**Causality**

In 20th century statistics classes, it was common to hear the statement: “You can never prove causality.” As a result, researchers published results saying “x is *associated* with y” as a way of circumventing the issue of causality yet implicitly suggesting that the association is causal. As an example from my former discipline, political science, there was an interest in determining how representative democracy works. Do politicians respond to voters, or do voters just update their policy beliefs to line up with the party they’ve always preferred? It turns out that this is a very difficult question to answer, so political scientists interested in publishing choose their language carefully and pronounce that policy “congruence” exists between voters and politicians. The upshot is that there now exists a scholarly literature on “voter-party congruence,” which tells you exactly nothing about how democracy works but allows democracy researchers to get their papers past peer review.

21st century understandings of causality, however, have evolved away from 20th century fatalism to reframe the question as:

* What assumptions need to be met in order to state that an association is causal?
* Under what conditions are those assumptions met?
* Can these assumptions be met even when we can’t perform randomization?

There are two conceptually different approaches to the problem:

* Donald Rubin’s (elaboration on Jerzy Neyman’s) *potential outcomes* framework.
* Judea Pearl’s (elaboration on Sewall Wright’s) *structural causal models* (SCMs).

The former is the dominant approach in applied statistics, but the latter approach can sometimes highlight unexpected results that inform the proper analysis of observational data. Before describing the SCM framework, the next section reviews the potential outcomes framework.

**Potential Outcomes**

Take a binary treatment \(D\_i \in \{0, 1\}\). Represent the outcome received by subject *i* as \(Y\_{iD}\). Then \(Y\_{i0}\) and \(Y\_{i1}\) are the potential outcomes. A subject is *either* \(Y\_{i0}\) *or* \(Y\_{i1}\), we don’t observe both. Yet we want to determine:

\(Y\_{i1} – Y\_{i0}\)

which is the causal effect of the intervention.

Although subjects receive *either* 0 *or* 1, but not the other, we may be able to identify the Average Treatment Effect (ATE).

\(\text{ATE} = \mathbb{E}\left[Y\_{i1} – Y\_{i0}\right]\)

To derive appropriate estimators for the ATE we need to make a few assumptions. Particularly important is that the treatment is independent of potential outcomes, written as:

\(Y\_{i0}, Y\_{i1} \perp\!\!\!\perp D\_i\)

Finding ways to make \(D\_i\) independent is at the heart of the potential outcomes framework. This leads to a few methodologies now commonplace in applied statistics:

1. Randomized experiments by definition make \(D\_i\) independent.
2. Propensity score matching or weighting make the treated and controls look the same on possible confounders so that the only differences must be random error.
3. Regression discontinuity designs where a cut-off on a continuous variable separates treated and control units.
4. Instrumental variables, where compliance is non-random but treatment *assignment* is random.
5. Longitudinal designs that use fixed effects or first differences to remove unit-level confounders affecting the treatment.

The key assumption is \(Y\_{i0}, Y\_{i1} \perp\!\!\!\perp D\_i\), termed *ignorability*. Judea Pearl has criticized how unintuitive the potential outcomes framework makes this assumption. He writes in *The Book of Why* (2018, pg. 279-280):

“Unfortunately, I have yet to find a single person who can explain what ignorability means in a language spoken by those who need to make this assumption or assess its plausibility in a given problem…If you think this sounds circular, I agree with you!”

Instead, Pearl has spent the last twenty years developing a different orientation that builds off his work in the 1990s on Bayesian networks. (It’s not necessary to know how Bayesian networks work to understand this post, but it does help clarify how his thinking on the problem of causality evolved over several decades.) This orientation is known as *structural causal models* (SCMs).

**Structural Causal Models**

SCMs are graphs with nodes, directed edges, and functions mapping exogenous variables to endogenous ones. Denote \(U\) as the set of exogenous variables, \(V\) as the set of endogenous variables, and \(F\) as the set of functions mapping \(U\) to \(V\).

A concrete example is:

\(U = \{X, Y\}\)

\(V = \{Z\}\)

\(F = \{f\_z\}\)

where \(f\_z\) is the function mapping \(X\) and \(Y\) onto \(Z\). This definition implies the following graph:

The arrows represent a generic causal relationship only, the actual function mapping \(X\) and \(Y\) onto \(Z\) can be anything we like.

These types of figures should be familiar to anybody who has previously encountered structural equation models (SEMs) in applied statistics. The primary difference is that SEMs are parametric, typically assuming a linear relationship:

\(Z = b\_0 + b\_1X + b\_2Y\)

but SCMs are defined without committing to a particular functional form.

We get around functional forms by talking about the variables in terms of joint probability functions and taking advantage of well-known rules for converting between joint, conditional, and marginal probabilities. Take the following graph:

Any (acyclic) graph has a joint distribution that is defined by multiplying all conditional probabilities, where conditioning is performed on the direct parent. For example, the joint distribution for the variables in the model is

\(P(X, Y, Z) = P(X)P(Y \vert X)P(Z \vert Y)\)

Understanding the conditional probabilities implied by a model will enable us to generate some rules for determining how causal effects can be identified from observational data. These rules provide surprising and important perspectives on how statistical modeling should be approached.

**Backdoor Paths and Colliders**

“You should control for everything you can. That is, after all, why we do regression.” – One of my methodology professors in the early 2000s.

No, you should not control for everything. In fact, depending on the causal model, some variables should explicitly *not* be controlled for. We’ll start out with when you *should* control for a non-treatment variable. Take the following graph:

We wish to know the effect of \(X\) on \(Z\), but \(Y\) is a common cause. Let’s say we could intervene in the world to set \(X\) at a given value. By doing so, we’d be removing the effect of \(Y\) on \(X\) and would be left with:

We can identify the causal effect by comparing the world in which we have control with the world in which we do not. In both scenarios, the probability that \(Z\) takes on a value is conditioned only on \(Y\) and \(X\), \(P(Z = z \mid Y, X)\), and the probability that \(Y\) takes on a given value is not conditional on anything.

We want to know the effect of \(X\) on \(Z\) if we could intervene on \(X\) and set its value. Pearl introduces the \(do(\cdot)\) operator to signify setting a variable \(X\) to a specific value \(x\).

\(P(Z = z \mid do(X = x))\)

Based on the intervention SCM,

\(P(Z = z \mid do(X = x)) = \sum\_z P(Z = z \mid Y = y, X = x)P(Y = y)\)

This is true because \(P(Z = z \mid do(X = x))\) is what we get after integrating out \(Y\). But we know from comparing the graphs that \(P(Z = z \mid Y = y, X = x)\) and \(P(Y = y)\) are the same in both worlds. Thus, we have all the information we need to calculate a causal effect such as

\(P(Z = z \mid do(X = 1)) – P(Z = z \mid do(X = 0))\)

Take a slightly more complicated model:

There are now two paths from \(X\) to \(Z\):

1. \(X \rightarrow Z\)
2. \(X \leftarrow W \rightarrow Y \rightarrow Z\)

These are read from left to right regardless of the direction of the arrows. However, the arrows identify the second path as a *backdoor path* because there is an arrow leading into \(X\). Backdoor paths are essential for identifying causal effects because they represent spurious associations. Pearl shows that causal effects can be identified if we can *block* the backdoor path. We do this by conditioning on any of the variables that lay on the backdoor path, meaning the conditioning set can be any of the following:

1. \(\{W\}\)
2. \(\{Y\}\)
3. \(\{W, Y\}\)

We don’t necessarily have to control for both, though we can. The key is that, by blocking a backdoor path, we remove the spurious association between the outcome and \(X\). After blocking, we do not necessarily need to control for subsequent variables on the backdoor path.

Now let’s flip the top arrows.

This fundamentally changes the conditioning set, which now *only* contains \(Y\). This occurs because \(W\) is a *collider variable*, which is defined as a variable that lies along a backdoor path with arrows pointing into it from multiple directions. We would write this backdoor path as

\(X \rightarrow W \leftarrow Y \rightarrow Z\).

When we write out the path in this manner, we can immediately identify collider variables as those with arrows pointing to the node from both directions. A collider variable blocks a backdoor path. The counter-intuitive result is that *conditioning on a collider opens the backdoor path*.

To identify the causal effect we need to block all backdoor paths from \(X\) to \(Z\). The *backdoor criterion* can be defined as (Pearl, Glymour, & Powell, 2016, p. 61):

Given an ordered pair of variables \((X,Z)\) in a directed acyclic graph \(G\), a set of variables \(V\) satisfies the backdoor criterion relative to \((X,Z)\) if no node in \(V\) is a descendant of \(X\), and \(V\) blocks every path between \(X\) and \(Z\) that contains an arrow into \(X\).

That is, we identify a set of nodes in \(\{V\}\) to condition on such that:

1. We block all spurious paths from \(X\) to \(Z\).
2. We leave all directed paths from \(X\) to \(Z\) unperturbed.
3. We do not inadvertantly create new spurious paths via conditioning on colliders or their descendants.

**Mediation**

Another example is *mediation*, as in the following figure:

We can get the *direct effect* of \(X\) on \(Z\) if we average over levels of \(M\), which is the standard approach to mediation. But what if we add a variable as follows?:

Now \(M\) is a collider, and we know that conditioning on a collider causes problems. Conditioning on \(M\) opens the path \(X \rightarrow M \leftarrow W \rightarrow Z\), allowing an indirect effect to interfere with the direct effect. But not conditioning on \(M\) leaves the indirect path \(X \rightarrow M \rightarrow Z\) open. How do we deal with this in a manner that allows us to recover the direct effect of \(X\) on \(Z\)? The answer is that we now intervene on *both* \(X\) and \(M\).

\(P(Z=z \mid do(X = x), do(M = m))\).

Intervening and conditioning are not the same thing. Conditioning averages over values of \(M\), intervening sets its value such that there are no longer the arrows \(X \rightarrow M\) and \(W \rightarrow M\).

The *conditional direct effect* is

\(CDE = P(Z=z \mid do(X = x), do(M = m)) – P(Z=z \mid do(X = x^{\prime}), do(M = m))\)

The *conditional* refers to the fact that the direct effect \(X \rightarrow Z\) may differ depending on the value to which the mediator is set.

The \(do(\cdot)\) operator is equivalent to removing an arrow from a graph. Reiterating the model:

There is no path to \(X\), so \(do(X) = x\), and the CDE is

\(CDE = P(Z=z \mid X = x, do(M = m)) – P(Z=z \mid X = x^{\prime}, do(M = m))\).

The last step is to rewrite the \(do(M = m)\) in terms of the observed world. To block the backdoor path \(M \leftarrow W \rightarrow Z\) we need to condition on \(W\). We are left with:

\[\begin{eqnarray}  
CDE = \sum\_i \left[P(Z=z \mid X = x, M = m, W = w) – \\  
P(Z=z \mid X = x^{\prime}, M = m, W = w)\right]P(W = w)  
\end{eqnarray}\]

There is a general result behind this (Pearl, Glymour, & Jewell, 2016, pg. 77):

The CDE of \(X\) on \(Z\) can be identified when a mediation variable \(M\) is present given:

1. There exists a set \(V\_1\) of variables that blocks all backdoor paths from \(M\) to \(Z\).
2. There exists a set \(V\_2\) of variables that blocks all backdoor paths from \(X\) to \(Z\) after deleting all arrows entering \(M\).

The second of these was met automatically given the lack of parents for \(X\).

These general rules make it possible to identify direct causal effects in contexts that were previously intractable, even if the researchers did not realize they were dealing with an intractable problem.

**The daggity R Package**

These models are all very simple, but graphs can be far more complex. Consider the following (adapted from Morgan & Winship, 2015, pg. 135):

A general approach to modeling these diagrams is to employ a tool called *d*-separation, defined as follows (Pearl, Glymour, & Powell, 2016, p. 47):

A path \(p\) is blocked by a set of nodes \(N\) iif:

1. \(p\) contains a chain of nodes \(A \rightarrow B \rightarrow C\) or fork \(A \leftarrow B \rightarrow C\) such that the middle node \(B\) is conditioned on, or
2. \(p\) contains a collider \(A \rightarrow B \leftarrow C\) such that the collision node \(B\) is not conditioned on, nor are any descendents of \(B\) conditioned on.

Fortunately, there is software that can help us algorithmically determine which variables are *d*-separated. The software (and R package) is called dagitty. To use the package, we start by declaring the SCM:

g <- dagitty('dag {

S [pos="0,0"]

T [pos="1,2"]

U [pos="0,4"]

V [pos="2,1"]

W [pos="2,3"]

X [pos="3,0"]

Y [pos="3,4"]

Z [pos="4,2"]

S -> X -> Z

S -> T

T -> V -> X -> Z

T -> V -> Z

T -> W -> Z

U -> Y -> Z

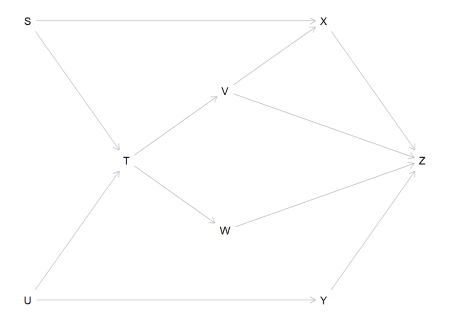
U -> T

Y -> Z

}')

The plot method confirms that it looks good.

plot(g)



We can now make some queries on the graph. For example, what are the paths from \(X\) to \(Z\)?

paths(g, "X", "Z")

## $paths

## [1] "X -> Z" "X <- S -> T -> V -> Z"

## [3] "X <- S -> T -> W -> Z" "X <- S -> T <- U -> Y -> Z"

## [5] "X <- V -> Z" "X <- V <- T -> W -> Z"

## [7] "X <- V <- T <- U -> Y -> Z"

##

## $open

## [1] TRUE TRUE TRUE FALSE TRUE TRUE TRUE

We can quickly see that there are seven paths, six of which are backdoor paths, linking \(X\) to \(Z\). Only the fourth is blocked by the collider at \(T\).

We wish to predict \(Z\) on the basis of \(X\). Using the rules for \(d\)-separation to remove spurious dependencies, what set of variables can we condition on to get the true causal effect of \(X\) on \(Z\)?

adjustmentSets(g, "X", "Z", type = "all") %>%

head(15)

## { S, V }

## { S, T, V }

## { S, U, V }

## { T, U, V }

## { S, T, U, V }

## { S, V, W }

## { S, T, V, W }

## { U, V, W }

## { S, U, V, W }

## { T, U, V, W }

## { S, T, U, V, W }

## { S, V, Y }

## { T, V, Y }

## { S, T, V, Y }

## { S, U, V, Y }

Notice that \(T\) is in some of these sets. If we unblock the path \(X \leftarrow S \rightarrow T \leftarrow U \rightarrow Y \rightarrow Z\), we need to reblock it by conditioning on another variable such as \(U\) or \(Y\).

This is a lot of options. Can we get something simpler?

adjustmentSets(g, "X", "Z", type = "minimal")

## { V, W, Y }

## { T, V, Y }

## { U, V, W }

## { T, U, V }

## { S, V }

Note two important points.

1. We don’t *have* to condition on all possible causes of \(Y\).
2. There are some combinations of variables we should *not* use as adjustors.

We’ll illustrate by generating some data consistent with the model. The SEM package [lavaan](http://lavaan.ugent.be/) makes generating data for simultaneous equations relatively easy.

lavaan\_model <- "Z ~ .8\*X + .6\*V + .6\*W + .6\*Y

X ~ .5\*S + .5\*V

Y ~ .5\*U

V ~ .5\*T

W ~ .5\*T

T ~ .5\*S + .5\*U"

set.seed(12345)

g\_tbl <- simulateData(lavaan\_model, sample.nobs=1000)

This creates a data.frame with 1000 observations. The effects of each exogenous variable on the endogenous variables are set to be non-zero. The code specifies a traditional SEM, meaning that the set of functions \(F\) in the SCM are all linear.

We can verify that our data conform to the model by first specifying the model without the known coefficients.

lavaan\_model <- "Z ~ X + V + W + Y

X ~ S + V

Y ~ U

V ~ T

W ~ T

T ~ S + U"

Next, fit the model using traditional SEM.

lavaan\_fit <- sem(lavaan\_model, data = g\_tbl)

Now look at the coefficients and verify that the path \(X \rightarrow Z\) has a coefficient of approximately 0.8.

parameterEstimates(lavaan\_fit)

## lhs op rhs est se z pvalue ci.lower ci.upper

## 1 Z ~ X 0.801 0.028 28.992 0 0.747 0.856

## 2 Z ~ V 0.602 0.032 18.661 0 0.538 0.665

## 3 Z ~ W 0.574 0.029 19.461 0 0.516 0.632

## 4 Z ~ Y 0.582 0.030 19.568 0 0.523 0.640

## 5 X ~ S 0.556 0.033 16.973 0 0.492 0.620

## 6 X ~ V 0.485 0.028 17.251 0 0.430 0.540

## 7 Y ~ U 0.489 0.031 15.980 0 0.429 0.549

## 8 V ~ T 0.505 0.027 18.699 0 0.452 0.558

## 9 W ~ T 0.489 0.026 18.971 0 0.439 0.540

## 10 T ~ S 0.510 0.030 16.987 0 0.452 0.569

## 11 T ~ U 0.488 0.030 16.038 0 0.428 0.548

## 12 Z ~~ Z 1.028 0.046 22.361 0 0.938 1.118

## 13 X ~~ X 1.052 0.047 22.361 0 0.959 1.144

## 14 Y ~~ Y 0.939 0.042 22.361 0 0.857 1.022

## 15 V ~~ V 1.035 0.046 22.361 0 0.944 1.126

## 16 W ~~ W 0.945 0.042 22.361 0 0.862 1.028

## 17 T ~~ T 0.928 0.041 22.361 0 0.846 1.009

## 18 S ~~ S 1.028 0.000 NA NA 1.028 1.028

## 19 S ~~ U -0.030 0.000 NA NA -0.030 -0.030

## 20 U ~~ U 1.002 0.000 NA NA 1.002 1.002

We want to estimate the effect of \(X\) on \(Z\). What do we get without adjustment?

lm(Z ~ X, data = g\_tbl) %>%

tidy()

## # A tibble: 2 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0939 0.0500 1.88 6.06e- 2

## 2 X 1.19 0.0369 32.2 1.26e-156

That’s NQR, the effect should be .8. What do we get if we also adjust on the collider \(T\)?

lm(Z ~ X + T, data = g\_tbl) %>%

tidy()

## # A tibble: 3 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0841 0.0456 1.85 6.52e- 2

## 2 X 0.974 0.0369 26.4 6.78e-117

## 3 T 0.597 0.0419 14.2 4.68e- 42

What if we condition using the sets dagitty told us to use?

Model 1:

lm(Z ~ X + S + V, data = g\_tbl) %>%

tidy()

## # A tibble: 4 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0675 0.0433 1.56 1.20e- 1

## 2 X 0.807 0.0422 19.1 1.31e-69

## 3 S 0.0997 0.0495 2.01 4.44e- 2

## 4 V 0.776 0.0427 18.2 5.03e-64

Model 2:

lm(Z ~ X + V + W + Y, data = g\_tbl) %>%

tidy()

## # A tibble: 5 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0227 0.0322 0.704 4.81e- 1

## 2 X 0.801 0.0278 28.8 3.81e-133

## 3 V 0.602 0.0322 18.7 4.64e- 67

## 4 W 0.574 0.0298 19.3 1.25e- 70

## 5 Y 0.582 0.0298 19.5 5.87e- 72

Model 3:

lm(Z ~ X + U + V + W, data = g\_tbl) %>%

tidy()

## # A tibble: 5 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0568 0.0364 1.56 1.19e- 1

## 2 X 0.800 0.0316 25.3 1.49e-109

## 3 U 0.337 0.0384 8.77 7.37e- 18

## 4 V 0.585 0.0371 15.8 3.62e- 50

## 5 W 0.557 0.0344 16.2 1.67e- 52

Model 4:

lm(Z ~ X + T + U + V, data = g\_tbl) %>%

tidy()

## # A tibble: 5 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0657 0.0403 1.63 1.03e- 1

## 2 X 0.820 0.0356 23.0 1.63e-94

## 3 T 0.248 0.0427 5.80 9.10e- 9

## 4 U 0.384 0.0442 8.68 1.60e-17

## 5 V 0.580 0.0430 13.5 3.48e-38

We get much closer to the true causal effect estimate whenever we use the conditioning sets suggested by daggity.

**Unobservable or Unmeasurable Variables**

Once again, take our model:

Let’s say that we can’t actually observe \(W\) or \(Y\). An old-school regressionista would say we are SOL. A modern causal-aware practitioner would not.

We can tell dagitty that these variables are unobserved, or *latent*.

g\_unobs <- g

latents(g\_unobs) <- c("W", "Y")

Compare the adjustment sets when we observe all variables (the DAG object we called g) with the adjustment sets after we tell daggity we can’t measure \(W\) or \(Y\).

adjustmentSets(g, "X", "Z", type = "minimal")

## { V, W, Y }

## { T, V, Y }

## { U, V, W }

## { T, U, V }

## { S, V }

adjustmentSets(g\_unobs, "X", "Z", type = "minimal")

## { T, U, V }

## { S, V }

We’re still okay! There is still a set of variables we can control for to recover the causal effect even when some of the variables along the full causal path can’t be measured.

**SEMs, SCMs, and p-hacking**

How do we know our SCM is correct? This raises an important concern.

Pearl writes in the Book of Why that SCMs are unfamiliar to statisticians. Although this *may* be true in their nonparametric form, linear SEMs have been popular ever since the software LISREL was released by a couple of Swedes in 1972.

However, linear SEMs (SCMs with linear functional forms) have been maligned by many statisticians over the last several decades because they have been so thoroughly abused that it’s become hard to take them seriously.

A typical approach:

* The effect of \(X\) on \(Z\) isn’t significant, my dissertation (or publication needed for tenure) is a failure!
* I know, I’ll add a variable \(M\_1\) between \(X\) and \(Z\), maybe there’s a mediated effect!
* Damn, no mediated effect. What if I add \(M\_2\) and \(M\_3\) to the model and keep moving around the directed arrows?
* Hey, something is eventually significant!

In other words, these models are rife with \(p\)-hacking.

A careful analysis of SCMs, however, closes off some of the models we may want to try out of desperation. This is because the conditioning we perform should render certain associations to be independent.

Take the model we just analyzed. As it stands, \(W\) and \(V\) are dependent because they have a common ancestor:

\(V \leftarrow T \rightarrow W\)

By the definition of *d*-separation, we know that conditioning on \(T\) should render \(W\) independent of \(V\). That is,

\(P(W = w \mid V = v) = P(W = w)\).

We can test this with a regression of \(W\) on \(V\) and \(T\). If the model is correct, the association between \(W\) and \(V\) should be zero.

To demonstrate, start by showing an association exists between \(W\) and \(V\).

lm(W ~ V, data = g\_tbl) %>%

tidy() %>%

mutate\_if(is.numeric, funs(round(., 3)))

## Warning: funs() is soft deprecated as of dplyr 0.8.0

## please use list() instead

##

## # Before:

## funs(name = f(.)

##

## # After:

## list(name = ~f(.))

## This warning is displayed once per session.

## # A tibble: 2 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.017 0.035 0.485 0.628

## 2 V 0.258 0.029 8.81 0

If our model is correct, controlling for \(T\) should render this association statistically indistinguishable from zero. Does it?

lm(W ~ V + T, data = g\_tbl) %>%

tidy() %>%

mutate\_if(is.numeric, funs(round(., 3)))

## # A tibble: 3 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.016 0.031 0.522 0.602

## 2 V 0.009 0.03 0.288 0.774

## 3 T 0.485 0.03 16.2 0

In fact, we can get all conditional independencies implied by the model.

impliedConditionalIndependencies(g) %>%

head(20)

## S \_||\_ U

## S \_||\_ V | T

## S \_||\_ W | T

## S \_||\_ Y

## S \_||\_ Z | V, W, X, Y

## S \_||\_ Z | T, V, X, Y

## S \_||\_ Z | U, V, W, X

## S \_||\_ Z | T, U, V, X

## T \_||\_ X | S, V

## T \_||\_ Y | U

## T \_||\_ Z | V, W, X, Y

## T \_||\_ Z | U, V, W, X

## T \_||\_ Z | S, V, W, Y

## T \_||\_ Z | S, U, V, W

## U \_||\_ V | T

## U \_||\_ W | T

## U \_||\_ X | S, V

## U \_||\_ X | S, T

## U \_||\_ Z | V, W, X, Y

## U \_||\_ Z | S, V, W, Y

We generated our data to intentionally be consistent with the model, so testing these conditional independencies will confirm them. When we don’t know if the model is correct, however, we can generate the conditional independencies and check each of them. If they are not correct, our model is wrong.

When \(\{V\}\) is large, the possible set of connections may not all be clearly dictated by theory, and the number of possible combinations of arrows is too large to test via a grid-search. Familiarity with Pearl’s earlier work on Bayesian networks is helpful here, since it led to algorithms for more efficient search rules. These algorithms are nonetheless still very computationally intensive, and there has been very little work testing out their utility in the social sciences.

**Counterfactuals**

Pearl also argues that SCMs, and their implied probabilities, can be used to address seemingly intractable questions. Specifically, they can address unit-specific *counterfactuals*. Whereas interventions, and determining ATEs, can be performed by averaging across a group of cases, specific counterfactuals relate to an individual case.

At first, counterfactuals seem unidentifiable. Think of a court case where there is an assertion that taking a drug caused a person’s death. There are two (potential) outcomes:

1. \(Z\_0\), the outcome when the person did not take the drug, i.e. \(X = 0\).
2. \(Z\_1\), the outcome when the person did take the drug, i.e. \(X = 1\).

The person took the drug and died, so we know \(Z\_1 = 1\) (\(1\) = death, \(0\) = no death). The defense would like to know \(P(Z\_0 \mid X = 1, Y = 1)\). But this seems like nonesense. We want to know the probability of an event under one hypothetical world while conditioning on another world, the one we observed.

The solution relies on establishing an SCM that explicitly includes error terms.

Each of the \(U \in \{UX, UY, UZ\}\) is an individual-specific value. After fitting the model using the observed data, we can get these values for a specific person. We then alter the graph by setting the value of \(X\) or \(Y\) to the counterfactual value and solve for \(Z\) using the error term value identified by the full regression.

In the most simplistic case, we are assuming that each person’s error term is determined exactly by the equations. Pearl’s texts also discuss working with stochastic errors to come up with bounds on possible counterfactuals.

**SCMs and ML**

Pearl (2018) makes the audacious claim that current machine learning models cannot ever assert causality because they cannot deal with interventions, let alone counterfactuals.

A machine learning model takes a set of features \(V = \{v\_1, v\_2, \ldots, v\_k\}\) and finds a function \(f\_z\) mapping this set onto an outcome \(Z\).

Using variations on statistical modeling, this amounts to modeling the joint distribution of all variables. However, using Pearl’s \(do(\cdot)\) operator, a joint distribution *changes* when we intervene on a variable.

For example, if we are given a data set without knowing where it came from, we can fit a regression model using the joint distribution. Yet nothing about the join distribution tells us whether \(X\) is randomized or not.

Causality requires knowing which conditional probabilities are invariant to changes in the structural model. ML is blind to this. ML as currently practiced throws a bunch of stuff into a blender and sees what comes out, akin to 20th century regression modeling that taught us to “control for everything.” This may not matter when we want to predict the presence of a dog, cat, or hot dog in a picture. It *will* matter if we want to:

1. Tell policymakers whether or not to increase the minimum wage.
2. Determine if admissions criteria at a university are racially biased.
3. Find a defendant guilty in a criminal trial.
4. Determine a counterfactual for an individual for whom existing data are not representative.

ML models are akin to the underwear gnome problem:

1. Features.
2. \(\dots\)
3. Prediction!

The black box hides the answer we need if we want to develop effective rules that lead to socially desirable outcomes.

**Limitations of the Pearlian Weltanschauung**

At the same time, Pearl’s dismissal of non-SCM approaches to modeling (potential outcomes, ML) are based on finding specific cases where these approaches fail, but he does not give a sense as to how often they fail. Take, for example, our apparently complicated model:

We can identify the *canonical set* of adjuster variables, which will be valid if any valid set exists.

adjustmentSets(g, "X", "Z", type = "canonical")

## { S, T, U, V, W, Y }

We see that we can in fact “control for everything”.

lm(Z ~ ., data = g\_tbl) %>%

tidy

## # A tibble: 8 x 5

## term estimate std.error statistic p.value

##

## 1 (Intercept) 0.0236 0.0322 0.732 4.64e- 1

## 2 X 0.815 0.0314 26.0 4.54e-114

## 3 Y 0.562 0.0332 16.9 1.39e- 56

## 4 V 0.604 0.0349 17.3 7.55e- 59

## 5 W 0.583 0.0332 17.5 4.11e- 60

## 6 T -0.0466 0.0395 -1.18 2.38e- 1

## 7 S -0.0106 0.0398 -0.266 7.90e- 1

## 8 U 0.0692 0.0395 1.75 8.05e- 2

We didn’t do too bad. The problem, of course, is that there are SCMs that do not have all IVs or features as a proper adjustment set. How bad our conclusions are will depend on how well our representation of reality is.

Indeed, reading Pearl’s (co-authored) introductory textbook *Causality: a Primer*, one can’t help but be struck by how many of the estimators look just like the types of formulas that Rubin and colleagues have developed using the potential frameworks approach. Is a complete re-orientation of applied statistics really going to result in different (and, presumably better) estimators? The jury is still out.

Finally, not all SCMs are identified, especially when stepping away from the world of linearity. Reverse causation plagues observational studies of social behavior, and unless you are satisfied with declaring “congruence”, not even SCMs may save you. At best, given complicated nonlinear and nonrecursive systems of equations, checking the model-implied conditional probability will rule out some models, but certainly not all candidates.