# Improving and generalizing Bayesian SCM estimation

Anthony Ozerov

2023-05-10

**Team**: Anthony Ozerov

**Mentor**: Justin Zhang

**Topic**: Bounding causal effect / sensitivity analysis + Estimation

## Contents

**1 Introduction** **1**
1.1 Prior work . . . . . 1
1.2 This paper . . . . . 2

**2 SCM preliminaries** **4**
2.1 Canonical SCMs . . . . . 4
2.2 Fully bidirected-connectedness . . . . . 4
2.3 Stacking and pushing down . . . . . 5

**3 Generative models** **7**
3.1 Known identification or partial-identification formulas . . . . . 8
3.2 Unoriented model . . . . . 9
3.3 Oriented model . . . . . 11
3.4 Discrete Function-Sampling model . . . . . 12
3.5 Probabilistic Function-Sampling model . . . . . 13

**4 Submodels and Supermodels** **14**
4.1 Submodels . . . . . 14
4.2 Heterogeneous data . . . . . 15

**5 Sampling** **16**
5.1 Moving away from Gibbs sampling . . . . . 16
5.2 Bringing the posterior closer to the bounds . . . . . 16
5.3 Verifying MCMC convergence to posterior . . . . . 17
5.4 Comparison of Generative Models . . . . . 18

**6 Conclusion** **19**

## 1 Introduction

### 1.1 Prior work

When a causal query on a particular DAG is identifiable from, say, the model's observational distribution, another layer of difficulty arises. In the real world, we typically have finite data, and therefore do not have perfect knowledge of the observational distribution. How do we apply an identification formula to finite data? And what is the resulting uncertainty in our estimate? There are, of course, many methods which solve the estimation problem. One approach is to use Bayesian inference, wherein there are priors on some parameters of the model, posteriors are

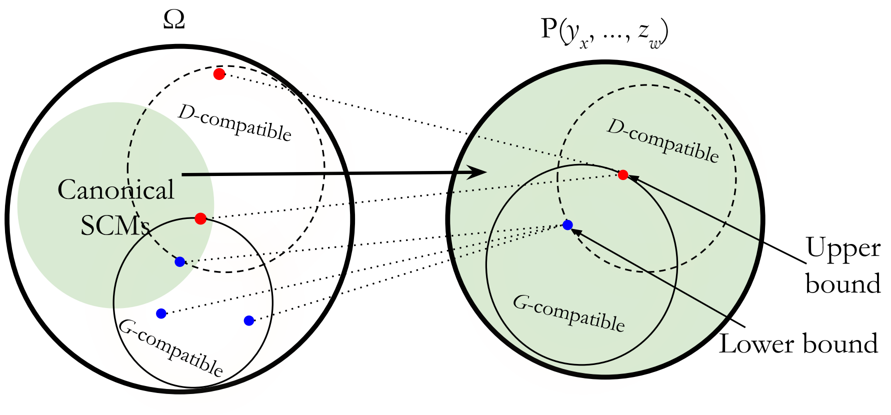


Figure 1: Diagram of the problem and the approach. $\Omega$ is the space of all SCMs, and $P(y_x, \ldots, z_w)$ is a counterfactual quantity $Q$ of interest, with a range of possible values. We are given a causal DAG $G$ and a finite dataset $D$, both of which constrain the set of SCMs and the possible values of the counterfactual quantity (the dashed line for the $D$-compatible set indicates that, due to sampling error, the boundary isn't tight). Different SCMs in $\Omega$ induce different values for $Q$. We would like to move around in the space of SCMs to obtain a lower and upper bound on $Q$. It is hard to move around in $\Omega$, but the space of canonical SCMs, which can induce any value of $Q$, is easy to move around in.

obtained upon observing the data, and a posterior distribution on the estimand is thus obtained [Rub78].

More recent work has applied this approach to the problem of partial identification, where a causal query cannot be identified from the data, but bounds can be placed upon it [CP96; IR97; RER11; SE16; ZTB22]. Understandably, much of this body of work focuses on the standard IV model. There are a few common features of this work:

- MCMC, and specifically the Gibbs sampler, is used to sample from the posterior.
- The convergence of the sampler is either poor or unexplored. One paper puts it rightly that "tuning a Gibbs sampler for the best results tends to be more of an art than a science"[CP96].
- The posterior within the identification bounds is highly sensitive to the prior. It is often interpreted as meaningless, though attempts have been made to reparameterize the model to make it less so [SE16].

There are, in general, many practical problems when using this general Bayesian approach to obtain the bounds.

Previous work has shown that, for partial identification problems with observational and interventional data, it is possible to express a canonical SCM and use a polynomial program to obtain the bounds for the query [ZTB22]. As this is comptuationally expensive, that work used a Bayesian method developed specifically for these canonical SCMs to obtain the bounds.

## 1.2 This paper

Our goals in this paper are to:

1. Investigate when and how the dimensionality of canonical SCMs can be reduced while preserving bounds (Section 2).
2. Develop several Bayesian models to infer the parameters of canonical SCMs (Section 3).
3. Apply these models to subgraphs and generalize them to solve problems in causal data fusion settings with heterogeneous data (Section 4).

4. Study different MCMC algorithms to quickly, reliably, and verifiably sample from the posterior distributions of these models and obtain bounds. (Section 5).

In terms of the long-term objective, we wish to solve problems in causal data fusion (i.e. provide estimates or bounds for arbitrary counterfactual queries when given data). We will generalize to take into account multiple datasets, some of which may:

1. Have stochastic interventions.
2. Have some variables left unmeasured.
3. Come from different populations.
4. Have selection bias.

We have investigated this in Section 4. Because solving such data fusion problems can involve even higher-dimensional distributions and pathological posteriors, this generalization effort is supported by the other threads, including theoretical ones:

- Defining several more efficient generative models for canonical SCMs (Section 3)
- To support and justify these generative models, developing theory about canonical SCMs (Section 2)

And practical/computational ones:

- Replacing Gibbs sampling with samplers that are more effective in high-dimensional problems. (Section 5)
- Using diagnostic metrics to check that the sampler did, in fact, converge to the posterior distribution. (Section 5)
- Reworking the model to make the posterior's high-probability region run right up to the bounds when the query is partially-identifiable. This has been done in previous work [RER11; SE16; ZTB22], though we believe that poor sampling may have caused this to be less effective than possible. (Section 5)

We have done these and report experiments on the IV model below.

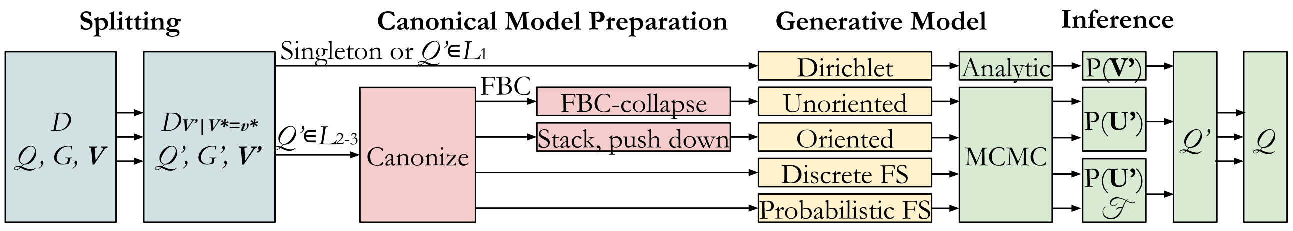


Figure 2: Diagram of the approach taken in this paper. Given data $D$, some query $Q$, a graph $G$, and a set of variables $\mathbf{V}$, we can sometimes, to increase efficiency, split this up into queries on smaller graphs $G'$ and with the data conditioned. Then, for each such smaller query, we want to answer it. If the graph is a singleton or it is a simple $\mathscr{L}_1$ query, we can skip making a canonical model and just use the Dirichlet generative model. Otherwise, we make a canonical model. If the graph is FBC, we collapse it and use the Unoriented generative model. Otherwise, we can use the Oriented generative model (in which case we need to apply the stack and push down operations to the canonical model), or the Discrete Function or Probabilistic Function generative models. Then we do inference, which can be done analytically for the Dirichlet model or using MCMC for the others. This gives us posteriors over $P(\mathbf{V}')$ for the Dirichlet model, $P(\mathbf{U}')$ for the Unoriented and Oriented models, and $P(\mathbf{U}')$ and $\mathscr{F}$ for the Function models. We can then provide a posterior distribution over $Q'$, then put these together to obtain a posterior for $Q$.

# 2 SCM preliminaries

## 2.1 Canonical SCMs

First let's start with a theorem from [ZTB22] on the existence of canonical SCMs, to motivate our discussion.We follow the definition of "canonical SCM" used in [ZTB22].

**Theorem 1.** (Existence of canonical SCMs) For any SCM $M = \langle \mathbf{V}, \mathbf{U}, \mathscr{F}, P(\mathbf{U}) \rangle$, $\exists$ a canonical SCM $N$ s.t.:

1. $\mathscr{G}_M = \mathscr{G}_N$
2. For any set of counterfactual variables $\mathbf{Y}_\mathbf{x}, \ldots, \mathbf{Z}_\mathbf{w}$, $P_M(\mathbf{Y}_\mathbf{x}, \ldots, \mathbf{Z}_\mathbf{w}) = P_N(\mathbf{Y}_\mathbf{x}, \ldots, \mathbf{Z}_\mathbf{w})$

Seeing as this is true, let's formalize an idea of "canonization," which we'll use to turn a pair $\langle G, \mathbf{V} \rangle$ into a set of possible canonical SCMs:

**Definition 1.** (Canonization) Given a pair $\langle G, \mathbf{V} \rangle$ of a causal DAG $G$ and a set of variables $\mathbf{V}$, we $\mathscr{L}_i$-canonize it by creating an equivalence class of canonical SCMs $\langle G, \mathbf{V}, \mathbf{U} \rangle$ where we let $\mathbf{U}$ contain one exogenous variable $U$ for each $U$ implied in $G$, with $|U| = d_U$ defined as:

$$d_U = \prod_{V \in \mathbf{C}(U)} \begin{cases} |\Omega_{\mathrm{Pa}_V} \mapsto \Omega_V| & \text{if } \mathscr{L}_3 \\ |\Omega_{\mathrm{Pa}_V} \times \Omega_V| & \text{if } \mathscr{L}_2 \end{cases}$$

This abstraction helps us think about the set of canonical models compatible with exogenous variables and a causal DAG.

## 2.2 Fully bidirected-connectedness

One thing that we are interested in is simplifications of the method when a $C$-component has a bidirected edge between every pair of variables, therefore leading to no independence constraints.

**Definition 2.** A $C$-component is *fully bidirected-connected* (FBC) if there is a bidirected edge between every variable in the $C$-component.

This definition is actually equivalent to a $C$-component being a "bidirected clique"[XPB22].

**Definition 3.** An SCM $M = \langle \mathbf{V}, \mathbf{U}, \mathscr{F}, P(\mathbf{U}) \rangle$ is *FBC-collapsed* if for every FBC $C$-component $C$ in the graph $\mathscr{G}_M$, there is only one $U \in \mathbf{U}$ in that $C$-component.

**Definition 4.** An SCM $M = \langle \mathbf{V}, \mathbf{U}, \mathscr{F}, P(\mathbf{U}) \rangle$ is *FBC-split* if there is exactly one $U \in \mathbf{U}$ corresponding to each bidirected arrow. (i.e. for each $C$-component $C$, $\forall V_i, V_j, V_k \in C$, if $i \neq j \neq k \neq i$, then $|U_{V_i} \cap U_{V_j}| = 1$ and $|U_{V_i} \cap U_{V_j} \cap U_{V_k}| = 0$)

Because we are interested in bounding, it would be good to show that FBC-collapsed and FBC-split SCMs are equivalent on $\mathscr{L}_3$, so that the bounds obtained for FBC-split SCMs are no wider or narrower than those of FBC-collapsed SCMs. First let's show that any SCM has an equivalent FBC-collapsed SCM:

**Lemma 1.** (FBC-collapsed generality) For any SCM $M = \langle \mathbf{V}, \mathbf{U}, \mathscr{F}, P(\mathbf{U}) \rangle$, there is an equivalent (i.e. matching in $\mathscr{L}_3$) FBC-collapsed SCM.

*Proof.* This is trivial by, for each FBC $C$-component, replacing $U_1, \ldots, U_k$ with a multidimensional $U = \langle U_1, \ldots, U_k \rangle$, with $P(U) = P(U_1) \cdot \ldots \cdot P(U_k)$. □

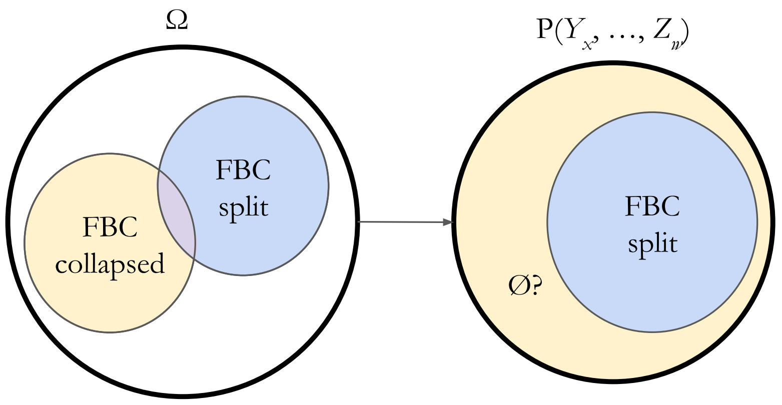


Figure 3: Diagram of the current understanding of the relationship between the sets of FBC-collapsed SCMs and FBC-split SCMs. The overlap region on the left is for SCMs only having $C$-components of size $\leq 2$. It is unknown if FBC-split SCMs can represent any $P(Y_x, \dots, Z_y)$.

Now we'd like to show that any FBC-collapsed SCM has an equivalent FBC-split SCM. Let's restrict our consideration to canonical SCMs.

**Conjecture 1.** (Canonical FBC-collapsed tightness) For any positive canonical FBC-collapsed SCM $N = \langle \mathbf{V}, \mathbf{U}, \mathscr{F}, P(\mathbf{U}) \rangle$, there is an equivalent canonical FBC-split SCM.

This has yet to be proven, but intuition suggests it may hold. To our knowledge [XPB22] effectively uses a version of this for NCMs.

The usefulness of FBC-collapsed canonical models is that they are far easier to sample from. This comes from the fact that, since the same $U$ is pointing to every endogenous variable in $U$'s $C$-component $C$, each of them only has one exogenous variable pointing to it. Since each variable $V$ has $d = |\Omega_{Pa_V} \mapsto \Omega_V|$ possible mechanisms, we can treat each variable as having a $d$-dimensional component of $U$ which directly selects one of its mechanisms. Altogether, this makes $U$ a $\prod_{V \in C(U)} |\Omega_{Pa_V} \mapsto \Omega_V|$-dimensional object which directly selects the mechanism of each of its children.

## 2.3 Stacking and pushing down

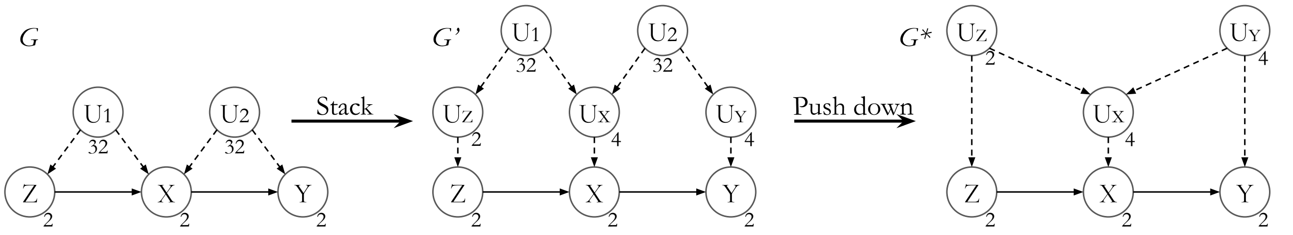


Figure 4: Stack and push-down applied to the double-bow graph. $G$: the double-bow causal DAG. $G'$: DAG after the stack operation. $G^*$: DAG after the push-down operation. The numbers below the nodes indicate their dimension. In this case $Z$, $X$, and $Y$ are binary, which places an upper bound of 32 on the dimensions of $U_1$ and $U_2$. Stacking to obtain $G'$ places some new exogenous variables between the existing ones and the endogenous variables. Pushing down lets us remove some (in this case, all) of the original exogenous variables.

Consider constructing a canonical SCM for the double-bow graph $Z \to X \to Y$ where each variable is binary. According to the SCM construction, if we are interested in $\mathscr{L}_2$ queries, we should make the two exogenous variables each have a dimension of:

$$|Z| \cdot (|Z| \cdot |X|) \cdot (|X| \cdot |Y|) = 2 \cdot (2 \cdot 2) \cdot (2 \cdot 2) = 2 \cdot 4 \cdot 4 = 32$$

It is not obvious how $X$ is determined by $U_1$, $U_2$, and $Z$, and there is no clear parameterization. Hence [ZTB22] uses function sampling to sample among all the ways that $X$ can depend on $U_1$ and $U_2$. As a different approach to get around this issue, we propose the concept of "stacking."

**Definition 5.** For an endogenous variable $V$, the *ordered set $F_V$ of canonical functions* is the set of functions $\{Pa_V \mapsto V\}$ with some ordering applied to it. We denote by $F_V^i$ the $i$th such function.

**Definition 6.** (Stacked equivalence class) Given a causal DAG $G$, set of endogenous variables $\mathbf{V} = \{V_1, \ldots, V_k\}$, and exogenous variables $\mathbf{U}$, we can create a stacked equivalence class $\langle G', \mathbf{V}, \langle \mathbf{U}, \mathbf{U}' \rangle \rangle$ of SCMs via the following algorithm:

1. First we can obtain $G'$ (which is no longer really a causal DAG) and $\mathbf{U}'$.
   (a) Let $U_V$ be the set of exogenous variables pointing to an endogenous variable $V$.
   (b) For each $V_i$, add a new exogenous variable $U_i$ with $|U_i| = |V_i|$.
2. Now let's parameterize the SCMs compatible in this equivalence class (which, since $\mathbf{V}$, $\mathbf{U}$, and $G$ were given, only differ in $P(\mathbf{U})$ and the mechanisms (but not their arguments):
   (a) For each $V_i$, set $f_{V_i}$ to $f(pa_{V_i}, u_i) = F_{V_i}^{u_i}(pa_{V_i})$
   (b) Let $P(\langle \mathbf{U}, \mathbf{U}' \rangle)$be determined by:

$$P(\mathbf{U}' = \mathbf{u}') = P(\mathbf{U} = \mathbf{u}) \prod_i P(U_i = u_i | U_V = u_V)$$

With $P(\mathbf{U} = \mathbf{u})$ and $P(U_i = u_i | U_V = u_V)$ being determined by parameters.

Having exogenous variables pointing to other exogenous variables makes this equivalence class abstraction *not at all* an SCM (which is why the DAG $G'$ is not a causal DAG). Stacked equivalence classes just end up being a useful abstraction for thinking about sets of canonical SCMs. The SCMs in the equivalence class differ *only* in $P(\mathbf{U} = \mathbf{u})$ and $P(U_i = u_i | U_V = u_V)$.

We'd like to know that SCMs in stacked equivalence classes are as expressive (and no more so) than the set of canonical SCMs with the same $G$, $\mathbf{V}$, and $\mathbf{U}$.

**Lemma 2.** Every canonical SCM $M$ has an SCM in its corresponding stacked equivalence class which is equivalent to it.

*Proof.* This is easy to see by setting:

$$P(U_i = u_i | U_v = u_V) = \begin{cases} 1 & \text{if } f_{V_i}|_{U_v = u_V} = F_{V_i}^{u_i} \\ 0 & \text{otherwise} \end{cases}$$
$$P(\mathbf{U}) = P_M(\mathbf{U})$$

□

**Lemma 3.** Going from a graph $G$ to a graph $G'$ does not affect any causal bounds. This is easy to see from the fact that we are just expressing the exogenous variables differently, and keeping all of the independence and dependence relations the same.

Hence the stacked equivalence classes induce the same bounds as canonical SCMs.

If we stack the double-bow graph (see 4), we see that the resulting dimension is quite high, and if we want to make estimates, $P(U_x|U_1, U_2)$ will have $32\times32\times3$ parameters! This is too many for such a simple problem. To reduce this dimension, we create the "push-down" procedure:

**Definition 7.** (Push-down) Given a stacked equivalence class $\langle G', \mathbf{V}, \langle \mathbf{U}, \mathbf{U}' \rangle\rangle$, we can "push it down" to create a pushed-down equivalence class $\langle G^*, \mathbf{V}, \langle \mathbf{U}^*, \mathbf{U}' \rangle\rangle$ as follows:

1. Let $G^* = G'$, $\mathbf{U}^* = \mathbf{U}$
2. For each $U_i \in \mathbf{U}'$:
   (a) If $U_i$ only has one "parent" (in $G^*$) $U_p$, and this parent $U_p$ is in $\mathbf{U}^*$, remove $U_p$ from $\mathbf{U}^*$ and add an edge from $U_i$ to any other children of $U_p$.
3. Return $\langle G^*, \mathbf{V}, \langle \mathbf{U}^*, \mathbf{U}' \rangle\rangle$.

**Theorem 2.** Every SCM in a stacked equivalence class has an equivalent SCM in the corresponding pushed-down equivalence class, and vice-versa.

*Proof.* The endogenous variables in both equivalence classes are modeled as only depending directly on $\mathbf{U}'$. So we just need to show that for any $P(\mathbf{U}')$ induced by an SCM in a stacked equivalence class, we can induce the same in the corresponding pushed-down equivalence class (the reverse is obvious). Let's denote the set of $\mathbf{U}'$ variables which have become ancestral via the push-down procedure as $\mathbf{U}^a$, and the rest $\mathbf{U}^b$. Let's also denote the variables in $\mathbf{U}$ which were removed in the push-down procedure (i.e. $\mathbf{U} - \mathbf{U}^*$) as $\mathbf{U}^\dagger$.

Given $P(\mathbf{U} = \mathbf{u})$ and, for all $U_i \in U'$, $P(U_i = u_i|U_V = u_V)$:

(i) For all $U \in \mathbf{U}^a$, set $P^*(U = u) = P(U = u)$.
(ii) Set $P^*(\mathbf{U}^* = \mathbf{u}^*)$ to $P(\mathbf{U}^* = \mathbf{u}^*)$
(iii) For all $U \in \mathbf{U}^b$, set $P^*(U = u|Pa_U = pa_u) = P(U = u|Pa_U = pa_u)$ (where on both sides the parents of $U_i$ are taken from $G^*$, not $G'$.

First let's note that $P^*(\mathbf{U}^a = \mathbf{u}^a) = P(\mathbf{U}^a = \mathbf{u}^a)$. This is because of (i) combined with the fact that, for any two $U_1, U_2 \in \mathbf{U}^a$, they must be $d$-separated in $G'$ (as they cannot have the same $U \in \mathbf{U}$ as a parent). Note also that $\mathbf{U}^* \perp\!\!\!\perp \mathbf{U}^a$. We can now show that $P^*(\mathbf{U}' = \mathbf{u}') = P(\mathbf{U}' = \mathbf{u}')$.

$$\begin{aligned}
P^*(\mathbf{U}' = \mathbf{u}') &= P^*(\mathbf{U}^a = \mathbf{u}^a)P^*(\mathbf{U}^b = \mathbf{u}^b|\mathbf{U}^a = \mathbf{u}^a) \\
&= P^*(\mathbf{U}^a = \mathbf{u}^a)\sum_{\mathbf{u}^*} P^*(\mathbf{U}^* = \mathbf{u}^*|\mathbf{U}^a = \mathbf{u}^a)P^*(\mathbf{U}^b = \mathbf{u}^b|\mathbf{U}^a = \mathbf{u}^a, \mathbf{U}^* = \mathbf{u}^*) \\
&= P^*(\mathbf{U}^a = \mathbf{u}^a)\sum_{\mathbf{u}^*} P^*(\mathbf{U}^* = \mathbf{u}^*) \prod_{U_i \in U^b} P^*(U_i = u_i|Pa_{u_i} = pa_{u_i}) \\
&= P(\mathbf{U}^a = \mathbf{u}^a)\sum_{\mathbf{u}^*} P(\mathbf{U}^* = \mathbf{u}^*) \prod_{U_i \in U^b} P(U_i = u_i|Pa_{u_i} = pa_{u_i}) \\
&= P(\mathbf{U}^a = \mathbf{u}^a)P(\mathbf{U}^b = \mathbf{u}^b|\mathbf{U}^a = \mathbf{u}^a) \\
&= P(\mathbf{U}' = \mathbf{u}')
\end{aligned}$$

□

## 3 Generative models

We introduce increasingly complex generative Bayesian models for the simplest case, when data is only observational. These models explicitly model the data as generated by random variables with specified distributions, making it clear how they can be implemented in a probabilistic programming language. The idea is that once we specify such a process, we can infer the parameters of the model, hence in practice inferring the parameters of a canonical SCM. This is effectively doing what Pearl calls *Abduction* [Pea09]. Then by Pearl's *Action* and *Prediction* we can answer any counterfactual query.

These models have three inputs: a causal DAG $G$, a set of endogenous variables $\mathbf{V}$, and a set of observational data $D$. We think of $D$ as being count data like the following:

| $x$ | $y$ | $z$ | $D(x, y, z)$ |
|---|---|---|---|
| 0 | 0 | 0 | 30 |
| 0 | 0 | 1 | 9 |
| 0 | 1 | 0 | 42 |
| 0 | 1 | 1 | 23 |
| 1 | 0 | 0 | 71 |
| 1 | 0 | 1 | 6 |
| 1 | 1 | 0 | 8 |
| 1 | 1 | 1 | 35 |

Which has a dimension of $\prod_{V \in \mathbf{V}} |V|$. Because we view our data as count data, there is actually no need to ever iterate over each data point (i.e. "sample 5 had $x = 0, y = 1, z = 1$"). We will also never need to infer the $U$'s for a particular data point. This will improve the complexity of our methods by a factor of $O(n)$ over previous work.

| Model | Variables | PCH Level | Inference | $C$-components |
|---|---|---|---|---|
| Dirichlet | $P(\mathbf{V})$ | 1 | Analytic | Fully-connected |
| Unoriented | $P(\mathbf{U})$ | 3 | MCMC | FBC |
| Oriented | $P(\mathbf{U})$ | 2 or 3 | MCMC | Arbitrary |
| Discrete FS | $P(\mathbf{U})$, $\mathscr{F}$ | 2 or 3 | MCMC | Arbitrary |
| Probabilistic FS | $P(\mathbf{U})$, $\mathscr{F}$ | 2 or 3 | MCMC | Arbitrary |

Table 1: The five main models presented. The Unoriented and Oriented count models each have two variations: exogenous-count and endogenous-count. When the PCH level of a model is "2 or 3", that indicates that, depending on the dimensions chosen for the exogenous variables, they can be limited to only being able to make $\mathscr{L}_2$ inferences.

## 3.1 Known identification or partial-identification formulas

If the queries are identifiable or partially-identifiable (from, let's say, observational data) by a known formula, then clearly one could apply said formula to data and obtain a point estimate. If the data is finite, then the point estimate will not be exact. But what is the uncertainty? We could compute the standard error of the estimates and whatnot, but these are typically done under assumptions of normality. One question a domain scientist might be interested in is, for example, the covariance structure of the estimates for the lower and upper bound when a query is partially identified.

Fortunately, when there is an identification or partial-identification formula available, we can sometimes construct a very simple Bayesian model to provide posterior estimates for the query. We can do this when every $C$-component in the assumed causal graph is fully-connected by either directed or bidirected arrows, which is often—but certainly not always—the case (Note: this is less strict than FBC). This is because, when $C$-components are fully-connected, it becomes much easier to model the independences encoded by the graph without modeling the exogenous variables.

For every $C$-component $C_i$ of $\mathbf{V}$, let $d_i$ be the product of the dimensions of the variables in $C_i$. We then model $P(C_i)$ as:

$$P(C_i|pa_{C_i}) \sim \text{Dirichlet}(\alpha_{C_i}^{(1)}, \dots, \alpha_{C_i}^{(d_i)})$$

We then have that:

$$P(\mathbf{V} = \mathbf{v}) = \prod_i P(C_i = c_i|Pa_{C_i} = pa_{C_i})$$

And finally our $n$ data points, when counted, come from the Multinomial distribution:

$$D_i \sim \text{Multinomial}(n, P(\mathbf{V}))$$

**Lemma 4.** This model will have a prior probability $> 0$ for any $P(\mathbf{V})$ compatible with the graph, and a prior probability of zero for others.

*Proof.* Any $P(\mathbf{V})$ compatible with the graph can be decomposed as:

$$P(\mathbf{V} = \mathbf{v}) = \prod_i P(C_i = c_i|Pa_{C_i} = pa_{C_i}),$$

and since in the model $P(C_i|pa_{C_i}) \sim \text{Dirichlet}(\alpha_{C_i}^{(1)}, \dots, \alpha_{C_i}^{(d_i)})$, any compatible value of $P(\mathbf{V})$ will have nonzero probability.

Any $P(\mathbf{V})$ not compatible with the graph must have a dependence not allowed by the graph, say $X \not\perp\!\!\!\perp Y|Z$. Since this is not allowed by the graph, $X$ and $Y$ must be in different $C$-components (as we assumed they are fully-connected). Also, since this is not allowed by the graph, this implies that $C_X \perp\!\!\!\perp C_Y|Z$, which means that there is no path from $C_X$ to $C_Y$ which isn't blocked by $Z$. This leads to two options:

1. There is a path, but it is blocked by $Z$.
2. There is no open path.

In either case, clearly, $C_X \perp\!\!\!\perp C_Y|Pa_{C_X}$ or $C_X \perp\!\!\!\perp C_Y|Pa_{C_Y}$. So because of the model formulation as $P(C_i|pa_{C_i})$, the product of the $P(C_i|Pa_{c_i})$ random variables cannot possibly generate a $P(\mathbf{V})$ where $C_X \not\perp\!\!\!\perp C_Y|Z$. $\square$

This model can be complicated further to account for the data fusion problems described above, but only under certain conditions which may warrant further research. Rather than pursuing this model further, we can make our model more general by explicitly modeling the exogenous variables.

## 3.2 Unoriented model

As we saw in Section 2, a canonical model which only has FBC $C$-components can be rewritten as an FBC-collapsed model. For such a model, we can present the simplest possible model for making $\mathscr{L}_2$ and $\mathscr{L}_3$ inferences from finite observational data.

This model is generalized later. We view our data as coming from a canonical SCM with, for all $U \in \mathbf{U}$:

$$d_U = \prod_{V \in \mathbf{C}(U)} |\Omega_{\text{Pa}_V} \mapsto \Omega_V|,$$

where $\Omega_X$ is the domain of variable(s) $X$, and $\mathbf{C}(U)$ is the set of variables in the $c$-component containing $U$. $U$ takes on as many different values as there are possible assignments of its

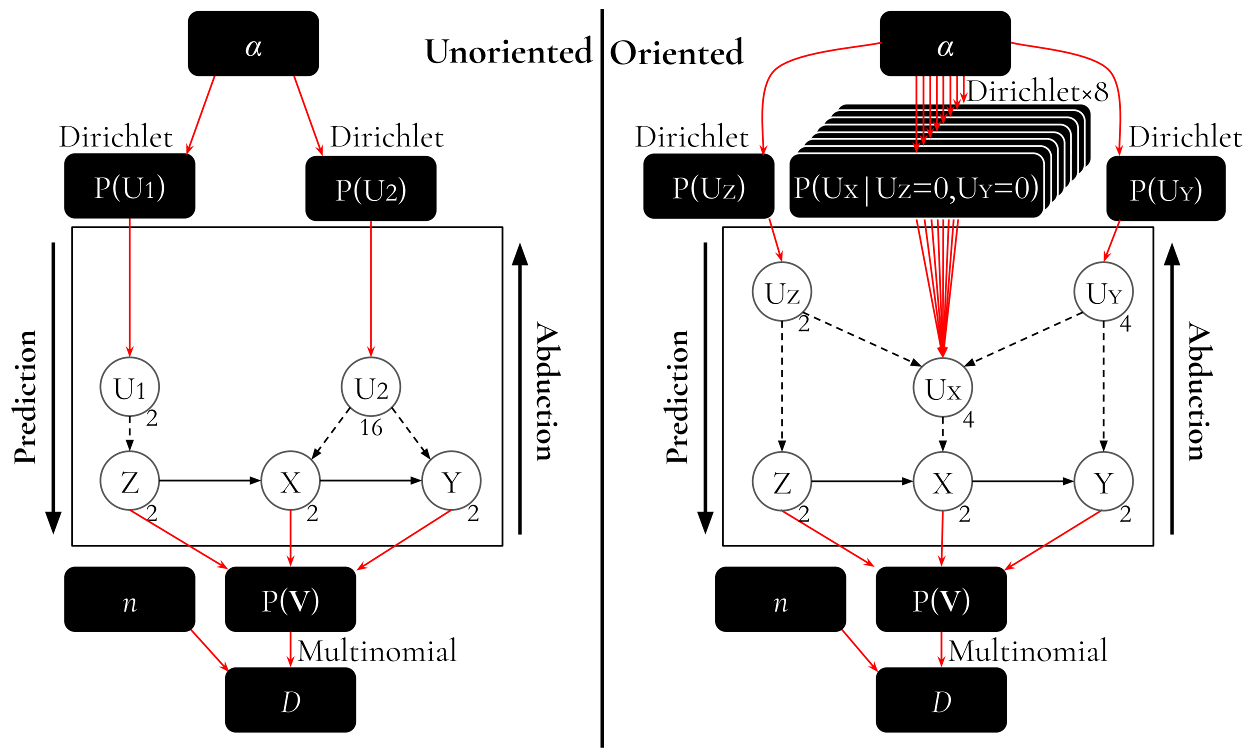


Figure 5: Left: The unoriented model applied to the standard IV graph. Right: the oriented model applied to the double-bow graph. These diagrams show how Bayesian generative models are placed around canonical SCMs. In this case, both models are inferring $P(\mathbf{U})$ from the data $D$, which is abduction. The SCM is used to deterministically go from $P(\mathbf{U})$ to $P(\mathbf{V})$, which is prediction. Both models are shown in their endogenous-count form.

children and each children's parents. In turn, for each $U$, we need a prior distribution over the possible values of $p_1, \ldots, p_{d_U}$, and we take, for some $\alpha$:

$$(p_1^{(U)}, \ldots, p_{d_U}^{(U)}) \sim \text{Dirichlet}(\alpha, \ldots, \alpha)$$

Here, as in the models that follow, we've assumed not only that all $d_U$ Dirichlet parameters for $U$ are the same, but that they are set the same for *all* of the exogenous $U \in \mathbf{U}$. Of course, neither of these necessarily has to be the case when specifying these models (and in some cases it might be advantageous to have different Dirichlet parameters), but for the simplicity of the discussion in this paper we will have only one $\alpha$.

We then have that:

$$P(\mathbf{U} = \mathbf{u}) = \prod_{U \in \mathbf{U}} P(U = u) = \prod_{U \in \mathbf{U}} p_u^{(U)}$$

How do the $\mathbf{U}$'s connect to the $\mathbf{V}$'s? Because of our FBC assumption, we can construct an equivalent canonical FBC-collapsed SCM where the $U$ for each $C$-component $C$ is thought of as $\langle U_{V_1}, \ldots, U_{V_k} \rangle$ for all $V_i \in C$, with each $U_{V_i}$ having a dimension of $|\Omega_{\text{Pa}_{V_i}} \mapsto \Omega_{V_i}|$. Hence $U_{V_i}$, which we think of as a component of $U$, trivially selects one of the possible functions from $V_i$'s parents to its domain. Having done this, it is then easy to compute $\mathbf{V}(\mathbf{u})$, the setting of $\mathbf{V}$ given a setting of the exogenous variables, for any $\mathbf{u}$.

Now there are two possible options to connect our model to the data $D$. In both cases, we represent our data as count data, and not as individual samples.

1. **Endogenous-count**.

For all possible realizations $\mathbf{v}$ of $\mathbf{V}$:

$$P(\mathbf{V}=\mathbf{v}) = \sum_{U\in\mathbf{U}}\sum_{u=1}^{d_U} P(\mathbf{U}=\mathbf{u})\mathbf{1}[\mathbf{V}(\mathbf{u})=\mathbf{v}]$$

And finally, our model generates the data $D$:

$$D \sim \text{Multinomial}(n, P(\mathbf{V})),$$

where $n$ is the number of observations. This last step may seem obvious, but we intend to be very clear about the data-generating process.

2. **Exogenous-count**.
Here, we instead treat our unobserved variables as count data $D_U$:

$$D_U \sim \text{Multinomial}(n, P(\mathbf{U})),$$

where $n$ is the number of observations. We then sum them up to obtain our observed data $D$:

$$D(\mathbf{V}=\mathbf{v}) = \sum_{\mathbf{u}} D_U(\mathbf{U}=\mathbf{u})\mathbf{1}[\mathbf{V}(\mathbf{u})=\mathbf{v}].$$

It is easy to see how these two alternatives are probabilistically equivalent. The only difference is that, in the exogenous-count model, rather than generating data from probabilities at the level of the endogenous variables, we can generate exogenous variables then compute the endogenous variables. But as, in the canonical SCM, the product of the dimensionalities of the exogenous variables will be at least that of the endogenous variables, $D_U$ will be a very high-dimensional object that may be more expensive to sample from. So through the rest of this paper we will be sticking to the endogenous-count versions.

## 3.3 Oriented model

Given a graph $G$ and endogenous variables $\mathbf{V}$, create a canonical model with, for each $U \in \mathbf{U}$:

$$d_U = \prod_{V\in\mathbf{C}(U)} \begin{cases} |\Omega_{\text{Pa}_V} \mapsto \Omega_V| & \text{if } \mathscr{L}_3 \\ |\Omega_{\text{Pa}_V} \times \Omega_V| & \text{if } \mathscr{L}_2 \end{cases},$$

Now stack and push down to obtain a pushed-down equivalence class $\langle G^*, \mathbf{V}, \langle \mathbf{U}^*, \mathbf{U}' \rangle\rangle$. Denote the set of $\mathbf{U}$'s which is ancestral in the graph $\mathbf{U}_a$ and all others as $\mathbf{U}_b$. For every $U \in \mathbf{U}_a$, specify:

$$(p_1^{(U)}, \ldots, p_{d_U}^{(U)}) \sim \text{Dirichlet}(\alpha, \ldots, \alpha),$$

and let $P(U=u) = p_u^{(U)}$. Now for every $U \in \mathbf{U}_b$, let its parents in $G$ be $\mathbf{U}_p$ and specify:

$$(p_1^{(U|\mathbf{U}_p=\mathbf{u}_p)}, \ldots, p_{d_U}^{(U|\mathbf{U}_p=\mathbf{u}_p)}) \sim \text{Dirichlet}(\alpha, \ldots, \alpha),$$

and let $P(U=u|\mathbf{U}_p=\mathbf{u}_p) = p_u^{(\mathbf{U}_p=\mathbf{u}_p)}$.

Recall that only the variables in $\mathbf{U}'$ are needed to determine $\mathbf{V}$, so we only need to speak of $P(\mathbf{U}')$. Because of how the pushed-down equivalence class is defined, $\mathbf{U}_b \subseteq \mathbf{U}'$. So we only need to consider the variables in $\mathbf{U}_b$ and $\mathbf{U}_a \cap \mathbf{U}' = \mathbf{U}'_a$, and can marginalize out $\mathbf{U}_a \cap \mathbf{U}^* = \mathbf{U}^*_a$. We get that:

$$P(\mathbf{U}'=\mathbf{u}') = \left(\prod_{U\in\mathbf{U}'_a} P(U=u'_a)\right) \sum_{\mathbf{u}^*_a\in\mathbf{U}^*_a} \left(\prod_{U\in\mathbf{U}^*_a} P(U=u^*_a)\right) \left(\prod_{U\in\mathbf{U}_b} P(U=u_b|Pa_U=pa_U)\right)$$

How do the $\mathbf{U}$'s connect to the $\mathbf{V}$'s? Because of the way we specified the pushed-down equivalence class, each $U_V \in \mathbf{U}'$ has a dimension of $|\Omega_{\mathrm{Pa}_V} \mapsto \Omega_V|$. Hence $U_V$ trivially selects one of the possible functions from $V$'s parents to its domain. Having done this, it is then easy to compute $\mathbf{V}(\mathbf{u}')$, the setting of $\mathbf{V}$ given a setting of the exogenous variables in $\mathbf{U}'$, for any $\mathbf{u}'$.

Now the rest of the steps are the same as in the unoriented model. Again, there are two possible options to connect our model to the data $D$: Endogenous-count and Exogenous-count. Here we'll finish with Endogenous-count. For all possible realizations $\mathbf{v}$ of $\mathbf{V}$:

$$P(\mathbf{V}=\mathbf{v}) = \sum_{U\in\mathbf{U}'}\sum_{u=1}^{d_U} P(\mathbf{U}'=\mathbf{u}')\mathbf{1}[\mathbf{V}(\mathbf{u}')=\mathbf{v}]$$

And finally, our model generates the data $D$:

$$D \sim \text{Multinomial}(n, P(\mathbf{V})),$$

where $n$ is the number of observations.

## 3.4 Discrete Function-Sampling model

We can also translate the model proposed in [ZTB22] into a clearer data-generating process and reduce the complexity of its sampling method by a factor of $O(n)$, where $n$ is the number of samples. This yields our Discrete Function-Sampling (Discrete FS) model.

We view our data as coming from a canonical SCM, and specify, for all $U \in \mathbf{U}$:

$$d_U = \prod_{V\in\mathbf{C}(U)} \begin{cases} |\Omega_{\mathrm{PA}_V} \mapsto \Omega_V| & \text{if } \mathscr{L}_3 \\ |\Omega_{\mathrm{PA}_V} \times \Omega_V| & \text{if } \mathscr{L}_2 \end{cases},$$

where $\Omega_X$ is the domain of variable(s) $X$, and $C(U)$ is the set of variables in the $c$-component containing $U$. The way the dimension is determined depends on whether our query is $\mathscr{L}_2$ or $\mathscr{L}_3$ [ZTB22]. For $\mathscr{L}_3$, $U$ effectively takes on as many different values as there are possible combinations of its children's mechanisms, and for $\mathscr{L}_3$ $U$ takes on as many different values as there are possible assignments of its children and each children's parents.

In turn, for each $U$, we need a prior distribution over the possible values of $p_1, \dots, p_{d_U}$, and we take, for some $\alpha$:

$$(p_1^{(U)}, \dots, p_{d_U}^{(U)}) \sim \text{Dirichlet}(\alpha, \dots, \alpha)$$

We then have that:

$$P(\mathbf{U}=\mathbf{u}) = \prod_{U\in\mathbf{U}} P(U=u) = \prod_{U\in\mathbf{U}} p_u^{(U)}$$

So far, we have largely kept to Unoriented model. Now, we include the aspect of [ZTB22] which the Unoriented model does not: sampling the functions $\boldsymbol{\mu}$. For every $V \in \mathbf{V}$, we can think of its mechanism $f_V$ as a value $\mu_V^{pa_V,u_V} \in V$ for each value of $pa_V$ and $u_V$. In the Discrete FS model, each of these is chosen uniformly from $V$'s domain:

$$\mu_V^{(pa_V,u_V)} \sim \text{Categorical}(1/|V|, \dots, 1/|V|).$$

Now that we have these mechanisms, $\mathbf{V}_{\boldsymbol{\mu}}(\mathbf{u})$ is easy to compute, as we can just pass in any $V$'s parents and the exogenous variables into $\mu_V$ to obtain the value of $V$. So for all possible realizations $\mathbf{v}$ of $\mathbf{V}$:

$$P(\mathbf{V}=\mathbf{v}) = \sum_{U\in\mathbf{U}}\sum_{u=1}^{d_U} P(\mathbf{U}=u)\textstyle\prod_{V\in\mathbf{V}}\mathbf{1}[\mu_V^{(pa_V,u_V)}=v]$$

And finally, our model generates the data $D$:

$$D \sim \text{Multinomial}(n, P(\mathbf{V})),$$

where $n$ is the number of observations.

In terms of dimensionality, for something like the IV model this Bayesian model is much larger than the Oriented model above, and will not be nearly as efficient to sample from. Moreover, since there is a large group of discrete latent variables $\boldsymbol{\mu}$, standard efficient MCMC methods like Metropolis-Hastings or Hamiltonian Monte Carlo won't work,[1] and different sampling methods like Gibbs sampling must be used. Regardless, this model represents a direct improvement over that presented in [ZTB22] as it is effectively the same, but treats the data as count data to make sampling constant rather than linear in the number of data points.

## 3.5 Probabilistic Function-Sampling model

The main problem with the Discrete FS model presented is that there is a large set of discrete latent variables $\mu$ which makes sampling from it impossible using any method that relies on making small jumps in the parameter space. We can, however, relax the determinism of the sampled mechanisms $\mu$ to make the model continuous. This yields the Probabilistic Function-Sampling (Probabilistic FS) model.

Just as in the Discrete FS model, we view our data as coming from a canonical SCM, and specify, for all $U \in \mathbf{U}$:

$$d_U = \prod_{V \in \mathbf{C}(U)} \begin{cases} |\Omega_{\text{PA}_V} \mapsto \Omega_V| & \text{if } \mathscr{L}_3 \\ |\Omega_{\text{PA}_V} \times \Omega_V| & \text{if } \mathscr{L}_2 \end{cases},$$

where $\Omega_X$ is the domain of variable(s) $X$, and $C(U)$ is the set of variables in the $c$-component containing $U$.

In turn, for each $U$, we need a prior distribution over the possible values of $p_1, \dots, p_{d_U}$, and we take, for some $\alpha$:

$$(p_1^{(U)}, \dots, p_{d_U}^{(U)}) \sim \text{Dirichlet}(\alpha, \dots, \alpha)$$

We then have that:

$$P(\mathbf{U} = \mathbf{u}) = \prod_{U \in \mathbf{U}} P(U = u) = \prod_{U \in \mathbf{U}} p_u^{(U)}$$

Now, the way that we will define $\boldsymbol{\mu}$ slightly differently.

$$\mu_V^{(pa_V, u_V)} \sim \text{Dirichlet}(\epsilon, \dots, \epsilon),$$

where $\epsilon$ is a small positive number and the dimension of $\mu_V^{(pa_V, u_V)}$ is $|V|$. Notice that, rather than being a single value, $\mu_V^{(pa_V, u_V)}$ is now a probability distribution. Treating $\mu_V$ as the mechanism of variable $V$, we see that it now no longer completely deterministic, but for any setting of its parents and exogenous variables returns a probability distribution over its domain. This will make the problem easier for MCMC samplers, which typically do not perform well when latent variables are discrete. Since the mechanism is nondeterministic and a probability vector, let's denote $\mu_V^{(pa_V, u_V)}(v)$ as the probability that $V = v$ when its inputs have value $(pa_V, u_V)$. Then, for all possible realizations $\mathbf{v}$ of $\mathbf{V}$:

$$P(\mathbf{V} = \mathbf{v}) = \sum_{U \in \mathbf{U}} \sum_{u=1}^{d_U} P(\mathbf{U} = u) \prod\nolimits_{V \in \mathbf{V}} \mu_V^{(pa_V, u_V)}(v)$$

[1]These sample by making small steps in the parameter space, which doesn't exactly work when part of the parameter space is discrete.

Finally, this model generates our data $D$ according to:

$$D \sim \text{Multinomial}(n, P(\mathbf{V})),$$

where $n$ is the number of observations.

## 4 Submodels and Supermodels

Now that we've stated these generating processes, we can discuss applying them to smaller submodels. We also provide a sketch of how to apply them to heterogeneous data by building on the original causal DAG to create more complicated models which take into account all of the data sources at once.

### 4.1 Submodels

Now that we've discussed these generative models, let's discuss how to reduce their dimensionality. Ideally, in a large SCM, we do not want to have to infer hundreds upon hundreds of parameters. Fortunately, we can actually do inference on every single $C$-component independently (or on partitions of the set of $C$-components, if we'd like). This is because if $U_1$ and $U_2$ are in different $C$-components, then $U_1 \perp\!\!\!\perp U_2 | \mathbf{V}$. In some cases, in a big graph, we will only actually need to do $\mathscr{L}_2$ or $\mathscr{L}_3$ inference on one $C$-component! Providing an accounting of when this is the case is outside of the scope of this paper, but we can at least state how to do inference after splitting the model into different components.

So, how do we actually do this? Suppose we have a submodel with endogenous variables $\mathbf{A}$ with variables $\mathbf{B}$ as parents, with no bidirected edges between them (which is the case if e.g. $A$ is a $C$-component). Let's denote by $D_{\mathbf{A}}|\mathbf{B} = \mathbf{b}$ the count data for $\mathbf{A}$ restricted to when $\mathbf{B} = \mathbf{b}$. Now, in any of the models above, rather than simply generating $D$ according to:

$$D \sim \text{Multinomial}(n, P(\mathbf{V})),$$

we generate $D_{\mathbf{A}}|\mathbf{B} = \mathbf{b}$ according to:

$$D_{\mathbf{A}}|\mathbf{B} = \mathbf{b} \sim \text{Multinomial}(n_{\mathbf{B}=\mathbf{b}}, P(\mathbf{A}|\mathbf{B} = \mathbf{b}))$$

and we include this for *all* levels of $\mathbf{b}$ in our generative model. How does now conditioning on $\mathbf{B} = \mathbf{b}$ affect the rest of the generative model? The precise details depend on which is used, but, as a sketch, we can still produce whatever abstraction (FBC-collapsed model, pushed-down equivalence class, or regular canonical model) for the whole graph $G$, then, when stating $P(\mathbf{A} = \mathbf{a}|\mathbf{B} = \mathbf{b})$, we just restrict our consideration to $\mathbf{A}$ while feeding in $\mathbf{B}$ into the mechanisms for $\mathbf{A}$. This was hand-wavy, but for the Oriented model, for example, we just get:

$$P(\mathbf{A} = \mathbf{a}|\mathbf{B} = \mathbf{b}) = \sum_{U \in \mathbf{U}'_{\mathbf{A}}} \sum_{u=1}^{d_U} P(\mathbf{U}' = \mathbf{u}')\mathbf{1}[\mathbf{A}(\mathbf{u}', \mathbf{b}) = \mathbf{a}]$$

Notice that we only consider the exogenous variables $\mathbf{U}'_A$ which directly plug into $\mathbf{A}$, as all others that would usually affect $\mathbf{A}$ are blocked by $\mathbf{B}$, and we feed in $\mathbf{b}$ in lieu of these ignored exogenous variables.

## 4.2 Heterogeneous data

*Note: this section was originally written for the midterm report, and the author's understanding has advanced significantly since. This section is to be updated.*

Suppose we have multiple different datasets (observational and interventional (possibly stochastic), from different populations, partially observed, and with a selection bias). How do we represent this? Let's think within the Unoriented Count model.

Suppose we have have datasets $D_1, \ldots, D_k$. $D_i$ comes from population $\pi_i$, has the intervention $\sigma_i$ (which may be stochastic), has variables $\mathbf{M}_i$ missing, and has the selection bias $\mathbf{S}_i = \mathbf{s}_i$. Let $\mathbf{A}_i = \mathbf{V}_i \backslash (\mathbf{M}_i \cup \mathbf{S}_i)$. Then each dataset is count data of the form $D_i = \left( \bigwedge_{\mathbf{a}} \#(\mathbf{A}_i = \mathbf{a}) = N_{\mathbf{a}} \right)$.

Now, starting with the set of canonical SCM exogenous variables $\mathbf{U}$ we for all $i \in 2, \ldots, n$, we add a new $U$ with children $V$ to the set of exogenous variables whenever $\pi_i$ differs from $\pi_1$ on at least one of the mechanisms in $V$ (equivalently, in a canonical SCM, this is represented by the exogenous variable just having a different distribution in that population). If $\pi_j$ has already been processed ($j < i$), and we know it has the same mechanism as $\pi_i$, then we don't add a redundant exogenous variable for $\pi_i$.[2]

Denote by $\mathbf{U}_\pi$ the subset of exogenous variables corresponding to population $\pi$. Now, just as before, for all $U \in \mathbf{U}$.

$$d_U = \prod_{V \in \mathbf{C}(U)} |\Omega_{\mathrm{Pa}_V} \mapsto \Omega_V|,$$

where $\Omega$ and $C(U)$ are defined as before. Again:

$$(p_1^{(U)}, \ldots, p_{d_U}^{(U)}) \sim \text{Dirichlet}(\alpha, \ldots, \alpha).$$

Now is where things will start to differ again. For dataset $D_i$, we can model its missing variables $\mathbf{M}_i$. For each $M_i \in \mathbf{M}_i$:

$$M_i = \mathcal{F}_{M_i}(\mathrm{pa}_{M_i}, U_{M_i}).$$

So $M_i$ is just a function of random variables and therefore itself a random variable. We can do this because, even though $M$ is not observed in the dataset, we still know its dimension and hence its mechanisms (as we are using a canonical SCM).

Now on to specifying the probability of the observed data. Let $\mathbf{A} = \mathbf{V} \backslash (\mathbf{M}_i \cup \mathbf{S}_i)$:

$$\begin{aligned} P(\mathbf{A}_{\sigma_i} = \mathbf{a} | \mathbf{S}_i = \mathbf{s}) &= \frac{P(\mathbf{A}_{\sigma_i} = \mathbf{a}, \mathbf{S}_i = \mathbf{s})}{P(\mathbf{S}_i = \mathbf{s})} \\ &= \frac{\sum_{U \in \mathbf{U}_{\pi_i}} \sum_{u=1}^{d_u} P(\mathbf{U}_{\pi_i} = \mathbf{u}) \sum_{\mathbf{m}} \mathbf{1}[\mathbf{V}_{\sigma_i}(\mathbf{u}) = (\mathbf{a}, \mathbf{s}, \mathbf{m})]}{\sum_{U \in \mathbf{U}_{\pi_i}} \sum_{u=1}^{d_u} P(\mathbf{U}_{\pi_i} = \mathbf{u}) \sum_{\mathbf{m}, \mathbf{a}} \mathbf{1}[\mathbf{V}_{\sigma_i}(\mathbf{u}) = (\mathbf{a}, \mathbf{s}, \mathbf{m})]}. \end{aligned}$$

By $\mathbf{V}_{\sigma_i}(\mathbf{u})$, we mean the $\mathbf{V}$ values realized when $\mathbf{U}_i = \mathbf{u}$ and there is an intervention $\sigma_i$ (which may be stochastic and hence depend on $\mathbf{u}$). This formula is quite similar to that presented in [ZTB22], with the main differences being the incorporation of stochastic interventions, specifying a subset of exogenous variables, and marginalizing over missing data. Finally, the model generates $D_i$ according to:

$$D_i \sim \text{Multinomial}(n, P(\mathbf{A}_{\sigma_i} | \mathbf{S}_i)).$$

[2]This procedure makes sense when written out, and can be implemented in code, but the present author struggles to write it more formally.

# 5 Sampling

## 5.1 Moving away from Gibbs sampling

Needless to say, actually applying this model to data from, say, several populations will significantly increase the dimension of the joint posterior. This makes sampling from the posterior slower and more difficult, which is a practical problem which must be addressed. Previous literature has consistently made use of Gibbs samplers, which sequentially sample different variables (or blocks of variables) according to their probabilities conditioned on all of the other variables and the data. This works in principle, but it can also be rather slow in practice. Gibbs sampling may also fail when applied to high-dimensional multimodal problems (especially if two variables are highly correlated). For future work which seeks to apply this or other methods to larger problems, or to extend it to continuous problems, sampling should be made more efficient. Specifying the model in a probabilistic programming language, like PyMC, allows different samplers (with highly efficient implementations) to be quickly tested and tuned.

What do these other sampling algorithms offer which Gibbs doesn't? In (blocked) Gibbs sampling, one must derive the distribution of each variable (or block of variables) when conditioned on all of the other variables. This is tedious in practice and can limit the complexity of the models implemented. When using other sampling algorithms, all that is needed is the conditional distribution of each random variable (including the data) given its parents in the model. This comes automatically when we define variables as having a certain distribution given their parents.

Gibbs sampling creates a chain of samples by sequentially sampling variables conditioned on the current value of all of the other variables. How do these other sampling algorithms sample from the posterior? Most MCMC algorithms are instances of the Metropolis-Hastings algorithm. Broadly, given a current point (i.e. a parameter vector), the Metropolis-Hastings algorithm proposes a new point from a specified *proposal distribution*, then accepts or rejects this jump based on the relative posterior probability of the data under this proposed parameter vector compared to the current one. The difference between the different Metropolis-Hastings-based algorithms is the proposal distribution chosen.

Below, we implement the model in PyMC, and experiment with using the No-U-Turn Sampler [HG+14]. The metrics in the following section indicate that it is effective even with the high-dimensional multimodal posterior that arises from the parameterization above.

For the Discrete FS method, Metropolis-Hastings cannot be directly used, so we propose a two step sampler which samples $\mu$ using Gibbs and the rest of the parameters using a Metropolis-Hastings method. We have not yet been able to implement such a sampler.

## 5.2 Bringing the posterior closer to the bounds

When using one of the Bayesian generative modles to obtain a point estimate, more data will simply make the posterior tighter and tighter around the true value. But when applied to partial identification, depending on the parameterization, the posterior can actually be tight around a small region that lies within the partial identification bounds. This is undesirable if we want to obtain the bounds themselves.

When obtaining the partial identification bounds for the IV model using a linear program, the bounds are only realized by very sparse distributions for the exogenous variables (i.e. the exogenous variables have a probability of 0 on several of their possible values). Such sparse distributions are highly unlikely to arise when the priors on the distributions of the exogenous variables are, say, Dirichlet$(1, 1, \ldots, 1)$.

We thus seek to reparameterize the Bayesian model to encourage sparsity and make the posterior fill in more of the bounds.[3] One method which appears to be effective is simply adjusting the prior:

$$P(U) \sim \text{Dirichlet}(1, 1, \ldots, 1) \rightarrow P(U) \sim \text{Dirichlet}(\alpha, \alpha, \ldots, \alpha); \quad \alpha \ll 1$$

This has been done in previous work, but we are not convinced that the samplers used were successfully able to sample from the resulting highly multimodal posteriors. Below are some experiments performed on the IV model examined in Experiment 7 of [ZTB22]. We have made this change and, with our choice of sampler, this appears to be effective:

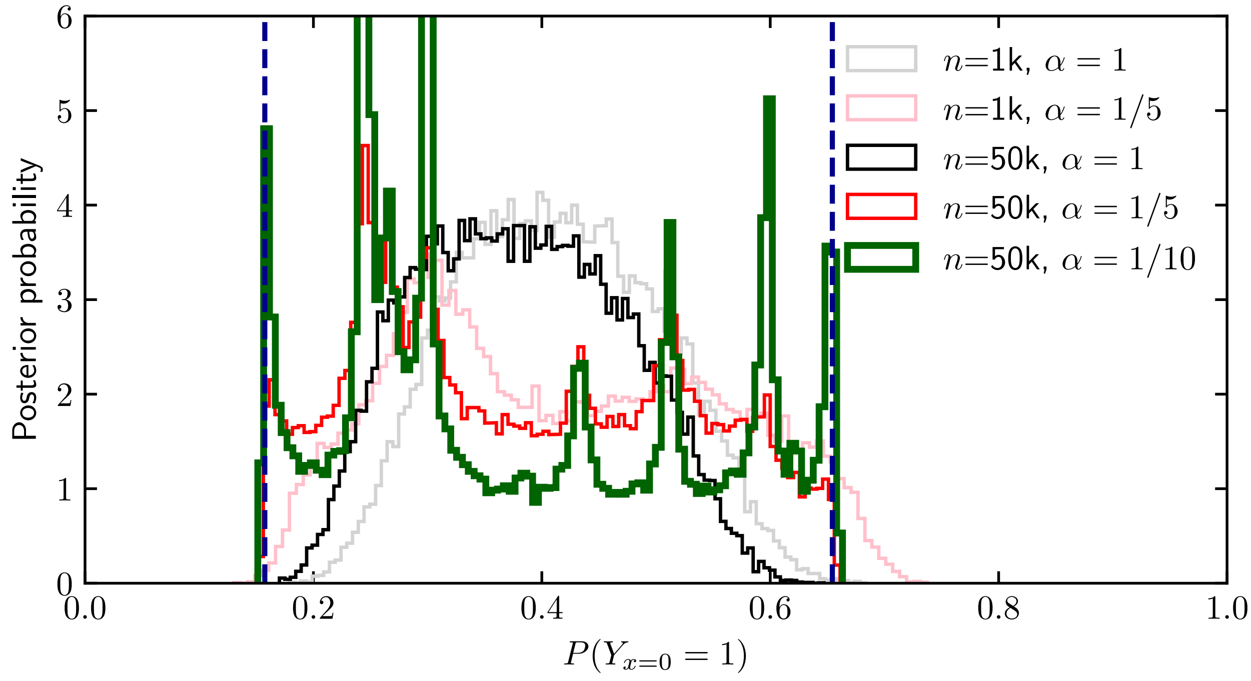


Figure 6: The parameterization we present applied to $n$ samples from Experiment 7 in [ZTB22]. Dashed lines are the bounds obtained by a linear program.

This experiment is rather promising. Notice that, as $\alpha$ decreases, the posterior becomes much higher at the bounds of the counterfactual query.

## 5.3 Verifying MCMC convergence to posterior

In previous work, the sampling is typically left unverified by any of the traditional methods used to verify the convergence of MCMC algorithms. Gibbs sampling relies on sampling from the full marginal posteriors rather than making small jumps, but it can still get stuck. The present author is not aware of good methods to verify Gibbs sampling, but for jump-based algorithms plenty of methods are available.

One useful metric is **effective sample size**. We do not go into the details here but, broadly, it is a measure of how many uncorrelated samples the autocorrelated samples that are produced by an MCMC algorithm are equivalent to, and it is calculated per-variable. In the experiment

[3]From the perspective of Bayesian inference, this is not exactly the most principled thing to do, as there is some sort of meaning in the fact that, in any obvious parameterization, the inferred model sometimes places almost no weight near the bounds. The bounds arise from the most extreme possible worlds we may be living in and, with any reasonable prior, it is unlikely that we are in such a world. But if we seek bounds, and not a posterior, then we ought to reparameterize to make such worlds less extreme.

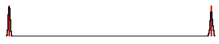
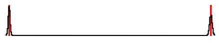
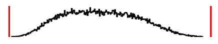
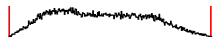
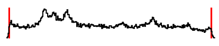
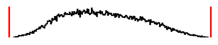
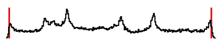
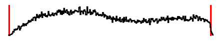
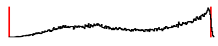
presented above for $\alpha = 1/5$, $n = 50k$, taking four chains of 50,000 samples each (1min35s on a 4-core i5-10210U CPU at 1.6GHz), the "bulk" and "tail" effective sample sizes ("bulk" and "tail" refer to different parts of the posterior) are >300 for each variable. Having effectively 300 *uncorrelated* samples for each variable (and many more for others) makes us more confident that our MCMC algorithm was able to walk around the distribution effectively.

Another useful metric is $\hat{\mathbf{R}}$, which is a measure of within-chain variance in the samples against between-chain variance in the samples. If mixing is good, these variances should be the same. In the experiment presented above, we achieve $\hat{R}$ values of $\leq 1.01$ on all variables, which indicates that we ran the algorithm for long enough to explore the distribution and not get stuck.

The effective sample size and $\hat{R}$ for the experiment where $\alpha = 1/10$ were not as promising given the same number of samples. This confirms the idea that, as $\alpha$ decreases, it becomes harder to sample from the posterior. But effective sample size and $\hat{R}$ are at least a way to *tell* when we might need to run MCMC for longer, rather than guessing and hoping that the number of samples is high enough.

## 5.4 Comparison of Generative Models

Table 2: Results for when the five models presented are used for bounding in the IV graph (4 chains of 10k samples each). Runtime is in seconds. The small unlabeled posterior plot has the true bounds labeled with vertical lines. The $y$-axis is posterior probability and the $x$ axis is the value for $P(Y_{x=0} = 1)$. For the posterior in the Dirichlet model, since that model does not obtain $P(Y_{x=0} = 1)$ but posteriors on the upper and lower bound, those posteriors are shown together in the same format. For the Oriented model, we choose to make $U_X$ ancestral in the push-down procedure. The time given for [ZTB22] is an estimate as running it to get the full set of posterior samples would take too long. The ES/s value given for [ZTB22] is upper-bounded by the 0.8 samples per second yielded, and presumably is much lower. The 2-step sampling method for the Discrete FS model has not been implemented yet.

| Model | Parameters | Sampler | Time | ESS | $\hat{R}$ | ES/s | Posterior |
|---|---|---|---|---|---|---|---|
| [ZTB22] | $\alpha = 1$ | Gibbs | ~135:00 | - | - | $\ll$0.8 | - |
| Dirichlet | $\alpha = 0.1$ | Analytic | - | - | - | - |  |
| Dirichlet | $\alpha = 1$ | Analytic | - | - | - | - |  |
| Unoriented | $\alpha = 1$ | NUTS | 0:31 | 17084 | 1.00 | 551 |  |
| Unoriented | $\alpha = 0.5$ | NUTS | 0:36 | 5874 | 1.00 | 163 |  |
| Unoriented | $\alpha = 0.2$ | NUTS | 0:33 | 798 | 1.00 | 24 |  |
| Oriented | $\alpha = 1$ | NUTS | 0:54 | 23499 | 1.00 | 435 |  |
| Oriented | $\alpha = 0.2$ | NUTS | 1:04 | 945 | 1.01 | 15 |  |
| Disc. FS | $\alpha = 0.1$ | 2-step | - | - | - | - | - |
| Prob. FS | $\alpha = 0.3, \epsilon = 0.2$ | NUTS | 8:14 | 7568 | 1.00 | 15 |  |
| Prob. FS | $\alpha = 0.3, \epsilon = 0.1$ | NUTS | 7:44 | 8562 | 1.00 | 18 |  |

In order to compare the models presented, we can first run them on data from an SCM compatible with the IV graph (specifically, the one used in [ZTB22]). Table 2 presents results for all of the models. We also include results for different settings of the hyperparameters, as these affect runtime, and for different samplers (each sampler's own parameters are tuned by hand to improve performance and only the best result is retained).

# 6 Conclusion

Starting with a goal of generalizing Bayesian methods for SCM estimation to work with heterogeneous data, we have largely focused on the task of working with regular observational data, as it turns out there is a lot of room for improvement. We start by introducing the concept of fully bidirected-connectedness, which allows us to collapse multiple exogenous variables into one, as well as stacking and pushing down, which allows us to reduce the dimension of a canonical SCM. We then move on to defining several generative models, each of which is a different way of representing how the data are generated by an SCM. We have made several advances here over previous work, the clearest which is an improvement by a linear factor in the number of samples in the dataset. The other advances are our models which don't require the functions for each endogenous variable to be sampled, which greatly improves sampling speed. We go on to discuss how different parts of an SCM can be inferred separately, which again greatly improves sampling speed, and sketch how heterogeneous data can be addressed. Finally, we get to the practical matters of how, computationally, to sample from SCMs. We argue that alternative MCMC algorithms to Gibbs sampling are far better, demonstrate how to set model priors so that the posterior density at the partial identification bounds is higher, discuss how to verify convergence, and benchmark the different generative models.

We think that together, the improvements we have made represent a significant advancement of the state of the art. Looking forwards, there are many interesting things for further investigation:

- More experiments! We need to do more experiments to see how the methods presented work on many different problems.
- Software development. Automating defining the generative models in a probabilistic programming language would greatly increase the speed with which different experiments could be run. Making this work into a neat package would also let others easily use the methods described to solve their own problems.
- In partial identification, we need a method to optimally select the prior distributions to maximize the posterior density close to the bounds which is more systematic than the heuristic of "make them as spiky as possible."
- Again in partial identification, we need a systematic method to decompose the query across different $C$-components.
- It seems like the Oriented model and the Discrete or Probabilistic FS models could be combined in the inference of one $C$-component, using function sampling for some but not all of the variables. This has not been investigated.
- Is there a custom MCMC method that could be made to not exactly sample from the posterior distribution, but stick close to the bounds in partial identification? Perhaps inference could be done with another likelihood term thrown in to emphasize high/low values for the query.
- There definitely needs to be some more formalizing. What exactly *are* those equivalence classes created by Stack and Push-down? And how can we describe the class of canonical SCMs that we're walking around? We're holding $\mathbf{V}$ and $\mathbf{U}$ fixed, but moving around in the space of $P(\mathbf{U})$ and (in the FS models) $\mathscr{F}$, though without changing the arguments of $\mathscr{F}$. Perhaps looking through the literature on Neural Causal Models which we were recently introduced to would be helpful, as they have a similar idea of optimizing within some class of models.
- The 2-step sampler for the Discrete FS model needs to be implemented.
- Just in terms of writing, there needs to be more exposition on the sampling methods used. These methods are extremely common in the applied Bayesian community but not super

common in the causal inference community, so explaining them well will be an important part of making a good paper.

## Acknowledgments

Thank you Justin for all the support! It was great talking with you about the problems I was facing and getting feedback on what direction to go in. And thank you Prof. Bareinboim for a wonderful year! I've learned so much about causality and the knowledge will be super useful in my future life as a statistician.

## References

[CP96] David Maxwell Chickering and Judea Pearl. "A clinician's tool for analyzing non-compliance". In: *Proceedings of the National Conference on Artificial Intelligence*. 1996, pp. 1269–1276.

[HG+14] Matthew D Hoffman, Andrew Gelman, et al. "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623.

[IR97] Guido W Imbens and Donald B Rubin. "Bayesian inference for causal effects in randomized experiments with noncompliance". In: *The annals of statistics* (1997), pp. 305–327.

[Pea09] Judea Pearl. *Causality*. Cambridge university press, 2009.

[RER11] Thomas S. Richardson, Robin J. Evans, and James M. Robins. "Transparent Parametrizations of Models for Potential Outcomes". In: *Bayesian Statistics 9*. Oxford University Press, Oct. 2011. ISBN: 9780199694587. DOI: `10.1093/acprof:oso/9780199694587.003.0019`. eprint: `https://academic.oup.com/book/0/chapter/141661815/chapter-ag-pdf/45787772/book\_1879\_section\_141661815.ag.pdf`. URL: `https://doi.org/10.1093/acprof:oso/9780199694587.003.0019`.

[Rub78] Donald B Rubin. "Bayesian inference for causal effects: The role of randomization". In: *The Annals of statistics* (1978), pp. 34–58.

[SE16] Ricardo Silva and Robin Evans. "Causal inference through a witness protection program". In: *Journal of Machine Learning Research* 17 (2016).

[XPB22] Kevin Xia, Yushu Pan, and Elias Bareinboim. "Neural Causal Models for Counterfactual Identification and Estimation". In: *arXiv preprint arXiv:2210.00035* (2022).

[ZTB22] Junzhe Zhang, Jin Tian, and Elias Bareinboim. "Partial Counterfactual Identification from Observational and Experimental Data". In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, July 2022, pp. 26548–26558. URL: `https://proceedings.mlr.press/v162/zhang22ab.html`.