CHEME 5760 Final Project

NAME: Peihong Liu

NETID: PL542

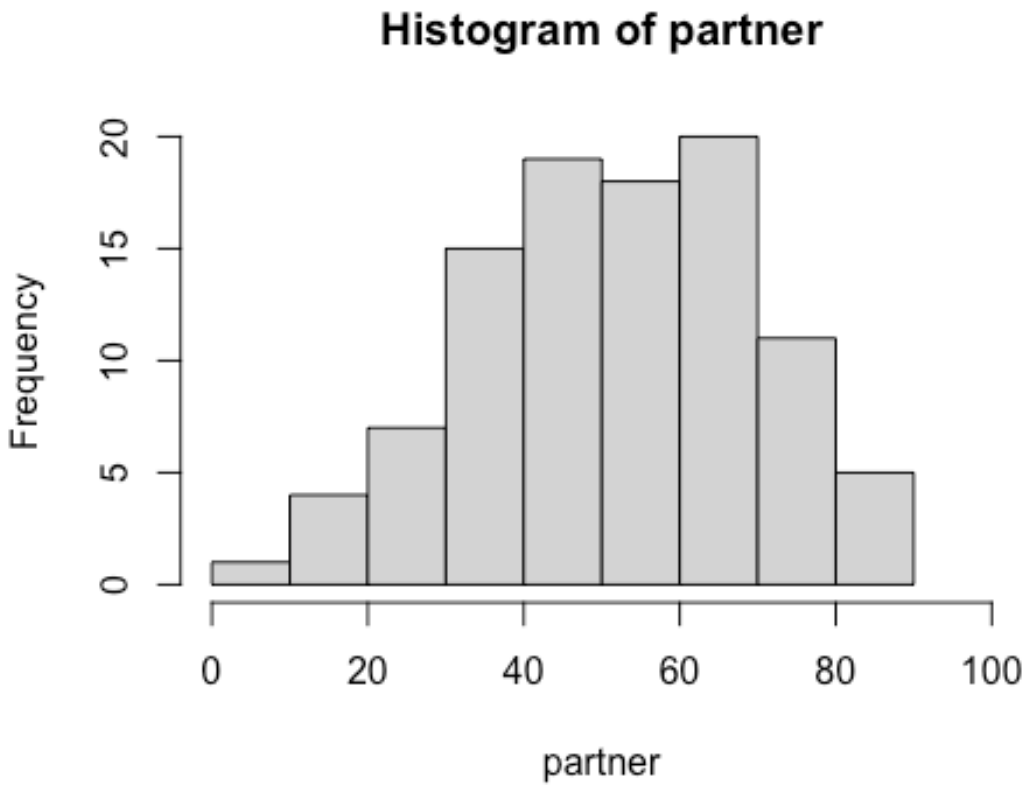
DUE DATE: December 12, 2023 by 11:59pm

Abstract

Part 1: First is not always best

1. Generate the partner sample

```
partner = round(rnorm(100, 50, 15))
hist(partner, breaks = 10, xlim = c(0,100))
```



2. When is the best time to start comparing? lemma 1: first is not always the best

```
opt = rep(0, 10000)

for (i in 1:10000) {
 partner = round(rnorm(100, 50, 15))
 first = partner[1]
 best = max(partner)
  opt[i] = first >= best
}
prob = sum(opt)/10000
prob

## [1] 0.0096
```

As this example shows, the chance of your first love being your best partner is around 1%

Lemma 2: Likewise, committing to the last date you would meet is also extremely sub-optimal.

Similarily, we can argue that the chance of the last person that you would date in your life being the most optimal candidate would be around 1%

Lemma 3: Comparing with first love only is not an optimal strategy.

```
opt = rep(0, 10000)

for (i in 1:10000) {
 partner = round(rnorm(100, 50, 15))
 first = partner[1]
 bingo =  Position(function(x) x>first, partner, nomatch = 100)
 choice = partner[bingo]
 best = max(partner)
  opt[i] = choice >= best
}
prob = sum(opt)/10000
prob

## [1] 0.0591
```

As the simulation result shows, using the one's first love as a calibration of best fit partner increased one's probability of finding the "best" partner to 5%, still a very low probability. ## Part 2: Theorem 1: The optimal strategy is start comparing after dating the 37th partner and commit to the next one that is better than all previous ones you have dated before.

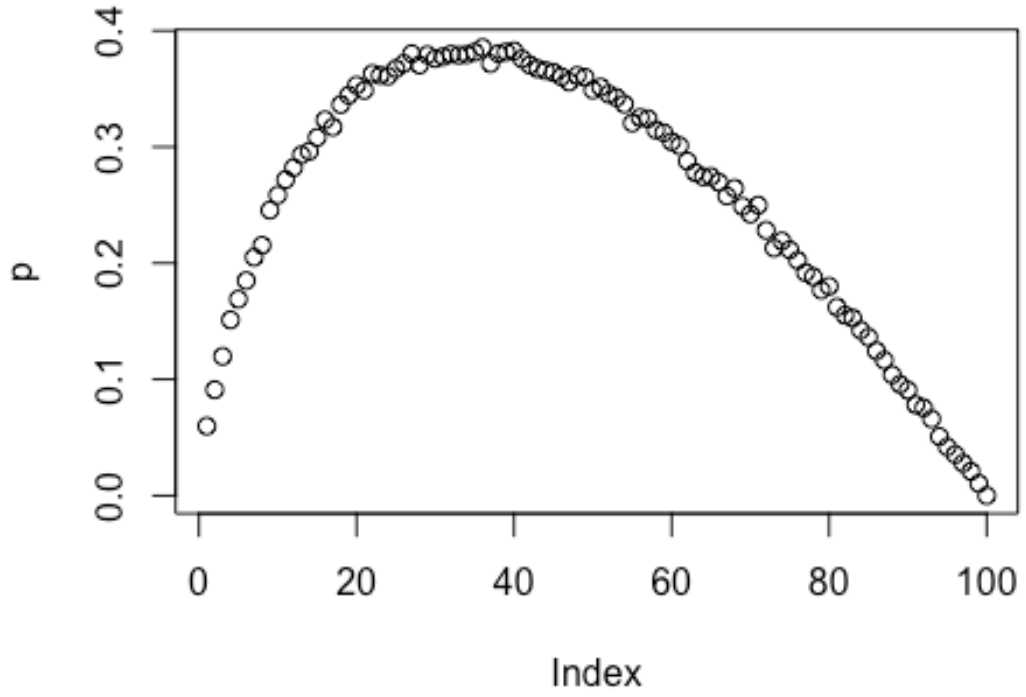
We will prove this using simulation and a plot.

```
opt = rep(0, 10000)
prob = function(n){
  m=n+1
for (i in 1:10000) {
 partner = round(rnorm(100, 50, 15))
 pool = partner[1:n]
 bingo =  Position(function(x) x>max(pool), partner[m:100], nomatch = 100)
if(bingo<100){
  choice = partner[bingo+n]
}else{
  choice = partner[100]
}

 best = max(partner)
  opt[i] = choice >= best
}
sum(opt)/10000
}

p = rep(0, 100)
for (i in 1:99){
  p[i]= prob(i)
}

plot(p)
```



As our simulation show, when following our strategy in theorem, the probability of finding the "best" partner is highest.

Implications?

Suppose we assume an average male starts having the intention to "date" at the age of 13, and the frequency and distribution of meeting potential candidates are constant until he reaches the age 60, then the optimal age to commit to marriage is around 30 years old. Similarily, for women, the optimal age to commit to marriage would be around 25 years old.

Part 3: A more diverse "Utility Function" of partner characteristics

Without Loss of Generality, assume when a person looks for a female partner, the person considers the following four categories: Height Weight, IQ, and EQ.

According to the CDC National Health Statistics Report:

Female mean weight is 77.3 kg with standard deviation of 21 kg

Female mean height is 161.5 cm with standard deviation 15.6 centimeters

IQ and EQ are by definition with mean = 100 and standard deviation = 15

Assume the person gives a score to each category that represents the person's preference, and that the weight or each category are identical, then the final score of candidate is the average of 4 numbers.

Scoring method is as follows:

Score for height: 100 goes to the partner who is 15cm shorter than you, for every 1 cm deviation, the score drops by 5

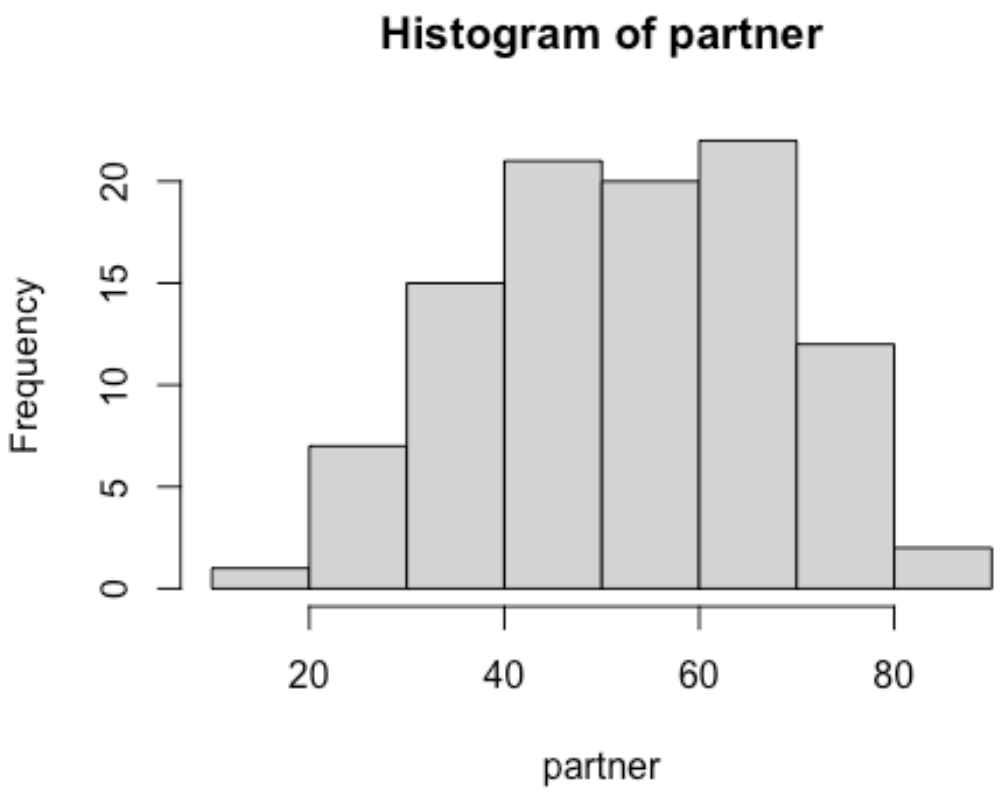
Score for weight: 100 goes to the partner whose weight = 60， for every 1 kg deviation, score drops by 2

Score for IQ: Percentile of IQ

Score for EQ: Percentile of EQ

```
score = rep(0,100)
height = rnorm(100, 161.5, 6.9)
weight = rnorm(100, 77.3, 21.2)
IQ = rnorm(100,100,15)
EQ = rnorm(100,100,15)

for (i in 1:100) {
score[i] = (100-5*abs(height[i]-166)+(100-2*abs(weight[i] - 60))+pnorm(IQ[i],
100, 15)*100+100*pnorm(EQ[i], 100, 15))/4
}
partner = (round(score))
hist(partner)
```



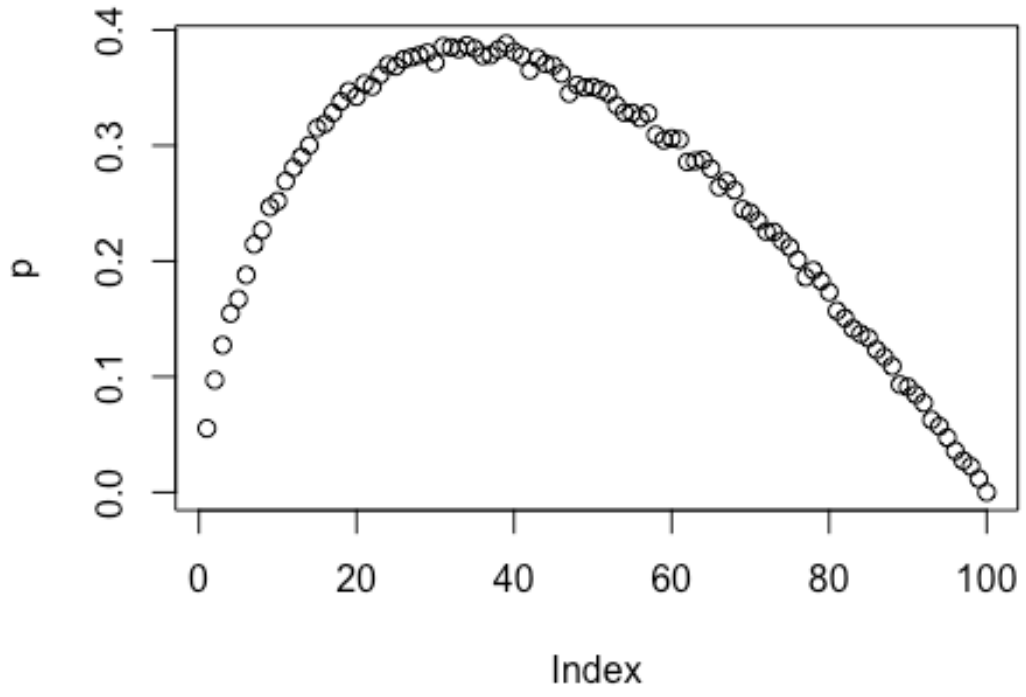
If we repeat this process for 10000 times

```
opt = rep(0, 10000)
prob = function(n){
  m=n+1
for (i in 1:10000) {
 partner = round((100-5*abs(rnorm(100, 161.5, 6.9)-181)+(100-2*abs(rnorm(100,
77.3, 21.2) - 60))+pnorm(rnorm(100,100,15), 100,
15)*100+100*pnorm(rnorm(100,100,15), 100, 15))/4)
 partner = pmax(partner, 0)
 pool = partner[1:n]
 bingo =  Position(function(x) x>max(pool), partner[m:100], nomatch = 100)
if(bingo<100){
  choice = partner[bingo+n]
}else{
  choice = partner[100]
}

 best = max(partner)
  opt[i] = choice >= best
}
sum(opt)/10000
}
```

```
p = rep(0, 100)
for (i in 1:99){
  p[i]= prob(i)
}

plot(p)
```



The result plot shows that even with a more complicated scoring system, 37% remains to be the magical number.