Causal models

Mark Andrews
Psychology Department, Nottingham Trent University

mark.andrews@ntu.ac.uk

Causal DAG

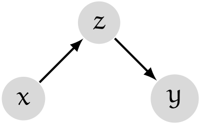
- A causal DAG is a DAG: A graph with vertices representing variables, and directed edges between the vertices and contain no cycles.
- It represents the causal relationships between variables: If one variable x causes another y, we write $x \to y$.
- The absence of an direct edge between x and y indicates no *direct* effect of x on y.
- The absence of directed edg between variables is as or more informative than the presence of directed edge.
- A causal DAG must have all relevant variables, all explanatory and outcome variables and *all their common causes*.

Causal DAG

- A causal DAG contains three key structures:
  - Chains
  - Forks
  - Colliders

Chains

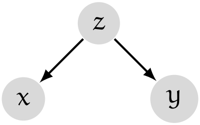
- Chains encode causal chains.



- Controlling for or blocking z blocks the causal effect of x on y.

Forks

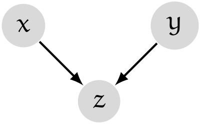
- Forks indicate confounders or spurious correlations.



- Controlling for or blocking z deconfounds the spurious correlation between x and y.

Colliders

- Colliders encode *endogenous associations*.



- Controlling for or blocking z *creates* a spurious correlation between x and y.

d-separation

- Two variables x and y in a causal DAG are d-separated by z if they are conditionally independent, conditional on z.
- We write this as follows:

$$x \perp\!\!\!\perp y | z$$

- If two variables are conditionally independent, they will have a partial correlation of zero.
- In general, we can have sets of variables being d-separated by set of other variables.
- Every causal DAG leads to list of d-separations.
- Each d-separation is a testable hypothesis.

d-separation: Alternative definitions

- Two variables x and y are d-separated given z iff *each* path from x to y is d-separated by z.
- A path is d-separated by z iff any of the following hold.
  - The path contains a chain $x \rightarrow z \rightarrow y$ or $x \leftarrow z \leftarrow y$.
  - The path contains a fork $x \leftarrow z \rightarrow y$.
  - The path contains a collider $x \rightarrow m \leftarrow y$, where m is neither z nor any of m's dependents is z.

d-separation: Alternative definitions

- The two variables x and y are d-connected if there is *any* active path between them, and are d-separated if *all* paths that connect them are inactive.
- A path between x and y is *blocked* by variable z if any of the following hold:
    1. x and y are connected by a chain in which z is a middle node.
    2. x and y are connected by a common cause, and z is that common cause.
    3. x and y are connected by a common effect (*collider*), but z is not that common effect, and z is not one of the effects of the common effect.

Deciding d-separation: Bayes ball

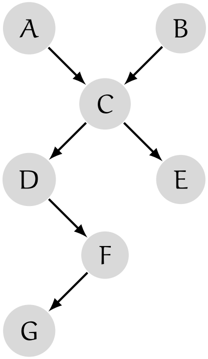
- We can decide if x and y are d-separated by z as follows:
- Shade z (or all nodes in z if z is a set).
- Place a "ball" in x, let it "bounce around" and see if it reaches y.
  1. The ball can pass through chain unless middle node is shaded.
  2. The ball can pass through a fork unless shaded in middle.
  3. The ball can *not* pass through collider unless middle node is shaded.
- If the ball can get from x to y then x and y are d-connected, otherwise they are d-separated.

Deciding d-separation: The "moral" and disoriented graph

- Draw the ancestral graph consisting only of the relevant variables (e.g., x, y, z) and all of their *ancestors* (parents, parents' parents, etc.)
- *Moralize* the ancestral graph by *marrying* the parents. For each pair of variables with a common child, draw an *undirected* edge between them. If a variable has more than two parents, draw lines between every pair of parents.
- *Disorient* the graph by replacing the directed edges with undirected edges.
- Delete the *givens* and their edges, i.e. erase the conditioning variables from the graph and erase all of their connections.

5. If the x and y variables are disconnected in this graph (no path between them), they are d-separated.

Example



d-separation queries

1. $A \perp\!\!\!\perp B | D, F$?
2. $A \perp\!\!\!\perp B$?
3. $A \perp\!\!\!\perp B | C$?
4. $D \perp\!\!\!\perp E | C$?
5. $D \perp\!\!\!\perp E$?
6. $D \perp\!\!\!\perp E | A, B$?
7. $D \perp\!\!\!\perp E | C$?
8. $D \perp\!\!\!\perp G | C$?

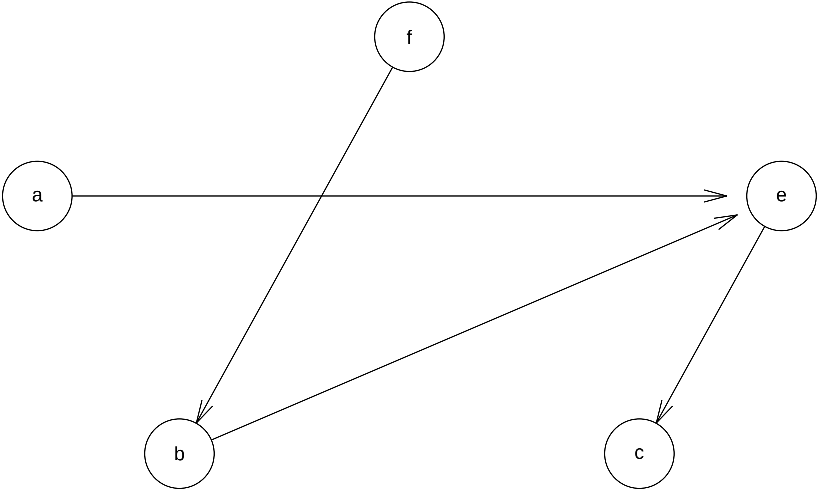
d-separation queries

1. $A \perp\!\!\!\perp B | D, F$? No
2. $A \perp\!\!\!\perp B$? Yes
3. $A \perp\!\!\!\perp B | C$? No
4. $D \perp\!\!\!\perp E | C$? Yes
5. $D \perp\!\!\!\perp E$? No
6. $D \perp\!\!\!\perp E | A, B$ No
7. $D \perp\!\!\!\perp E | C$? Yes
8. $D \perp\!\!\!\perp G | C$? No

Using bnlearn

```
library(bnlearn)
bnet <- model2network("[a][b|f][c|e][e|a:b][f]")

plot(bnet)
```



Using bnlearn

```r
# Is a d-separated from b given c?
dsep(bnet, "a", "b", "c")
```

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

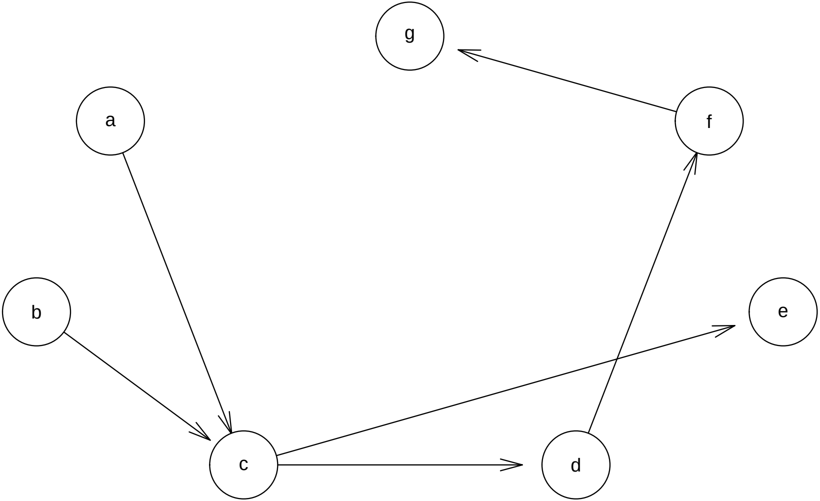
```r
# Is a d-sepated from b given f
dsep(bnet, 'a', 'b', 'f')
```

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

Using bnlearn

```
bnet <- model2network("[a][b][c|a:b][d|c][e|c][f|d][g|f]")

plot(bnet)
```



Using bnlearn

```r
# a d-separated from e given {}
dsep(bnet, 'd', 'e')
```

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

```r
# a d-separated from e given {a,b}
dsep(bnet, 'd', 'e', c('a', 'b'))
```

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

Using bnlearn

```r
# d d-separated from e given c
dsep(bnet, 'd', 'e', 'c')
```

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

```r
# a d-separated from e given {}
dsep(bnet, 'd', 'e')
```

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

```r
# a d-separated from e given {a,b}
dsep(bnet, 'd', 'e', c('a', 'b'))
```

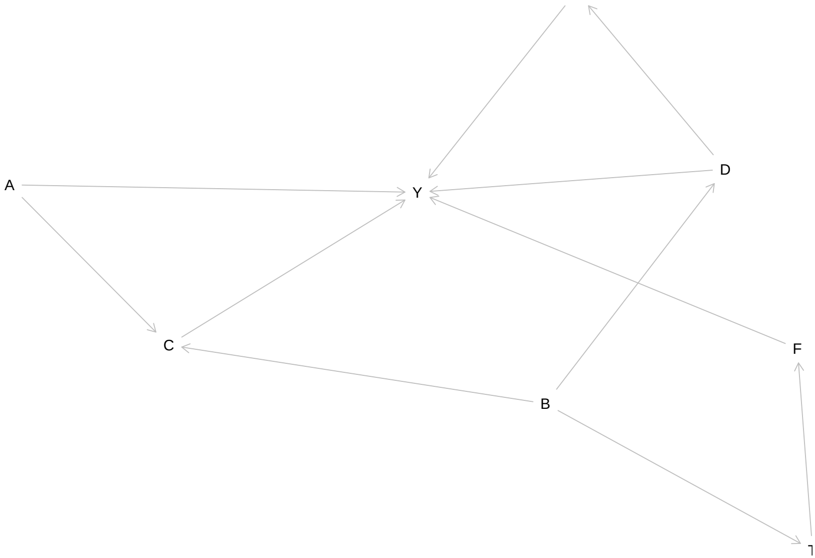
```
## [1] FALSE
```

Using daggity

```r
library(dagitty)
g1 <- dagitty( "dag {
    Y <- A
    Y <- C
    Y <- F <- T <- B
    Y <- E <- D <- B
    Y <- D <- B
    C <- A
    C <- B
}")
```

Using daggity

```r
plot(graphLayout(g1))
```



Using dagitty

- The *basis set* of d-separations.

```
impliedConditionalIndependencies(g1, type = 'basis.set')
```

```
## A _||_ B, D, E, F, T
## B _||_ A
## C _||_ D, E, F, T | A, B
## D _||_ A, C, F, T | B
## E _||_ A, B, C, F, T | D
## F _||_ A, B, C, D, E | T
## T _||_ A, C, D, E | B
## Y _||_ B, T | A, C, D, E, F
```