Section 5

Observational Studies 2

Sooahn Shin

GOV 2003

Oct 14, 2021

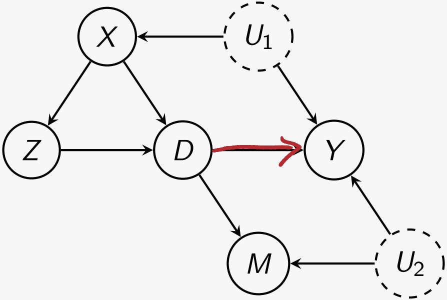
Overview

- Logistics:
  - **Pset 5 released!** Due at 11:59 pm (ET) on Oct 20

- Today's topics:
  1. Directd Acyclic Graphs

Motivation

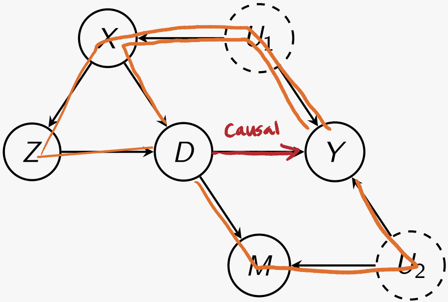
- Assume that you've come out with a DAG based on your expertise.



- Suppose you want to identify a causal effect of D on Y.

Motivation

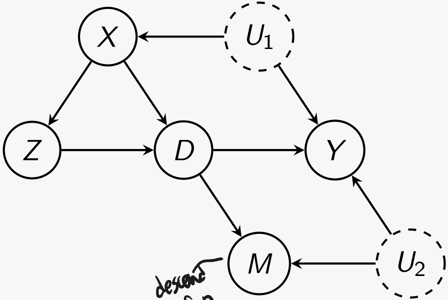
- Assume that you've come out with a DAG based on your expertise.



- Suppose you want to identify a causal effect of D on Y.
  - In a nutshell, what you might want to do is to block all the paths that yields statistical associations between D and Y.

Motivation

- Assume that you've come out with a DAG based on your expertise.



descent of D

- Suppose you want to identify a causal effect of D on Y.
  - In a nutshell, what you might want to do is to block all the paths that yields statistical associations between D and Y.
  - Thus, you want to find a set of nodes **S** such that X, Z, M ~~U_1~~, ~~U_2~~
    - once we condition on **S**, **no unmeasured confounding** holds and
    - any descend of D is not in **S** $\rightsquigarrow$ **no post-treatment bias**.
- $\rightsquigarrow$ Use backdoor criterion!

DAG

- Things we have to know to check the backdoor criterion:

DAG

- Things we have to know to check the backdoor criterion:
  - Three common structures:
    - confounder (fork): $A \leftarrow C \rightarrow B$
    - collider (inverted fork): $A \rightarrow C \leftarrow B$
    - mediator (chain): $A \rightarrow C \rightarrow B$

DAG

- Things we have to know to check the backdoor criterion:
  - Three common structures:
    - confounder (fork): $A \leftarrow C \rightarrow B$
    - collider (inverted fork): $A \rightarrow C \leftarrow B$
    - mediator (chain): $A \rightarrow C \rightarrow B$
  - How to block a path between A and B
    - If $A \leftarrow C \rightarrow B$: condition on C.
    - If $A \rightarrow C \leftarrow B$: **do not** condition on C.
    - If $A \rightarrow C \rightarrow B$: condition on C.

DAG

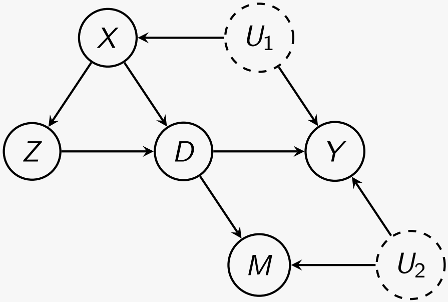
- Things we have to know to check the backdoor criterion:

  - Three common structures:
    - confounder (fork): $A \leftarrow C \rightarrow B$
    - collider (inverted fork): $A \rightarrow C \leftarrow B$
    - mediator (chain): $A \rightarrow C \rightarrow B$

  - How to block a path between A and B
    - If $A \leftarrow C \rightarrow B$: condition on C.
    - If $A \rightarrow C \leftarrow B$: **do not** condition on C.
    - If $A \rightarrow C \rightarrow B$: condition on C.

  - **D-separation**: $A \perp\!\!\!\perp B \mid C$
    1. Find all paths between A and B.
    2. Check if each path is **blocked**.
    3. If all paths are blocked, then A is **d-separated** from B by C

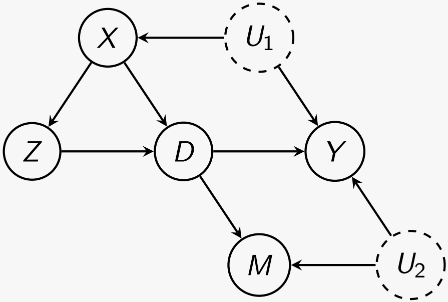
Backdoor criterion example



1. List all of the **backdoor** paths between D and Y.

- $D \leftarrow X \leftarrow U_1 \rightarrow Y$
- $D \leftarrow Z \leftarrow X \leftarrow U_1 \rightarrow Y$

Backdoor criterion example



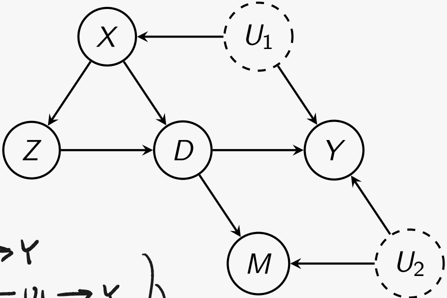
1. List all of the **backdoor** paths between D and Y.
2. List all the possible set of nodes **S** that you can condition on.

$\emptyset$ $\{X\}$ $\{Z\}$ $\{M\}$

$\{X, Z\}$ $\{X, M\}$ $\{Z, M\}$

$\{X, Z, M\}$

Backdoor criterion example



- $D \leftarrow \boxed{X} \leftarrow U_1 \rightarrow Y$
- $D \leftarrow \boxed{Z} \leftarrow \boxed{X} \leftarrow U_1 \rightarrow Y$

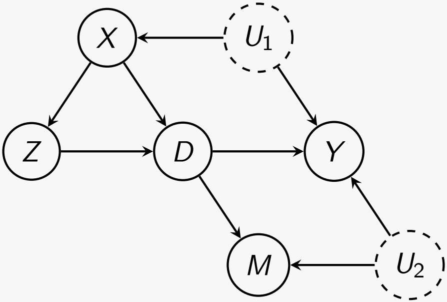
1. List all of the **backdoor** paths between D and Y.
2. List all the possible set of nodes **S** that you can condition on.
3. List all the **S** such that **blocks** all the backdoor paths.

$\emptyset$ ✓$\{X\}$ $\{Z\}$ $\{M\}$

✓$\{X, Z\}$ $\{X, M\}$ $\{Z, M\}$

$\{X, Z, M\}$

Backdoor criterion example



1. List all of the **backdoor** paths between D and Y.
2. List all the possible set of nodes **S** that you can condition on.
3. List all the **S** such that **blocks** all the backdoor paths.
4. Among those **S**, drop the sets which include a descend of D.

M

R package: `dagitty`

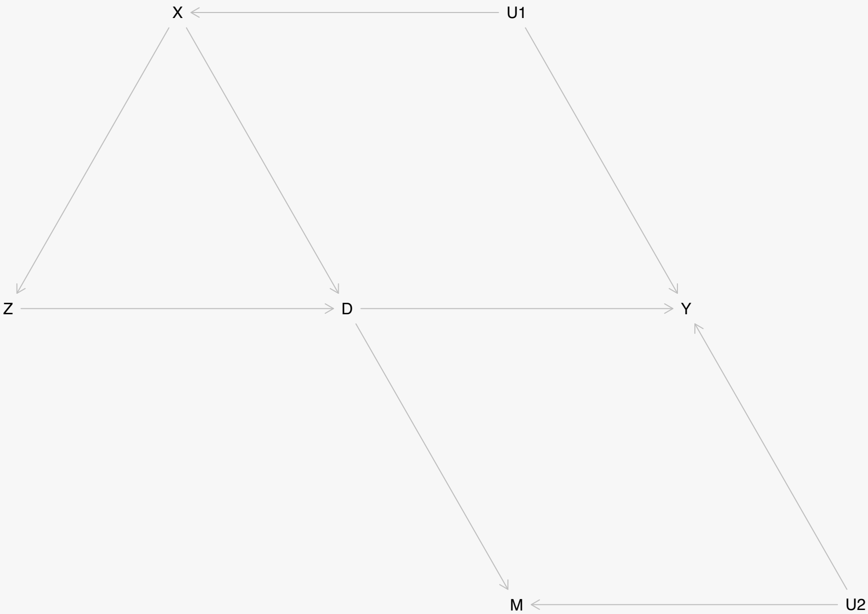
- DAGitty: www.dagitty.net

```
library(dagitty)
g <- dagitty('dag {
    X [pos="1,-1.5"]
    Y [pos="4,0"]
    Z [pos="0,0"]
    M [pos="3,1.5"]
    D [pos="2,0"]
    U1 [pos="3,-1.5"]
    U2 [pos="5,1.5"]

    X -> Z -> D -> Y
    X -> D -> M
    M <- U2 -> Y
    X <- U1 -> Y
}')
latents(g) <- c("U1", "U2")
```

R package: `dagitty`

```
plot(g)
```



R package: `dagitty`

```r
parents(g, "D")
```

```
## [1] "X" "Z"
```

```r
ancestors(g, "D")
```

```
## [1] "D"  "Z"  "X"  "U1"
```

```r
children(g, "D")
```

```
## [1] "M" "Y"
```

```r
descendants(g, "D")
```

```
## [1] "D" "Y" "M"
```

R package: dagitty

```r
paths(g, "D", "Y")$paths
```

```
## [1] "D -> M <- U2 -> Y"      "D -> Y"                 "D <- X <- U1 -> Y
## [4] "D <- Z <- X <- U1 -> Y"
```

```r
paths(g, "D", "Y", directed = TRUE)$paths # only causal path(s)
```

```
## [1] "D -> Y"
```

R package: dagitty

```r
dseparated(g, "Z", "D", c("X")) # because of Z -> D
```

```
## [1] FALSE
```

```r
dseparated(g, "Z", "M", c("D"))
```

```
## [1] TRUE
```

* latent(g) ← c(V1, V2)

```r
impliedConditionalIndependencies(g)
```

```
## M _||_ X | D
## M _||_ Z | D
## Y _||_ Z | D, X
```

$\cancel{V_1}$, $\cancel{V_2}$

R package: dagitty

```r
dseparated(g, "Z", "D", c("X")) # because of Z -> D
```

```
## [1] FALSE
```

```r
dseparated(g, "Z", "M", c("D"))
```

```
## [1] TRUE
```

```r
impliedConditionalIndependencies(g)
```

```
## M _||_ X | D
## M _||_ Z | D
## Y _||_ Z | D, X
```

R package: dagitty

```r
adjustmentSets(g, "D", "Y", type="minimal")

## { X }

# Caveat: adjustmentSets may include unobserved variables
#         which we cannot actually condition on.
S = adjustmentSets(g, "D", "Y", type="all")
S[!grepl("U1|U2", S)]

## { X }
## { X, Z }

# Note that this implements a slightly more general criterion
# (sometimes it may contain descendants)
```

X Z

* Frontdoor Criterion

$$p(Y(d)) = \sum_m \{ p(M_i = m \mid T_i = t) \sum_{t'} P(Y \mid T = t', M_i = m) P(T_i = t') \}$$