 1, 1, 1, 1$
- independent and identically distributed: 같은 분포에서 독립적으로 수 추출(추출하는 과정에서 서로 연관 주지 않음)
- $\hat{\theta}$: sample estimator, θ: estimator, true parameter, $\bar{\theta}$: sample mean
- $E(X_1 + X_2) = E(X_1) + E(X_2), E(X_1 - X_2) = E(X_1) - E(X_2)$

Monte Carlo Method

- Monte Carlo Method is a class of computational algorithm that rely on repeated random sampling to compute results.
- Monte Carlo methods, or Monte Carlo experiments, are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results.
- Developed in the 1940’s by Neumann, Ulam, and Metropolis, while they were working on Manhattan Project in the Lost Alamos National Laboratory.
- Named from the Monte Carlo casino.
- Also called parametric bootstrap.

Monte Carlo Method

- MC method for estimation in Simulation situation.
- We know every information from population even the true parameter of distribution.
- The purpose of this method is to compare the several estimators to find the best efficiently estimator for true parameter.
- We consider the making sample with repeating a lot from the population with known distribution.
- We will focus how to estimate the true parameter.

Monte Carlo Method

Algorithm: Estimation of Standard Error (Standard Deviation) of $\hat{\theta}$.

1. For $j = 1, \dots, m$ and the distribution of population $f(x)$,

(1) Generate a sample $\boldsymbol{x}^{(j)} = \left(x_1^{(j)}, \dots, x_n^{(j)}\right) \sim f(x)$.

(2) Compute $\hat{\theta}^{(j)}$

2. $\widehat{SE}\left(\hat{\theta}\right) = \sqrt{\frac{\sum_{j=1}^{m}(\hat{\theta}^{(j)} - \bar{\hat{\theta}})^2}{m-1}}$, where $\bar{\hat{\theta}} = \frac{1}{m}\sum_{j=1}^{m}\hat{\theta}^{(j)}$.

$\Rightarrow$ Standard Deviation of $\left(\hat{\theta}^{(1)}, \dots, \hat{\theta}^{(m)}\right)$

Example

Suppose that X_1, X_2 have independent and identically distributed from $N(0,1)$.

We want to estimate $\theta = E(|X_1 - X_2|)$.

Algorithm

1. For $j = 1, \ldots, m$,

(1) Generate a sample $\left(x_1^{(j)}, x_2^{(j)}\right) \sim N(0,1)$

(2) Compute $\hat{\theta}^{(j)} = \left|x_1^{(j)} - x_2^{(j)}\right|$.

2. Compute $\bar{\hat{\theta}}$ and $\widehat{SE}(\hat{\theta})$.

Example


```r
MCtheta.R = function(m){
  theta = numeric(m) # same as theta = [] with m elements
  for (i in 1:m){
    x = rnorm(2,0,1) #random 2 numbers
    theta[i] = abs(x[1]-x[2])
  }
  mean_theta = mean(theta)
  s.e.theta = sqrt(sum(theta-mean_theta)^2/(m-1))
  return(s.e.theta)
}
MCtheta.R(1000)
#Theoretical value = 0
```

```
> MCtheta.R(1000)
[1] 5.549893e-16
> MCtheta.R(1000)
[1] 2.058378e-15
> MCtheta.R(1000)
[1] 3.020828e-16
> MCtheta.R(1000)
[1] 2.409637e-15
> MCtheta.R(1000)
[1] 7.20081e-16
```

```
> MCtheta.R(5000)
[1] 1.38339e-15
> MCtheta.R(5000)
[1] 5.682733e-15
> MCtheta.R(5000)
[1] 5.004385e-15
> MCtheta.R(5000)
[1] 4.509757e-15
> MCtheta.R(5000)
[1] 1.713142e-15
```

```
> MCtheta.R = function(m){
+   theta = numeric(m) # same as theta = [] with m elements
+   for (i in 1:m){
+     x = rnorm(2,0,1) #random 2 numbers
+     theta[i] = abs(x[1]-x[2])
+   }
+   mean_theta = mean(theta)
+   s.e.theta = sqrt(sum(theta-mean_theta)^2/(m-1))
+   return(s.e.theta)
+ }
> MCtheta.R(1000)
[1] 1.531489e-15
```

```
> MCtheta.R(10000)
[1] 5.988842e-15
> MCtheta.R(10000)
[1] 9.054321e-15
> MCtheta.R(10000)
[1] 6.724957e-15
> MCtheta.R(10000)
[1] 9.892582e-15
> MCtheta.R(10000)
[1] 8.017321e-15
> #Theoretical value = 0
```

Time Complexity: $O(m)$

Space Complexity: $O(m)$

Monte Carlo Method

- MC method for estimation in Data analysis situation.
- We only do not know the one parameter from population.
- The purpose of this method is to estimate the unknown parameter with assumption for distribution.
- We consider the making sample with repeating a lot from the pseudo-distribution by replacing unknown parameter with sample parameter.
- We will focus how to estimate the unknown parameter.

Example

Suppose that $X_1, \dots, X_{300}$ have independent and identically distributed from $N(73, \sigma^2)$.

We want to estimate $\theta = \sigma^2$ by using $\widehat{Var}(\hat{\theta})$.

Algorithm

1. For $j = 1, \dots, m$,

(1) Generate a sample $\left(x_1^{(j)}, \dots, x_{300}^{(j)}\right) \sim N(73, S^2)$, assume $S^2 = 285$

(2) Compute $\hat{\theta}^{(j)} = \frac{1}{300}\sum_{i=1}^{300}(x_i^{(j)})^2 - (\frac{1}{300}\sum_{i=1}^{300} x_i^{(j)})^2$.

2. Compute $\widehat{Var}(\hat{\theta})$.

Example


```r
MCvar.R = function(n,m){
  theta = numeric(m) #same as theta = [] with m elements
  for (i in 1:m){
    x = rnorm(n,73,sqrt(285)) #replace σ^2 with sample variance = 285
    theta[i] = var(x) * (n-1)/n
  }
  var_theta = var(theta)
  return(var_theta)
}
MCvar.R(300,10000)
```

```
> MCvar.R(300,10000)
[1] 542.7698
> MCvar.R(300,10000)
[1] 528.5641
> MCvar.R(300,10000)
[1] 533.8688
> MCvar.R(300,10000)
[1] 532.6673
> MCvar.R(300,10000)
[1] 545.2126
```

Consider when running this code.

Q1. What are the time complexity and space complexity of this code?

A1. Time complexity: $O(mn)$, Space complexity: $O(m)$.

Q2. What is the problem of this code?

A2. It may buffer to print the result.

From this result, we can say that the population variance would be 535, that is, standard deviation is 23.13.

Bootstrap Method

- Bootstrap is a class of nonparametric Monte Carlo Methods that estimate the distribution of estimators by resampling, that is, sampling with replacement.
- Introduced by Efron in 1979.
- Consider an observed sample as a finite population and random samples are generated (resampled) from the observed sample.
- Often used when the distribution of target population is not specified (i.e., only information is sample).

Bootstrap Method

- Bootstrap is impossible to have a simulation situation but data analysis situation.
- Data analysis situation means that we do not know the population distribution.
- Also Bootstrap cannot assume the distribution of population.
- We only consider “resampling” with producing pseudo-distribution by generated samples.

Example

We have 5 observed data from population as follows:

$$0, 95, 30, 70, 80$$

We want to estimate $\theta = Var(X)$ by $\widehat{Var}(\hat{\theta})$.

min: 0, max: 95, mean = 70, otherwise.

Algorithm

1. For $j = 1, \dots, m$,

(1)Generate a sample $\boldsymbol{x}^{(j)} = \left(x_1^{(j)}, \dots, x_5^{(j)}\right)$ by resampling.

(2)Compute $\hat{\theta}^{(j)} = \frac{1}{5}\sum_{i=1}^{5}(x_i^{(j)})^2 - (\frac{1}{5}\sum_{i=1}^{5} x_i^{(j)})^2$.

2. Compute $\widehat{Var}(\hat{\theta})$.

Example


```r
Bootstrap.R = function(m){
  theta = numeric(m)
  x = c(0,95,30,70,80)
  for (j in 1:m){
    i = sample(1:5, 5, replace = TRUE) #resampling data from x
    theta[j] = var(x[i]) * 0.8
  }
  var.theta = var(theta)
  return(var.theta)
}
Bootstrap.R(10000)
```

```
> Bootstrap.R(10000)
[1] 225767.1
> Bootstrap.R(10000)
[1] 227704.4
> Bootstrap.R(10000)
[1] 224909.4
> Bootstrap.R(10000)
[1] 228757.3
> Bootstrap.R(10000)
[1] 230728
```

Consider when running this code.

Q1. What are the time complexity and space complexity of this code?

A1. Both are $O(m)$.

Q2. Is this code fine?

A2. No, because the estimation of variance is too large, it is not good.

Example

We have 7 observed data from population as follows:

$$70,45,35,60,85,55,55$$

We want to estimate $\theta = Var(X)$ by $\widehat{Var}(\hat{\theta})$.

max: 85, mean = 70, my score = 55, otherwise.

Algorithm

1. For $j = 1, \dots, m,$

(1)Generate a sample $\boldsymbol{x}^{(j)} = \left(x_1^{(j)}, \dots, x_7^{(j)}\right)$ by resampling.

(2)Compute $\hat{\theta}^{(j)} = \frac{1}{7}\sum_{i=1}^{7}(x_i^{(j)})^2 - (\frac{1}{7}\sum_{i=1}^{7} x_i^{(j)})^2$.

2. Compute $\widehat{Var}(\hat{\theta})$.

Example


```r
Bootstrap_var.R = function(m){
  theta = numeric(m)
  x = c(70,45,35,60,85,55,55)
  for (j in 1:m){
    i = sample(1:7, 7, replace = TRUE) #resampling data from x
    theta[j] = var(x[i]) * 6/7
  }
  var_theta = var(theta)
  return(var_theta)
}
est_var = Bootstrap_var.R(10000); est_var
sqrt(est_var)
```

```
> est_var = Bootstrap_var.R(10000); est_var
[1] 9270.191
> sqrt(est_var)
[1] 96.28183
> est_var = Bootstrap_var.R(10000); est_var
[1] 9318.102
> sqrt(est_var)
[1] 96.53032
> est_var = Bootstrap_var.R(10000); est_var
[1] 9265.349
> sqrt(est_var)
[1] 96.25668
> est_var = Bootstrap_var.R(10000); est_var
[1] 9154.518
> sqrt(est_var)
[1] 95.67925
> est_var = Bootstrap_var.R(10000); est_var
[1] 9244.814
> sqrt(est_var)
[1] 96.14996
```

Consider when running this code.

Q1. What are the time complexity and space complexity of this code?

A1. Both are $O(m)$.

Q2. Is this code fine?

A2. It still produces the large variance, but better.

Conclusion

- So, we learned about time complexity and space complexity for efficiency of codes.
- Monte Carlo Method is useful with estimators for population. (Evaluation for estimators is hard to cover!)
- Bootstrap Method is also useful without any assumptions for populations.

Thank you