PART 2: MONTE CARLO SIMULATIONS FOR QUANTIFYING CONFOUNDING BIAS AND COLLIDER-STRATIFICATION BIAS

Elizabeth Rose Mayeda, PhD, MPH

UCSF Epi Tools Workshop

Quantitative Bias Analysis

March 4, 2021

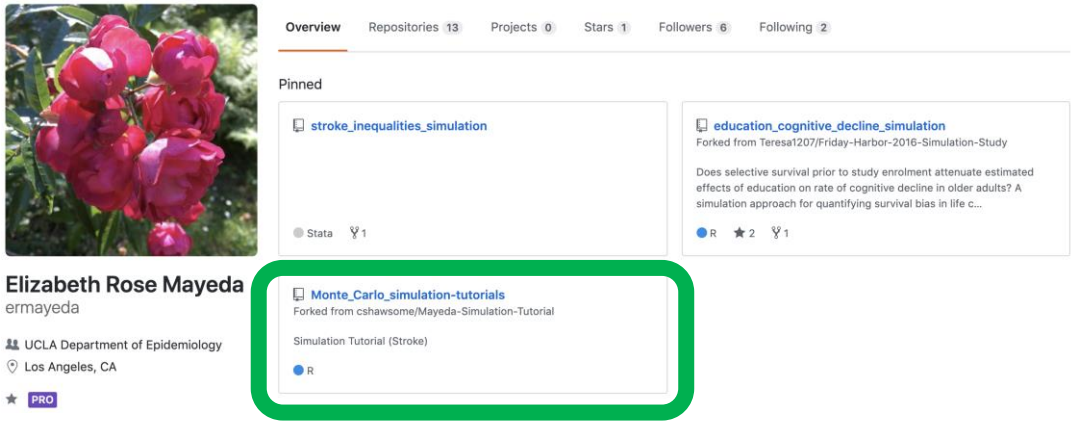
ermayeda@ph.ucla.edu

@er_mayeda

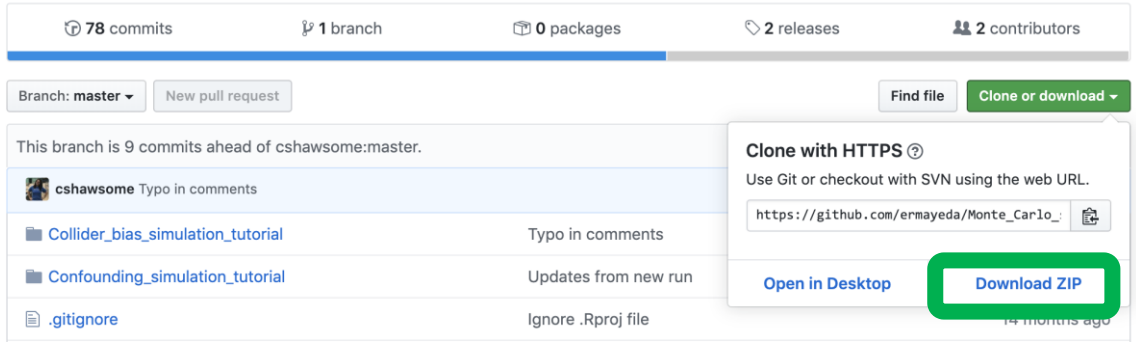
To download the simulation tutorials from GitHub…

1. Navigate to github.com/ermayeda

2. Click on the link to the Monte_Carlo_simulation_tutorials repo



3. Click clone or download and choose Download ZIP



The whole repo is now in your computer's downloads folder

Outline

- Part 2: Monte Carlo simulations for quantifying confounding bias and collider-stratification bias
  - Why simulate?
  - Simulation steps
  - Simulation exercise: confounding bias
  - Simulation exercise: collider-stratification bias
  - Briefly: extension to simulating information bias

Why simulate?

- Simulation studies provide a means for quantifying expected magnitude of bias
- Why might you simulate vs. use bias equations for quantitative bias analysis?
  - Flexible
  - Complex data-generating processes
  - One tool that can be used for quantifying multiple sources of bias in your study (i.e., confounding, selection bias, information bias)
  - Allows for comparison of alternative analytic tools

Steps for using Monte Carlo simulations for quantitative bias analysis from a simulated data set

1. Express causal structure as a DAG

2. Specify data generating rules corresponding with the DAG

Since we are generating the data, we are generating the causal effect of the exposure on the outcome (β)

3. Generate data according to specified rules

4. Estimate exposure-outcome association ($\hat{\beta}$)

5. Quantify magnitude of bias: Compare estimated exposure-outcome association with the causal effect ($\hat{\beta}$ vs. β)

Repeat!
Varying causal structure and parameterizations

Run multiple* iterations of sample generation, compare $\bar{\hat{\beta}}$ vs. β

*e.g. 1,000

Simulation tips

- K-I-S-S
- Start simple, gradually add complexity
- Sanity checks:
  - Check distributions of data: Did you generate the data you thought you did?
  - Consider a base scenario with no bias anticipated
- Check work in one iteration of sample generation (with large n)
- Check work by running a few iterations of sample generation (with desired n) prior to running desired # iterations of sample generation
- Make sure to set random number seeds so your results are reproducible

For today's simulations involving data generation from scratch, we have three Stata do files (with parallel files in R)

1. Preamble file: Sets parameters
   - Why set parameters in a separate file?: You can have separate preamble files for different sets of parameter inputs for alternative causal structures, but one data generation and analysis file

2. Data generation and analysis file: Generates one data set, analyzes the data set, and stores results as scalars

3. Run simulation file (runs Monte Carlo simulations): Calls the data generation and analysis file to generate and analyze B iterations of sample generation, stores the results from each sample, summarizes the results across the B samples, and stores the summarized results

R simulation tutorial code courtesy of Crystal Shaw

- Crystal Shaw, MS, Doctoral Student, UCLA Department of Biostatistics

Macro variables in Stata: Variables you are storing in Stata memory that you don't see in your data set

- Local macro variables
  - `local x 1 //creates a local macro variable named x with value = 1`
  - To call the local macro variable, surrounded variable by a specific set of quotation marks: `` `x' ``
  - A local macro variable is temporarily defined and only exists within the program in which it is defined
- Global macro variables
  - `global x 1 //creates a global macro variable named x with value = 1`
  - To call the global macro variable, put a "$" before the variable name: `$x`
  - A global macro variable is available to all programs in your Stata session

Generating continuous and binary variables in Stata

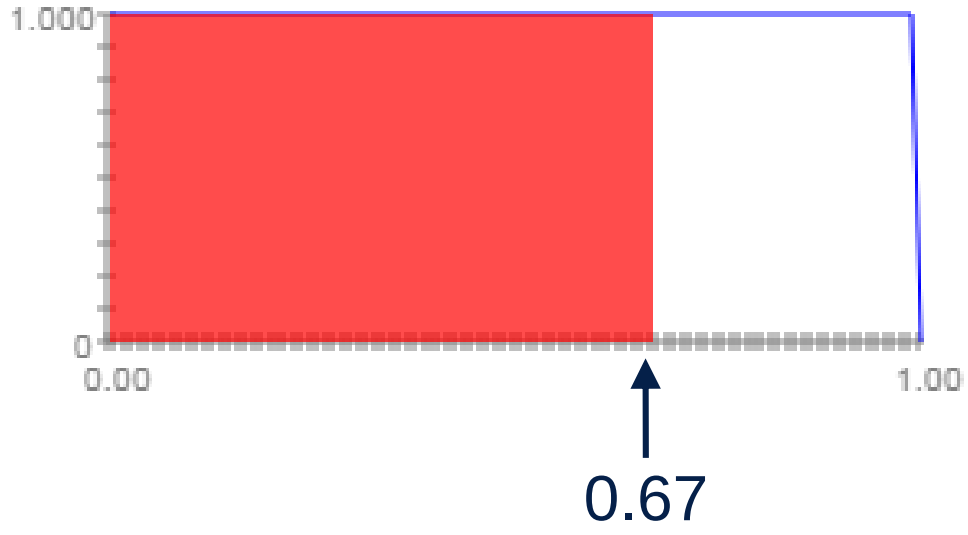
- Generating continuous variables:

```
*Generate A, where A~N(0,1)
  gen A = rnormal()
```

- Generating binary variables:

```
*Specify Probability(C)
  Gen P_C = 0.67
*Generate C
  gen C = runiform()< P_C
```

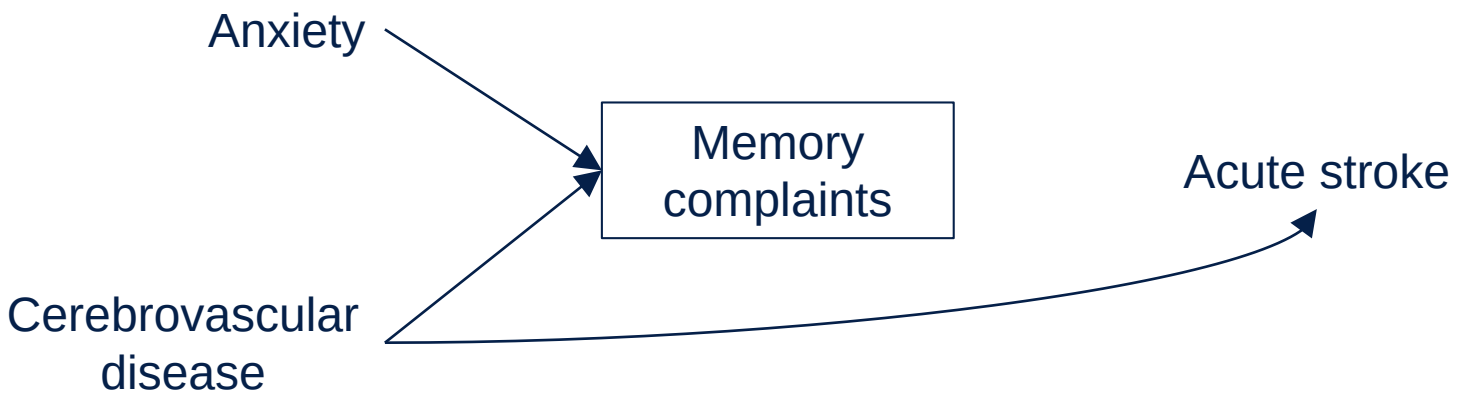
runiform() produces random numbers over [0,1)



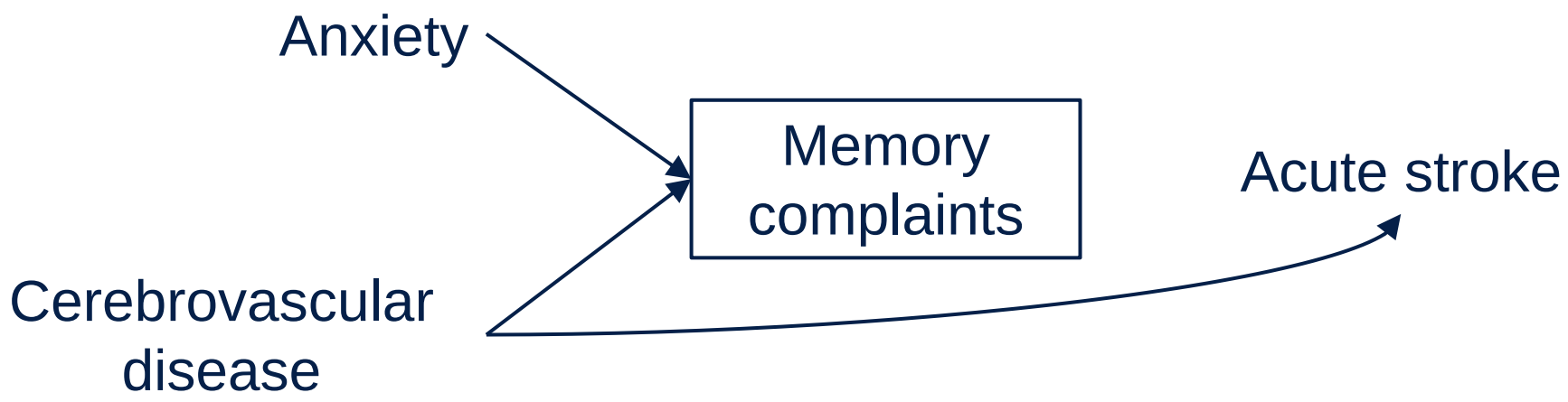
Documentation on Stata's random number generators:
http://blog.stata.com/2012/07/18/using-statas-random-number-generators-part-1/

Example

- Causal question: Does anxiety influence risk of stroke?
  - Study: Cohort study of adults with memory complaints
    - Exposure: Anxiety
    - Outcome: Acute stroke during the next 5 years
    - Selection factor/Collider: Memory complaints
    - U: Underlying cerebrovascular disease



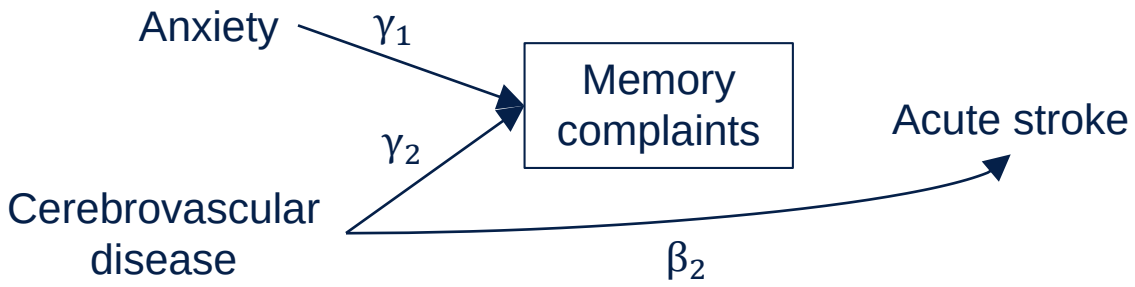
1. Express causal structure as a DAG



2. Specify data generating rules corresponding with the DAG

For each variable in the DAG, specify the causal data-generating model, generating each variable as a function of its parents

- Binary exposure: Anxiety ~ Bernoulli(0.2)
- Continuous common cause of selection and outcome: Cerebrovascular disease ~ $N(0,1)$



2. Specify data generating rules corresponding with the DAG

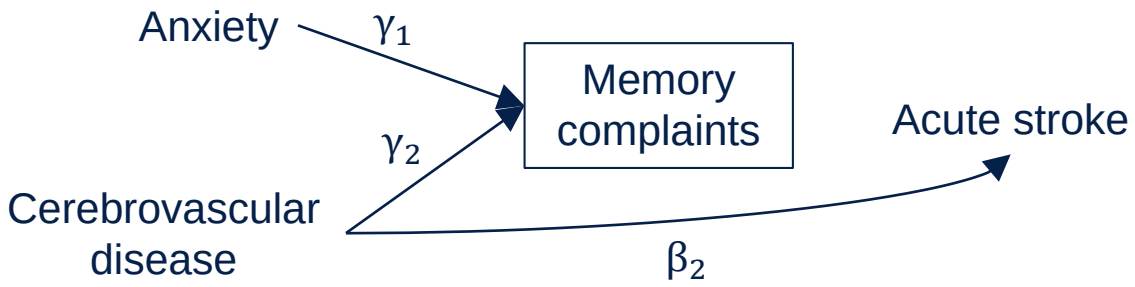
- Binary selection factor: Memory complaints

$$P(\text{memory complaints}) = \frac{\exp(\gamma_0+\gamma_1 \text{anxiety}+\gamma_2 \text{cerebrovascular disease})}{1+\exp(\gamma_0+\gamma_1 \text{anxiety}+\gamma_2 \text{cerebrovascular disease})}$$

- Binary outcome: Acute stroke

$$P(\text{acute stroke}) = \frac{\exp(\beta_0+\beta_1 \text{anxiety}+\beta_2 \text{cerebrovascular disease})}{1+\exp(\beta_0+\beta_1 \text{anxiety}+\beta_2 \text{cerebrovascular disease})}$$

*In our DAG $\beta_1 = 0$ (no effect of the exposure on the outcome)



3. Generate data according to specified rules

- Download folder: “<your file path>\Collider_bias_simulation_tutorial”
- Open Stata file:
  2_simulation_workshop_collider_bias_data_gen_analysis.do

3. Generate data according to specified rules

- Generate a data set with n=5,000 observations
- Apply the data-generating rules sequentially to the data set, starting with the exogenous variables and then for each of their children

id	anxiety	cerebrovascular disease	P(memory complaints)	memory complaints	P(acute stroke)	acute stroke
1	0	-0.630	0.039	0	0.019	0
2	1	-1.018	0.097	0	0.010	0
3	0	-0.147	0.081	0	0.040	0
…						
5,000	1	-1.420	0.054	0	0.005	0

3. Generate data according to specified rules

Start with a blank data set (no observations)

```
*Start with a blank data set (no observations)
set more off
clear
*set seed 67208113  //For multiple iterations of sample generation, the seed
                    //is set in the run simulation file
```

We will set a seed for one iteration of sample generation so we can check our work. Before we run multiple iterations of sample generation, we'll need to comment this line of code back out.

```
*Start with a blank data set (no observations)
set more off
clear
set seed 67208113   //For multiple iterations of sample generation, the seed
                    //is set in the run simulation file
```

3. Generate data according to specified rules

Generate a data set with n=5,000 observations

```
*call preamble file
do 1_simulation_workshop_collider_bias_preamble.do

/*****************************************************/
/********   Step A. Create blank data set                ********/
/*****************************************************/

set obs 5000 //creates blank dataset with desired # observations
gen id = _n
```

3. Generate data according to specified rules

Set parameters

```
/*****************************************************/
/********        Step B. Set parameters         ********/
/*****************************************************/

*Specify prevalence of A (exposure)
local P_A = $P_A

*Parameters for odds of S (selection)
local g0 = $g0   //log odds of S for ref group (A=0 and U=0)
local g1 = $g1   //log OR for effect of A on log odds of selection (OR=5.0)
local g2 = $g2   //log OR for effect of U on log odds of selection (OR=5.0)
local g3 = $g3   //log OR for interaction between A and U on  (OR=1.0)


*Parameters for odds of Y (outcome)
local b0 = $b0   //log odds of Y for ref group (A=0, U=0, and S=0)
local b1 = $b1   //log OR for effect of A on log odds of Y (OR=1.0)
local b2 = $b2   //log OR for effect of U on log odds of Y (OR=5.0)
```

3. Generate data according to specified rules

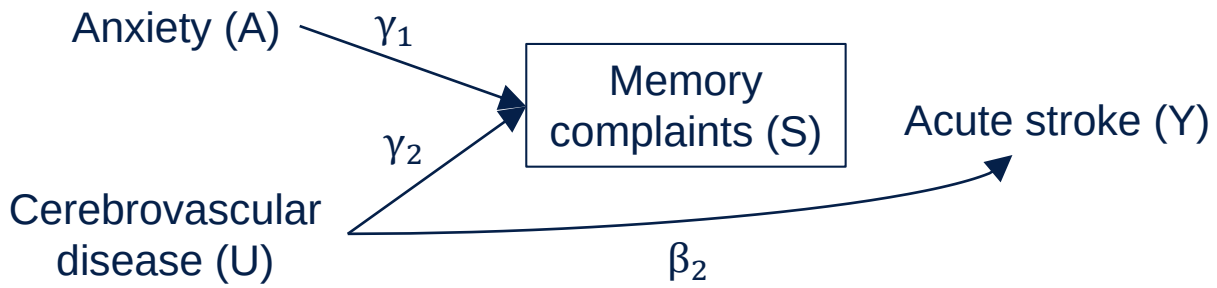
- View parameter inputs:

  1_simulation_workshop_collider_bias_preamble.do

3. Generate data according to specified rules

Generate binary Anxiety (*A*), where *A*~Bernoulli(Probability(A))

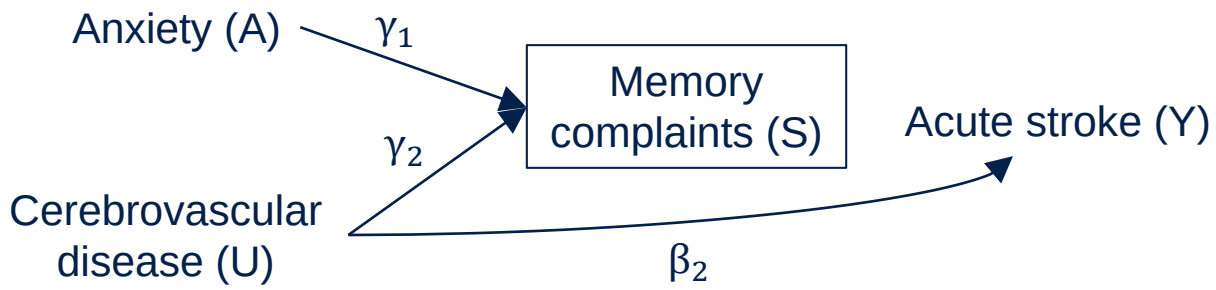
```
/*****************************************************/
/********    Step C. Generate data             ********/
/*****************************************************/

*Generate A
gen A = 0
replace A = runiform()<`P_A'
```



3. Generate data according to specified rules

Generate continuous Cerebrovascular disease (*U*), where *U*~N(0,1)

```
*Generate U, where U~N(0,1)
gen U = rnormal()
```

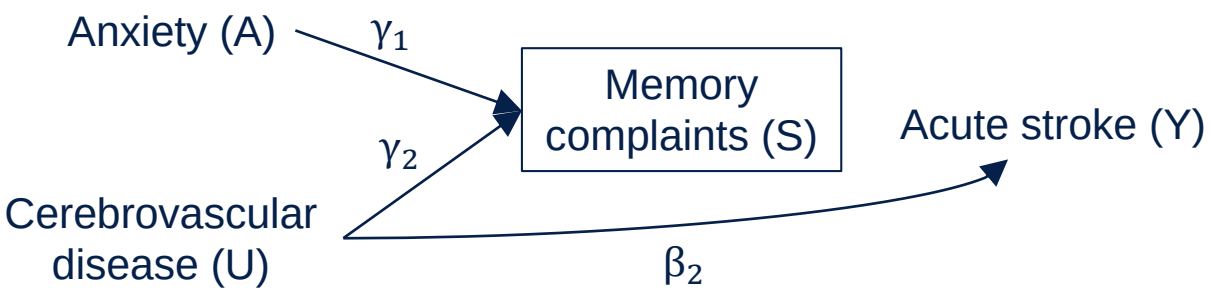


3. Generate data according to specified rules

Generate binary Memory complaints (Selection factor, *S*)

Generate P(S)

$$P(S) = \frac{\exp(\gamma_0 + \gamma_1 A + \gamma_2 U + \gamma_3 A * U)}{1 + \exp(\gamma_0 + \gamma_1 A + \gamma_2 U + \gamma_3 A * U)}$$



Generate S for each person *i*:

- Generate a random number from a uniform distribution ($R1_i \sim U[0,1]$)
- Assign person *i* S=1 if $P(S)_i > R1_i$; otherwise S=0

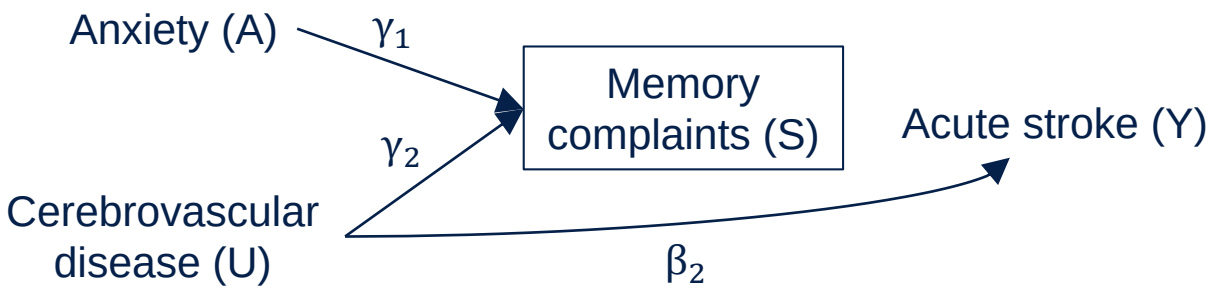
```
*Generate S
*First, generate P(S)
*Next, assign each person S status based on P(S)
gen P_S = exp(`g0' + `g1'*A + `g2'*U + `g3'*U*A)/(1 + exp(`g0' + `g1'*A
+ `g2'*U + `g3'*U*A))
gen S = runiform()<P_S
```

3. Generate data according to specified rules

Generate the Acute Stroke (Y)

Generate P(Y)

$$P(Y) = \frac{\exp(\beta_0 + \beta_1 A + \beta_2 U)}{1 + \exp(\beta_0 + \beta_1 A + \beta_2 U)}$$



*In our DAG $\beta_1 = 0$ (no effect of the A on Y)

Generate Y for each person *i*:

- Generate a random number from a uniform distribution ($R2_i \sim U[0,1)$)
- Assign person *i* Y=1 if $P(Y)_i > R2_i$; otherwise Y=0

```
*Generate Y
*First, generate P(Y)
*Next, assign each person Y status based on P(Y)
gen P_Y = exp(`b0' + `b1'*A + `b2'*U)/(1 + exp(`b0' + `b1'*A + `b2'*U))

gen Y = runiform()<P_Y
```

3. Generate data according to specified rules

Take a look at the data you generated!

id	anxiety	cerebrovascular disease	P(memory complaints)	memory complaints	P(acute stroke)	acute stroke
1	0	-0.630	0.039	0	0.019	0
2	1	-1.018	0.097	0	0.010	0
3	0	-0.147	0.081	0	0.040	0
…						
5,000	1	-1.420	0.054	0	0.005	0

4. Estimate exposure-outcome association ($\hat{\beta}$)

Before estimating exposure-outcome association, check out your results: Did you generate the data you intended to?

```
/*****************************************************/
/********     Step D. Look at data              ********/
/*****************************************************/

/********Check distributions of variables and store results********/

*Check proportions of people with: exposure (A) = 1, selection (S) = 1, outcome
(Y) = 1
*Check mean U
foreach x in A U S Y {
summarize `x', meanonly
       scalar p_`x' = round(r(mean),0.001)
}
scalar mean_U = p_U
       scalar drop p_U
```

4. Estimate exposure-outcome association ($\hat{\beta}$)

Variable	Obs	Mean	Std. Dev.	Min	Max
A	5,000	.2028	.402125	0	1

Variable	Obs	Mean	Std. Dev.	Min	Max
U	5,000	.0119757	1.017345	-4.06433	3.239829

Variable	Obs	Mean	Std. Dev.	Min	Max
S	5,000	.2238	.4168313	0	1

Variable	Obs	Mean	Std. Dev.	Min	Max
Y	5,000	.0988	.2984231	0	1

4. Estimate exposure-outcome association ($\hat{\beta}$)

```
/********Look at associations and store results********/

*Check ORs for A and Y (whole population). No bias anticipated
logistic Y A
       matrix list r(table)
       matrix matrix2 = r(table)
       scalar OR_AY_all = matrix2[1,1]
       scalar lb_OR_AY_all = matrix2[5,1]
       scalar ub_OR_AY_all = matrix2[6,1]

*Estimates of primary interest: Estimated ORs for A and Y among S=1 (store ORs and
95% CI limits)
logistic Y A if (S==1)
       matrix list r(table)
       matrix matrix3 = r(table)
       scalar OR_AY_S1 = matrix3[1,1]
       scalar lb_OR_AY_S1 = matrix3[5,1]
       scalar ub_OR_AY_S1 = matrix3[6,1]
*Look at stored results
scalar list
```

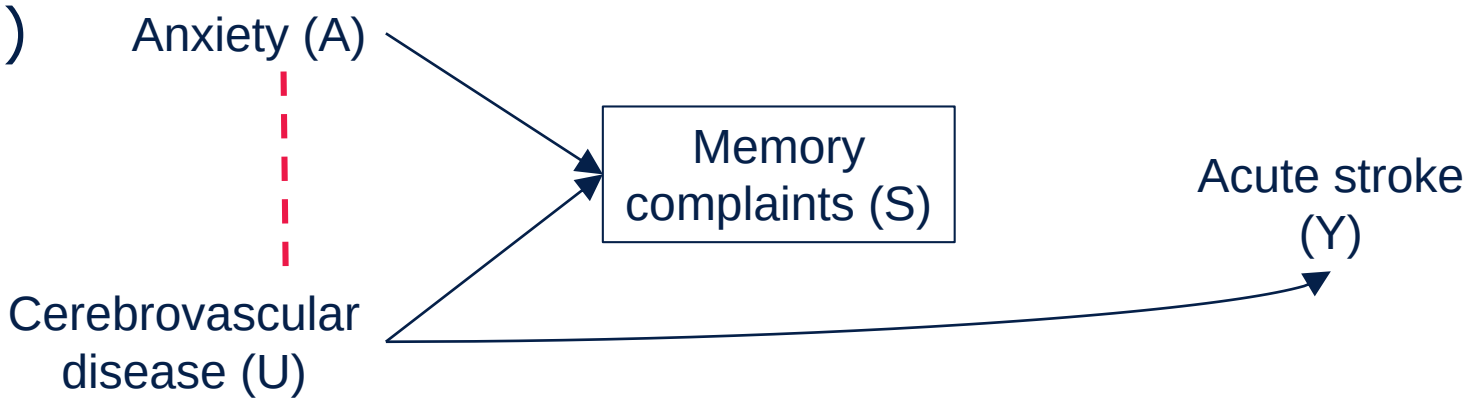
4. Estimate exposure-outcome association ($\hat{\beta}$)

- Among people with memory complaints = 1, estimate effect of anxiety on risk of acute stroke:
  - $\text{Logit}(\text{acute stroke}|\text{anxiety}, \text{memory complaints} = 1) = \exp(\hat{\beta}_0 + \hat{\beta}_1 \text{anxiety})$
  - $\widehat{\text{OR}}_{\text{AY}|\text{S}=1} = \exp(\hat{\beta}_1)$= 0.69 (95% CI: 0.51, 0.94)

- In full sample, estimate effect of anxiety on risk of acute stroke
  - $\text{Logit}(\text{acute stroke}|\text{anxiety}) = \exp(\hat{\beta}_0 + \hat{\beta}_1 \text{anxiety})$
  - $\widehat{\text{OR}}_{\text{AY}} = \exp(\hat{\beta}_1)$= 1.04 (95% CI: 0.83, 1.31)

5. Quantify magnitude of bias: Compare estimated exposure-outcome association with the causal effect ($\hat{\beta}$ vs. β)

- $\widehat{\mathrm{OR}}_{\mathrm{AY|S=1}} = \exp(\hat{\beta}_1) = 0.69$ vs. $\mathrm{OR}_{\mathrm{AY}} = \exp(\beta_1) = 1.00$
- In our simulated world, there is no causal effect of anxiety on acute stroke ($\mathrm{OR}_{\mathrm{AY}}=1.00$)
- In our hypothetical cohort of people with memory complaints, people with anxiety are less likely to have underlying cerebrovascular disease, inducing a spurious protective association between anxiety and acute stroke ($\widehat{\mathrm{OR}}_{\mathrm{AY|S=1}}<1.0$)



Run multiple iterations of sample generation

Run multiple (several hundred to several thousand) iterations of sample generation, compare $\bar{\hat{\beta}}$ vs. β

- Don't forget to comment out the seed in the data generation and analysis file or else you will generate the same data set over and over again!

```
*Start with a blank data set (no observations)
set more off
clear
set seed 67208113    //For multiple iterations of sample generation, the seed
                     //is set in the run simulation file
```

```
*Start with a blank data set (no observations)
set more off
clear
*set seed 67208113   //For multiple iterations of sample generation, the seed
                     //is set in the run simulation file
```

Run simulation file

- File name: 3_simulation_workshop_collider_bias_run.do
- Call the data generation and analysis file B times to generate and analyze B data sets
- Summarize results across the B replications
- Stores the results from the B replications (one replication = one row in the file)

Run simulation file

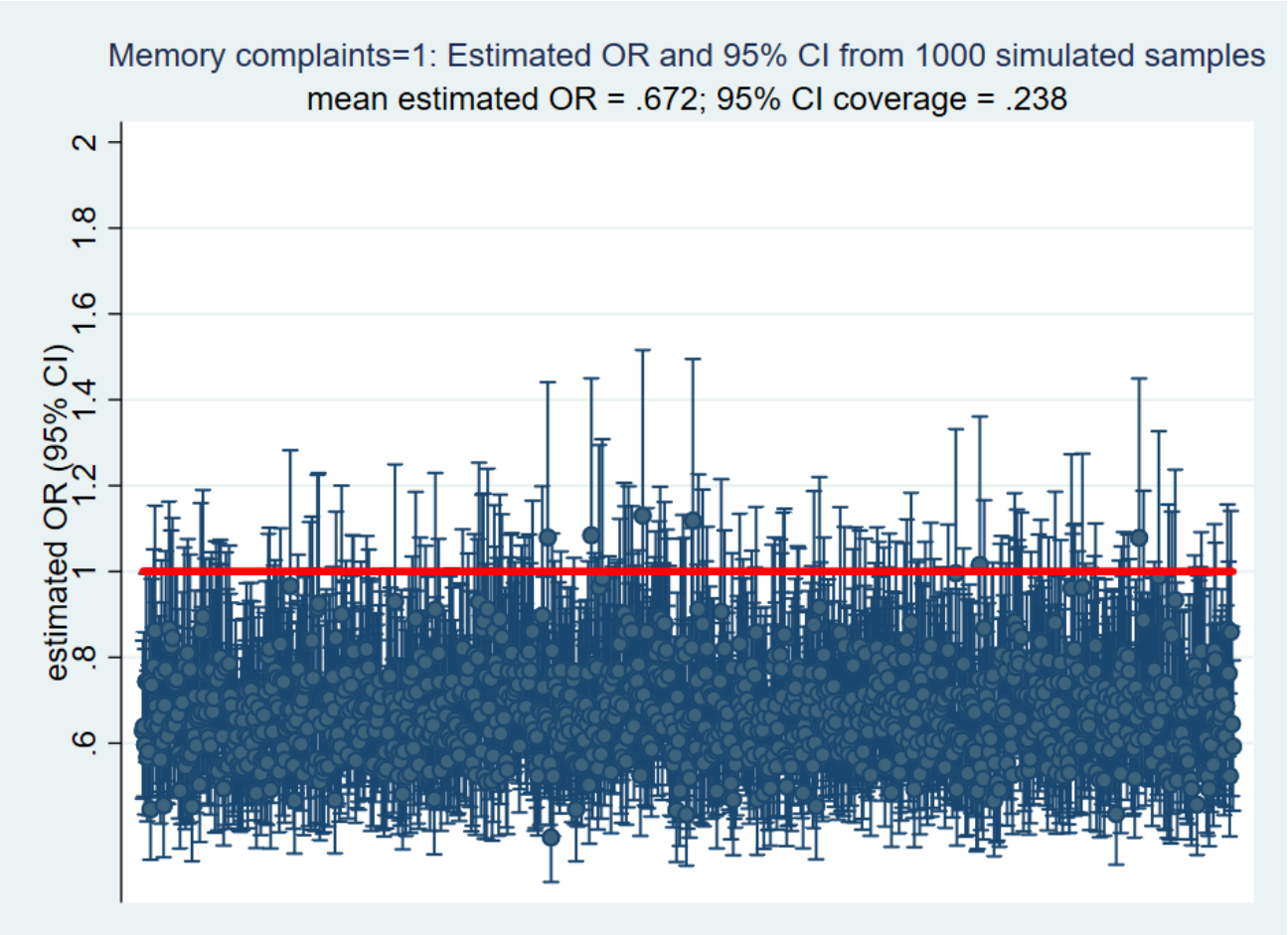
- The simulate command performs Monte Carlo-type simulations
- Basic form of the code is:
  - `simulate [list of scalars stored from each replication], reps(B) seed(#): do [data generation and analysis do file]`
  - This will replicate B data sets using the data generating code and perform the analyses in the do file. Then it will pull the results from each replication we stored as scalars so we can summarize results across all B repetitions.
  - `"reps(B)" specifies the number of replications`
  - `"seed(#)" specifies the random number seed`
  - Simulate documentation: http://www.stata.com/manuals13/rsimulate.pdf

Simulation results

Sample	OR_{S1}	Upper CL OR_{S1}	Lower CL OR_{S1}	OR_{all}	Upper CL OR_{all}	Lower CL OR_{all}	Mean U	…
1	0.63	0.84	0.47	1.06	1.32	0.85	0.02	
2	0.64	0.86	0.48	0.95	1.19	0.76	-0.01	
3	0.60	0.82	0.43	0.91	1.14	0.72	0.01	
…								
1,000	0.59	0.79	0.44	0.99	1.23	0.80	0.02	

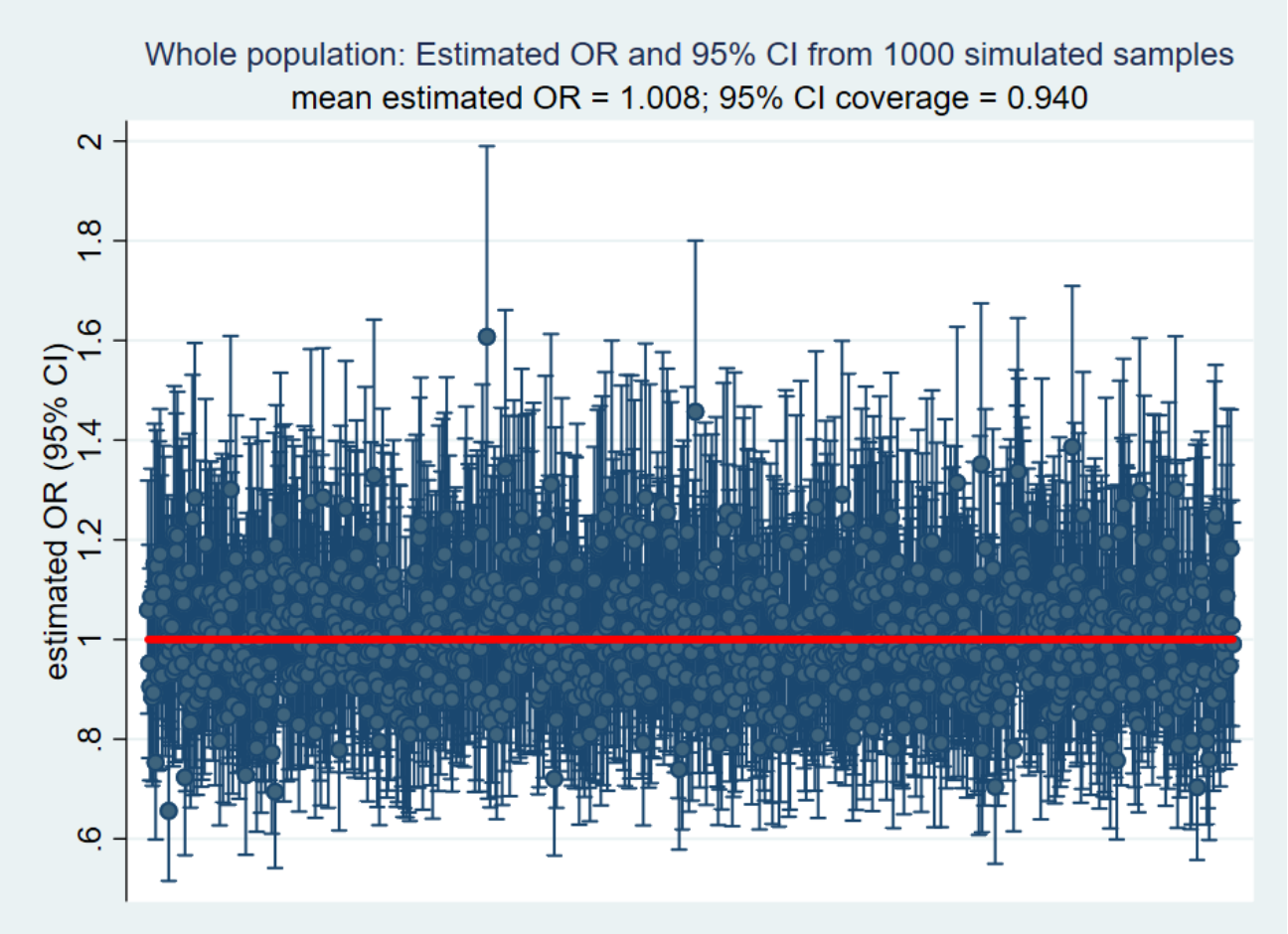
Simulation results: $\widehat{\mathrm{OR}}$ and 95% CI among Memory complaints = 1

$\overline{\widehat{\mathrm{OR}}}_{\mathrm{AY|S=1}}$ = 0.672; 23.8% of 95% CI's include $\mathrm{OR}_{\mathrm{AY}}$ = 1.0



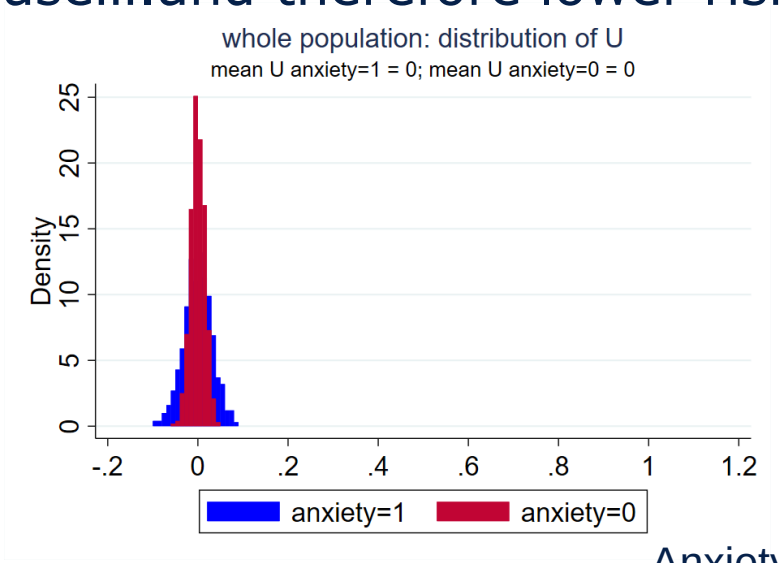
Simulation results: $\widehat{\mathrm{OR}}$ and 95% CI in whole sample

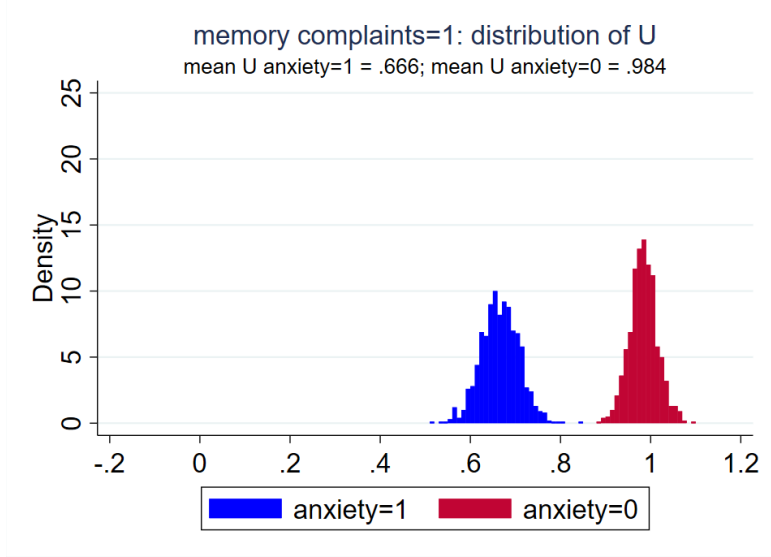
$\overline{\widehat{\mathrm{OR}}}_{\mathrm{AY}}$ = 1.008; 94.0% of 95% CI's include $\mathrm{OR}_{\mathrm{AY}}$ = 1.0

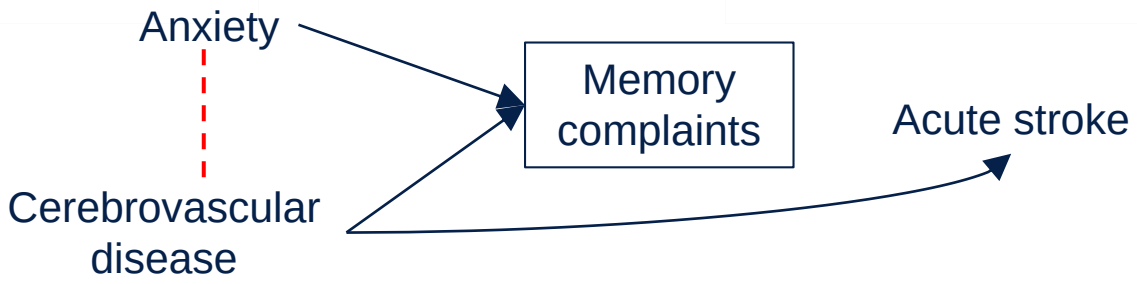


Spurious association between anxiety and stroke induced by conditioning on memory complaints

- Because our hypothetical study was restricted to people with memory complaints, participants with anxiety have lower burden of underlying cerebrovascular disease….and therefore lower risk of acute stroke



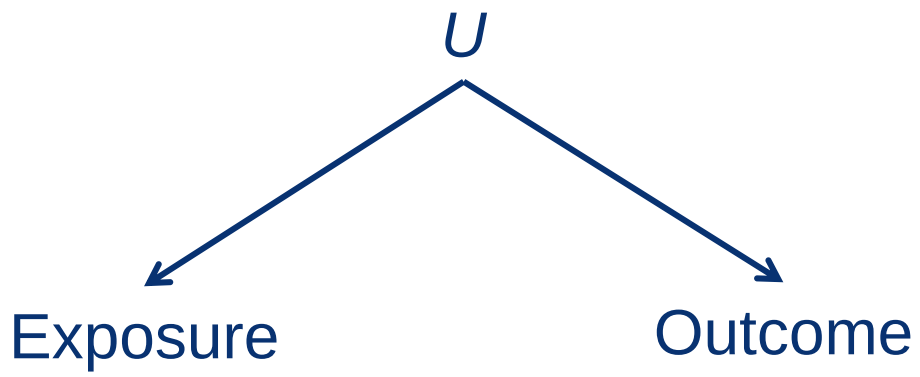




Simulation exercise: Confounding bias

Example: Hypothetical cohort study with an unmeasured confounder

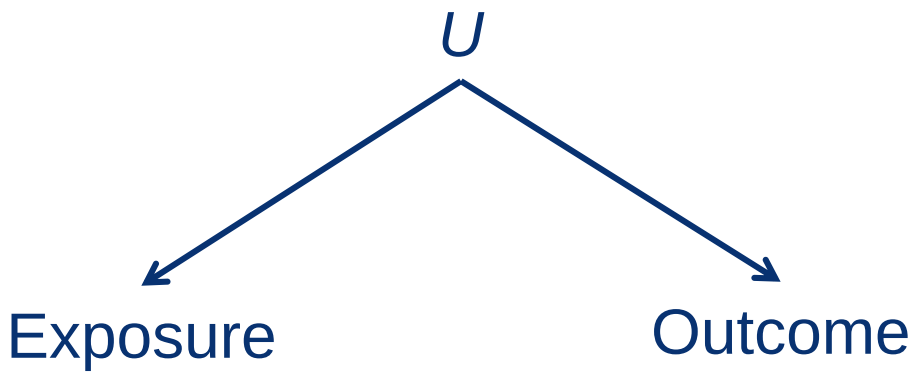
- Causal question: Does “exposure” cause “outcome”?
- Hypothetical cohort study of 5,000 people
- Binary exposure
- Binary outcome
- Unmeasured continuous confounder *U*



- Estimand of interest: odds ratio for the effect of exposure on the outcome
- For simplicity, we will simulate under the sharp null in this exercise (no effect of exposure on the outcome for any individual in the population) and assume no other sources of bias

1. Express causal structure as a DAG

Draw a DAG representing the exposure, the outcome, and confounder



2. Specify data generating rules corresponding with the DAG

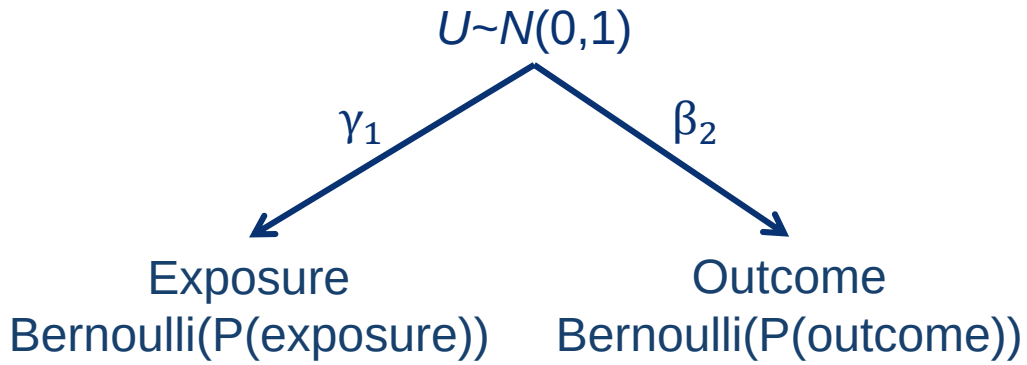
For each variable in the DAG, specify the causal data-generating model, generating each variable as a function of its parents

- Continuous confounder: $U \sim N(0,1)$
- Binary exposure:

$$\mathrm{P}(\text{exposure}) = \frac{\exp(\gamma_0 + \gamma_1 U)}{1 + \exp(\gamma_0 + \gamma_1 U)}$$

- Binary outcome:

$$\mathrm{P}(\text{outcome}) = \frac{\exp(\beta_0 + \beta_1 \text{exposure} + \beta_2 \mathrm{U})}{1 + \exp(\beta_0 + \beta_1 \text{exposure} + \beta_2 \mathrm{U})}$$



*In our DAG $\beta_1 = 0$ (no effect of the exposure on the outcome)

3. Generate data according to specified rules

- Download folder: “<your file path>\Confounding_simulation_tutorial”
- Open Stata file:
  2_simulation_workshop_confounding_data_gen_analysis.do

3. Generate data according to specified rules

- Generate a blank data set with n=5,000 observations
- Apply the data-generating rules sequentially to the data set, starting with the exogenous variables and then for each of their children

id	*U*	P(Exposure)	Exposure	P(Outcome)	Outcome
1	1.829	0.470	1	0.283	1
2	-0.487	0.151	0	0.073	0
3	0.391	0.247	0	0.127	1
…					
5,000	-2.345	0.047	0	0.021	0

3. Generate data according to specified rules

Start with a blank data set (no observations)

```
*Start with a blank data set (no observations)
set more off
clear
*set seed 67208113  //For multiple iterations of sample generation, the seed
                    //is set in the run simulation file
```

We will set a seed for one iteration of sample generation so we can check our work. Before we run multiple iterations of sample generation, we'll need to comment this line of code back out.

```
*Start with a blank data set (no observations)
set more off
clear
set seed 67208113   //For multiple iterations of sample generation, the seed
                    //is set in the run simulation file
```

3. Generate data according to specified rules

Generate a data set with n=5,000 observations

```
*call preamble file
do 1_simulation_workshop_confounding_preamble.do

/*****************************************************/
/********   Step A. Create blank data set             ********/
/*****************************************************/

set obs 5000 //creates blank dataset with desired # observations
gen id = _n
```

3. Generate data according to specified rules

Set parameters

```
/*************************************************************/
/********    Step B. Set parameters                  ********/
/*************************************************************/

*Parameters for odds of exposure
local g0 = $g0 //log odds [p/(1-p)] of exposure for ref group (U=0)
local g1 = $g1 //effect of one-unit higher U on log odds of exposure

*Parameters for odds of outcome
local b0 = $b0 //log odds [p/(1-p)] of outcome for ref group (U=0,exp=0)
local b1 = $b1 //effect of exposure on log odds of outcome
local b2 = $b2 //effect of one-unit higher U on log odds of outcome
```

3. Generate data according to specified rules

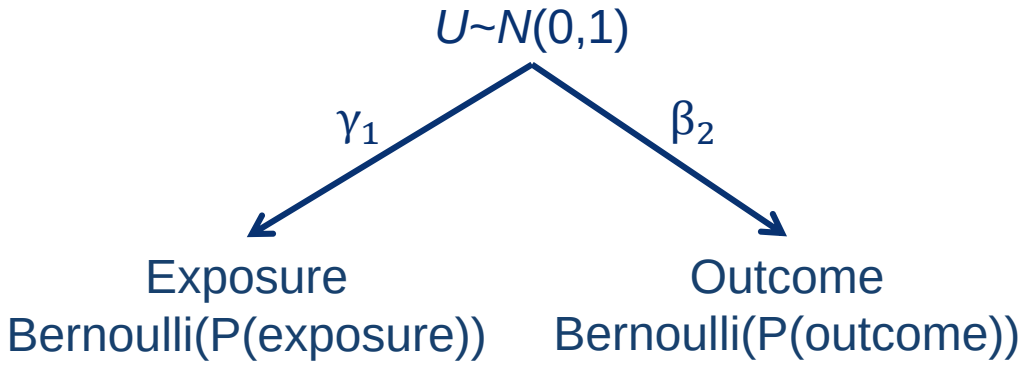
- View parameter inputs:

  1_simulation_workshop_confounding_preamble.do

3. Generate data according to specified rules

Generate continuous confounder *U*, where *U*~N(0,1)

```
/*****************************************************************/
/********    Step C. Generate data                        ********/
/*****************************************************************/

*Generate U, where U~N(0,1)
gen U = rnormal()
```

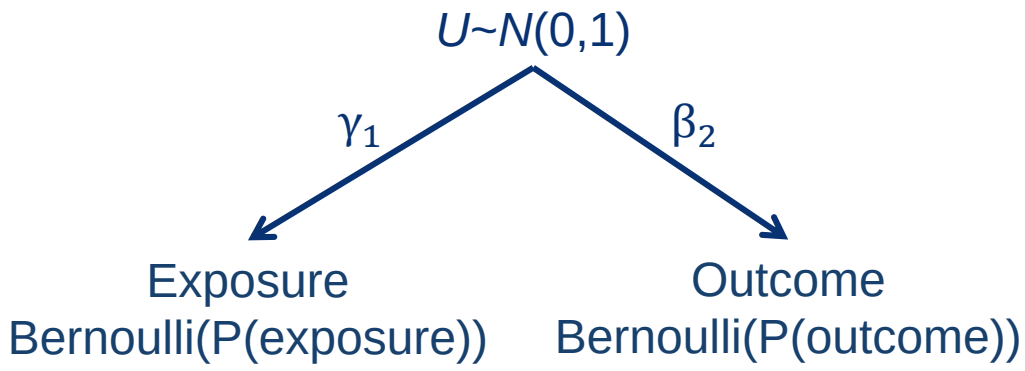


3. Generate data according to specified rules

Generate binary exposure

Generate P(exposure)

$$\mathrm{P(exposure)} = \frac{\exp(\gamma_0 + \gamma_1 \mathrm{U})}{1 + \exp(\gamma_0 + \gamma_1 \mathrm{U})}$$



Generate exposure for each person *i*:

- Generate a random number from a uniform distribution ($R1_i \sim U[0,1]$)
- Assign person *i* exposure=1 if $P(exposure)_i > R1_i$; otherwise exposure=0

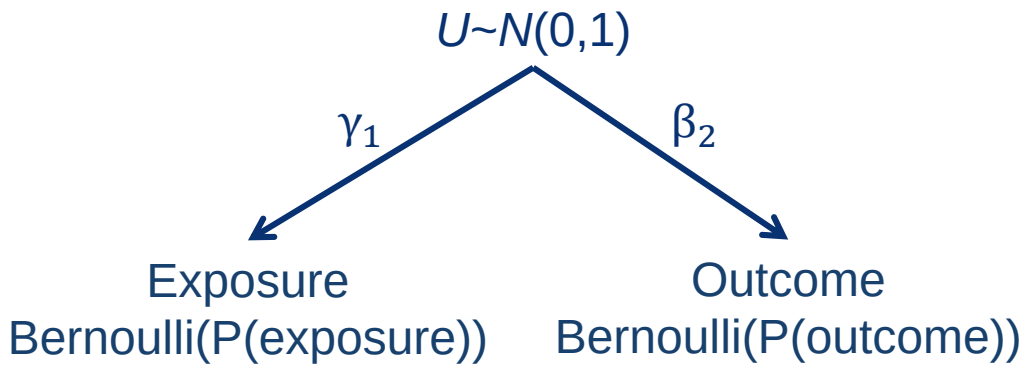
```
*Generate exposure
*First, generate P(exposure)
*Next, assign each person exposure status based on P(exposure)
gen Pexposure = exp(`g0' + `g1'*U )/(1 + exp(`g0' + `g1'*U))

gen exposure= runiform()<Pexposure
```

3. Generate data according to specified rules

Generate binary outcome

Generate P(outcome)

$$\mathrm{P(outcome)} = \frac{\exp(\beta_0 + \beta_1 \mathrm{exposure} + \beta_2 \mathrm{U})}{1 + \exp(\beta_0 + \beta_1 \mathrm{exposure} + \beta_2 \mathrm{U})}$$



*In our DAG $\beta_1 = 0$ (no effect of the exposure on the outcome)

Generate outcome for each person *i*:

- Generate a random number from a uniform distribution ($R2_i \sim U[0,1]$)
- Assign person *i* outcome=1 if $P(outcome)_i > R2_i$; otherwise outcome=0

```
*Generate outcome
*First, generate P(outcome)
*Next, assign each person outcome status based on P(exposure)
gen Poutcome = exp(`b0' + `b1'*exposure + `b2'*U)/(1 + exp(`b0' +
`b1'*exposure + `b2'*U))
gen outcome= runiform()<Poutcome
```

3. Generate data according to specified rules

Take a look at the data you generated!

id	*U*	P(Exposure)	Exposure	P(Outcome)	Outcome
1	1.829	0.470	1	0.283	1
2	-0.487	0.151	0	0.073	0
3	0.391	0.247	0	0.127	1
…					
5,000	-2.345	0.047	0	0.021	0

4. Estimate exposure-outcome association ($\hat{\beta}$)

Before estimating exposure-outcome association, check out your results: Did you generate the data you intended to?

```
/*******************************************************/
/********     Step D. Analyze data & store results     ********/
/*******************************************************/

*Store results as scalar variables
*Scalar variables store single numbers or  stings, not variables in the dataset


/**** Check distributions and store results ***/

*Check mean value of U and proportion of people with exposure=1 and outcome=1
foreach x in U exposure outcome {
summarize `x'
scalar p_`x' = round(r(mean),0.001)
}
scalar mean_U = p_U
scalar drop p_U
```

4. Estimate exposure-outcome association ($\hat{\beta}$)

Before estimating exposure-outcome association, check out your results: Did you generate the data you intended to?

```
 Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
           U |      5,000    .0097761    1.014481   -4.06433   3.791389

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
    exposure |      5,000        .2344    .4236656          0          1

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     outcome |      5,000        .1112    .3144111          0          1
```

4. Estimate exposure-outcome association ($\hat{\beta}$)

```
/*** Look at associations and store results ***/

*Estimated OR & 95% CI for exposure-outcome association
*Adjusting for U (confounder)
logistic outcome i.exposure c.U
        matrix list r(table)
        matrix logit1 = r(table)
        scalar OR_exposure_Uyes = logit1[1,2]
        scalar lb_OR_exposure_Uyes = logit1[5,2]
        scalar ub_OR_exposure_Uyes = logit1[6,2]

*Estimated OR & 95% CI for exposure-outcome association
*Without adjusting for U (confounder)
logistic outcome i.exposure
        matrix list r(table)
        matrix logit2 = r(table)
        scalar OR_exposure_Uno = logit2[1,2]
        scalar lb_OR_exposure_Uno = logit2[5,2]
        scalar ub_OR_exposure_Uno = logit2[6,2]

*Look at stored results
scalar list
```

5. Quantify magnitude of bias: Compare estimated exposure-outcome association with the causal effect ($\hat{\beta}$ vs. β)

- In our simulated world, there is no causal effect of the exposure on the outcome ($\exp(\beta_1) = \text{OR}=1.0$)
- $\widehat{\text{OR}}_{\text{crude}} = \exp(\hat{\beta}_1) = 1.75$ (95% CI: 1.44, 2.11)
- $\widehat{\text{OR}}_{\text{adjusted}} = \exp(\hat{\beta}_1) = 1.14$ (95% CI: 0.93, 1.40)

Run multiple iterations of sample generation

Run multiple (several hundred to several thousand) iterations of sample generation, compare $\bar{\hat{\beta}}$ vs. β

- Don't forget to comment out the seed in the data generation and analysis file or else you will generate the same data set over and over again!

```
*Start with a blank data set (no observations)
set more off
clear
set seed 67208113    //For multiple iterations of sample generation, the seed
                     //is set in the run simulation file
```

```
*Start with a blank data set (no observations)
set more off
clear
*set seed 67208113   //For multiple iterations of sample generation, the seed
                     //is set in the run simulation file
```

Run simulation file

- File name: 3_simulation_workshop_confounding_run.do
- Call the data generation and analysis file B times to generate and analyze B data sets
- Summarize results across the B replications
- Stores the results from the B replications (one replication = one row in the file)

Run simulation file

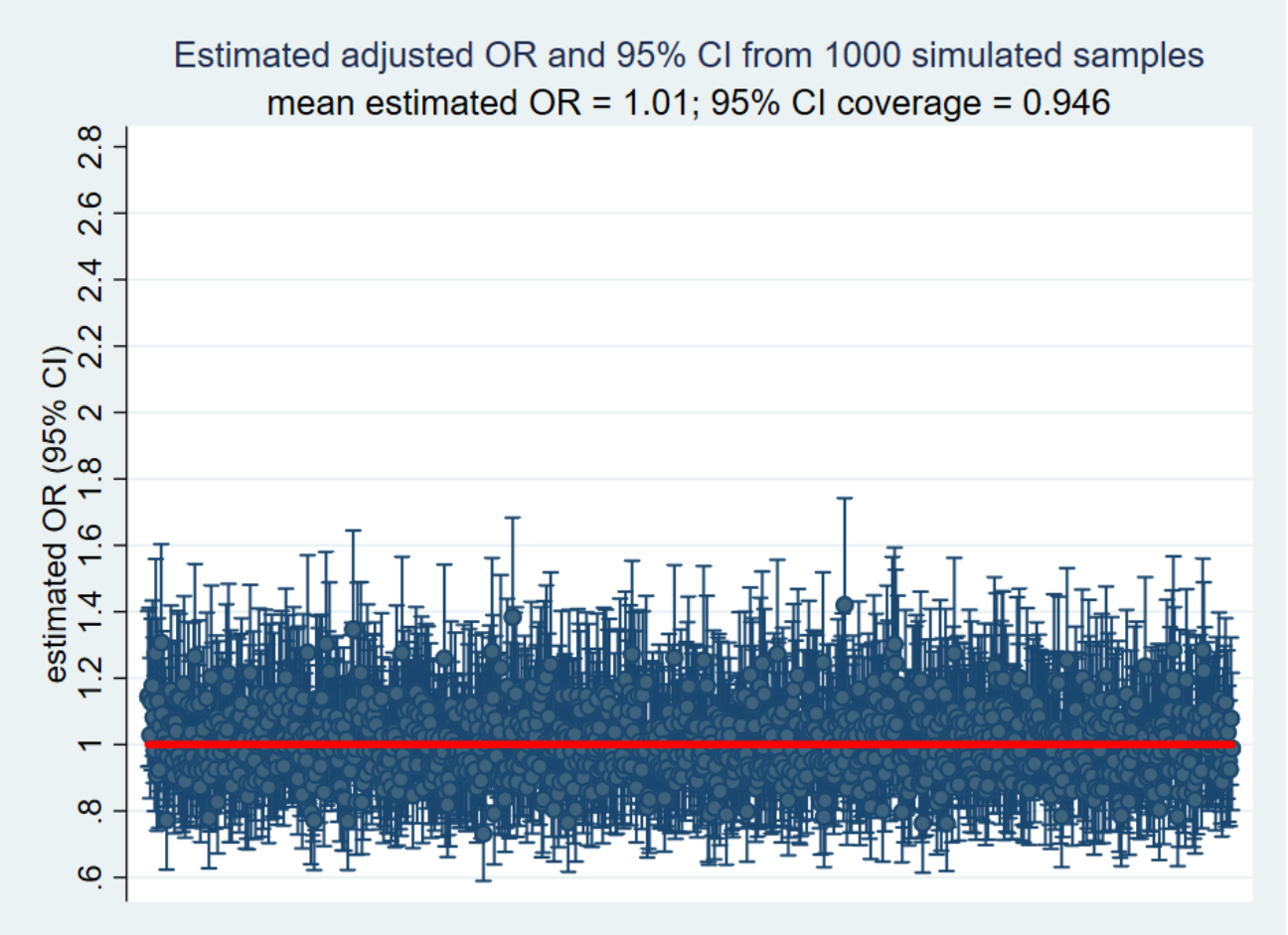
- The simulate command performs Monte Carlo-type simulations
- Basic form of the code is:
  - `simulate [list of scalars stored from each replication], reps(B) seed(#): do [data generation and analysis do file]`
  - This will replicate B data sets using the data generating code and perform the analyses in the do file. Then it will pull the results from each replication we stored as scalars so we can summarize results across all B repetitions.
  - `"reps(B)" specifies the number of replications`
  - `"seed(#)" specifies the random number seed`
  - Simulate documentation: http://www.stata.com/manuals13/rsimulate.pdf

Simulation results

Sample	Mean U	Prev exposure	Prev outcome	OR_{adj}	OR_{crude}	Lower CL OR_{adj}	Upper CL OR_{adj}	Lower CL OR_{crude}	Upper CL OR_{crude}	95% CI covers truth OR_{adj}	95% CI covers truth OR_{crude}
1	0.010	0.234	0.111	1.14	1.75	0.93	1.40	1.44	2.11	1	0
2	-0.006	0.208	0.115	1.15	1.70	0.94	1.41	1.40	2.06	1	0
3	-0.003	0.219	0.118	1.03	1.57	0.84	1.26	1.30	1.90	1	0
…											
1,000	-0.026	0.216	0.115	0.99	1.44	0.80	1.22	1.19	1.76	1	0

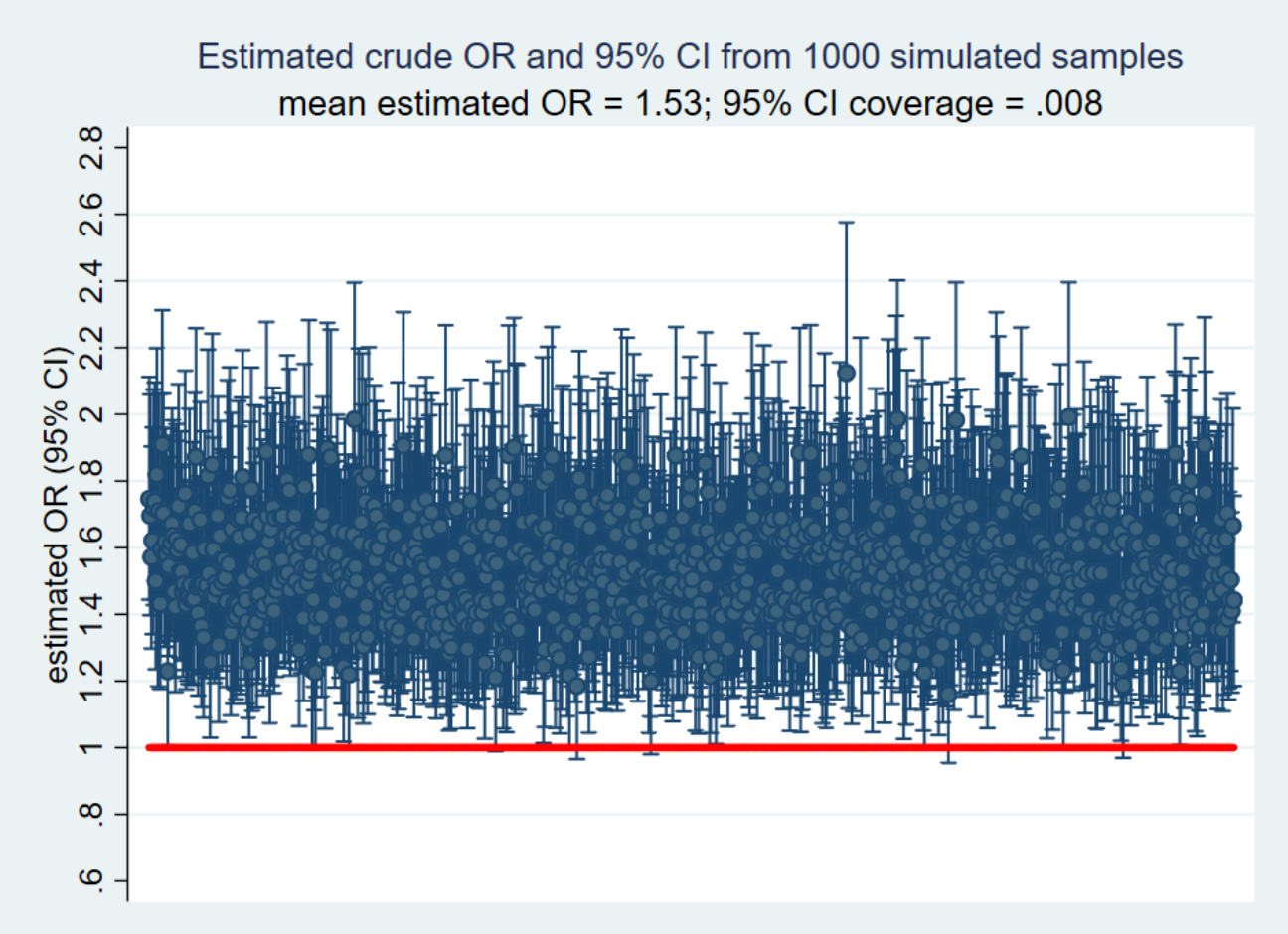
Simulation results: Adjusted $\widehat{\mathrm{OR}}$ and 95 % CI

$\overline{\widehat{\mathrm{OR}}}_{\mathrm{adjusted}}$ = 1.01, 95% CI coverage = 0.946



Simulation results: Crude $\widehat{\mathrm{OR}}$ and 95 % CI

$\overline{\widehat{\mathrm{OR}}}_{\mathrm{crude}}$ = 1.53, 95% CI coverage = 0.008



Simulating information bias (briefly)

- Measurement error for continuous variable *X*
- Generate variable that represents "true" value of *X* (X_true)
- Generate variable that represents measurement error for *X* (X_error)
- Generate variable that represents "measured" value of *X*
  X_measured = X_true + X_error

- For simplicity, we only looked at a few simulation results today, but in practice, we recommend:
  - Many sanity checks (did you simulate the data you intended to?)
  - Simulation results: Calculate mean(estimate), bias, % bias, mean(estimated SE), empirical SE, root mean square error (RMSE), 95% CI coverage, estimate median and 2.5^{th} and 97.5^{th} percentiles
- For simplicity, we only considered one causal structure and set of parameter inputs for each source of bias, but in practice, we recommend considering several causal structures/sets of input parameters