# RCT Simulation Homework

*Ray Lu & Zarni Htet*

## Simulate Building Blocks of DAG

Simulation used methods based on random numbers to simulate of a process of interest on computer. The goal of this exercise is to learn to simulate Directed Acyclic Graph (DAG).

First, let's start with some normal simulations. The following code is a single draw, from a normal distribution with mean 0 and standard deviation 1.

```
set.seed(100)
rnorm(n=1,mean=0,sd=1)
```

```
## [1] -0.5021924
```

Think of this code as defining a *normal experiment*. You are drawing a observation once from N(0,1). Similarly, you can simulate any other distributions based on same logic .

Then, lets simulate a simple DAG structure (A -> C -> B) by following step. In this diagram, C is a child of A and A is a parent of C. C is parent of B and B is a child of C.

**Step One:** Find "parent" for each vertex (parents means direct causation) and detect vertex without parents

- Parents of A: NULL
- Parents of C: A
- Parents of B: C

We will start simulation with vertex without parent. In this example, we will simulate A ~N(0,1) with 1000 observations.

```
A<-rnorm(n=1000,mean=0,sd=1)
```

**Step Two:** Find *children* for vertex defined in step one and simulate its corresponding value.

Children for A is C. Therefore, we can simulate C based on value of A. There are several approaches that to complete this task. For example, we can assume C is linearly related with A with effective size 0.5 and random errors following N (0,1).

```
C<-0.5*A+rnorm(n=1000,mean=0,sd=1)
```

**Step Three:** Find *children* for vertex defined in step two and simulate its corresponding value. Children for C is B. Similarly, we can simulate B

```
asB<-0.5*C+rnorm(n=1000,mean=0,sd=1)
```

**Step Four:** If there are no children for vertex in previous step. The simulation is finished.

Now, it's your turn to simulate following DAG structure

(1): A→C←B (2): A← C→ B

Answer Key

```
set.seed(5123)
C<-rnorm(n=1000,mean=0,sd=1)
A<-0+0.5*C+rnorm(n=1000,mean=0,sd=1)
B<-0+0.5*C+rnorm(n=1000,mean=0,sd=1)
```

```r
set.seed(5123)
A<-rnorm(n=1000,mean=0,sd=1)
B<-rnorm(n=1000,mean=0,sd=1)
C<-0+0.5*A+0.5*B+rnorm(n=1000,mean=0,sd=1)
```

## Simulate fake data to run causal inference with lm

Show a treatment effect of 5 unit increase in Causal Inference Quiz I Score take II after the intervention of a home tutor program that is randomly assigned to students taking Professor Hill's class in Fall of 2018 .The home tutors (treatments) are equally talented in **All Ways** of graduate students by the name of **Bossy** Andrea, **Raven** Ray and **Witty** Zarni.

### Assumptions

Class size N is 100 The pre-treatment Causal Quiz Score is N(80, 10) Treatment assigned is completely random.

### Conditions on Quiz 1 Take 2

On average, both treated and non-treated student scores improves!

### Expected Output

A Linear Regression model for Causal Inference. Coefficient may round.

### Hint

Build a simulated data set first

### Libraries

```r
library(dplyr)
```

### Simulated Data Set

Creating the Vectors for the Data Set

```r
set.seed(1234)
#Create 100 students
students <- c(1:100)
#Create pre-treatment test scores for everyone
pretest <- rnorm(n=100, mean = 80, sd = 3)
#Create a vector of 0 and 1
zero <- rep(0,50)
one  <- rep(1,50)
zone <- c(zero, one)
#Creat a Randomly Assigned Treatment Variable
treat <- sample(zone, 100, replace = FALSE)
#Create a non-treated test score post-intervention
postest_nt <- rnorm(n=50, mean = 82, sd = 3)
#Create a treated test score post-intervention
postest_t <- rnorm(n =50, mean = 87, sd = 3)
```

```r
#Combining the posttest values to one vector
postest <- c(postest_nt, postest_t)
```

### Gluing the Data Together

```r
#Create a data frame containing studentID, pretest score and treatment factor variable
df <- data.frame(students, treat, pretest)
#Join the teh post-treatment scores conditioned on the treatment factor variable
df <- df %>% arrange(treat)
```

```
## Warning: package 'bindrcpp' was built under R version 3.3.2
```

```r
df$postest <- postest
head(df)
```

```
##   students treat  pretest  postest
## 1        3     0 83.25332 80.86829
## 2        7     0 78.27578 82.29286
## 3        8     0 78.36010 86.91623
## 4        9     0 78.30664 79.37322
## 5       12     0 77.00484 82.36528
## 6       13     0 77.67124 86.08639
```

### Running the LM model

```r
print(summary(lm(formula = postest ~ treat + pretest, data = df )))
```

```
##
## Call:
## lm(formula = postest ~ treat + pretest, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.3248  -1.6862   0.0161   1.5385   8.6994
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  92.4697     8.6802  10.653  < 2e-16 ***
## treat         5.4338     0.6532   8.319 5.61e-13 ***
## pretest      -0.1292     0.1089  -1.186    0.238
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.266 on 97 degrees of freedom
## Multiple R-squared:  0.4223, Adjusted R-squared:  0.4103
## F-statistic: 35.45 on 2 and 97 DF,  p-value: 2.781e-12
```