Directional relationship

Summary:

At the beginning I generate a population similar to the one on the paper with the size of the population equals to 100 and their ages have a normal distribution and their gender have a Bernoulli distribution. And I also generated an outcome for each pair of individuals using outcome = beta0 + beta1 * gender1 + beta2 * gender2 + beta3 * age1 + beta4 * age2 + beta5 * $link_{ij}$. $Link_{ij}$ denotes whether there is a link between individual i and j or not. And this link_ij was decided by whether the difference between the benefit from this link and the cost from the link is great than zero or not.

In the paper, it gives a way to calculate variance (5 by 5 matrix) and I calculated confidence interval based on the diagonal variances and ignored the covariance. And also sometimes the method presents in the paper gives negative variances. I also used the standard way to calculate confidence intervals, and compute the coverage for each method. Then I draw the graph for confidence interval computed in both ways.

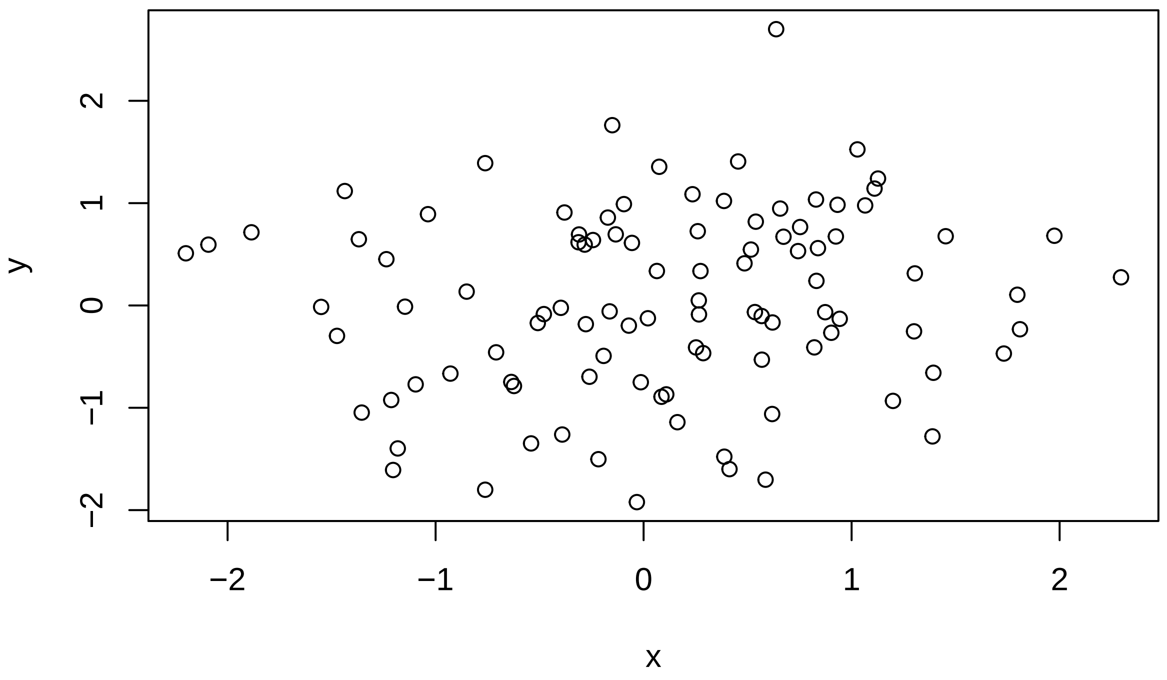
The coverage for regular beta1 to beta5 is 47/50, 48/50, 45/50, 42/50, 48/50. And the coverage for adjusted beta1 to beta5 is 45/50, 43/50, 38/50, 41/50, 48/50. It turns out that the fraction of times the true value of these parameters are contained in the estimate interval calculated by standard confidence interval is pretty high, most of the time the fraction will be from 42/50 to 48/50, the fraction of times the true value of these parameters are contained in the estimate interval calculated by the given method in the paper is really high as well, most of the time the fraction will be from 41/50 to 48/50, but sometimes can be as low as 38/50.

Simmulate a social network

```r
rm(list = ls())
setwd('/Users/yuezhou/Desktop/TYLER')
library(car)
```

Simulate a position in a two-dimentional space

```r
n <- 100
x <- rnorm(n, mean = 0, sd = 1)
y <- rnorm(n, mean= 0, sd = 1)
plot(x, y)
```



Compute the n*n matrix of Euclidean distances between individuals

```
A <- cbind(x, y)
distance <- dist(A)
distance <- as.matrix(distance)
```

Create symmetric matrixs A, B and C as constant that times distance to simulate benefit and cost from a link

```
a_0 <- 10
b_0 <- 5
c_0 <- 5

a <- rnorm(n*n, mean = a_0, sd = 1)
A <- matrix(a, nrow = n)
for(i in 1:n){
  for(j in 1:n){
    A[j,i] <- A[i,j]
  }
}

b <- rnorm(n*n, mean = b_0, sd = 1)
B <- matrix(b, nrow = n)
for(i in 1:n){
  for(j in 1:n){
    B[j,i] <- B[i,j]
  }
}

c <- rnorm(n*n, mean = c_0, sd = 1)
C <- matrix(c, nrow = n)
for(i in 1:n){
  for(j in 1:n){
    C[j,i] <- C[i,j]
  }
}

B_1 <- A * distance
B_0 <- B * distance
cost <- C * distance
benefit <- B_1 - B_0
relation <- benefit - cost
```

Creat a matric L represent the relationship between pairs of individuals. 0 means there is no relationship between individuals i and j; 1 means there is a relationship beteen individuals i and j.

```
L <- matrix(nrow = n, ncol = n)
for(i in 1:n){
  for(j in 1:n){
    if(relation[i,j] > 0){
      L[i,j] <- 1
    }else{
      L[i,j] <- 0
    }
  }
}
```

Directional relationship means when dyadic relationship is directional, $Y_{ij} = Y_{ji}$. Use this relationship to construct residuals. Then simulate an outcome Y for each pair of individuals. Write a forloop that runs the simulation trials times and get different confidence interval.

```
beta0 <- 10
beta1 <- 2
beta2 <- 3
beta3 <- 4
beta4 <- 5
beta5 <- 6

trials <- 50

countbeta1 <- 0
countbeta2 <- 0
countbeta3 <- 0
countbeta4 <- 0
countbeta5 <- 0
countbeta_1 <- 0
countbeta_2 <- 0
countbeta_3 <- 0
countbeta_4 <- 0
countbeta_5 <- 0

confidence_1 <- matrix(NA, nrow = trials, ncol = 4)
```

```r
confidence_2 <- matrix(NA, nrow = trials, ncol = 4)
confidence_3 <- matrix(NA, nrow = trials, ncol = 4)
confidence_4 <- matrix(NA, nrow = trials, ncol = 4)
confidence_5 <- matrix(NA, nrow = trials, ncol = 4)
```

Generate matrix age_1 that has n rows with each row repeats the same age and age_2 that has n columns with each column repeats the same age. Then generate matrix $gender_1$ that has n rows with each row repeats the same gender and $gender_2$ that has n columns with each column repeats the same gender, so that when generating the outcome. The outcome with each pair of individual can have a corresponding age, gender and link.

```r
for(h in 1:trials){
  age1 <- runif(n, min = 0, max =100)
  repage1 <- rep(age1, time = n)
  age_1 <- matrix(repage1, nrow = n)
  age_2 <- matrix(repage1, nrow = n, byrow = T)

  gender1 <- rbinom(n, 1, 1/2)
  repgender1 <- rep(gender1, time = n)
  gender_1 <- matrix(repgender1, nrow = n)
  gender_2 <- matrix(repgender1, nrow = n, byrow = T)

  fittedoutcome <- beta0 + beta1*gender_1 + beta2*gender_2 + beta3*age_1+ beta4*age_2 + beta5*L

  rep_gender2 <- rep(gender1, each = n)
  gender.2 <- matrix(rep_gender2, ncol = 1)

  rep_gender1 <- rep(gender1, time = n)
  gender.1 <- matrix(rep_gender1, ncol = 1)

  rep_age2 <- rep(age1, each = n)
  age.2 <- matrix(rep_age2, ncol = 1 )

  rep_age1 <- rep(age1, time = n)
  age.1 <- matrix(rep_age1, ncol = 1 )

  link_ij <- matrix(L, ncol = 1, byrow = T)

  X <- cbind(rep_gender1, rep_gender2, rep_age1, rep_age2, link_ij)

  someoutcome1 <- rnorm(n*n, mean = fittedoutcome, sd = 20)
  trueoutcome1 <- matrix(someoutcome1, nrow = n)
  fitted_outcome <- matrix(fittedoutcome, ncol = 1, byrow = T)
  reg <- lm(someoutcome1 ~ gender.1 + gender.2 + age.1 + age.2 + link_ij)
  reg_1 <- matrix(fitted(reg), ncol = n)
  u1 <- trueoutcome1 - reg_1

  compareCoefs(reg)
  coef <- matrix(compareCoefs(reg))
  X_prime <- t(X)
  Y <- solve(X_prime%*%X)
```

```r
m_ijkl <- 0
suminside <- 0
N <- n*n
for (i in 1:n) {
  for (j in 1:n) {
    for (k in 1:n) {
      for (l in 1:n) {
        if (i == k || j == l || i == l || j == k) {
          m_ijkl <- 1
          X_ij <- X[(i-1) * n + j, ]
          u_ij <- u1[i,j]
          u_kl <- u1[k,l]
          X_kl <- X[(k-1) * n+l, ]
          inside <- m_ijkl *u_ij * u_kl * X_ij  %*% t(X_kl)
          suminside <- inside + suminside
        }
      }
    }
  }
}
suminside1 <- as.numeric(suminside)

var <- Y %*% suminside %*% Y

conf_gender1l <- coef[2,1] - 1.96 * var[1,1]^0.5
conf_gender1u <- coef[2,1] + 1.96 * var[1,1]^0.5
c(conf_gender1l, conf_gender1u)

conf_gender2l <- coef[3,1] - 1.96 * var[2,2]^0.5
conf_gender2u <- coef[3,1] + 1.96 * var[2,2]^0.5
c(conf_gender2l, conf_gender2u)

conf_age1l <- coef[4,1] - 1.96 * var[3,3]^0.5
conf_age1u <- coef[4,1] + 1.96 * var[3,3]^0.5
c(conf_age1l, conf_age1u)

conf_age2l <- coef[5,1] - 1.96 * var[4,4]^0.5
conf_age2u <- coef[5,1] + 1.96 * var[4,4]^0.5
c(conf_age2l, conf_age2u)

conf_linkl <- coef[6,1] - 1.96 * var[5,5]^0.5
conf_linku <- coef[6,1] + 1.96 * var[5,5]^0.5
c(conf_linkl, conf_linku)

a_0 <- confint(reg, "(Intercept)")
a0 <- matrix(a_0)

a_1 <-confint(reg, "gender.1")
a1 <- matrix(a_1)

a_2 <-confint(reg, "gender.2")
a2 <- matrix(a_2)
```

```
  a_3 <- confint(reg, "age.1")
  a3 <- matrix(a_3)

  a_4 <- confint(reg, "age.2")
  a4 <- matrix(a_4)

  a_5 <-confint(reg, "link_ij")
  a5 <- matrix(a_5)

  a_a <- cbind(a0, a1, a2, a3, a4, a5)

  confidence_1[h,1] <- conf_gender1l
  confidence_1[h,2] <- conf_gender1u
  confidence_1[h,3] <- a1[1,1]
  confidence_1[h,4] <- a1[2,1]

  confidence_2[h,1] <- conf_gender2l
  confidence_2[h,2] <- conf_gender2u
  confidence_2[h,3] <- a2[1,1]
  confidence_2[h,4] <- a2[2,1]

  confidence_3[h,1] <- conf_age1l
  confidence_3[h,2] <- conf_age1u
  confidence_3[h,3] <- a3[1,1]
  confidence_3[h,4] <- a3[2,1]

  confidence_4[h,1] <- conf_age2l
  confidence_4[h,2] <- conf_age2u
  confidence_4[h,3] <- a4[1,1]
  confidence_4[h,4] <- a4[2,1]

  confidence_5[h,1] <- conf_linkl
  confidence_5[h,2] <- conf_linku
  confidence_5[h,3] <- a5[1,1]
  confidence_5[h,4] <- a5[2,1]
}
```

First plot two kinds of confidence intervals for these betas using different method on the same graph.

Confidence intervals on the left are generated using the method presents in the paper which is appropriately large.

Confidence intervals on the right are generated using the standard variance which is appropriately large and most of them cover the true value of beta1. And when using the method on the paper, it sometimes gives negative variances.

The plots of the following codes are also in Github (file name: figure__k.pdf, figure__k__1.pdf, and figure__k__2.pdf for beta__k) z <- 3 * beta1 plot(NA, xlim = c(0, z), ylim = c(0, trials), xlab = "beta1 paper vs confint", ylab = "trials") abline(v = beta1, col = "red") abline(v = beta1 + beta1, col = "red") n__1 <- 0 for(h in 1:trials){ segments(x0 = confidence__1[h,1], y0 = n__1, x1 = confidence__1[h,2], y1 = n__1) segments(x0 = confidence__1[h,3] + beta1, y0 = n__1, x1 = confidence__1[h,4] + beta1, y1 = n__1) if (beta1 < confidence_1[h,2] && beta1 > confidence_1[h,1]) { countbeta1 <- countbeta1 + 1 } if (beta1 < confidence_1[h,4] && beta1 > confidence_1[h,3]) { countbeta_1 <- countbeta_1 + 1 } n_1 <- n_1 + 1 }

plot(NA, xlim = c(min(min(confidence_1[,1]), beta1),max(max(confidence_1[,2]), beta1)), ylim = c(0,trials), xlab = "confidence interval for beta1 paper",ylab = "trials") abline(v = beta1, col = "red") n_2 <- 0 for(h in 1:trials){ segments(x0 = confidence_1[h,1], y0 = n_2, x1 = confidence_1[h,2], y1 = n_2) n_2 <- n_2 + 1 }

plot(NA, xlim = c(min(min(confidence_1[,3]),beta1), max(max(confidence_1[,4]),beta1)), ylim = c(0,trials), xlab = "confidence interval for beta1 confint",ylab = "trails") abline(v = beta1, col = "red") n_3 <- 0 for(h in 1:trials){ segments(x0 = confidence_1[h,3], y0 = n_3, x1 = confidence_1[h,4], y1 = n_3) n_3 <- n_3 + 1 } "‘

Beta2

```
z <- 3 * beta2
plot(NA, xlim = c(0, z),
     ylim = c(0, trials), xlab = "beta 2 paper vs confint", ylab = "trials")
abline(v = beta2, col = "red")
abline(v = beta2 + beta2, col = "red")
n_1 <- 0
for(h in 1:trials){
  segments(x0 = confidence_2[h,1], y0 = n_1, x1 = confidence_2[h,2], y1 = n_1)
  segments(x0 = confidence_2[h,3] + beta2, y0 = n_1, x1 = confidence_2[h,4] + beta2, y1 = n_1)
  if (beta2 < confidence_2[h,2] && beta2 > confidence_2[h,1]) {
    countbeta2 <- countbeta2 + 1
  }
  if (beta2 < confidence_2[h,4] && beta2 > confidence_2[h,3]) {
    countbeta_2 <- countbeta_2 + 1
  }
  n_1 <- n_1 + 1
}

plot(NA, xlim = c(min(min(confidence_2[,1]), beta2),max(max(confidence_2[,2]),beta2)),
     ylim = c(0,trials), xlab = "confidence interval for beta2 paper",ylab = "trails")
abline(v = beta2, col = "red")
n_2 <- 0
for(h in 1:trials){
  segments(x0 = confidence_2[h,1], y0 = n_2, x1 = confidence_2[h,2], y1 = n_2)
  n_2 <- n_2 + 1
}

plot(NA, xlim = c(c(min(confidence_2[,3]),max(confidence_2[,4]))), ylim = c(0,trials), xlab = "confiden
     ylab = "trails")
abline(v = beta2,col = "red")
n_3 <- 0
for(h in 1:trials){
  segments(x0 = confidence_2[h,3], y0 = n_3, x1 = confidence_2[h,4], y1 = n_3)
  n_3 <- n_3 + 1
}
```

Beta3

```r
z <- 3 * beta3
plot(NA, xlim = c(0, z),
     ylim = c(0, trials), xlab = "beta3 paper vs confint", ylab = "trials")
abline(v = beta3, col = "red")
abline(v = beta3 + beta3, col = "red")
n_1 <- 0
for(h in 1:trials){
  segments(x0 = confidence_3[h,1], y0 = n_1, x1 = confidence_3[h,2], y1 = n_1)
  segments(x0 = confidence_3[h,3] + beta3, y0 = n_1, x1 = confidence_3[h,4] + beta3, y1 = n_1)
  if (beta3 < confidence_3[h,2] && beta3 > confidence_3[h,1]) {
    countbeta3 <- countbeta3 + 1
  }
  if (beta3 < confidence_3[h,4] && beta3 > confidence_3[h,3]) {
    countbeta_3 <- countbeta_3 + 1
  }
  n_1 <- n_1 + 1
}

plot(NA, xlim = c(min(min(confidence_3[,3]), beta3),max(max(confidence_3[,4]),beta3)),
     ylim = c(0,trials), xlab = "confidence interval for beta3 paper",ylab = "trails")
abline(v = beta3, col = "red")
n_2 <- 0
for(h in 1:trials){
  segments(x0 = confidence_3[h,1], y0 = n_2, x1 = confidence_3[h,2], y1 = n_2)
  n_2 <- n_2 + 1
}

plot(NA, xlim = c(min(min(confidence_3[,3]), beta3),max(max(confidence_3[,4]),beta3)),
     ylim = c(0,trials), xlab = "confidence interval for beta2 confint",ylab = "trails")
abline(v = beta3, col = "red")
n_3 <- 0
for(h in 1:trials){
  segments(x0 = confidence_3[h,3], y0 = n_3, x1 = confidence_3[h,4], y1 = n_3)
  n_3 <- n_3 + 1
}
```

Beta4

```r
z <- 3 * beta4
plot(NA, xlim = c(0,z),
     ylim = c(0, trials), xlab = "beta4 paper vs confint", ylab = "trials")
abline(v =beta4, col = "red")
abline(v = beta4 + beta4, col = "red")
n_1 <- 0
for(h in 1:trials){
  segments(x0 = confidence_4[h,1], y0 = n_1, x1 = confidence_4[h,2], y1 = n_1)
  segments(x0 = confidence_4[h,3] + beta4, y0 = n_1, x1 = confidence_4[h,4] + beta4, y1 = n_1)
  if (beta4 < confidence_4[h,2] && beta4 > confidence_4[h,1]) {
    countbeta4 <- countbeta4 + 1
  }
  if (beta4 < confidence_4[h,4] && beta4 > confidence_4[h,3]) {
    countbeta_4 <- countbeta_4 + 1
  }
  n_1 <- n_1 + 1
}

plot(NA, xlim = c(min(min(confidence_4[,1]), beta4),max(max(confidence_4[,2]),beta4)),
     ylim = c(0,trials), xlab = "confidence interval for beta4 paper",ylab = "trails")
abline(v = beta4, col = "red")
n_2 <- 0
for(h in 1:trials){
  segments(x0 = confidence_4[h,1], y0 = n_2, x1 = confidence_4[h,2], y1 = n_2)
  n_2 <- n_2 + 1
}

plot(NA, xlim = c(min(min(confidence_4[,3]), beta4),max(max(confidence_4[,4]), beta4)),
     ylim = c(0,trials), xlab = "confidence interval for beta4 confint",ylab = "trails")
abline(v = beta4,col = "red")
n_3 <- 0
for(h in 1:trials){
  segments(x0 = confidence_4[h,3], y0 = n_3, x1 = confidence_4[h,4], y1 = n_3)
  n_3 <- n_3 + 1
}
```

Beta5

```
z <- 3 * beta5
plot(NA, xlim = c(0, z),
     ylim = c(0, trials), xlab = "beta5 paper vs confint", ylab = "trials")
abline(v = beta5, col = "red")
abline(v = beta5 + beta5, col = "red")
n_1 <- 0
for(h in 1:trials){
  segments(x0 = confidence_5[h,1], y0 = n_1, x1 = confidence_5[h,2], y1 = n_1)
  segments(x0 = confidence_5[h,3] + beta5, y0 = n_1, x1 = confidence_5[h,4] + beta5, y1 = n_1)
  if (beta5 < confidence_5[h,2] && beta5 > confidence_5[h,1]) {
    countbeta5 <- countbeta5 + 1
  }
  if (beta5 < confidence_5[h,4] && beta5 > confidence_5[h,3]) {
    countbeta_5 <- countbeta_5 + 1
  }
  n_1 <- n_1 + 1
}

plot(NA, xlim = c(min(min(confidence_5[,1]), beta5),max(max(confidence_5[,2]),beta5)),
     ylim = c(0,trials), xlab = "confidence interval for beta5 paper",ylab = "trails")
abline(v = beta5, col = "red")
n_2 <- 0
for(h in 1:trials){
  segments(x0 = confidence_5[h,1], y0 = n_2, x1 = confidence_5[h,2], y1 = n_2)
  n_2 <- n_2 + 1
}

plot(NA, xlim = c(c(min(confidence_5[,3]),max(confidence_5[,4]))), ylim = c(0,trials), xlab = "confidenc
     ylab = "trails")
abline(v = beta5,col = "red")
n_3 <- 0
for(h in 1:trials){
  segments(x0 = confidence_5[h,3], y0 = n_3, x1 = confidence_5[h,4], y1 = n_3)
  n_3 <- n_3 + 1
}
```