Introduction Causal Inference

Maarten Vonk

August 12, 2021

Contents

1 Introduction to Causal Graphs

Before introduction of the do-operator, it is necessary to introduce the graphical terminology that lie underneath the probabilistic models. These give rise to the

Bayesian networks, which form the basics of causal models. Also, this will help provide intuition to understanding the rules of do-calculus. The assumptions that combine probability theory and graph theory are stated explicitly. Many have been extracted from the Causal Textbook of Brady Neal [3].

1.1 Graph Terminology

A graph $G = (V, E)$ where V is the set of vertices and E the set of edges. A graph can be *directed* when every edge has a direction, *undirected* when no edge has a direction or *partially directed* when some edge have a direction. A graph can contain a *cycle* when there exists a directed path from a node to itself. When there is no such path and the graph is directed, we call this a *directed acyclic graph* or DAG. Finally we distinguish a couple of different directed graph structure: We call the structure $X \to Z \to Y$ a *chain*, $X \leftarrow Z \to Y$ a *fork* and $X \to Z \leftarrow Y$ a *v-structure*. In the last case Z is called the *collider*.

1.2 Bayesian Network

We now extend this graph knowledge to introduce probabilistic models. According to chain rule of probability, we can write the distribution $P(x_1, \dots x_n)$ in terms of its factors:

$$P(x_1, \dots x_n) = P(x_1) \prod_i P(x_i \mid x_{i-1} \dots x_1) \qquad \textit{none}$$

but we can simplify this if we assume the following:

Assumption 1 ***Local Markov Assumption.*** *Given a DAG G, a node X is conditionally independent from all variables except its parents* $\mathrm{pa}_i(G)$ *and children.*

This assumption allows us to write the joint probability in a much more tractable way:

$$P(x_1, \dots x_n) = P(x_1) \prod_i P(x_i \mid \mathrm{pa}_i(G)). \qquad \textit{none}$$

This is called the *Bayesian Network Factorization* and we call P and G *Markov compatible*. Now in order to completely connect graphs to the probability distribution, we need one more assumption that incorporates dependencies:

Assumption 2 ***Adjacency-Faithfulness Assumption.*** *Given a DAG G and suppose that two variables are adjacent in the G. Then they are not independent conditional on any subset of other variables.*

Both assumptions an be combined to obtain the minimality assumption:

Assumption 3 ***Minimality Assumption.*** *Given a DAG G and suppose that distribution P is Markov to G. Then no proper subgraph of G is also Markov to P.*

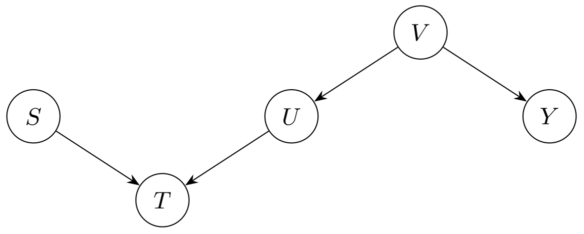

Figure 1: DAG

In other words, the remaining edges imply dependency between variables. Because we are dealing with directed edges, we these dependencies do not have to be bilateral with can be one-sided:

Assumption 4 ***Causal Edges Assumption.*** *Given a DAG G, every directed edge from X to Y in G implies that X has a causal effect on Y.*

1.3 Dependencies in Graph Structures

We create distributions according to a graph that contains a chain, fork and v-structure. In each of these cases we will derive the dependencies and the conditional dependencies of the variables. We include the python code to illustrate the true distribution in this Colab Notebook, but first we share the graph that lie underneath the distributions in Figure 1.

Based on this graph we create distribution where T depends on S and U, U depends on V and Y depends on V:

Listing 1: Distributions

```
N = 5000
s = np.random.uniform(size=N)
v = np.random.normal(size=N)
u = 2. * v + 0.1 * np.random.normal(size=N)
t = 2. * s + u + 0.1 * np.random.normal(size=N)
y = np.random.binomial(1., p=1./(1. + np.exp(-5. * v)))
df = pd.DataFrame({'S': s, 'T': t, 'U': u, 'V': v, 'Y': y})
```

When can check dependence by simple Pearson Correlation test to obtain the following results shown in Figure 3. As we expect according to our distributions: S and T are dependent, but S is independent from all other variables since the collider T blocks all association. The variable Y is dependent on all other variables except S since association goes through the fork structure $U \leftarrow V \rightarrow Y$.

Note that all that we have checked so far is unconditional dependence. We are also interested in dependencies when we condition on variables. The various structures; chain, fork and v-structure behave different under conditional dependence. First the dependence under the chain structure $V \rightarrow U \rightarrow T$ when we do not condition (we know we have dependency, but we check again to show the difference in results) and then we check again when we condition on the mediator node U. We do the same exercise for the fork and the v-structure to obtain the following:

Listing 2: p-values Chain Dependence (rounded to 6 decimals)

```
V and T are unconditionally dependent cause p-value<0,05:   0.0
V and T conditional U are independent cause p-value>0,05:   0.664282
```

Listing 3: p-values Fork Dependence (rounded to 6 decimals)

```
U and Y are unconditionally dependent cause p-value<0,05:   0.0
U and Y conditional V are independent cause p-value>0,05:   0.42431
```

Listing 4: p-values v-structure Dependence (rounded to 6 decimals)

```
S and U are unconditionally independent cause p-value>0,05: 0.312844
S and U conditional T are dependent cause p-value<0,05:       1.5e-05
```

The conclusion of this exercise is the following: in chain and forks we have unconditional dependence between the distant variables, but independence conditional on the middle variable. In v-structure we have unconditional independence between the distant variables, but conditional dependence on the middle, collider variable (or any descendent of the collider variable). This concept gives rise to the definition of *d-separation*.

Definition 1 ***d-separation*** *A path p is blocked by a set of nodes Z if and only if*

1. *p contains a a chain $T \to U \to V$ or fork $T \leftarrow U \to V$ where U is contained in Z.*

2. *p contains a a v-structure $T \to U \leftarrow V$ and the collider node U or any of the descendants of the collider is in Z.*

If Z blocks every path p between two nodes X and Y, then X and Y are d-separated by Z: $X \perp\!\!\!\perp Y \mid Z$.

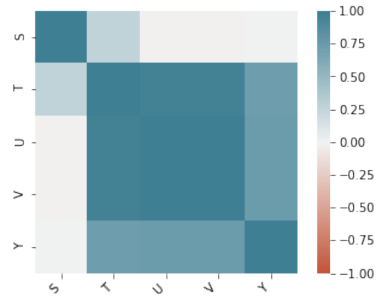

Figure 2: Correlation Plot

Given the graph in Figure 1, S and U are d-separated given the empty set, but S and U are not d-separated given T. The formally state the connection between probability independence and graph independence we have the following Theorem.

Theorem 1 ***Global Markov Assumption*** *Given that P is Markov with respect to G, if X and Y are d-separated in G conditioned on Z, Then X and Y are independent in distribution P conditioned on Z.*

$$X \perp\!\!\!\perp_G Y \mid Z \Rightarrow X \perp\!\!\!\perp_P Y \mid Z.$$

Even though this is framed as a theorem, this can also be framed as an assumption since it is derived from the local Markov assumption.

2 Introduction to do-operator

In order to say something about causal effects, we should first say something about intervention and introduce the do-operator and do-calculus that goes along with that created by Judea Pearl. When we condition on a certain variable, say $T = t$, we are saying we are restricting ourselves to look at the subset of observations where $T = t$. However when we say we intervene on variable $T = t$, we say we enforce t to the entire population. See Figure from Brady Neil's textbook for a clear distinction.

The intervention is specified by the do-operator and is written as $do(T = t)$. We call the distribution following from the intervention the *intervention distribution:* $P(Y \mid do(T = t))$ or simply $P(Y \mid do(t))$. Similarly we call a distribution *observable* when there is no do-operator in the distribution. The relation between these two observable is described in the following definition:

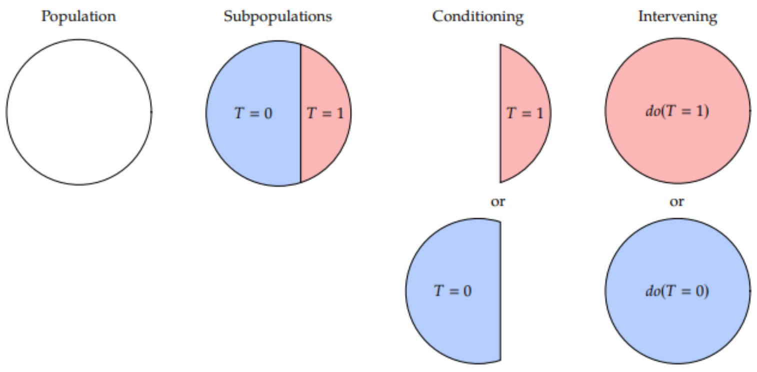

Figure 3: Difference Intervention and Conditioning

Definition 2 *Suppose Q is an expression containing a do-operator. Then Q is identifiable if we can replace the expression containing the do-operator with an expression without a do operator.*

We will get back at how to rewrite the expression containing a do-operator in an expression without do-operator when we introduce the do-calculus. We first have to introduce an additional assumption describing the locality of the intervention:

Assumption 5 *Modularity Assumption. Suppose we intervene on a subset of nodes S, then for all nodes i we have*

1. *If $i \notin S$, then $P(x_i \mid pa_i)$ remains unchanged*
2. *If $i \in S$, then $P(x_i \mid pa_i) = 1$ with x_i being the value set by the intervention and 0 otherwise.*

Note that the modularity assumptions encodes that the intervention is local in the sense that only the conditional distribution of X_i given its causes (parents) is affected by the intervention on X_i. Note that the modularity assumptions and bayesian network factorization allows us to rewrite an intervention distribution in the following way:

Theorem 2 *Assume that P is Markov to G and satisfies modularity. Given a set of intervention nodes S and x being consistent with S, we can write*

$$P(x_1, \ldots, x_n \mid do(S = s)) = \prod_i P(x_i \mid pa_i)$$

and of course $P(x_1, \ldots, x_n \mid do(S = s)) = 0$ if x is not consistent.

Using this new way of writing interventional distributions, we can bolster the intuition behind Figure 3 by formalizing the difference between intervening and conditioning on a variable. For this consider the causal graph in Figure 4. Assume that P is Markov to this graph, then according to the bayesian network assumption we can write the joint distribution.

$$P(y, t, x) = P(x)P(t \mid x)P(y \mid t, x)$$

We we start intervening, we can use Theorem 2 to rewrite the distribution as follows:

$$P(y, x \mid do(t)) = P(x)P(y \mid t, x)$$

and by marginalizing x out we receive

$$P(y \mid do(t)) = \sum_x P(x)P(y \mid t, x)$$

While the conditional probability $P(y \mid t)$ can be reframed as

$$P(y \mid t) = \sum_x P(y, x \mid t) = \sum_x P(x \mid t) P(y \mid t, x).$$

Hence the difference between interventional distribution and conditional distribution is the difference between $P(x)$ and $P(x \mid t)$ as is illustrated in Figure 3.

2.1 Intuition do-calculus

Now we can combine the concept of d-separation together with Markov assumption to arrive at the rules of do-calculus intuitively. For a proof of the rules see [1] Consider a graph depicted in Figure 5. When we condition on Z, this means that X and Y become d-separated. As we have seen in the previous chapter, this implies that X and Y become independent in distribution when P is Markov to graph G. This means:

$$P(y \mid x, w) = P(y \mid w) \text{ if } Y \perp\!\!\!\perp X \mid Z.$$

This can then be generalized to form the first rule of do-calculus.

$$\textbf{Rule 1: } P(y \mid do(x), z, w) = P(y \mid do(x), w) \text{ if } (Y \perp\!\!\!\perp Z \mid X, W)_{G_{\overline{X}}}.$$

Now we are going back to Figure 4. Note that when we condition on X, the only relation there is between T and Y is a direct path or in other words: T become d-separated from Y when we remove all the arrows going from T, the last directed path. In this way intervening is the same as conditioning hence:

$$P(y \mid do(t), x) = P(y \mid t, x) \text{ if } (Y \perp\!\!\!\perp T \mid X)_{G_{\underline{T}}}.$$

which can again be generalized to the second rule

$$\textbf{Rule 2: } P(y|do(z), do(t), x) = P(y|do(z), t, x) \text{ if } (Y \perp\!\!\!\perp T|X, Z)_{G_{\overline{X}\underline{T}}}.$$

Rule 3 can also be intuitively derived, but can be a little harder. Suppose the graph of Figure 6. In this graph T and Y are d-separated when we intervene

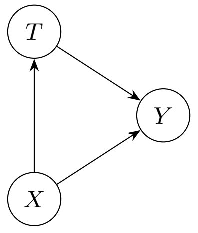

Figure 4: Simple Causal Graph

on T because this would break the arrow from V to T. This d-separation would suggest that intervening on T does not have affect on Y. Also, in this example conditioning on W does still keep this d-separation intact, which would suggest the following:

$$P(y \mid do(t), w) = P(y \mid w) \text{ if } (Y \perp\!\!\!\perp T \mid W)_{G_{\overline{T}}}.$$

However, this is **NOT** true! This can be understood by taking a look at the modification of the graph in Figure 7. In this graph conditioning on W would imply that U and Y become dependent variables because W is the descendent of a collider. Consequently, intervening on T would break that dependence and therefore we cannot just remove the do-operator in this equation. Instead we can check if we can remove the do-operator by checking those nodes in T that are not ancestors of the W we are conditioning on. In this way, we make sure that conditioning on W does not create additional dependencies that we consequently break with the intervention. Hence the correct way of formulating this is:

$$P(y \mid do(t), w) = P(y \mid w) \text{ if } (Y \perp\!\!\!\perp T \mid W)_{G_{\overline{T(W)}}}.$$

where T(W) are the nodes of T that are not ancestors of W. Again, generalizing this yields the last rule:

Rule 3: $P(y \mid do(z), do(t), w) = P(y \mid do(z), w)$ if $(Y \perp\!\!\!\perp T \mid W, Z)_{G_{\overline{T(W)Z}}}$.

The rules of do-calculus shed a new light on Definition 2, meaning the rules are the toolkit needed to rewrite an expression containing a do-operator to an expression not containing a do-operator. In fact, Pearl [**?**] proved that the rules of do-calculus are complete. This means that there is no identifiable expression containing a do-operator who cannot be rewritten to an expression without a do-operator using the three rules of do-calculus

3 Application of the do-calculus and estimation

Now consider the example of Figure 4 again. In this chapter we are going to use the do-calculus to identify the causal estimate and we use rules from do-

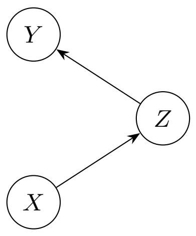

Figure 5: Graph

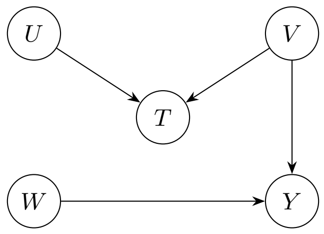

Figure 6: Causal Graph 1 for intuition rule 3

calculus to create two different estimates: the Inverse probability weight (IPW) estimator and the conditional outcome model (COM).

We define the average causal treatment effect ATE for binary treatment T on outcome Y as:

$$\mathbb{E}(Y \mid do(T=1)) - \mathbb{E}(Y \mid do(T=0)).$$

Suppose now we have the following observations: we have N datapoints with covariates X where X is uniformly distributed between $[0,1]$: $x \sim U(0,1)$. Treatment T has binomial distribution $t \sim B(1, \frac{1}{1+e^{-5x}})$. Also suppose we have outcome based on covariate X and treatment T: $y \sim 2x + t + 0.1\mathcal{N}(0,1)$.

3.1 Application do-calculus for identifiability

We can now prove that in our causal graph, Y is identifiable by treatment T. First we marginalize out x by rules of probability:

$$P(Y|do(T=t)) = \sum_z P(y|x, do(t))P(x|do(t)) \tag{1}$$

Now we we observe that $((Y \perp\!\!\!\perp T)|X)_{G_{\underline{T}}}$ and we apply the second rule of do-calculus to rewrite the first term:

$$P(y|x, do(t)) = P(y|x,t). \tag{2}$$

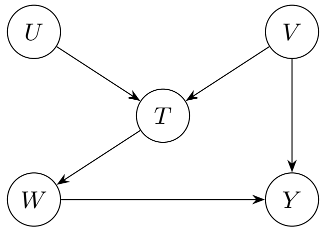

Figure 7: Causal Graph 2 for intuition rule 3

Similarly, we observe that $(X \perp\!\!\!\perp T)_{G_{\overline{T}}}$ to apply the third rule of do-calculus to obtain

$$P(x|do(t)) = P(x). \tag{3}$$

We can rewrite our first expression than to:

$$P(Y|do(T=t)) = \sum_x P(y|x,t)P(x). \tag{4}$$

Since the last expression is 'do-operator free', this is now an application of the do-calculus to prove identifiability. In this case the identifiablility is called the *back-door theorem.* There is also a *front-door theorem.* The derivation for this can be found in this colab notebook. (estimation still under construction)

3.2 Application IPW algorithm for estimation

In this subsection we use the IPW algorithm to solve calculate the Average treatment effect (ATE). The python code is in this Colab Notebook. Note that we want to know

$$P(y|do(T=t)) = \sum_z P(y|t,z)P(z) = \sum_z \frac{P(y|t,z)P(t|z)P(z)}{P(t|z)} = \sum_z \frac{P(y,t,z)}{P(t|z)}$$

where the first identity follows from the do-calculus as shown above. This means we can obtain the intervention distribution $P(y|do(T=t))$ by accounting for the propensity score $P(t|z)$. So now we need to train a model $\hat{e}$ to predict T from Z. This is done by logistic regresion. We apply this models to obtain weights for each data point: $w_i = \frac{1}{P(T_i=t|z_i)}$.

Using this weights we will now sample from the intervention distribution $P(y|do(T=t))$ by sampling from our data given the weights we created to adjust for the covariate Z. Data points with higher weights w_i are more likely to get picked from the data than datapoints with lower weights. Note that there will be duplicates in our new data when we create datasamples of equal length to the original data. Since we had more observations of $T=1$ in our original dataset, observations of $T=0$ received higher weights and therefore the new dataset will be balanced.

Now given the causal relation is not confounded anymore, we can calculate the causal effect. Note for the expectation first that

$$\begin{aligned}\mathbb{E}(Y \mid do(T)) &= \sum_y \sum_z \frac{YP(y,t,z)}{P(t|z)} = \sum_y \sum_z \frac{yP(y)\mathbb{1}_{T=t,Z=z}}{P(t|z)} \\ &= \mathbb{E}(\sum_z \frac{Y\mathbb{1}_{T=t,Z=z}}{P(t|z)}) = \mathbb{E}(\frac{Y\mathbb{1}_{T=t}}{P(t|Z)}).\end{aligned}$$

So we we implement that to calculate the causal effect we have

$$\mathbb{E}(Y \mid do(T=1)) - \mathbb{E}(Y \mid do(T=0)) = \mathbb{E}(\frac{Y\mathbb{1}_{T=1}}{P(t|Z)}) - \mathbb{E}(\frac{Y\mathbb{1}_{T=0}}{1-P(t|Z)})$$

We can then estimate the causal effect by the following estimand:

$$\hat{\tau} = \frac{1}{n}\sum_i (\frac{y_i\mathbb{1}_{T=1}}{\hat{e}(z_i)} - \frac{y_i\mathbb{1}_{T=0}}{1-\hat{e}(z_i)})$$

where $\hat{e}$ is the model used to estimate $P(t|Z)$. Since we have already applied that estimate in our python function, this reduces to

$$\hat{\tau} = \frac{1}{n}\sum_i (y_i\mathbb{1}_{T=1} - y_i\mathbb{1}_{T=0}).$$

3.3 Application COM algorithm for estimation

There is another way of calculating the average causal treatment effect that also makes use of the back-door theorem, but makes use of the conditional expectation method. The details python code is described in this Colab Notebook. The mathematics that lead to this estimator is described in this chapter:

Note that we want to know thanks to the back-door theorem:

$$P(y|do(T=t)) = \sum_z P(y|t,z)P(z) \tag{5}$$

but we are after the causal effect of T on Y, so we are after $\mathbb{E}(Y \mid do(T = 1)) - \mathbb{E}(Y \mid do(T=0))$, so using the backdoor theorem we can rewrite this to:

$$\mathbb{E}(Y \mid do(T=1)) - \mathbb{E}(Y \mid do(T=0)) = \mathbb{E}_Z(\mathbb{E}(Y \mid T=1,Z)) - \mathbb{E}(Y \mid T=0,Z)).$$

So we can use any sort of model $\hat{\mu}$ to estimate the conditional expectation and then approximate the outer expectation by taking the sample mean:

$$\begin{aligned}\mathbb{E}(Y \mid do(T=1)) - \mathbb{E}(Y \mid do(T=0)) &= \mathbb{E}_Z(\mathbb{E}(Y \mid T=1,Z) - \mathbb{E}(Y \mid T=0,Z)) \\ &= \frac{1}{n}\sum_i^n \hat{\mu}(1,z_i) - \hat{\mu}(0,z_i).\end{aligned}$$

Note that both estimation should yield the same results of course.

4 Extraction Causal Graph

So far we have been working with predefined causal effects and apply the do-calculus from there on. One of questions then remaining is how we get the causal graph. This is called *Causal Discovery*. Using the machinery of dependencies of graph structures defined in subsection 1.3 it is possible to extract causal graphs from raw data. There are variants of causal discovery and they all revolve around to what degree the assumptions are satisfied. We start by laying out those assumptions and then explain the algorithms that exists for various variants of causal discovery.

4.1 Assumptions and Definitions

We start by one of the assumptions that is the reverse of the global Markov assumption:

Assumption 6 *Faithfulness*

$$X \perp\!\!\!\perp_G Y \mid Z \Leftarrow X \perp\!\!\!\perp_P Y \mid Z.$$

This means that independence in the probability also implies independence in the graph. Of course, this is necessary when considering the observational data and converting that to a causal graph, but the drawback is that the assumption is much stronger. Some methods also imply another assumption called causal sufficiency:

Assumption 7 *Causal Sufficiency*
All confounders of variables have been taking into account.

Note that this is also a pretty strong assumption. Imagine dealing with observational data, then it is really hard to be sure that all the relevant variables have been taken into account. For this reason we will consider also methods that let go of this assumption. But first we introduce Markov equivalence classes. We go back to graph example of subsection 1.3 but now consider the case where we have only three variables and their observational data and the goal is to retrieve the causal graph structure from here.

Suppose that we have that all variables are unconditionally dependent, but conditional on only one of the variables the others are independent: $A_1 \perp\!\!\!\perp_P A_2 \mid A_3$. Then the results of the independence testing of subsection tell us that it either one of the graphs portrayed by Figure 8.

Because by testing for independence we can never be really sure which graphs it is, we say that all the graphs of Figure 8 belong to the same *Markov Equivalence Class*, because they entail the same conditional dependencies. Since we cannot direct the edges based on independence testing, in this scenario we can only derive the undirected graph or *skeleton* of the graph. The one thing we do know for certain is that we are not dealing with the graph structure as in Figure 9, since we have seen that this one behaves very differently under independence testing. Unlike the other graph, this graph can be directed and then belongs to a different Markov equivalence class. Using independence testing, we can extract the causal graph up to its Markov equivalence class thanks to the following theorem by Pearl [9] and Frydenberg [11]:

Theorem 3 *Markov Equivalence Theorem*

Graphs are Markov equivalent if and only if they have the same skeleton and v-structures.

This means we can extract the partially directed acyclic graph (PDAG) based on independence testing. Note how the faithfulness theorem is central in this process. Two central algorithms to do this are PC and GES. We will discuss PC in detail:

4.2 PC algorithm for Causal Discovery

PC is named after Peter Spirtes and Clark Glymour [6] and is one of the central algorithms for causal discovery. It basically consists of three steps, skeleton identification, v-structure identification and rest-orientation where information from each step is saved to use in the following steps. We apply PC to extract the graph of Figure 10 to see how it works in practice.

$$\begin{tikzcd}
Y1 \arrow[r] & Y2 \arrow[r] & Y3 \\
X1 & X2 \arrow[l] & X3 \arrow[l] \\
Z1 & Z2 \arrow[l] \arrow[r] & Z3
\end{tikzcd}$$

Figure 8: Markov Equivalent Graphs

$$\begin{tikzcd}
W1 \arrow[r] & W2 & W3 \arrow[l]
\end{tikzcd}$$

Figure 9: V-Structure

$$\begin{tikzcd}
X \arrow[dr] & & Y \arrow[dl] \\
& W \arrow[d] & \\
& Z &
\end{tikzcd}$$

Figure 10: To be extracted graph

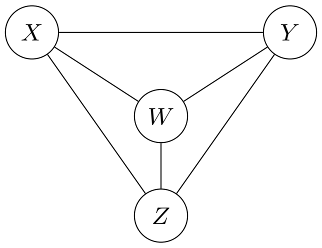

Figure 11: Markov Equivalent Graphs

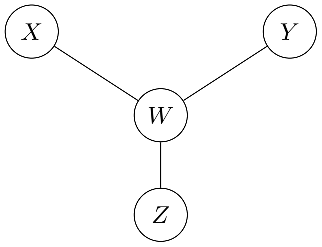

Figure 12: Graph obtained after step 1

4.2.1 Skeleton Identification

The first thing we do is start with the complete graph pointed out in Figure 11 and remove edges based on independence testing. Now we are going to start with unconditional independence testing. In this way, we break the relation between X and Y since they are unconditional independent and we can remove that arrow. Note that the arrow between X and Z remains since they are unconditional dependent. The algorithm then continues with conditional dependence testing of size 1. When we testing dependence between X and X conditional on W we will see that they are indeed independent and we can remove that arrow between the two nodes. The PC algorithm then continues with independence testing of conditional size 1 to remove the arrow between X and Z and to achieve a interesting result for dependence testing between X and Y conditional on W, namely they are dependent. The resulting graph from this procedure is Figure 13. In this case we have the result of the first step, but normally PC will continue.

4.2.2 V-structure Identification

In the previous step we accidentally came across a very interesting result: X and Y are unconditionally independent, but conditionally dependent on W. The only way this is possible is that X, W and Y form a v-structure, so we can now direct the arrows from X and Y to W. This step tests if there are any more v-structures based on the results of step 1 and the remaining conditional independence tests. In our example there are none, so we can continue to step

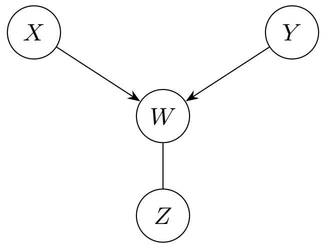

Figure 13: Graph obtained after step 2

3.

4.2.3 Rest-orientation

Note that we have only one edge that is not yet directed: the edge between W and Z. Note that in a result of the PC algorithm, a partially directed acyclic graph, we are also dealing with undirected edges. However in the case of our example, we can direct that edge: if the edges was directed from Z to W, we would have a two more v-structures, but that contradicts our result from step 1 that X and Z are independent conditional W. Hence the arrow should be directed from W to Z and we achieve graph from Figure 10. This step aims to direct all arrows that contradict the v-structure conditional dependence.

4.3 Scoring algorithms and Notes

First we describe how scoring algorithms work and discuss one briefly, then we add some notes about causal discovery and special cases.

4.3.1 Scoring Methods and GES

Besides the PC-algorithm, we also have the GES algorithm from [8], which we will not explain in detail, but needs to be mentioned because it is a score based method. Score based methods basically assume one of the models from a markov equivalence class and fit it to the data. Then it scores the fit based on a scoring system (e.g. BIC score) and check for a model from a slightly different markov equivalence class. When the score from the second model is higher than it continues with the higher score model. The problem that arises is the enormous search space that exists. This is where the GES (Greedy equivalence Search) comes in.

It has a forward and a backwards phase. The goal is to start with an empty model. The forward phase keeps keeps adding edges which improves the score most. Then, when no edges can be added that improve the score, the backwards phase starts with removing edges that improve the score most. When no edge can be removed that improves the score, the algorithm ends.

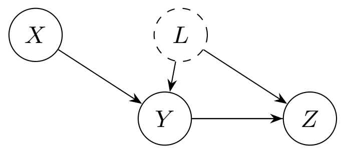

Figure 14: Causal Graph with Larent Variable

4.4 Notes

One very important thing to mention is that many algorithms for causal discovery make use of conditional independence testing and this is very hard and requires many statistical data to achieve usable results [4] . That is one of the main practical problems of causal discovery.

Up to now we have only been able to do causal discovery up to its Markov equivalence class. However, when we assume additional conditions, we are able to add other edge orientation. This is the case when we assume linear non-Gaussian noise or when we assume nonlinear additive noise. We refer to [5] and [7] for the specifics for these cases in detail respectively.

As stated at the beginning of this chapter. There exist algorithms that can also deal with latent variables. We describe the specifics in the following subsection.

5 Causal discovery with latent variables

Because the causal sufficiency algorithm can be pretty strong assumption, there are many researchers that drop this assumption and consider the case where there can be missing confounders or latent variables. One of the most central in this discussion is the FCI algorithm (Fast Causal Inference) [6], which makes use of the CI algorithm(causal inference). We might discuss this algorithm to in the same way as we did with the PC algorithm, but this would call for a vast introduction of more graph-related concepts (induced paths, semi-colliders, definite discriminating path), so we save the details for later.

However, we illustrate the importance of such an algorithm by showing how PC extracts erroneous causal graphs when latent variables are involved: suppose we are dealing with the graph as depicted in Figure 14.

Imagine applying the PC algorithm to the observed variables here. In the first iteration of the first step, the PC algorithm checks for unconditional dependence and finds the complete graph containing X, Y and Z. In the second iteration, it checks for conditional dependence, in this case the dependence of Z and X conditioned on Y. Now, if we were not dealing with the latent variable Y, this would yield an independence, hence we would remove the arrow between X and Y, but because Y functions here as a collider, we receive dependency in the latent variable case. In this way PC would yield a complete graph as in Figure 15, which is false. Recall that the direction of the arrow from X to Y

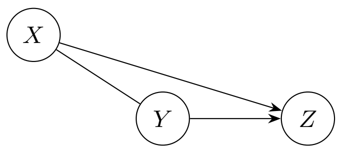

Figure 15: Result PC algorithm

cannot be known.

The FCI algorithm deals with this complication by removing edges in a later phase.

5.1 Better algorithms

Even though the FCI can be seen as the foundation of causal discovery when latent variables are involved, there have been many progressions. Colombo et al [10] have developed a new algorithm called RFCI, because the FCI algorithm becomes computational infeasible for large graphs. Spirtes [12] has also made advances on the FCI algorithm, making it better to perform for small sample sizes. This algorithm is called GFCI.

6 Optimization problem

Having introduced the do-calculus and a simple application, we can start how generalizing this idea brings us to the optimization exercise that lies ahead: consider the following graph in Figure 16 were the treatment and outcome variables have already been specified. The causal effect of the treatment variable can be defined as the causal effect

$$\mathbb{E}(Y \mid do(T=1)) - \mathbb{E}(Y \mid do(T=0))$$

for treatment variable T and outcome variable Y if T is binary. And we define the (marginal) causal effect of treatment T on outcome variable Y when T continuous as:

$$\frac{\mathbb{E}(Y \mid do(t)) - \mathbb{E}(Y \mid do(t-\Delta t))}{\Delta t}.$$

Thanks to the do-calculus, if the outcome is identifiable given the treatment, we can rewrite this expression containing a do-operator to an expression not containing a do-operator. Once we have such an expression, we can use statistical techniques to approximate this expression. In the example of the previous chapter, the objective was obvious, because we have one treatment variable and it was binary. But considering the graph in Figure 16, we are not merely interested in the causal effect of one binary variable on the outcome, but we are interested in knowing what variables to intervene on and what to set as the intervention value in order to yields the most desirable outcome given a maximal number of variables to intervene on. Mathematically, this means we are interested in:

$$\max_{S,t_i} \mathbb{E}(Y \mid \bigcup_{T_i \in S} do(T_i = t_i))$$

where $\mid S \mid \leq n$.

Let for simplicity first consider the case were we are interested in finding the intervention value that maximizes the expected outcome given a certain intervention T:

$$\max_{t} \mathbb{E}(Y \mid do(T = t))$$

.

Then we can consider multiple interventions and find the intervention values that maximize the expected outcome:

$$\max_{t_1,\ldots,t_n} \mathbb{E}(Y \mid do(T_1 = t_1, \ldots, T_n = t_n))$$

.

Now we consider the case were we do not only have to find the intervention value, but also the variable to intervene on, considering at most one variable to intervene on:

$$\max_{i,t_i} \mathbb{E}(Y \mid do(T_i = t_i))$$

Of course, the goal is to find multiple intervention variables and the intervention values that maximizes our objective with the constraint of only n interventions:

$$\max_{S,t_i} \mathbb{E}(Y \mid \bigcup_{T_i \in S} do(T_i = t_i))$$

where $\mid S \mid \leq n$.

References

[1] Pearl, J, (1995) Causal diagrams for empirical research: Rejoinder to Discussions of 'Causal diagrams for empirical research', In *Biometrika, Volume 82, Issue 4* Pages 702–710

[2] Shpitser, I and Pearl, J, (2006) Identification of Joint Interventional Distributions in Recursive SemiMarkovian Causal Models. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2.* Pages 1219–1226

[3] Neal, B (2020) Introduction to Causal Inference. $https://www.bradyneal.com/Introduction_{t}o_{C}ausal_{I}nference - Dec17_{2}020 - Neal.pdf$

[4] Shah, R., Peters, J. (2020). The hardness of conditional independence testing and the generalised covariance measure. *The Annals Of Statistics, 48(3). doi: 10.1214/19-aos1857*

[5] Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. (2006) A Linear Non-Gaussian Acyclic Model for Causal Discovery. In: *Journal of Machine Learning Research 7.72*, pp. 2003–2030

[6] Spirtes, P., Glymour, C., Scheines, R. (2000). Causation, prediction, and search. Cambridge, Mass.: MIT Press.

[7] Kun Zhang and Aapo Hyvärinen. (2009) On the Identifiability of the Post-Nonlinear Causal Model. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence.* UAI '09. Montreal, Quebec, Canada: AUAI Press, 2009, pp. 647–655

[8] Chickering, David Maxwell. (2002) Optimal structure identification with greedy search. In *Jorunal of machine learning research 3*

[9] Thomas Verma and Judea Pearl. (1990) Equivalence and Synthesis of Causal Models. In: *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence. UAI '90. USA: Elsevier Science Inc.*

[10] Colombo, D., Maathuis, M., Kalisch, M., Richardson, T. (2012). Learning high-dimensional directed acyclic graphs with latent and selection variables. In *The Annals Of Statistics, 40(1). doi: 10.1214/11-aos940*

[11] Morten Frydenberg. (1990) The Chain Graph Markov Property. In: *Scandinavian Journal of Statistics 17.4*

[12] JM, O., P, S., J, R. (2021). A Hybrid Causal Search Algorithm for Latent Variable Models. Retrieved 11 August 2021, from https://pubmed.ncbi.nlm.nih.gov/28239434/

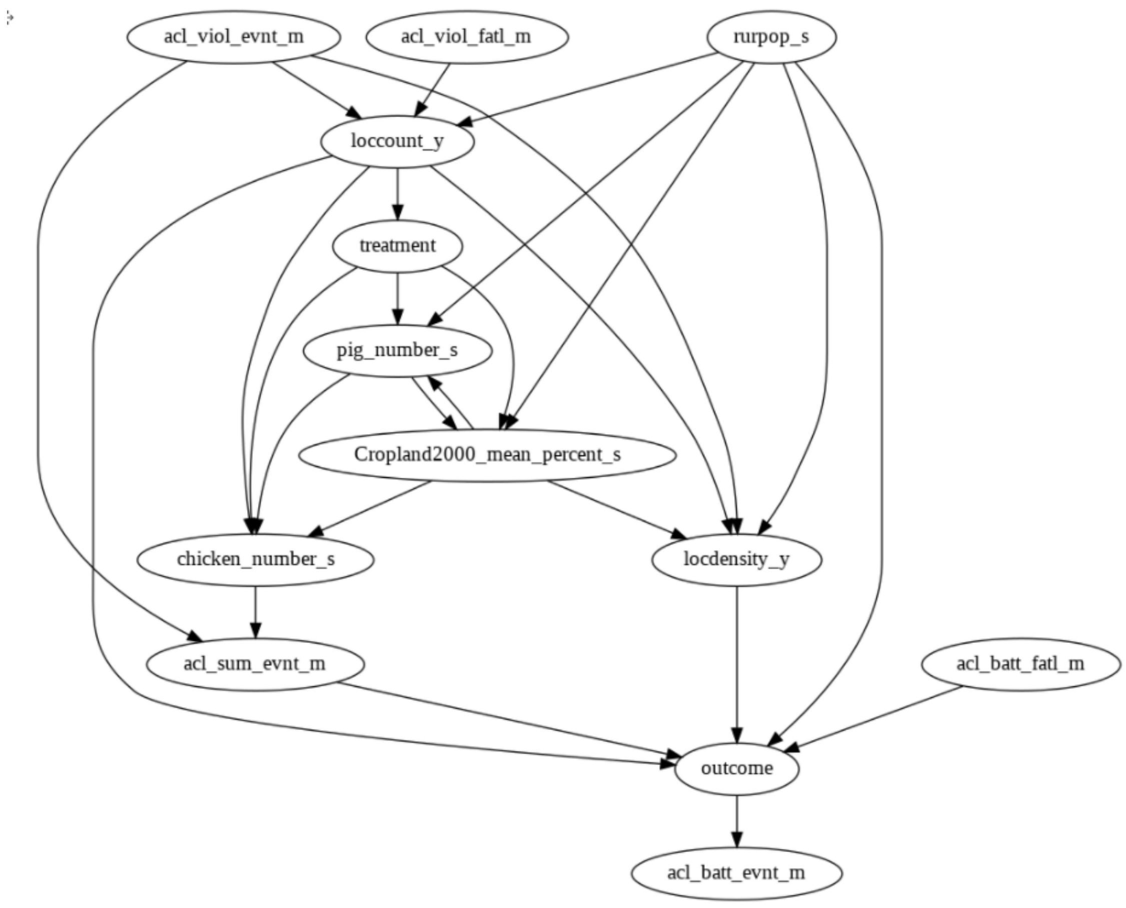

Figure 16: Causal Inference with multiple variables