Directed Acyclic Graphs (DAGs)

EC 607 Metrics, Tutorial 7

Philip Economides
Spring 2021

Today

- Recap DAGs
- Simple DAG
- Adding Chains
- Collider Bias

Recap

- Graphical representation of a chain of causal effects.
- Explain causality in terms of counterfactuals.
- Nodes for random variables, random variables assumed to follow some data-generating process.
- Causal effects defined as comparison between two states of world; one state when intervention occured on some value and another state where it did not occur under some other intervention.
- Effects are eiether direct or mediated by a third variable.

Simple DAG

```
p_load(dagitty,ggdag)

g <- dagitty("dag{
  a -> b ;
  b -> c ;
  d -> c
 }")
#plot( graphLayout(g))
```

```
# Evaluating linkages
parents(g, "c")
```

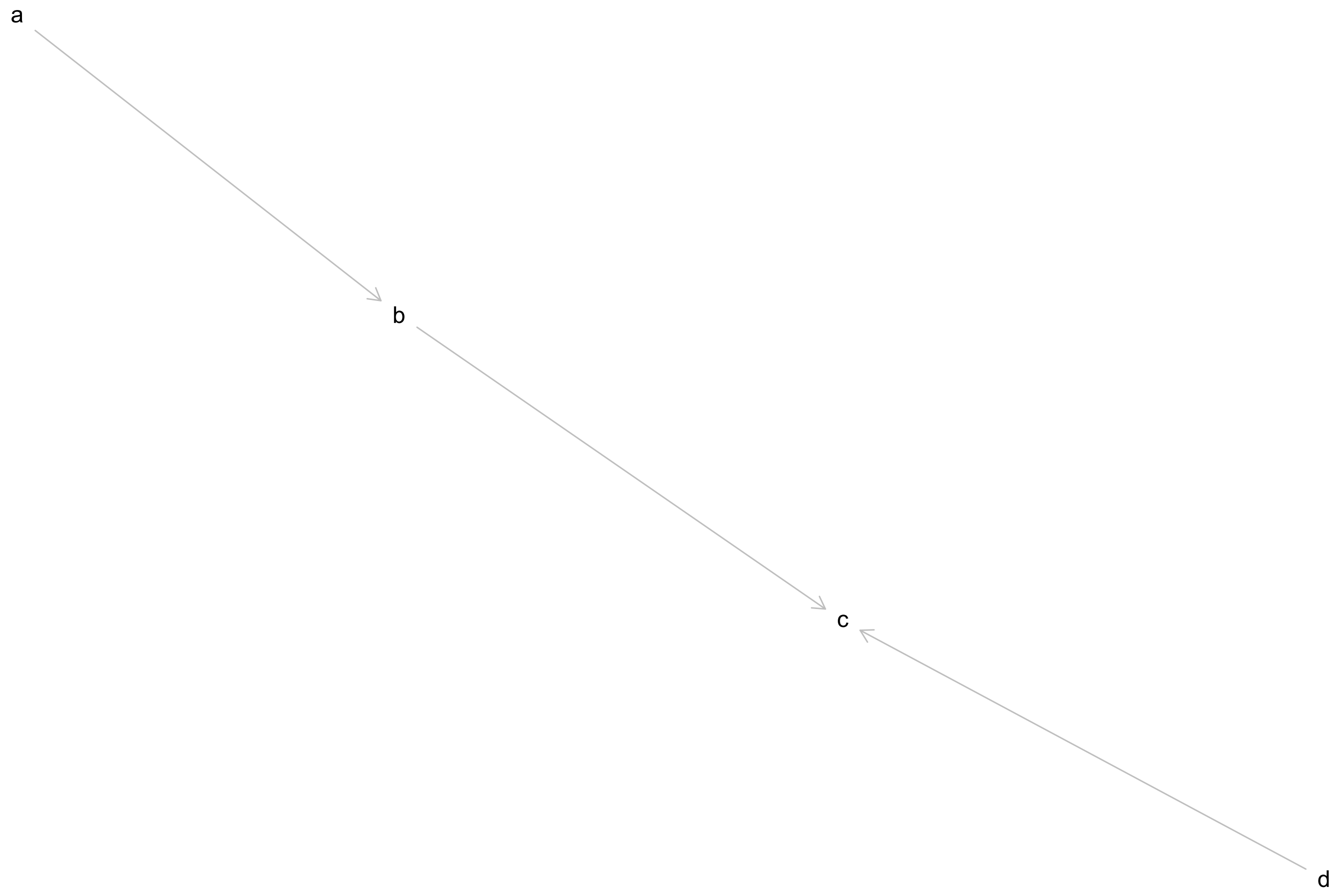
```
#> [1] "b" "d"
```

```
children(g, "b")
```

```
#> [1] "c"
```

Simple DAG

Simple DAG



Simple DAG

First, assign your random variable linkages and their coordinates.

```r
p_load(data.table)

dag_full = dagify(
  Y ~ D,
  Y ~ W,
  D ~ W,
  coords = tibble(
    name = c("Y", "D", "W"),
    x = c(1, 3, 2),
    y = c(2, 2, 1)  ) )
```

Simple DAG

We can include information for segments in `data.table` format.

```r
# Convert to data.table
dag_dt = dag_full %>% ggplot2::fortify() %>% setDT()
dag_dt[, `:=`(
  path1 = (name == "D" & to == "Y") | (name == "Y"),
  path2 = (name == "D" & to == "W") | (name == "W" & to == "Y") | (name == "Y")
)]
```

name	x	y	direction	to	xend	yend	circular	path1	path2
D	3	2	->	Y	1	2	FALSE	TRUE	FALSE
W	2	1	->	D	3	2	FALSE	FALSE	FALSE
W	2	1	->	Y	1	2	FALSE	FALSE	TRUE
Y	1	2					FALSE	TRUE	TRUE

Simple DAG

Having set up our mapping, we just need co-ordinates for causal arrows.

```
# Shorten segments
mult = 0.15
dag_dt[, `:=`(
  xa = x + (xend-x) * (mult),
  ya = y + (yend-y) * (mult),
  xb = x + (xend-x) * (1-mult),
  yb = y + (yend-y) * (1-mult)
)]
```

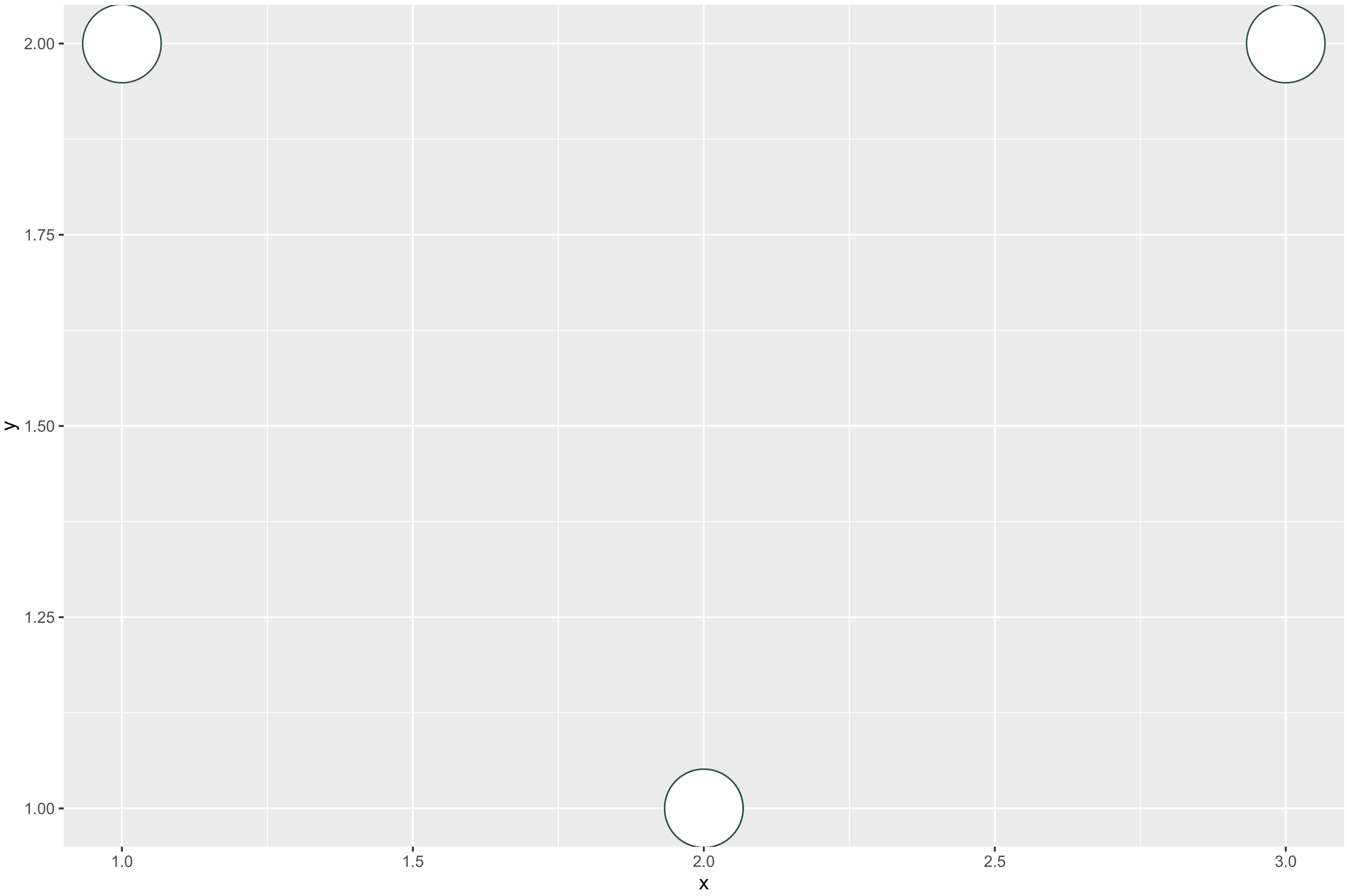
name	x	y	direction	to	xend	yend	circular	path1	path2	xa	ya	xb	yb
D	3	2	->	Y	1	2	FALSE	TRUE	FALSE	2.7	2	1.3	2
W	2	1	->	D	3	2	FALSE	FALSE	FALSE	2.15	1.15	2.85	1.85
W	2	1	->	Y	1	2	FALSE	FALSE	TRUE	1.85	1.15	1.15	1.85
Y	1	2					FALSE	TRUE	TRUE				

Simple DAG

Using ggplot and the segment points, we can begin visualizing.

```r
# Plot the full DAG
p1 = ggplot(
  data = dag_dt,
  aes(x = x, y = y, xend = xend, yend = yend)
) +
geom_point(
  size = 20,
  fill = "white",
  color = slate,
  shape = 21,
  stroke = 0.6
)
```

Simple DAG

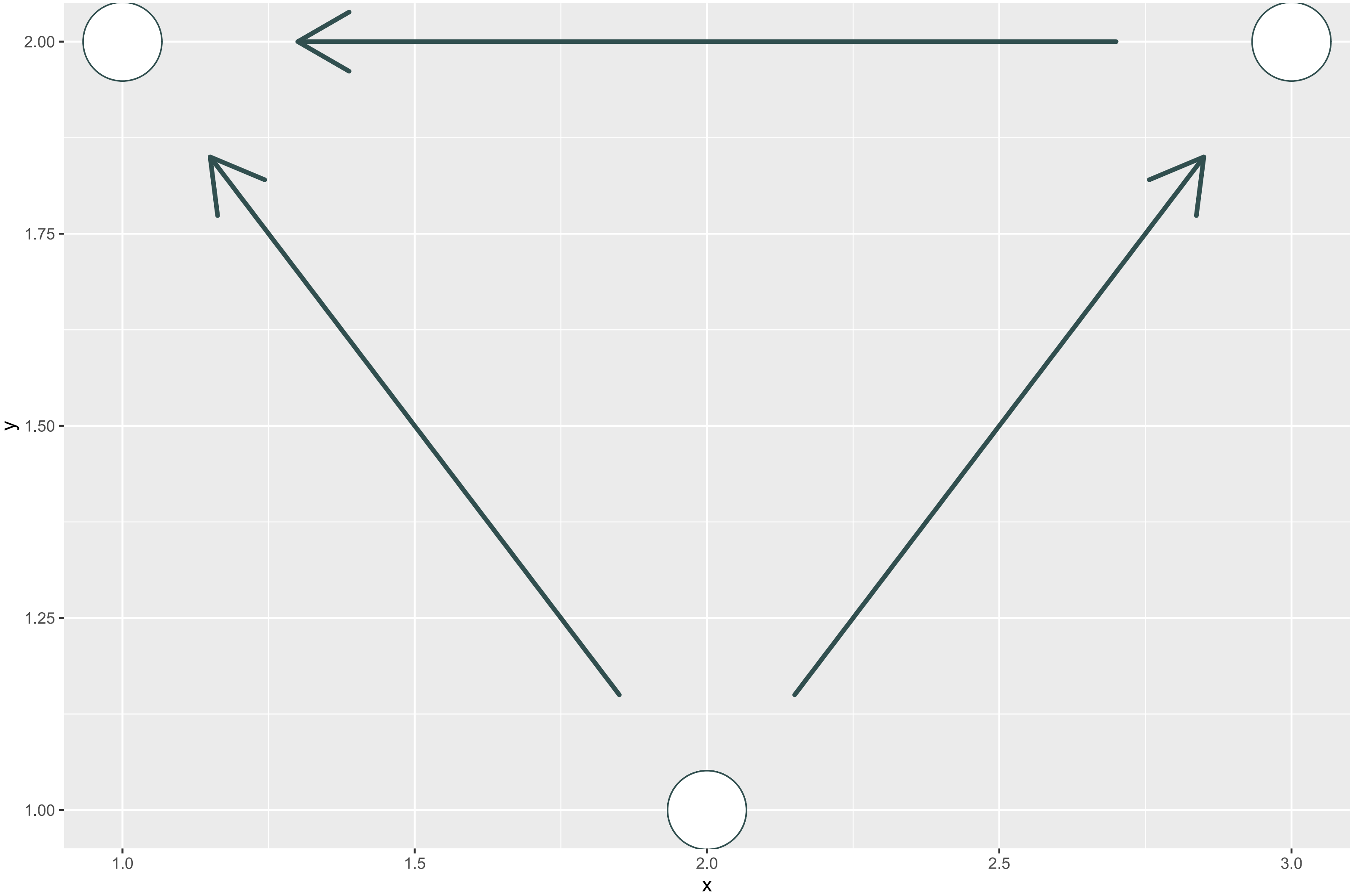


Simple DAG

We then add our causal arrows using segment info.

```
p2 = p1+
geom_curve(
  aes(x = xa, y = ya, xend = xb, yend = yb),
  curvature = 0,
  arrow = arrow(length = unit(0.07, "npc")),
  color = slate,
  size = 1.2,
  lineend = "round"
)
```

Simple DAG

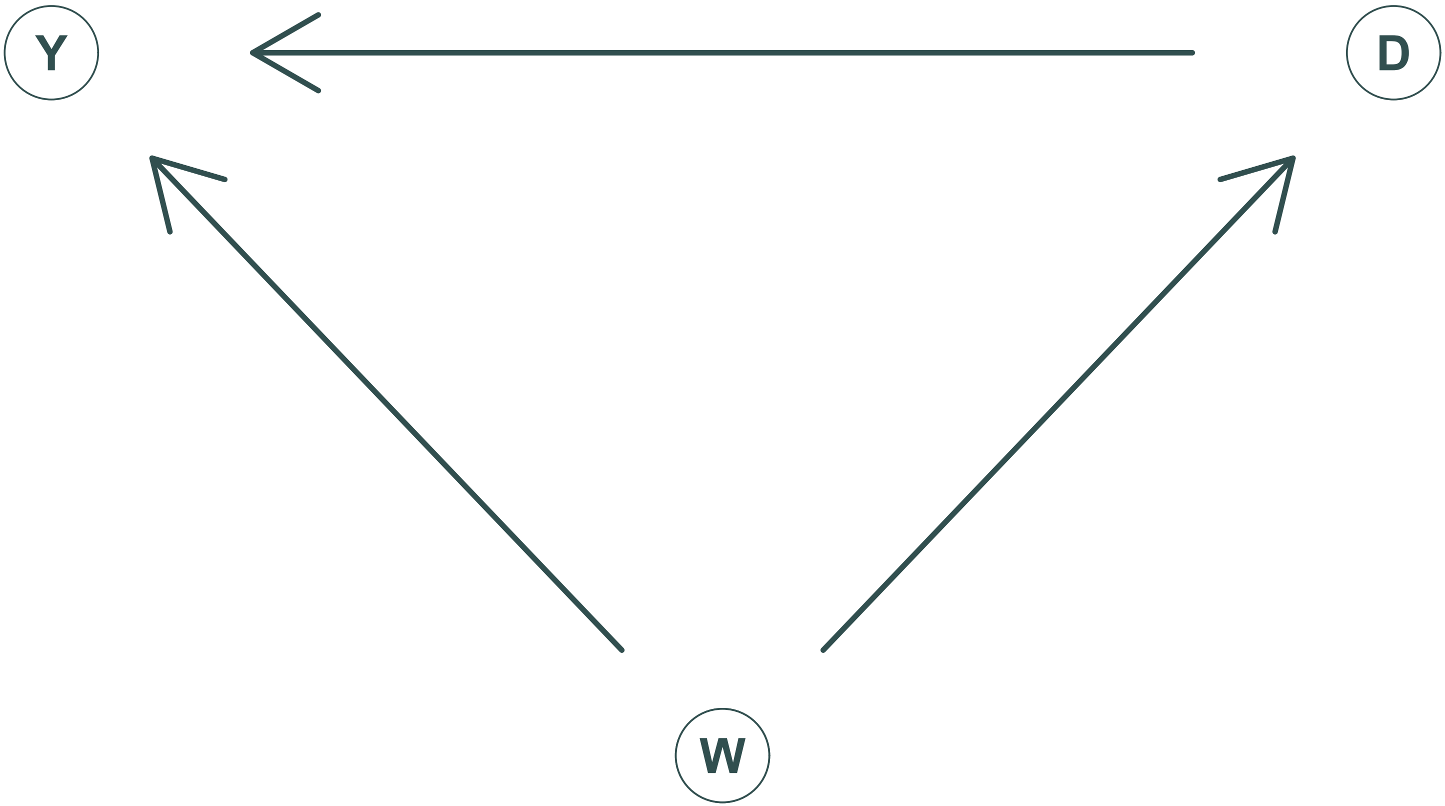


Simple DAG

Lastly, update the nodes with corresponding variables and adjust appearance.

```r
p3 = p2+
geom_text(
  data = dag_dt[,.(name,x,y,xend=x,yend=y)] %>% unique(),
  aes(x = x, y = y, label = name),
  family = "Fira Sans Medium",
  size = 8,
  color = slate,
  fontface = "bold"
) +
theme_void() +
theme(
  legend.position = "none",
) +
coord_cartesian(
  xlim = c(dag_dt[,min(x)]*0.95, dag_dt[,max(x)]*1.05),
  ylim = c(dag_dt[,min(y)]*0.8, dag_dt[,max(y)]*1.1)
)
```

Simple DAG



Chains

Let us denote *associated* changes of A on C through B using a DAG.

```r
bb3_ex = dagify(
  B ~ A, C ~ B,
  coords = tibble(
    name = LETTERS[1:3],
    x = -1:1, y = 0   )
)
# Convert to data.table
bb3_dt = bb3_ex %>% fortify() %>% setDT()
bb3_dt
```

name	x	y	direction	to	xend	yend	circular
A	-1	0	->	B	0	0	FALSE
B	0	0	->	C	1	0	FALSE
C	1	0					FALSE

Chains

Again we'll plot our coordinates for the segments.

```
# Shorten segments
mult = 0.25
bb3_dt[, `:=`(
  xa = x + (xend-x) * (mult),
  ya = y + (yend-y) * (mult),
  xb = x + (xend-x) * (1-mult),
  yb = y + (yend-y) * (1-mult)
)]
```

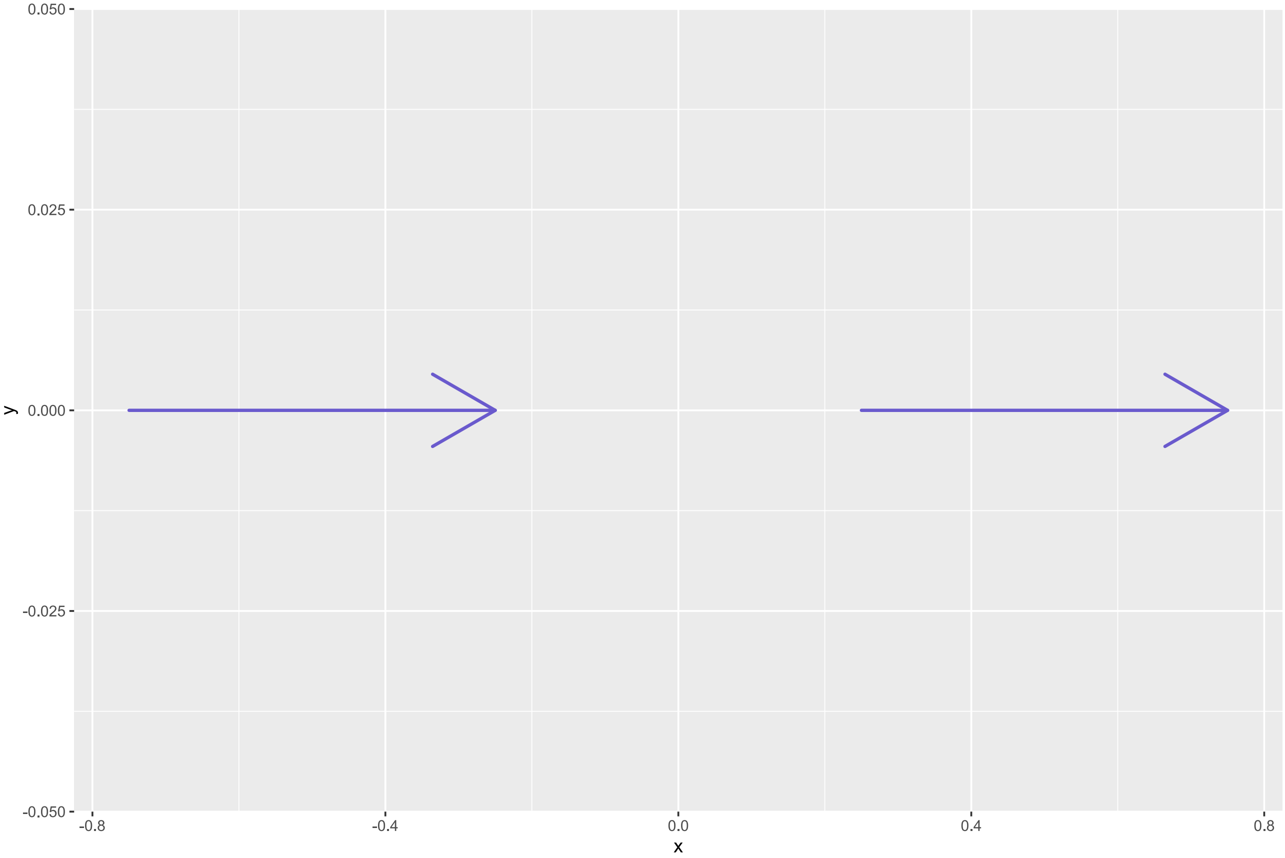
name	x	y	direction	to	xend	yend	circular	xa	ya	xb	yb
A	-1	0	->	B	0	0	FALSE	-0.75	0	-0.25	0
B	0	0	->	C	1	0	FALSE	0.25	0	0.75	0
C	1	0					FALSE				

Chains

We can begin visualizing. We'll start simple with causal arrows.

```r
# Plot the DAG
gg_chain = ggplot(
  data = bb3_dt,
  aes(x = x, y = y)
) +
geom_curve(
  aes(x = xa, y = ya, xend = xb, yend = yb),
  curvature = 0,
  arrow = arrow(length = unit(0.09, "npc")),
  color = purple,
  size = 0.9,
  lineend = "round"
)
```

Chains

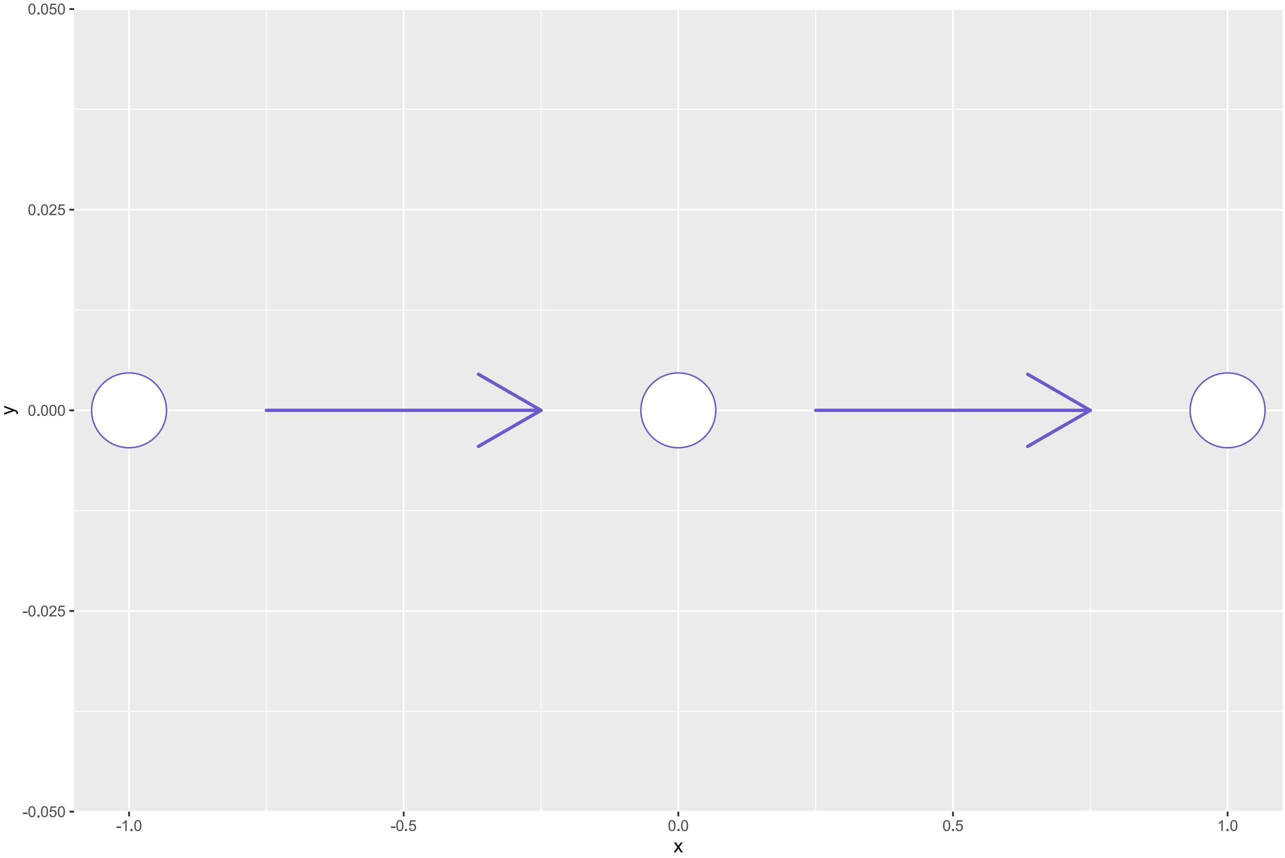


Chains

Adding our nodes.

```
gg_chain = gg_chain +
geom_point(
  size = 20,
  fill = "white",
  color = purple,
  shape = 21,
  stroke = 0.6
)
```

Chains

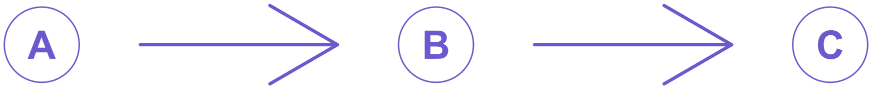


Chains

Node text and cleaning up the appearance.

```r
gg_chain = gg_chain   +
geom_text(
  aes(x = x, y = y, label = name),
  family = "Fira Sans Medium",
  size = 8,
  color = purple,
  fontface = "bold"
) +
theme_void() +
theme(
  legend.position = "none",
) +
coord_cartesian(
  xlim = c(-1.5, 1.5),
  ylim = c(-1, 0.5)
)
```

Chains



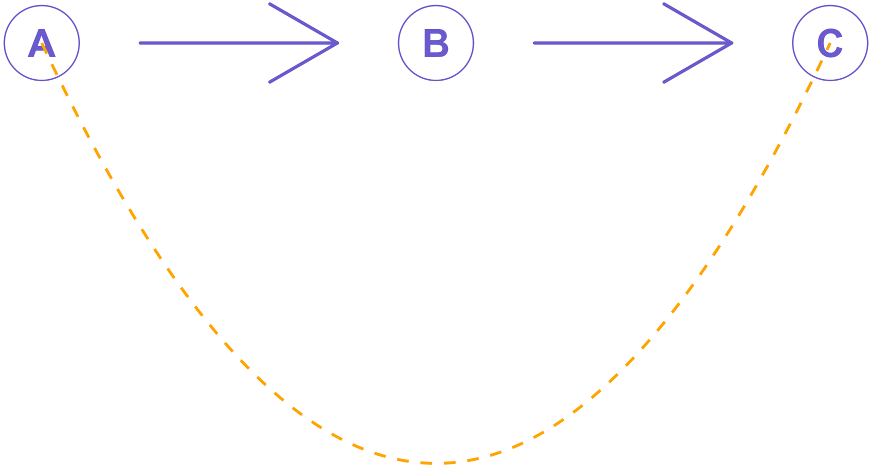
Chains

The chain is easily added to this chart.

```r
curve_dt = tibble(
  x = c(-1, 0, 1),
  y = c(0, -0.8, 0)
) %>% spline(n = 101) %>% as.data.table()

gg_chain +
geom_line(
  data = curve_dt,
  color = orange,
  linetype = "dashed",
  size = 0.8
)
```

Chains



Notice anything off?

Collider Bias

```r
p_load(tidyverse, jtools)

tb <- tibble(
  female = ifelse(runif(10000)>=0.5,1,0),
  ability = rnorm(10000),
  discrimination = female,
  occupation = 1 + 2*ability + 0*female - 2*discrimination + rnorm(10000),
  wage = 1 - 1*discrimination + 1*occupation + 2*ability + rnorm(10000)
)
```

How would we represent these random variables in a DAG?

$$\begin{gathered} F \implies D, \; D \implies O, \; D \implies Y \\ A \implies O, \; A \implies Y, \; O \implies Y \end{gathered}$$

Using an example from Causal Inference: The Mixtape

Collider Bias

$$\begin{gathered} F \implies D,\ D \implies O,\ D \implies Y \\ A \implies O,\ A \implies Y,\ O \implies Y \end{gathered}$$

```
# F, O, D, Y , A   (X range: 0,3 , Y range: 0,5)
bb3_ex = dagify(
  D ~ F, O ~ D, Y ~ D,
  O ~ A, Y ~ A, Y ~ O,
  coords = tibble(
    name = c("F","O", "D", "Y", "A"),
    x =  c(0, 0, 2, 3, 3),
    y = c(3, 0, 5, 3, 0)
  )
)
# Convert to data.table
bb3_dt = bb3_ex %>% fortify() %>% setDT()
# Shorten segments
mult = 0.15
bb3_dt[, `:=`(
  xa = x + (xend-x) * (mult),
  ya = y + (yend-y) * (mult),
  xb = x + (xend-x) * (1-mult),
  yb = y + (yend-y) * (1-mult)
```

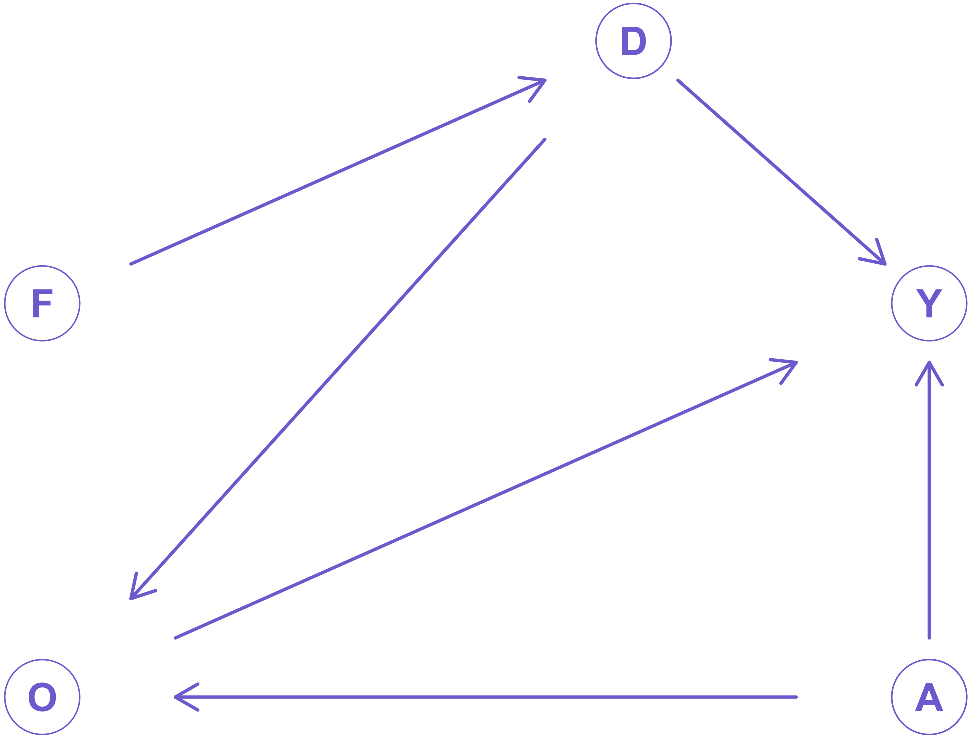
Collider Bias

$$F \implies D, \; D \implies O, \; D \implies Y$$
$$A \implies O, \; A \implies Y, \; O \implies Y$$

name	x	y	direction	to	xend	yend	circular	xa	ya	xb	yb
A	3	0	->	O	0	0	FALSE	2.55	0	0.45	0
A	3	0	->	Y	3	3	FALSE	3	0.45	3	2.55
D	2	5	->	O	0	0	FALSE	1.7	4.25	0.3	0.75
D	2	5	->	Y	3	3	FALSE	2.15	4.7	2.85	3.3
F	0	3	->	D	2	5	FALSE	0.3	3.3	1.7	4.7
O	0	0	->	Y	3	3	FALSE	0.45	0.45	2.55	2.55
Y	3	3					FALSE				

Collider Bias

$$F \implies D,\ D \implies O,\ D \implies Y$$
$$A \implies O,\ A \implies Y,\ O \implies Y$$



Collider Bias

	Biased Unconditional	Biased	Unbiased Conditional
(Intercept)	2.02 ***	0.23 ***	0.99 ***
	(0.06)	(0.02)	(0.02)
female	-3.00 ***	0.59 ***	-0.96 ***
	(0.08)	(0.03)	(0.03)
occupation		1.80 ***	1.01 ***
		(0.01)	(0.01)
ability			1.98 ***
			(0.02)
N	10000	10000	10000
R2	0.11	0.91	0.95

*** p < 0.001; ** p < 0.01; * p < 0.05.