TÉCNICO LISBOA

Differentiable Causal Discovery

under latent interventions

Gonçalo Rui Alves Faria

Thesis to obtain the Master of Science Degree in

Data Science and Engineering

Supervisor(s): Prof. André Filipe Torres Martins

Prof. Mário Alexandre Teles de Figueiredo

Examination Committee

Chairperson: Prof. Full Name

Supervisor: Prof. Full Name 1 (or 2)

Member of the Committee: Prof. Full Name 3

October 2021

Esta dissertação é dedicada às pessoas que me incentivaram ao longo da minha educação. A elas expresso o meu agradecimento por tornarem possível viver esta aventura até ao fim.

Acknowledgments

I would like to express my appreciation to the supervisors, André Martins and Mário Figueiredo, for the valuable guidance, discussions, and support. I also would like to thank the DeepSPIN/MAIA research group for providing the computational resources that allowed me to adequately experiment with and evaluate the proposed methods.

I also would like to acknowledge the python community Van Rossum and Drake [1], for developing the core set of tools that enables this work, including Pytorch from Li et al. [2], Numpy from Harris et al. [3], matplotlib from Hunter [4], and seaborn Waskom [5].

Resumo

Trabalhos recentes têm mostrado resultados promissores na descoberta de relações causais, através de dados de intervenções com métodos baseados em redes neuronais, mesmo quando são desconhecidas as variáveis intervencionadas. No entanto, trabalhos anteriores pressupõem que a correspondência entre amostras e intervenções é conhecida, o que muitas vezes é irreal. Consideramos um cenário com um extenso conjunto de dados amostrados de múltiplas distribuições de intervenção e uma distribuição observacional, no entanto, não sabemos qual distribuição originou cada amostra e como a intervenção afetou o sistema; *ou seja,* as intervenções são inteiramente latentes. Desta forma, proponho um método computacional, baseado em redes neuronais e inferência variacional, que enquandra aprendizagem de estruturas causais como um problema de otimização continua com restrições. Experências com dados sintéticos e reais mostram que a nossa abordagem e sua variante semi-supervisionada são capazes de descobrir relações causais neste cenário desafiante.

Palavras-chave: Descoberta causal, intervenções latentes, inferência variacional, processo Dirichlet

Abstract

Recent work has shown promising results in causal discovery by leveraging interventional data with gradient-based methods, even when the intervened variables are unknown. However, previous work assumes that the correspondence between samples and interventions is known, which is often unrealistic. We envision a scenario with an extensive dataset sampled from multiple intervention distributions and one observation distribution, but where we do not know which distribution originated each sample and how the intervention affected the system, *i.e.*, interventions are entirely latent. We propose a method based on neural networks and variational inference that addresses this scenario by framing it as learning a shared causal graph among an infinite mixture (under a Dirichlet process prior) of intervention structural causal models . Experiments with synthetic and real data show that our approach and its semi-supervised variant are able to discover causal relations in this challenging scenario.

Keywords: causal discovery, latent interventions, variational inference, Dirichlet process.

Contents

List of Tables

List of Figures

Acronyms

DAG directed acyclic graph

SCM structural causal models

RCT randomized control trials

MEC Markov equivalence class

ICM independence of cause and mechanism

MAP maximum a posteriori

ELBO evidence lower bound

KL Kullback–Leibler

ST straight-through

DP Dirichlet process

NN neural network

DSF deep sigmoidal flows

ER Erdős-Rényi

SHD structural hamming distance

SGD stochastic gradient descent

Chapter 1

Introduction

1.1 Motivation

Even the most mundane activities in the everyday life of a human individual require planning and decision making, using predictions of the consequences of actions. Since actions, by their very nature, interact with reality and change the state of affairs, assessing the consequences of actions not yet taken requires understanding realities' true underlying mechanisms to some level of sophistication. This is because only models of these underlying mechanisms can make sound predictions under unforeseen circumstances. The connection between cause-effect relations and robust prediction under unexpected circumstances makes discovering causal structures essential for intelligent planning, decision-making, and model interpretation in numerous settings. This can be illustrated by considering a University of Winnipeg study that showed that "shallowness" correlated with heavy text messaging in teens. As noted in [6], media outlets jumped on this as proof that texting causes teenagers to be more shallow. Under this reasoning, if we train a machine learning model that learns to predict the shallowness given how prolific a teen is, if we then, for instance, transfer the student to a far off location without a cell phone signal, our model should in principle generalize, and therefore accurately predict the teen's expected shallowness under this new conditions, right? The flaw with this reasoning should, hopefully, be evident. The study proved nothing of the sort. *Correlation is not causation*. It might be the case that shallowness makes teens more drawn to texting or that a common factor causes both (see Figure 1.1). In either case, the value of learning causal mechanisms should be clear. We can capture principles with our models that generalize for out-of-distribution data, in this case, catapulting the teen to a place without cell phone towers, which for our model is an unprecedented scenario unseen in the training distribution.

1.2 Causal discovery

However, how can we obtain this causal knowledge? We could use experts' domain expertise to encode it in our model. Still, this is not generally possible for the big problems usually tackled with Machine Learning methods or small problems with insufficient domain expertise. An ambitious alternative

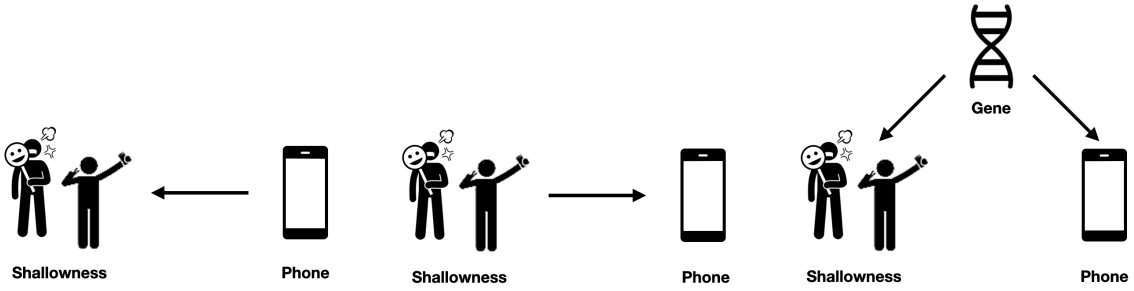

Figure 1.1: Causal hypothesis for the phenomena observed in the University of Winnipeg study.

is creating data-driven methods of discovering the causal structure. Not surprisingly, this is by itself a tremendously difficult problem of fundamental interest in science in many fields of research and a hurdle in integrating causality and machine learning. Data-driven causal discovery is a challenging problem. Even if we disregard the practicality of solving it in finite time, only with strong assumptions can we safeguard that we will always detect distinct causal structures by the observations they produce (even with an infinite amount thereof). Yet, in science, tremendous progress has already been accomplished through the application of the scientific method. Namely, generating hypotheses, doing experiments, and collecting data. Which is to say, interventions (actions) themselves are a powerful beam of light that enables us to distinguish causal structures. Therefore, countless works combine observations with data obtained after performing interventions, and given enough interventions, they theoretically guarantee and experimentally validate that it is possible to distinguish structures by the joint data they produce.

While many of the fields of science can perform interventions of the system of claim, for instance, in genomics, recent advances in gene-editing technologies [7] have given rise to high-throughput methods for interventional gene expression data, for the types of problems Machine Learning has been shown to excel, performing interventions is complex, expensive and sometimes unethical. This motivates the setup of this dissertation. We want to bridge the gap between the observational and joint observational and interventional cases. We envision a scenario with an extensive dataset sampled from multiple intervention distributions and one observation distribution, but where we do not know which distribution originated each sample and how the intervention affected the system,i.e., interventions are entirely latent.

1.3 Applicability

However, how realistic are latent interventions? There are a few scenarios where the system somehow was intervened, and the samples generated under this condition were not properly identified. For instance, consider a software platform that collects analytics regarding in-app user experience and regularly performs small-scale new features tests. It is reasonable to assume that some information was lost, and the platform no longer knows which samples were subject to the feature tests. Another scenario is discovering causal relations in a historical dataset where there might be significant events (interventions) in the system of interest not yet uncovered. Nevertheless, from a machine learning perspective, we

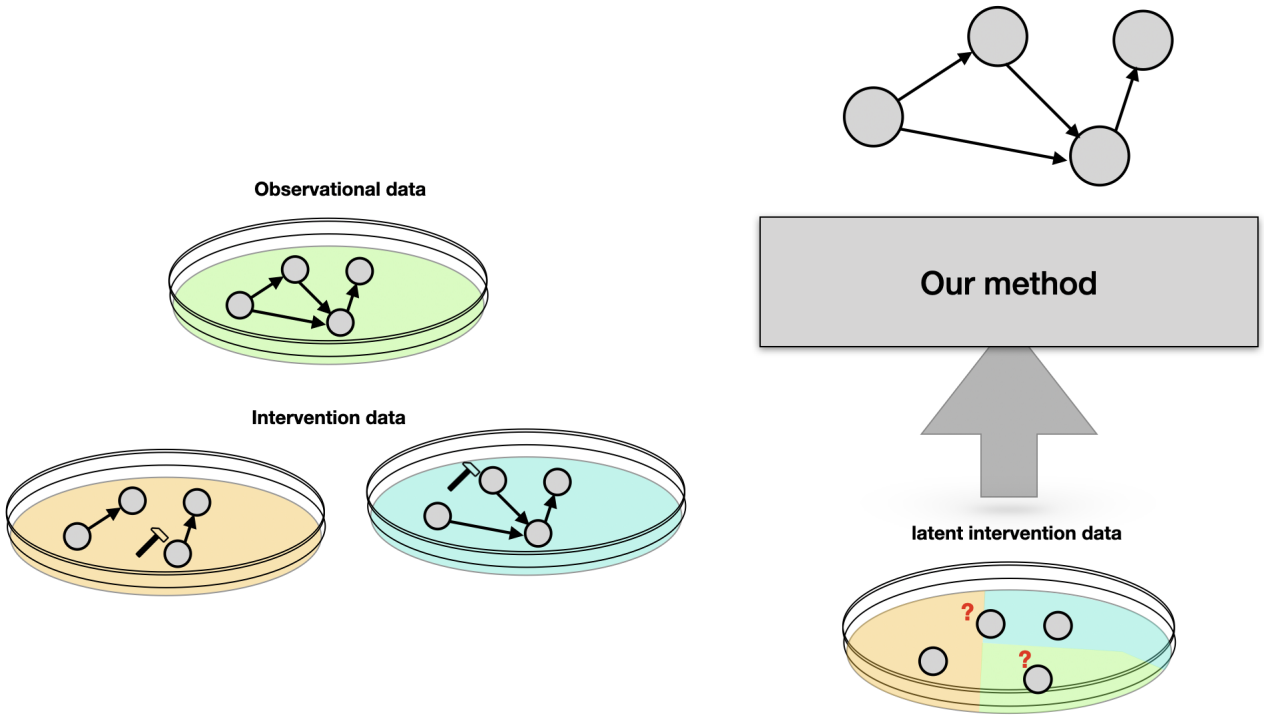

Figure 1.2: Illustrative example of the causal discovery scenario we consider.

pose the existence of latent interventions, primarily because we assume we are dealing with big datasets collected across different geographic locations and times. Under these conditions, we are bound to deal with what is generally considered a nuisance to the practitioner, changes in the distribution due to non-stationarity, and uncontrolled external interventions. We aspire to use these non-stationarities as a training signal in the search for the underlying causal relations in real-world problems.

1.4 Contributions

This dissertation contributes to the field of causality by considering the problem of data-driven causal discovery when we have a dataset from a system subjected to latent interventions. We identify the main contributions of this dissertation as follows:

- We propose a method for identifying the latent interventions when given the correct causal graph.
- We formulate causal discovery under latent interventions as searching for the shared causal graph among an infinite mixture of intervention structural causal models.
- We develop a semi-supervised variant of our method; for when we know the correspondence for a subset of the samples.
- We open-sourced our Pytorch implementation; the code is available at https://github.com/goncalorafaria/latent-causal-discovery.
- We creaded a package, named PyCausal. for defining, intervening, and sampling from structural causal models; the code is available at https://github.com/goncalorafaria/PyCausal.

Part of this dissertation has originated a scientific paper submitted to the 1st Conference on Causal Learning and Reasoning, currently under revision.

1.5 Dissertation Outline

This dissertation is organized as follows. Chapter 2 starts by reviewing causality and causal models; then, we present the related work on causal discovery, particularly some of the methods we build upon; and we finish with latent variable models and normalizing flows. Chapter 3 proposes a method based on variational inference for identifying latent interventions when given the correct causal graph. Chapter 4 proposes a gradient-based data-driven causal discovery method for datasets with latent interventions by building upon the method proposed in Chapter 3. Appendix A gives a derivation for the variational inference lower bounds and the Kullback–Leibler divergences of the models proposed in this dissertation. Appendix C presents the Python package we developed and open-sourced for defining structural causal models, intervening on them, and sampling. Appendix D describes in detail how we generate the synthetic data sets used for experiments.

Chapter 2

Background and Related Work

This chapter provides the necessary background to understand both the problem and our main contributions. We begin, in Section 2.1, by presenting the topic of causality, particularly *structural causal models*, introducing the concepts of interventions and causal structure learning. Section 2.2, we provides a concise background on latent variable models, approximate inference, and gradient estimation techniques; our goal is to present the theory behind these models and structured prediction using amortized variational inference, tools used in this dissertation for causal discovery and estimation of latent interventions.

2.1 Causality and Causal Models

The simplest and most popular causal model is the *Markovian* model. It consists of a *directed acyclic graph* (DAG) $\mathcal{G}$, called a *causal graph*, over a set of vertices $X = \{x_1, x_2, \ldots, x_d\}$, representing the variables of interest, and a set of directed edges connecting these vertices (see Fig. 2.1 for an example). This model has both a probabilistic and a causal interpretation. The former builds upon the framework of Bayesian networks and views the graph structure as a representation of conditional independence statements, and hence it is a description of a compact factorization of the joint distribution of the variables. On the other hand, the latter entails viewing the arrows as a representation of a cause-effect relation, where the compact factorization still holds, but the factors are further assumed to represent autonomous data-generating processes.

In this dissertation, we will use an extension of the Markovian model, denominated *structural causal*

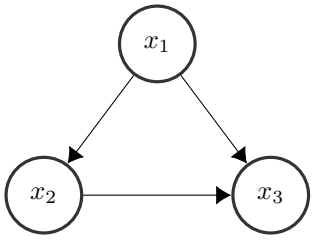

Figure 2.1: Example of a causal graph.

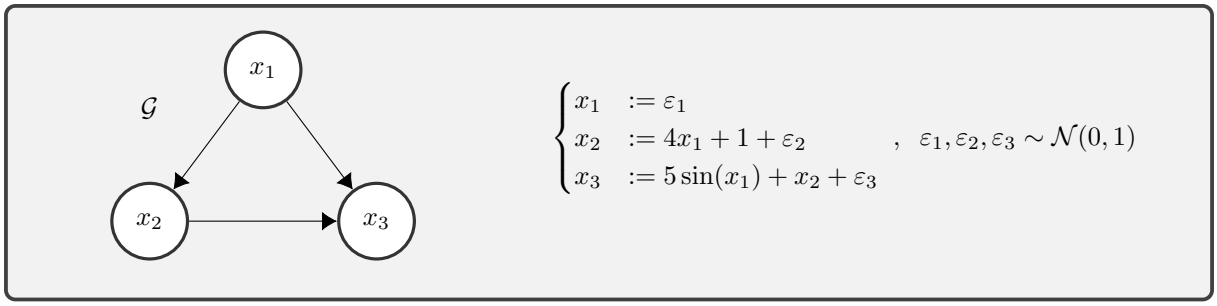

Figure 2.2: Example of an SCM and the corresponding causal graph.

models (SCM) [8, 9], which considers, more concretely, the functional form of the autonomous data-generating processes. We will use SCMs to relate causal and probabilistic statements. The notation and terminology used in this section is heavily inspired by the one presented in [9].

2.1.1 Structural Causal Models

Let $X = \{x_1, \ldots, x_d\}$ be a set of d *endogenous* variables of interest, $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_d\}$ be a set of d *exogenous* variables, $\mathcal{F}$ be a set d of functions $\mathcal{F} = \{\zeta_1, \zeta_2, \ldots \zeta_d\}$, $\mathcal{G}$ a DAG, and $\mathbf{PA}_j^{\mathcal{G}} \subseteq X \backslash \{x_j\}$ be the set of *parents* of x_j according to $\mathcal{G}$, *i. e.*, $x_i \in \mathbf{PA}_j^{\mathcal{G}}$ if and only if there is a directed edge $x_i \to x_j$ in $\mathcal{G}$. We define a collection of d assignments S as

$$x_j := \zeta_j(\mathbf{PA}_j^{\mathcal{G}}, \varepsilon_j) \ , \qquad j = 1, \ldots, d. \tag{2.1}$$

An SCM $\mathcal{M} := (S, p(\mathcal{E}))$ consists of a collection S of d assignments and a independent noise distribution, of exogenous variables $p(\mathcal{E}) = \prod_{j=1}^{d} p(\varepsilon_j)$. An SCM $\mathcal{M}$ defines a unique joint distribution for X, usually referred to as the **entailed distribution** $p_{\mathcal{M}}(X)$, which can be factorized as

$$p_{\mathcal{M}}(X) = \prod_{j=1}^{d} p_{\mathcal{M}}(x_j | \mathbf{PA}_j^{\mathcal{G}}), \tag{2.2}$$

where $p_{\mathcal{M}}(x_j | \mathbf{PA}_j^{\mathcal{G}})$ is the conditional distribution of x_j, given its parents. The particular case where the SCM has only two variables ($d = 2$) is often called a *cause-effect pair*. Figure 2.2 contains an example of an SCM with three variable and its corresponding causal graph. We can sample from SCMs via ancestral sampling. In this case, we start with the root node x_1, then compute x_2, given x_1, followed by x_3, given x_1 and x_2.

2.1.2 Interventions

An ubiquitous tool for describing statistical models is the concept of conditional probability. For instance, given the SCM described in Figure 2.2, $p_{\mathcal{M}}(x_3|x_2 = 2)$ denotes the conditional distribution of the variable x_3, given $x_2 = 2$. Which is to say: if we restrict our focus to samples where $x_2 = 2$, $p_{\mathcal{M}}(x_3|x_2 = 2)$ encodes the distribution of the observed values of x_3. However, how can we describe perturbing a system in such a way as forcing the entire population to have the value x_2 be 2? Interventions and the mathematical operator "do" (introduced by Pearl [10]) allow doing precisely that: acting (intervening) on the data generating process and finding the resulting distribution of the system's variables. In the previous example (Figure 2.2), we denote by $p_{\mathcal{M}}\big(x_3|do(x_2 = 2)\big)$ the intervention distribution of x_3 if we force x_2 to be 2. In general, the intervention distributions and conditional distributions will be distinct. For instance, since x_1 is a common cause of x_2 and x_3, conditioning on $x_2 = 2$ gives information about the distribution of x_1. By intervening, this will not be the case: just by forcing x_2 to be 2 "says" nothing about the distribution x_1 and, in fact, x_1 and x_2 actually become independent from each other, since the intervention block the effect of x_1 on x_2.

The foundations of interventions rest on Judea Pearl's seminal work on *do-calculus* [10–12] . They are a mathematical construction that describes performing idealized experiments in a SCM. Given an SCM $\mathcal{M}$, we constructs an *interventioned* SCM $\tilde{\mathcal{M}}$ by replacing one (or several) of the structural assignments. Let I the set of variables that were targeted by the intervention; if $I = \emptyset$, $\tilde{\mathcal{M}} = \mathcal{M}$. For each variable $j \in I$, the interventions consists in one or multiple of the following actions: replacing the assignment function ζ_j by $\tilde{\zeta}_j$; replacing the parents $\mathbf{PA}_j^{\mathcal{G}}$ by $\tilde{\mathbf{PA}}_j^{\mathcal{G}}$; changing the noise variable ϵ_j by $\tilde{\epsilon}_j$. The SCM $\tilde{\mathcal{M}}$ generally has a different entailed distribution, called the *intervention* distribution:

$$p_{\tilde{\mathcal{M}}}(X) = \prod_{j \notin I} p_{\mathcal{M}}(x_j|\mathbf{PA}_j^{\mathcal{G}}) \prod_{j \in I} p_{\mathcal{M};\mathsf{do}\left(x_j := \tilde{\zeta}_j(\tilde{\mathbf{PA}}_j^{\mathcal{G}}, \tilde{\varepsilon}_j)\right)}(x_j|\mathbf{PA}_j^{\mathcal{G}}). \tag{2.3}$$

If there are K possible interventions, we denote the corresponding sets of target variables as $I^{(k)}$, for $k = 1, ..., K$, and the corresponding SCMs by $\tilde{\mathcal{M}}^{(k)}$.

We divide the types of interventions into: *atomic*, if the target variable x_j is set to a constant value; *stochastic*, if x_j is set to a random variable $\tilde{\varepsilon}_j$; *imperfect* (or *soft*), if the intervention embedding and the set of parents are changed, as long as it does not become empty. We do not consider interventions that are able to add new elements to $\mathbf{PA}_j^{\mathcal{G}}$. This means that the intervention graph only differs from the observational by the removal of edges.

Figure 2.4 contains an example of an intervention to the SCM from Figure 2.2. In Figure 2.5, we present scatter plots from sampling according to these SCMs.

Intervention example. A prototypical example of intervention in the real world is a randomized control trials (RCT). In a RCT, a group of subjects is randomly selected from the population and is subject to a given treatment. Generally, the study designers intend to understand if the given treatment *causes* a particular response. While the study designers calculate various statistics between the treatment and the response with samples from the intervention distribution, they are confident that any correlation

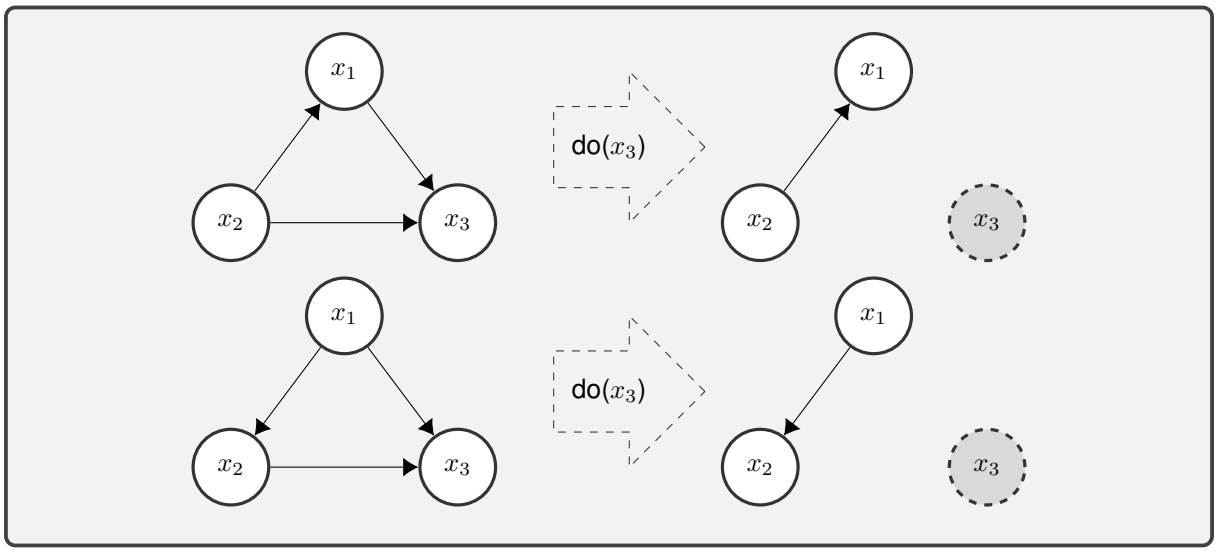

Figure 2.3: On the left, Example of two $\mathcal{I}$-Markov equivalent graphs where $\mathcal{I}$ is composed of a single intervention on the node x_3. On the right, the graphs resulting from performing the interventions from $\mathcal{I}$ to the graphs on the left. As expected, the graph on the right are Markov equivalent graphs.

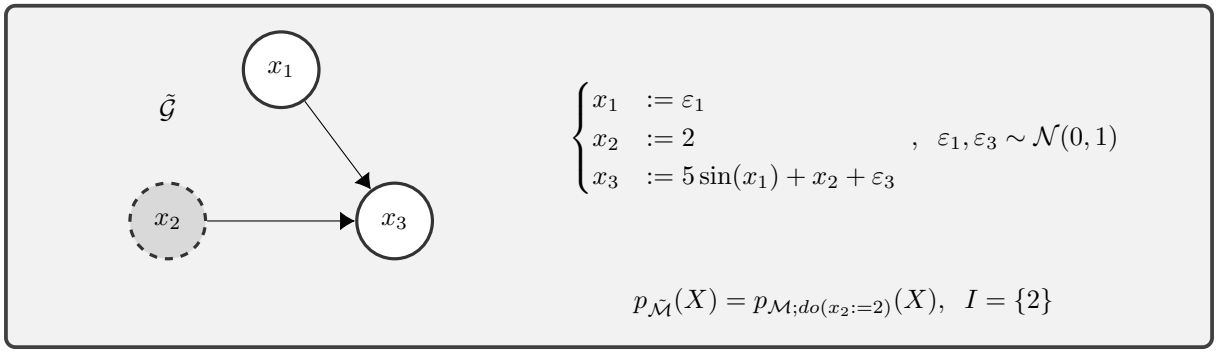

Figure 2.4: Example of an intervention SCM and the corresponding causal graph. The source SCM is the one present in Fig. 2.2. This is an instance of an atomic intervention.

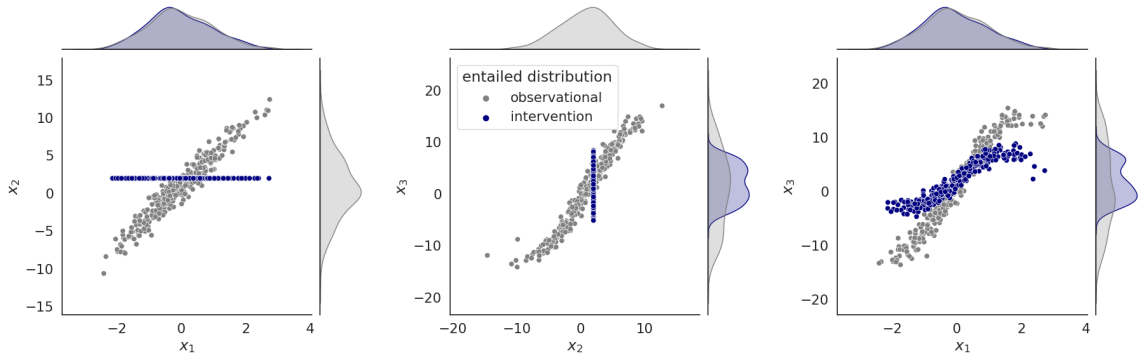

Figure 2.5: Samples from the observational SCM in Figure 2.2, and intervention SCM in Figure 2.4.

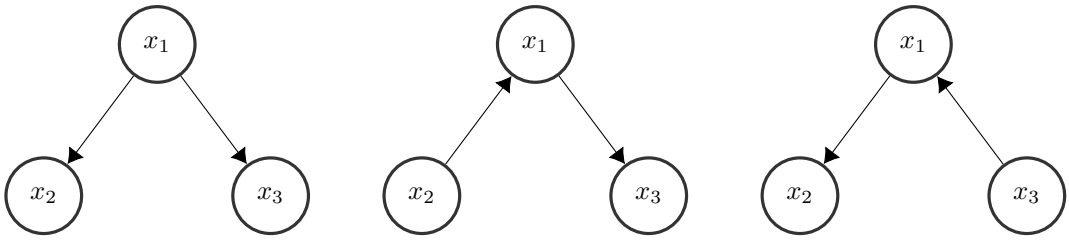

Figure 2.6: Example of three Markov-equivalent graphs.

found implie causation. This is because the treatment was attributed indeed at random and hence is not confounded by hidden factors. This stands in stark contrast with calculating the same statistics, but on the observational distribution where a person's decision to incur or not in treatment may be due to unknown external factors that could correlate with the response. In our SCM framework, the factors would be the incoming edges from $\mathbf{PA}_j^{\mathcal{G}}$.

2.1.3 Faithfulness and Markov equivalence classes

Given a set $\mathcal{F}$ where each element f_i is *sufficiently* dependent on all of the elements of its input $\mathbf{PA}_i^{\mathcal{G}}$, we obtain an SCM $\mathcal{M}$ whose computations strictly follow the structure of $\mathcal{G}$. In this scenario, $\mathcal{G}$ and $p_{\mathcal{M}}(X)$ are said to be mutually *faithful* since $\mathcal{G}$ encodes all and only the conditional independencies that hold in the entailed distribution. Therefore, $\mathcal{G}$ is both a compact description of the set of conditional independence properties, and the high-level causal dependencies present in the SCM $\mathcal{M}$. In essence, the Markovian model mentioned before. The set of faithful graphs that could entail a particular joint distribution is denominated the *Markov equivalence class* (MEC) [13]. In other words, the set of faithful graphs that share exactly the same conditional independence assumptions. Figure 2.6 contains an example of three Markov equivalent graphs. If there is access to intervention data (in a set of interventions $\mathcal{I}$), it is possible to shrink the MEC to the so-called $\mathcal{I}$-MEC [14]: the subset of graphs in the MEC that have the same conditional independencies after applying the interventions in $\mathcal{I}$. Figure 2.3 contains an example of two $\mathcal{I}$-Markov equivalent graphs.

Independence of cause and mechanisms.

We defined the intervention distribution in equation 2.3 as the product of conditional distributions entailed in $\mathcal{M}$ for the unperturbed variables and distinct distributions for the perturbed ones. Like in the source SCM, the entailed distribution of the intervention SCM is defined as the product of all $p_{\tilde{\mathcal{M}}}(x_j|\mathbf{PA}_j^{\mathcal{G}})$. However, interventions are defined as infinitely precise, so we can swap out $p_{\tilde{\mathcal{M}}}(x_j|\mathbf{PA}_j^{\mathcal{G}})$ by $p_{\mathcal{M}}(x_j|\mathbf{PA}_j^{\mathcal{G}})$ for the unaffected variables. This is nevertheless a very strong assumption. We consider that the underlying data generating process is modular and composed of autonomous units, modeled with assignments, that do not influence each other.

The assumption is what is generally referred to as *independence of cause and mechanism*(ICM) [9, 15, 16]. It enables us to define interventions that exchange some of the mechanisms leaving the remaining ones intact. Even though this is intuitive and inspired by physical reality, note however that it

might not be how some phenomena operate, and some popular examples do not follow this assumption, as noted in [9]. However, we follow this assumption in this dissertation.

2.1.4 Identifiability

Causal questions, such as "does variable x_2 cause x_3?", can be answered easily once we know the correct causal graph. We only need to check if there is a directed path in the causal graph connecting variable x_2 to x_3. However, discovering a faithful directed acyclic graph from samples of the entailed distribution is a challenging combinatorial problem, primarily due to the intractable search space, which is super-exponential in the number of variables. Furthermore, if our goal is to find the causal graph of the SCM that originated the data, just a faithful graph, which is to say an element of the MEC, does not suffice. Interventions are therefore a way to inspect the data generating process and generally allow us to shrink the MEC to the $\mathcal{I}$-MEC. When we perform enough interventions, the graph becomes exactly identifiable, hence with a consistent method we have guarantees of identifying the structure of the true underlying causal model.

With a limited number of interventions, the graphs that share the MEC are impossible to distinguish using observational data, even with an unlimited supply of data. The graphs that share the $\mathcal{I}$-MEC are impossible to distinguish, even when we consider the interventions distributions obtained after applying the interventions in $\mathcal{I}$. The work in [13, 17] explores this topic in detail and present all of the technical conditions that we mostly gloss over for brevity. In the context of this thesis, we highlight as particularly relevant the following results:

- For any SCM $\mathcal{M}$, $d-1$ single node interventions (of any type) are sufficient, and necessary in the worst case to obtain full identifiability [18].
- For any SCM $\mathcal{M}$, with no restriction on the number of intervention targets per intervention and let c be the size of the largest clique, $\lfloor \log_2(c) \rfloor + 1$ interventions are sufficient, and necessary in the worst case to obtain full identifiability [19].
- For any SCM $\mathcal{M}$, where we consider interventions that affect up to k variables, where $k < \frac{n}{2}$, we only need $\lceil \frac{d}{k} \rceil \log_{\lceil \frac{d}{k} \rceil}(d)$ interventions to obtain full identifiability [20].

More recently, within this framework of SCMs, when we confine ourselves to assumptions such as non-Gaussianity [21], nonlinearity [22, 23], or equal noise variances [24], we can guarantee full identifiability of structures in the limit of infinite data from the observational distribution.

2.1.5 Causal Structure Learning

There are generally three approaches for discovering the causal structure when given independent and identically distributed samples from the observational distribution and/or intervention distributions: constrained-based methods, score-based methods, and hybrid methods. We describe each of them bellow.

Constrained-based methods test for conditional independence using the samples from the entailed distribution or intervention distributions and propose a set of constraints that the underlying causal structure must meet. Then, using various graph construction procedures, these methods identify the structures that are congruent with all of the constraints and identify some or all of the elements of the MEC. However, testing for independence is not simple, and spurious dependencies that are an artifact of the independence hypothesis testing procedure may lead to failure regardless of the guarantees around the effectiveness of the graph construction algorithm. Typical examples of constrained-based methods that work with samples from the observation distribution are the *Peter and Clark*(PC) algorithm [25], *Spirtes, Glymour and Scheines* (SGS) [25], *Inductive Causation* (IC) [26], and *Fast Causal Inference* (FCI) [27, 28] algorithms.

Methods that incorporate data from the intervention distributions include COmbINE [29], and HEJ [30], which rely as a graph construction procedure on Boolean satisfiability solvers, and [31] that does this by extending the FCI algorithm. Additionally, the *Joint causal inference framework*(JCI) enables one to incorporate intervention data into existing algorithms designed for causal discovery with observational data, even when a priori, the interventions' targets are unknown. Lastly, another family of constrained-based methods that support intervention distributions is the *Invariant Causal Prediction*(ICP) [32, 33], which relies mainly on the invariance of causal mechanisms across intervention distributions, and is exclusively concerned with uncovering the causal parents of a particular variable of choice.

Score-based methods formulate the problem of structure estimation as the by-product of learning a statistical model. Each method defines a hypothesis space of potential structures. For simplicity, consider the space of DAGs and a scoring function $\mathcal{S}$ that measures how well the statistical model fits the data when given a particular structure. The computational task is to find the graph that achieves the highest score. This search problem generally has an intractable search space, super-exponential in the number of variables. For that reason, even with a consistent scoring function that effectively gives a higher score to the structures in the MEC and the $\mathcal{I}$-MEC, this problem is still an profoundly challenging problem, and most approaches resort to heuristic search. More concretely, the estimated DAG $\hat{\mathcal{G}}$ is given by

$$\hat{\mathcal{G}} = \underset{\mathcal{G} \in \mathsf{DAG}}{\arg\max}\, \mathcal{S}(\mathcal{G}). \tag{2.4}$$

There are many score function methods with guarantees of convergence to the MEC and some for the $\mathcal{I}$-MEC. Popular score functions include BIC [34], MDL [35], BDe(u) [36] and BGe [37]. For

small problems there are algorithms for solving Equation 2.4 to global optimality [38–43]. There is a vast literature on approximate algorithms based on order search [44] and more classical methods such as greedy search [36, 45, 46] and coordinate descent [47–49]. In order search, we first search over the space of topological orderings. As a result, we avoid the acyclicity constraint since, given a correct ordering in the form of a permutation matrix P, we can write any DAG's adjacency matrix as $P^T W P$, where W is an upper triangular matrix. Notable examples of score-based greedy-search algorithms that support perfect interventions are GIES [14] and CAM [50]. More recently, a new family of approximate algorithms for structure learning initiated in [51] treats the problem in 2.4 as a continuous constrained optimization problem. What enables the use of continuous optimization is fundamentally the smooth characterization of the acyclicity constraint. The work of [52], the DCDI algorithm, extends this contiguous constrained formulation to intervention data, with perfect and imperfect interventions, and also supports interventions with unknown intervention targets.

Hybrid methods combine constraint and score-based approaches. Recent hybrid methods that support data from interventions include IGSP [53, 54], which optimize a score based on conditional independence tests. It also supports interventions with unknown targets with the extensions from UT-IGSP [55].

Continuous constrained optimization for structure learning

As we have mentioned in the previous exposition of the available causal discovery methods, a new line of research initiated by [51] uses continuous constrained optimization to solve the score-based formulation. This work and the work from [52], which extends this approach for when we have data from intervention distributions, form the basis of our work. For this reason, we will briefly present how they characterize the problem and their solutions in some detail.

In general, these methods adopt the *maximum a posteriori* (MAP) criterion (a.k.a. penalized maximum likelihood). Based on a generative/sampling model $p(\mathcal{D}|\mathcal{G}, \theta)$ for data $\mathcal{D}$, given the graph structure $\mathcal{G}$ and parameters θ, and on a *prior* $p(\mathcal{G})$ over graphs, they seek a graph that maximizes the score function

$$\mathcal{S}(\mathcal{G}) := \max_{\theta} \log p(\mathcal{D}|\mathcal{G}, \theta) + \log p(\mathcal{G}). \tag{2.5}$$

The prior $p(\mathcal{G})$ penalizes graph complexity to avoid over-fitting. A typical choice is $p(\mathcal{G}) \propto \exp(-\lambda|\mathcal{G}|)$, for $\lambda > 0$ and $|\mathcal{G}|$ is some graph complexity measure (*e.g.*, number of edges). With finite data, exact independence seldom occurs, thus graphs maximizing $\log p(\mathcal{D}|\mathcal{G}, \theta)$ alone would almost always be fully connected. If $\mathcal{D}$ is a collection of i.i.d. observations, then $p(\mathcal{D}|\mathcal{G}, \theta) = \prod_{i=1}^{n} p(x_i|\mathcal{G}, \theta)$.

Central to this class of methods is the weighted adjacency matrix $W_{\mathcal{G}} \in \mathbb{R}_{\geq 0}^{d \times d}$, where $(W_{\mathcal{G}})_{ij} > 0$ is equivalent to $(i, j) \in \mathcal{G}$, which is treated as a parameter itself or as a function of the parameters. To

ensure the estimated graph is a DAG, Zheng et al. [51] proposed the constraint

$$\text{trace}\big(e^{W_{\mathcal{G}}}\big) - d = 0, \tag{2.6}$$

where $e^{W_{\mathcal{G}}}$ is the matrix exponential, and the trace function is the sum of the elements of the main diagonal. The intuition is that if $(W_{\mathcal{G}}{}^n)_{ii} > 0, \forall n \in \mathbb{N}$, the i^{th} element of the main diagonal of the matrix after performing the n^{th} power operation is greater than zero, than exists a n steps loop in $\mathcal{G}$. Since the exponential operation is a weighted sum of all power matrices including the one raised to the power 0 if $(e^{W_{\mathcal{G}}})_{ii} > 1$ there exists at least one loop in $\mathcal{G}$.

The work from NOTEARS [51], assume a linear Gaussian model with equal variances where $\theta \in \mathbb{R}^{d\times d}$ is the matrix of regression coefficients, the prior distribution is proportional to $\exp(-\lambda|\theta|)$ and $W^{\mathcal{G}} = \theta \odot \theta$. Altogether the joint optimization problem takes the following form:

$$\max_{\theta} \log p(\mathcal{D}|\theta) - \lambda||\theta||_1 \;\text{ s.t. }\; \text{trace}\big(e^{W_{\mathcal{G}}}\big) - d = 0, \tag{2.7}$$

where the sampling model $\log p(\mathcal{D}|\theta)$ takes the form :

$$\log p(\mathcal{D}|\theta) = -\frac{1}{2\sigma^2}(X - X\theta)^\intercal(X - XW) + c$$

where $c \in \mathbb{R}$ is a constant.

Several other methods apply non-linear models such as neural networks [56, 57] and define $W_{\mathcal{G}}$ differently. The works from [52, 58, 59] treat the adjacency matrix as a random variable and relax the score from Equation 2.5 in the following way:

$$\mathcal{S}^\star(\Lambda) := \max_{\theta} \mathbb{E}_{\mathcal{G}\sim\text{Bern}\left(\mathcal{G};\sigma(\Lambda)\right)}\Big[\log p(\mathcal{D}|\mathcal{G},\theta) + \log p(\mathcal{G})\Big], \tag{2.8}$$

where $\sigma(\Lambda)$ is the sigmoid transformation applied element-wise to the parameter matrix $\Lambda \in \mathbb{R}^{d\times d}$, $\text{Bern}\big(\mathcal{G};\sigma(\Lambda)\big)$ is a distribution over graphs, with mutually independent edges, with expected value $\sigma(\Lambda)$. This score tends asymptotically to $\mathcal{S}(\mathcal{G})$ as $\sigma(\Lambda)$ progressively concentrates its mass on a single DAG $\mathcal{G}$.

Constrained Optimization Most works, and particularly [51], solve the constrained problem using the *augmented Lagrangian* procedure. The augmented Lagrangian procedure [60–62] consists of transforming, for instance Equation 2.7, into a sequence of unconstrained optimization sub-problems which are then solved with a simple method such as stochastic gradient descent. By denoting the acyclicity constraint by $h(\Lambda) = 0$, each unconstrained sub problem t is written as follows:

$$\Lambda_t^*, \theta_t^* = \arg\max_{\theta_t,\Lambda_t} \log p(\mathcal{D}|\theta_t) - \lambda\|W\|_1 - \varphi_t h(\Lambda_t) - \frac{\mu_t}{2}h(\Lambda_t)^2. \tag{2.9}$$

Each sub-problem t is initialized using the previous sub-problem's solution (Λ_{t-1}^*,θ_{t-1}^*). The parame-

ters γ_t and μ_t are updated in the end of each sub-problem as follows:

$$\begin{aligned} \varphi_{t+1} &\leftarrow \varphi_t + \mu_t h(\Lambda_t^*), \\ \mu_{t+1} &\leftarrow \begin{cases} \eta\mu_t & \text{if } h(\Lambda_t^*) > \delta h(\Lambda_{t-1}^*) \\ \mu_t & \text{otherwise} \end{cases}. \end{aligned} \tag{2.10}$$

The optimization stops when $h(\Lambda_t^*) \leq 10^{-8}$. The parameter $\delta \in]0, 1[$ controls the schedule of μ_t, more concretely, when the relative reduction in the constraint is higher than δ, μ_t is multiplied by $\eta > 0$ hence the squared penalization has a greater impact on the objective function. The parameters $(\varphi_0, \mu_0, \eta, \delta)$ are respectively set to $(0.0, 10^{-8}, 2.0, 0.9)$.

2.2 Latent Variable Models

The fundamental construct behind latent variable models is that besides the random variable representing the observed data x, we assume the existence of *latent variable* z. Formally, this means that the model is a probability distribution over the joint $p(x, z; \theta)$, where θ are the model parameters.

The motivation behind such an assumption is that there could be an underlying process happening, and our observed variables are just a manifestation of such process. We are then able to make assumptions about this underlying process (through the prior $p(z; \theta)$) and about how it influences the observable data (through the conditional $p(x|z; \theta)$).

As an example, consider the case where x is a sequence of discrete random variables $x = (x_1, \ldots, x_D)$ with $x_t \in \{1, \ldots, V\}$. We can make the assumption that each x_t is conditionally independent from each other given a common categorical *latent* random variable $z \in \{1, \ldots, N\}$ associated with prior $p(z; \theta)$. This translates to model

$$p(x_1, \ldots, x_D, z; \theta) = p(z; \theta) \prod_{t=1}^{D} p(x_t|z; \theta). \tag{2.11}$$

This is known as the *Naive Bayes* model, and while it makes very strong independence assumptions, it allows us to construct from a simple joint distribution a sophisticated distribution over the observed variables.

2.2.1 Learning under Exact Inference

With a joint distribution $p(x, z; \theta)$ over the observed and latent variables, an exciting problem surfaces: how can we learn the model's parameters θ without data for the latent variables? We can start by noting that we can obtain a distribution over the observed variables alone, the **evidence**, by marginalizing over the latent variable space, i.e.:

$$p(x; \theta) = \int p(x, z; \theta) dz = \mathbb{E}_{z \sim p(z)}[p(x|z; \theta)]. \tag{2.12}$$

Here the integral can be swapped for a summation when discrete latent variables are used. The goal

of a learning algorithm would be, in this case, to find the parameters to maximize the **log-evidence** over the training set $x^{(1:N)} = [x_1, \ldots, x_N]$:

$$\mathcal{L}(\theta) = \log p(x^{(1:N)}; \theta) = \sum_{i=1}^{N} \log p(x^{(i)}; \theta), \tag{2.13}$$

$$\theta^* = \arg\max_{\theta} \mathcal{L}(\theta). \tag{2.14}$$

Typically, if the model is differentiable with respect to the parameters (as is the case with latent variable models parameterized by neural networks), we can run gradient-based optimization to solve this optimization problem. In case we have access to the latent variable's posterior, we can write the gradient of this log-evidence as the following (derivation in Appendix A.1):

$$\nabla_\theta \mathcal{L}(x, \theta) = \sum_{i=1}^{N} \mathbb{E}_{z \sim p(z|x^{(i)};\theta)} \left[\nabla_\theta \log p(x^{(i)}, z; \theta) \right]. \tag{2.15}$$

2.2.2 Variational Inference

As mentioned before, maximizing the true log-evidence requires either differentiating over a marginalization or, if it is possible to do exact *posterior inference*, calculate the expectation of the gradient of the **log-likehood** over the latent variable's posterior. However, such computation is generally intractable when we are using deep neural network since we cannot find an analytical formulation for the posterior.

One way to overcome this is to instead do *aproximate* inference. To do so, we start by introducing a new (parameterized) distribution, $q(z|x;\phi) \approx p(z|x;\theta)$, the *approximate posterior*. We can then decompose the log-evidence into two terms (derivation in Appendix A.2):

$$\log p(x; \theta, \phi) = \mathbb{E}_{z \sim q(z|x;\phi)} \left[\log \frac{p(x, z; \theta)}{q(z|x; \phi)} \right] + D_{KL}\big(q(z|x;\phi) || p(z|x;\theta)\big). \tag{2.16}$$

The first term is known as the **evidence lower bound (ELBO)**, the second is a measure of the distance between the true posterior and the proposed approximate posterior. Since the Kullback-Leibler divergence is always positive, the first term is a lower bound on the log-evidence (hence the name), i.e.:

$$\log p(x; \theta, \phi) \geq \underbrace{\mathbb{E}_{z \sim q(z|x;\phi)} \left[\log \frac{p(x, z; \theta)}{q(z|x; \phi)} \right]}_{\text{ELBO}(x,\theta,\phi)}. \tag{2.17}$$

For different choices of $q(z|x;\phi)$ we get different ranges of approximations to evidence, and in the case where this distribution is equal to the true posterior, $q(z|x;\phi) = p(z|x;\theta)$, the approximation is perfect (however it is assumed that such approximation is intractable).

By observing Equation (2.16), we notice that maximizing the ELBO has two effects: it maximizes the log-likelihood, our end-goal, and minimizes the Kullback–Leibler divergence between the approximate posterior and the true posterior, leading to a better approximation of the latter by the former. Thus, we

define the objective function (parameterized by θ and ϕ) that we want to jointly optimize to be given by

$$\mathcal{L}(x, \theta, \phi) = \sum_{i=1}^{N} \text{ELBO}(x^{(i)}, \theta, \phi). \tag{2.18}$$

We can write the gradients of the ELBO with respect to the latent variable parameters and the variational parameters as:

$$\begin{aligned} \nabla_{\theta,\phi} \text{ELBO}(x, \theta, \phi) &= \nabla_{\theta,\phi} \mathbb{E}_{z\sim q(z|x;\phi)} \left[\log \frac{p(x, z; \theta)}{q(z|x; \phi)} \right] \\ &= \nabla_{\theta,\phi} \, \mathbb{E}_{z\sim q(z|x;\phi)} \left[\log p(x, z; \theta) - \log q(z|x; \phi) \right]. \end{aligned} \tag{2.19}$$

Gradient Estimation

We could try to compute these gradients directly by marginalizing over the approximate posterior distribution $q(z|x, \phi)$. However, this is impossible for most cases. If we use a continuous latent variable space, for all but the simplest models, the integral is intractable, whereas if we have a discrete space, the summation grows exponentially with the latent variable's dimension size, as is the case for causal structures we consider in this work.

For the model parameters θ, we can however obtain low-variance estimates for the gradients. Notice that the approximate posterior does not depend on the latent variables parameters θ, thus we get $\nabla_\theta \, \mathbb{E}_{z\sim q(z|x;\phi)} \left[\log q(z|x; \phi) \right] = 0$, and we can move the gradient inside the expectation of the log joint probability, i.e:

$$\nabla_\theta \, \mathbb{E}_{z\sim q(z|x;\phi)} \left[\log p(x, z; \theta) \right] = \mathbb{E}_{z\sim q(z|x;\phi)} \left[\nabla_\theta \log p(x, z; \theta) \right]. \tag{2.20}$$

We can use a simple Monte Carlo estimate of this gradient by sampling from the posterior, i.e, simply sample a few samples from the posterior and use the average resulting gradient. However getting a good estimate for the gradients with respect to the variational parameters ϕ, is a bit trickier. For this computation, we cannot move the gradient inside the expectation as that is taken with respect to the approximate posterior that explicitly depends on ϕ.

One way to go around this is to instead make use of the **score-function estimator** (also known as **REINFORCE** [63])(derivation in Appendix A.3):

$$\nabla_\phi \, \mathbb{E}_{z\sim q(z|x;\phi)}[f(z)] = \mathbb{E}_{z\sim q(z|x;\phi)}[f(z) \nabla_\phi \log q(z|x; \phi)], \tag{2.21}$$

where f is an arbitrary function. This allows us to write the gradient w.r.t. the variational parameters ϕ as the expectation:

$$\begin{aligned} \nabla_\phi \, \mathcal{L}(x, \theta, \phi) &= \mathbb{E}_{z\sim q(z|x;\phi)}[\big(\log p(x, z; \theta) - \log q(z|x; \phi) \big) \nabla_\phi \log q(z|x; \phi)] \\ &= \mathbb{E}_{z\sim q(z|x;\phi)} \left[\frac{\log p(x, z; \theta)}{\log q(z|x; \phi)} \nabla_\phi \log q(z|x; \phi) \right]. \end{aligned} \tag{2.22}$$

This gradient estimation is reminiscent of the policy gradient theorem [64] from the reinforcement learning literature. Therefore, the gradient estimation methods developed in that context such as using

Monte Carlo, popularized as the REINFORCE [63, 65], or other score function estimators, such as using baselines to reduce the variance of the estimator [66, 67], are commonly used.

Finding different methods for estimating these gradients has been a recent topic of interest in the literature. An exceedingly popular approach is the **pathwise gradient estimator** (or the reparametrization trick) [68, 69]. The idea is that we assume that $q(z|x;\phi)$ is a reparameterization $\mathcal{T}(\epsilon;\phi)$ of some base distribution $q_0(\epsilon)$. Since the expectation with the respect to $q_0(\epsilon)$ there is no dependence on the parameters ϕ we can write the gradient as follows :

$$\nabla_\phi \mathcal{L} = \sum_{i=1}^{N} \mathbb{E}_{\epsilon\sim q_0(\epsilon)} \left[\nabla_z \log \frac{p\big(z=\mathcal{T}(\epsilon;\phi), x_i;\theta\big)}{q\big(z=\mathcal{T}(\epsilon;\phi)|x_i,\phi\big)} \nabla_\phi \mathcal{T}(\epsilon;\phi) \right]. \tag{2.23}$$

The gradient is also estimated using Monte Carlo but notice that now we differentiate through the joint likelihood. This gradient estimation method has empirically been shown to yield much lower variance estimators compared to the score function methods [70]. Note that by doing this variable transformation we are able to differentiate through random operations.

However, such reparametrizations are only applicable to continuous latent variables. For the discrete case, we are forced to use the score-function estimator, apply a continuous relaxation [71–74] or construct the latent distribution such that it results in a sparse expectation [75].

Continuous Relaxations

Since the introduction of the reparametrization trick, there has been significant interest in applying it for discrete latent variables, particularly in the context of structured prediction where sometimes we cannot even evaluate a density (as required by the score-function estimator) due to complex partition functions. We now present two continuous relaxations that we employ in our work.

Gumbel-softmax. The Gumbel-softmax distribution [71, 73] also called concrete distribution is a continuous relaxation of a categorical distribution. It is based on the Gumbel-max trick [76, 77] that allows for sampling efficiently a categorical variable z with class probabilities π, in the following manner:

$$z = \arg\max_i \left(g_i + \log \pi_i\right) \quad , \tag{2.24}$$

where $g_1, \ldots, g_k$ are i.i.d samples drawn from the distribution Gumbel(0,1). However, instead of using arg max, a continuous approximation thereof is adopted, the softmax function. Therefore, the Gumbel-max trick coupled with the softmax and a temperature parameter τ can be written as follows:

$$y_i = \frac{\exp\Big(\big(g_i + \log(\pi_i)\big)/\tau\Big)}{\sum_{j=1}^{k} \exp\Big(\big(g_j + \log(\pi_j)\big)/\tau\Big)} \quad \text{for } i = 1, \ldots, k. \tag{2.25}$$

The samples from this distribution lie in the k-th dimentional probability simplex. As τ goes to zero, the softmax becomes the arg max function, and all the samples lie in the corners of the simplex. In this

extreme case, the gradient estimate's variance grows significantly. Typically, the temperature is annealed such that it is high at the beginning of training and close to zero at the end.

Straight-Through Gumbel-softmax Some sampling models require that the samples are discrete. In cases like this, where we cannot use a relaxed sample, we use the Straight-Through (ST) gradient estimator [78]. During the forward phase, we use a discretized version of the samples from the Gumbel-softmax distribution, and during the backward phase, we use the samples themselves. Therefore, given a sample y from the Gumbel-softmax distribution, we obtain the ST sample $\hat{y}$ as follows:

$$\hat{y} := \text{discretize}(y) + (y - \text{grad-block}(y)), \tag{2.26}$$

where the grad-block function is such that grad-block$(z) = z$ and ∇_zgrad-block$(z) = 0$ and discretize$(y)_i = 1$ if $y_i \geq \max_j y_j$ and is zero otherwise.

2.3 The Dirichlet Process

The Dirichlet process, first introduced in [79], DP(α, P_0), is a distribution of distributions. It has two parameters, $\alpha > 0$ (the scaling parameter) and P_0 (the base distribution). A random distribution P of the variable η is distributed according to a Dirichlet process if, for all natural numbers k and partitions $\{A_0, A_1, A_2, \ldots, A_k\}$, the following holds:

$$\big(P(A_0), P(A_1), \ldots, P(A_k)\big) \sim \text{Dir}\big(\alpha P(A_0), \alpha P(A_1), \ldots, \alpha P(A_k)\big). \tag{2.27}$$

To obtain a distribution over n variables η, we can marginalize w.r.t G as follows:

$$p(\eta_0, \eta_1, \ldots, \eta_n) = \int p(P) \prod_{i=0}^{n} p(\eta_i|P) dP. \tag{2.28}$$

The marginalization, inevitably creates a dependency between successive ηs. The resulting joint distribution factorizes as the product of the following full conditionals:

$$p(\eta|\eta_0, \eta_1, \ldots, \eta_n) = \frac{\alpha}{\alpha + n} p(\eta|P_0) + \frac{1}{\alpha + n} \sum_{i=0}^{n} \delta_{\eta_i}(\eta). \tag{2.29}$$

This is what is generally called the *Chinese restaurant process* [80] representation of the Dirichlet process. Under this interpretation, for each new costumer that is given a new table, here denoted by η, the particular table is drawn according to the base distribution P_0. However, if the costumer does not go to a new table, the probability of the costumer going to a particular existing table is proportional to the number of costumers already in that table. This evidently creates a clustering effect. The probability of the costumer being drawn a new table, if we denote by n the number of existing tables, is $\frac{\alpha}{\alpha+n}$. This probability shrinks as more costumers are already in the restaurant.

While this formulation of the Dirichlet process is intuitive, in order to use variational inference with it,

we would have to use Kullback–Leibler divergences between distributions that are both continuous and discrete. For this reason, we employ the *stick-breaking process* representation from [81]. Consider two infinite collection of independent random variables $v_k \sim \text{Beta}(1, \alpha)$ and $\eta_k \sim P_0$, for $k = \{0, 1, 2, \ldots\}$. The stick-breaking process representation of P is as follows:

$$\begin{aligned} \beta_k(v_k, v_{k-1}, \ldots, v_1) &= v_k \prod_{k'=0}^{<k} (1 - v_{k'}); \\ G &= \sum_{k=0}^{\infty} \beta_k(v_k, v_{k-1}, \ldots, v_1)\delta_{\eta_k}(\eta). \end{aligned} \tag{2.30}$$

With this representation, we can see that P is discrete (with probability one), and the support consists of a countably infinite set of atoms drawn independently from P_0. For now on, we use β_k to refer to $\beta_k(v_k, v_{k-1}, \ldots, v_1, v_0)$. We will denote the set of random variables $\{v_0, v_1, v_2, \ldots, v_k\}$ by $\mathcal{V}^k$.

The Dirichlet process mixture model Under this hierarchical Bayesian model, the DP is used as a non-parametric prior. The collection of variables $\{\beta_0, \beta_1, \ldots\}$ act as the mixing proportions and $\{\eta_0, \eta_1, \ldots\}$ are the atoms representing the mixture components. If we then define variable $z^{(i)}$ to be an assignment variable of the mixture component with which the data point $x^{(i)}$ is associated, the sampling process can be described as follows:

1. Draw $v_k|\alpha \sim \text{Beta}(1, \alpha)$, $k = \{0, 1, \ldots\}$
2. Draw $\eta_k \sim P_0$, $k = \{0, 1, \ldots\}$
3. For the i^{th} sample:
   - Draw $z^{(i)}|\{v_0, v_1, \ldots\} \sim \text{Cat}(\beta)$
   - Draw $\eta^{(i)}|z, \eta_0, \eta_1, \ldots \sim \sum_{k=0}^{\infty} \delta_{\eta_k}(\eta^{(i)})^{z_k}$
   - Draw $x^{(i)}|\eta^{(i)} \sim p(x^{(i)}|\eta^{(i)})$

The joint distribution can be factorized as follows:

$$\begin{aligned} p(x, z, \eta, &\eta_0, \eta_1, \ldots, v_0, v_1, \ldots|\alpha, P_0) = \\ &p(x|\eta)p(\eta|z, \eta_0, \eta_1, \ldots)p(z|v_0, v_1, \ldots) \prod_{k=0}^{\infty} p(v_k|\alpha) \prod_{k=0}^{\infty} p(\eta_k|P_0) \end{aligned} \tag{2.31}$$

The Dirichlet process mixture model is popular as a model-based approach for clustering. Primarily because it can be used when the number of clusters is unknown and is to be inferred from data.

Truncated Dirichlet process For practical purposes, it might not be possible to work with the stick-breaking formulation due to the infinity of mixture components. For this reason we can truncate the Dirichlet process to have only $K+1$ components by defining an atom with subscript $_{>K}$, that aggregates

and models all of the components greater than K. The probability of this component is given by:

$$\beta_{>K} = \sum_{k=K+1}^{\infty} \beta_k = \prod_{k=0}^{K} (1 - v_{k'}) = \beta_K \frac{(1 - v_K)}{v_K}. \tag{2.32}$$

Chapter 3

Intervention Recovery

In this chapter, we consider the problem and propose an algorithmic solution for identifying latent interventions in an unsupervised manner when given the underlying causal graph. We begin by defining a joint distribution over the unobserved intervention assignments and the collected data. We then use variational inference to approximate the posterior over intervention assignments and to learn the parameters of an SCM. Lastly, we experimentally validate our algorithm using a range of synthetic data sets.

3.1 A joint distribution for latent interventions

3.1.1 Mixture of intervention distributions

We assume that the dataset $\mathcal{D}$ is produced by a mixture of SCMs, each resulting from an intervention applied to a base SCM $\mathcal{M}$. More specifically, $\mathcal{D}$ is partitioned into $K+1$ exchangeable groups, with group k containing i.i.d. samples from the intervention SCM $\tilde{\mathcal{M}}^{(k)}$ resulting from applying the k^{th} intervention to the base SCM $\mathcal{M}$ with causal graph $\mathcal{G}$; the index $k = 0$ indicates the absence of intervention, *i.e.*, $\tilde{\mathcal{M}}^{(0)} = \mathcal{M}$ (observational model). We denote by $\tilde{\mathcal{M}} = (\tilde{\mathcal{M}}^{(0)}, \dots, \tilde{\mathcal{M}}^{(K)})$ the ensemble of SCMs.

The latent variables $z^{(i)} \in \{0, \dots, K\}$ indicate which SCM generated each sample: $z^{(i)} = k$ if and only if $x^{(i)}$ is a sample of the SCM $\tilde{\mathcal{M}}^{(k)}$. We denote by $\mathcal{Z} = (z^{(1)}, \dots, z^{N})$ the set of *correspondences*. We call the scenario where both $\tilde{\mathcal{M}}$ and $z^{(i)}$ are unknown as *fully latent interventions*.

Marginalizing with respect to the latent $z^{(i)}$ yields the mixture model

$$p(x^{(i)}|\tilde{\mathcal{M}}) = \sum_{k=0}^{K} p(z^{(i)} = k)\; p(x^{(i)}|z^{(i)} = k, \tilde{\mathcal{M}}) = \sum_{k=0}^{K} \tau_k \; p_{\tilde{\mathcal{M}}^{(k)}}(x^{(i)}), \tag{3.1}$$

where we define $\tau_k = p(z^{(i)} = k)$. Conditioning on $z^{(i)}$ and invoking Equation 2.3 leads to

$$p(x^{(i)}|z^{(i)}, \tilde{\mathcal{M}}) = \sum_{k=0}^{K} \mathbb{I}(z^{(i)} = k) \prod_{j \notin I^{(k)}} p_{\mathcal{M}}(x_j^{(i)}|\mathbf{PA}_j^{\mathcal{G}}) \prod_{j \in I^{(k)}} p_{\mathcal{M};do\left(x_j := \tilde{\zeta}_j^k(\mathbf{PA}_j^{\mathcal{G}}, \tilde{\varepsilon}_j)\right)}(x_j^{(i)}|\mathbf{PA}_j^{\mathcal{G}}).$$

We also consider that, like the group memberships, the set of targets $I^{(k)}$ of each intervention k is unknown (except for $I^{(0)} = \emptyset$).

3.1.2 Intervention embeddings and shared intervention space

We use density estimators, *e.g.*, neural networks and normalizing flows [82], to model the conditional densities in both the observational and interventional distributions. With this goal in mind, we use an appropriate encoding of the changes in the intervened assignments and the intervention targets. The set of targets in the k^{th} intervention $\tilde{\mathcal{M}}^{(k)}$ is indicated by a d-dimensional binary vector $r_k = [r_{k1}, \ldots, r_{kd}]$, where $r_{kj} = 1$ if and only if $j \in I^{(k)}$, and $r_{kj} = 0$ otherwise. Since $I^{(0)}$ has no targets (it corresponds to the observational SCM $\mathcal{M}$), we have $r_0 = [0, 0, \ldots, 0]$. To encode the type of intervention, we introduce the *intervention embedding vector* $u_k \in \mathbb{R}^h$, where h is an hyper-parameter. The vector u_k represents the changes in the affected assignments for intervention $\tilde{\mathcal{M}}^{(k)}$. We denote by $\mathcal{R} = [r_0, r_1, \ldots, r_K] \in \mathbb{R}^{K \times d}$ the matrix of intervention targets, and by $\mathcal{U} = [u_0, u_1, \ldots, u_K] \in \mathbb{R}^{K \times h}$ the matrix of intervention embeddings. We use $(\mathcal{R}, \mathcal{U})$ as the representation of the interventions $\tilde{\mathcal{M}}$. We represent the causal graph $\mathcal{G}$ via the adjacency matrix $A^{\mathcal{G}} \in \{0, 1\}^{d \times d}$.

Putting everything together, when given the graph $\mathcal{G}$, interventions $\tilde{\mathcal{M}}$, indicator z, and assuming the intervention is imperfect, the log-probability of single data-point x is given by

$$
\begin{aligned}
&\log p(x|z, \tilde{\mathcal{M}}, \mathcal{G}; \theta) = \\
&= \sum_{k=0}^{K} \mathbb{I}(z = k) \sum_{j=1}^{d} (1 - r_{kj}) \log \underbrace{g_j(x_j | A_j^{\mathcal{G}} \odot x, u_0; \theta_j)}_{=:\, p_{\mathcal{M}}(x_j | \mathbf{PA}_j^{\mathcal{G}})} + r_{kj} \log \underbrace{g_j(x_j | A_j^{\mathcal{G}} \odot x, u_k; \theta_j)}_{=:\, p_{\tilde{\mathcal{M}}^{(k)}}(x_j | \mathbf{PA}_j^{\mathcal{G}})} \\
&= \sum_{j=1}^{d} \log g_j(x_j | A_j^{\mathcal{G}} \odot x, (e_z^\top \mathcal{R})_j \; (e_z^\top \mathcal{U} - u_0) + u_0; \theta_j),
\end{aligned}
\tag{3.2}
$$

where $e_z \in \mathbb{R}^K$ is a one-hot vector indicating z, and $A_j^{\mathcal{G}} \odot x$ is the Hadamard (elementwise) product between the j^{th} column of the adjacency matrix of $\mathcal{G}$ and x, which is equivalent to selecting the entries of x in $\mathbf{PA}_j^{\mathcal{G}}$. The parameters $\theta = (\theta_1, \ldots, \theta_d)$ parameterize the conditional densities $g_1, \ldots, g_d$. We will consider in the sequel several forms for these conditional densities, *e.g.*, using parametric families and normalizing flows. Crucially, each conditional density g_j is a distribution of x_j, and the parameters θ_j are *shared* between all of the interventions—only the intervention-specific intervention embedding vector u_k changes depending on the intervention. This enables dealing with an unlimited number of interventions, as we shall see.

Conditional density for perfect interventions. If we assume that the intervention is perfect (stochastic or atomic), the log-probability of single data-point is given by

$$
\log p(x|z, \tilde{\mathcal{M}}, \mathcal{G}; \theta) = \sum_{j=1}^{d} \log g_j(x_j | (e_z^\top \mathcal{R})_j \; \left(A_j^{\mathcal{G}} \odot x\right), (e_z^\top \mathcal{R})_j \; (e_z^\top \mathcal{U} - u_0) + u_0; \theta_j),
\tag{3.3}
$$

that is, we make the conditional density of x_j be only dependent on u_k when intervened.

3.1.3 Modeling the conditional densities

In Section 3.1.2, we described the conditional probability of particular data-point x in terms of the parametric conditional densities $g_1, \ldots, g_d$. There is no limitation in our formulation that would restrict us to a particular family of sampling models g_j.

A simple nonlinear model for the conditional densities g_j can be constructed using neural networks. We use a neural network $\text{NN}([u_k, A_j^{\mathcal{G}} \odot x]; \theta_j) : \mathbb{R}^{(h+d)} \to \mathbb{R}^m$, a parametric non-linear mapping, parameterized by θ_j, that receives the concatenation of the parents of x_j and the intervention embedding u_k and outputs m parameters of some distribution $f(x_j; \text{NN}([u_k, A_j^{\mathcal{G}} \odot x]; \theta_j))$ of the variable x_j. There are many possible choices for the distribution f, depending on the problem at hand and on whether x_j is discrete or continuous: Poisson ($m = 1$), Bernoulli ($m = 1$), univariate Gaussian ($m = 2$), categorical, etc. In this paper, we focus on three density families in $\mathbb{R}$, which we experiment with in Section 4.5.

Linear Gaussian We use a neural network $\text{NN}(u_k; \theta_j)$ to output coefficients $\tilde{a}_j \in \mathbb{R}^d$, and $\tilde{\sigma}_j, \tilde{b}_j \in \mathbb{R}$. Then, we use these as parameters of a Gaussian distribution whose mean is an affine transformation of the values of the parents of x_j:

$$g_j(x_j | A_j^{\mathcal{G}} \odot x, u_k) = \mathcal{N}\big(\tilde{a}_j^\top \big(A_j^{\mathcal{G}} \odot x\big) + \tilde{b}_j, \tilde{\sigma}_j^2\big).$$

Non-Linear Gaussian We use a neural network $\text{NN}([u_k, A_j^{\mathcal{G}} \odot x]; \theta_j)$ to output coefficients $\tilde{\mu}_j \in \mathbb{R}$ and $\tilde{\sigma}_j \in \mathbb{R}$, given the values of the parents of x_j as input. Then, we use these as parameters of a Gaussian distribution:

$$g_j(x_j | A_j^{\mathcal{G}} \odot x, u_k) = \mathcal{N}\big(\tilde{\mu}_j, \tilde{\sigma}_j^2\big).$$

Normalizing flows To model non-linear non-Gaussian conditional densities, we employ normalizing flows. A normalizing flow [82] is a transformation of a base probability density (in our case a Gaussian) through a sequence of invertible mappings $\tau(x_j; \tilde{W}_j) = \tau_l \circ \tau_{l-1} \cdots \circ \tau_1(x_j; \omega_1)$, where $\tilde{W}_j = \{\omega_1, \ldots, \omega_l\}$. We use a model introduced in [83], called deep sigmoidal flows (DSF), where each of the invertible mappings has the following form:

$$\tau_l(x) = \sigma^{-1}(w_l^\top \sigma(a_l x + b_l)), \qquad w_l \in \triangle_{F-1},\ a_l \in \mathbb{R}_+^F,\ b_l \in \mathbb{R}^F.$$

where $\triangle_{F-1}$ is the probability simplex and F is an hyper-parameter. We use a neural network $\text{NN}([u_k, A_j^{\mathcal{G}} \odot x]; \theta_j)$ to output the parameters $\tilde{W}_j$. With the former, we obtain a controllable flow $\tau(x_j; \tilde{W}_j)$ that when given x_j, outputs the parameters of a Gaussian distribution $\tilde{\mu}$, $\tilde{\sigma}$. Altogether, the joint density has the

following form:

$$g_j(x_j|A_j^{\mathcal{G}} \odot x, u_k) = \left| \det \left(\frac{\partial \tau(x_j; \tilde{W}_j)}{\partial x_j} \right) \right| \mathcal{N}\left(\tilde{\mu}, \tilde{\sigma}^2\right).$$

3.1.4 Modeling latent interventions with Dirichlet processes

To obtain a complete statistical description of the data-generating process, we still need to design a prior distribution for the latent interventions, apart from the sampling model g_j. Namely, we need a prior distribution for the correspondence z, the intervention embeddings $\mathcal{U}$, and the intervention targets $\mathcal{R}$. Furthermore, while experimentally, we can design scenarios where K, the number of latent interventions, is known, in general, given a data set, it will not be clear what the number of latent interventions is. Therefore, we will formulate the model to support a potentially non-specified number of latent interventions K.

We do this by using a *Dirichlet process prior* with a stick-breaking process representation (as described in Section 2.3) as the prior distribution of the variables associated with the latent interventions. We consider the base distribution $P_0(u_k, r_k)$, with the following form:

$$P_0(u_k, r_k) \doteq \mathcal{N}(u_k; 0, I_h) \prod_{j=1}^{d} \text{Bern}\big(r_{kj}; \sigma(\gamma)\big). \tag{3.4}$$

In this way, associated with each mixture component k, or intervention distribution, we have both the intervention targets r_k, which are assumed to come from d independent Bernoulli and a intervention embedding u_k, which is assumed to come from a h-dimensional standard Gaussian. With the stick-breaking weights $v_0, v_1, \ldots$ we obtain the mixing coefficients $\beta_0, \beta_1, \ldots$ which serve as the prior distribution for the categorical variable z.

The generative story is as follows. For $k = 0, 1, \ldots,$ we sample the variables u_k and r_k, associated with each intervention $\tilde{\mathcal{M}}^{(k)}$, as well as the probability β_k of picking that intervention as a stick-breaking process, with scaling parameter $\alpha > 0$ (which controls the clustering effect of the Dirichlet process) and hyperparameter γ (which controls the sparsity of the intervention targets), as follows:

$$\begin{aligned} u_k &\sim \mathcal{N}(0, I_h) \\ r_{kj} &\sim \text{Bern}(\sigma(\gamma)), \quad j = 1, \ldots, d \\ \beta_k &= v_k \prod_{k'=0}^{k-1} (1 - v_{k'}), \quad \text{with } v_k \sim \text{Beta}(1, \alpha). \end{aligned} \tag{3.5}$$

Then, to generate the data, we first sample the intervention index $z^{(i)}$, and then sample a point $x^{(i)}$

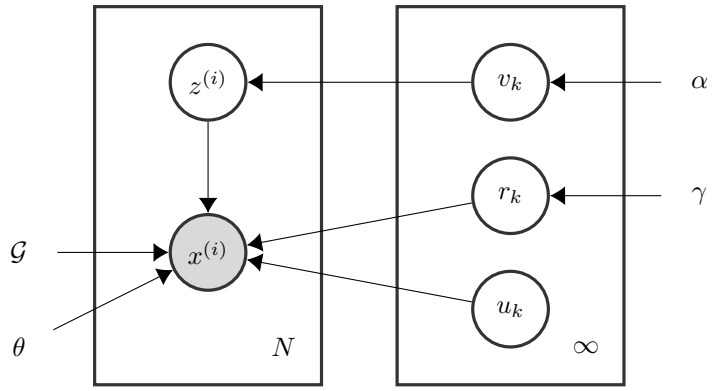

Figure 3.1: Graphical model representation of the Dirichlet Process Mixture model as applied to the latent intervention identification problem.

conditioning on the corresponding intervention $\tilde{\mathcal{M}}^{(z^{(i)})}$:

$$
\begin{aligned}
z^{(i)} &\sim \mathrm{Cat}(\beta_0, \beta_1, \ldots, \beta_k, \ldots) \\
x_j^{(i)} &\sim \begin{cases} g_j(x_j | A_j^{\mathcal{G}} \odot x^{(i)}, u_0) & \text{if } r_{z^{(i)}j} = 0 \\ g_j(x_j | A_j^{\mathcal{G}} \odot x^{(i)}, u_{z^{(i)}}) & \text{if } r_{z^{(i)}j} = 1. \end{cases}
\end{aligned} \tag{3.6}
$$

Figure 3.1 contains a graphical model representation of this joint distribution. The joint distribution this model is written as:

$$
p(\mathcal{D}, \mathcal{Z}, \tilde{\mathcal{M}} | \mathcal{G}; \theta) = \prod_{i=1}^{N} \underbrace{p(x^{(i)} | z^{(i)}, \tilde{\mathcal{M}}, \mathcal{G}; \theta)}_{p_{\tilde{\mathcal{M}}^{(z^{(i)})}}(x^{(i)})} \underbrace{p(z^{(i)} | \beta_0, \beta_1, \ldots)}_{p(z^{(i)} | \tilde{\mathcal{M}})} \underbrace{\prod_{k=0}^{\infty} p(u_k) p(r_k | \gamma) p(v_k | \alpha)}_{p(\tilde{\mathcal{M}})}. \tag{3.7}
$$

3.2 Approximate posterior inference

Given a collection of data $\mathcal{D} = \{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$, our goal is to recover the associated set of intervention correspondences $\mathcal{Z} = \{z^{(1)}, z^{(2)}, \ldots, z^{(N)}\}$, and the intervention family $\tilde{\mathcal{M}}$. In general, the posterior $p(\mathcal{Z}, \tilde{\mathcal{M}} | \mathcal{D}, \mathcal{G})$ is impossible to obtain in closed form, thus we resort to variational inference [84]. Following Section 2.2.2, we take an optimization approach by seek an approximate distribution $q^\star(\mathcal{Z}, \tilde{\mathcal{M}})$, the closest (in the Kullback–Leibler – KL – sense) approximation to the posterior in a variational family $\mathcal{Q}$. We design the family $\mathcal{Q}$ of tractable variational distributions $q(\mathcal{Z}, \tilde{\mathcal{M}}) = p(\tilde{\mathcal{M}}) \prod_{i=1}^{N} q(z^{(i)} | \tilde{\mathcal{M}})$. Where for the variables associated with latent interventions $\tilde{\mathcal{M}}$ we propose the fully factorized and finite variational distribution

$$
q(\tilde{\mathcal{M}}) = \prod_{k=0}^{K} \Big(\prod_{j=1}^{d} q_R(r_{kj}) \Big) \Big(\prod_{l=1}^{h} q_U(u_{kl}) \Big) q_V(v_k), \tag{3.8}
$$

where here the hyper-parameter K defines the truncation level of the variational approximation, and the distributions associated with v_k, u_{kl}, and r_{kj} take the following form:

$$\begin{aligned} q_V(v_k; \rho_k, w_k) &= \mathsf{Beta}\big(\rho_k w_k, (1-\rho_k) w_k\big); \\ q_U(u_{kl}; \mu_{kl}, \sigma_{kl}) &= \mathcal{N}(\mu_{kl}, \sigma_{kl}^2); \\ q_R(r_{kj}; \pi_{kj}) &= \mathsf{Bern}(\pi_{kj}). \end{aligned} \tag{3.9}$$

where $\pi_{kj}, \mu_{kl}, \sigma_{kl}, \rho_k, w_k$ are free parameters to be optimized, for each $k \in \{0, \ldots, K\}$, $l \in \{1, \ldots, h\}$ and $j \in \{1, \ldots, d\}$. For the distribution of interventional assignments z, we propose the following variational posterior:

$$q_Z(z) \propto \exp\left(\frac{u_z^\top \mathsf{NN}(x; \phi_Z)}{\sqrt{h}}\right), \qquad k = 1, \ldots, K\ , \tag{3.10}$$

where $\mathsf{NN}(x; \phi_Z) : \mathbb{R}^d \to \mathbb{R}^h$ is a neural network. In Appendix B we describe the architecture of the neural networks used in our experiments. We use the shorthand ϕ to denote the vector of all variational parameters, which includes $\phi_Z, \mu_{kl}, \sigma_{kl}, \rho_k, w_k, \pi_{kj}$ for all $k \in \{0, \ldots K\}$, $l \in \{1, \ldots, h\}$, and $j \in \{1, \ldots, d\}$.

For the choice of variational posterior $q(\tilde{\mathcal{M}})$, we opted by the mean-field variational family for tractability reasons. However, why choose the variational posterior $q_Z(z)$ as stated in Equation 3.10?

Statistical arguments Without making further assumption, the true posterior $p(z^{(i)}, |\tilde{\mathcal{M}}, x^{(i)}, \mathcal{G}; \theta)$ should be considered dependent not only on $x^{(i)}$, but all of the variables associated with $\tilde{\mathcal{M}}$. So, having an approximate posterior that depends on the latent embeddings, not exclusively on $x^{(i)}$, should help reduce the gap between the ELBO and the log-likelihood.

Computational arguments The approximate distribution $q_Z(z)$ is constructed to allow a shared global model for distinct interventions, whose parameters do not scale with the truncation level K, in the same manner attention weights are computed in sequence modeling. In this analogy, the sequence's elements are the intervention embeddings. Furthermore, like attention mechanisms, if we reorder $\mathcal{U}$, the discrete distribution $q_Z(z)$ remains the same. Meaning, that the vectors u_k, in the first ordering, and $u_{k'}$, in the second ordering, retain the same probability (it is a operation on sets).

Modeling arguments We interpret the transformation $\mathsf{NN}(x; \phi_Z)$ from Equation 3.10, as a prediction for the intervention embedding $\hat{u}_i$ that generated x_i. Therefore, the probability of intervention k being assigned to the i^{th} sample is higher the more compatible $\hat{u}_i$ and u_k are, as measured using the dot-product kernel. We conjecture, that the free parameters μ_k, and σ_k associated with the variational

posterior $q_U\big(u_k; \mu_k, \text{diag}(\sigma_k)\big)$ of the k^{th} intervention, will converge to the equilibrium point:

$$\mu_k = \Big(\sum_{i=1}^{N} q_Z(k; x_i)\ \text{NN}(x_i; \phi_Z) \Big), \qquad \sigma_k^2 = \Big(\sum_{i=1}^{N} q_Z(k; x_i) \text{NN}(x_i; \phi_Z)^2 \Big) - \mu_k^2. \tag{3.11}$$

So we are implicitly doing fuzzy clustering in the intervention latent space.

Given the ingredients above, we obtain the following lower bound for the marginal likelihood from Equation 4.2:

$$\log p(\mathcal{D}|\mathcal{G}; \theta) \geq \sum_{i=1}^{N} \underbrace{\mathbb{E}_{z^{(i)}, \tilde{\mathcal{M}} \sim q(z^{(i)}, \tilde{\mathcal{M}}; \phi)} \big[\log p(x^{(i)}|z^{(i)}, \tilde{\mathcal{M}}, \mathcal{G}; \theta) \big] - D_{KL}\big[q(z^{(i)}, \tilde{\mathcal{M}}) || p(z^{(i)}, \tilde{\mathcal{M}}) \big]}_{\text{ELBO}_{q(z^{(i)}, \tilde{\mathcal{M}}; \phi)}(x^{(i)}, \mathcal{G}; \theta)}. \tag{3.12}$$

For different choices of ϕ, we get different lower bound approximations to the marginal likelihood. By maximizing the ELBO w.r.t. ϕ we minimize the approximation gap, which equals he Kullback–Leibler divergence between the approximate and the true posterior.

The Kullback–Leibler divergence present in Equation 3.12 can be written in the following way :

$$\begin{aligned}
&D_{KL}\big[q(\tilde{\mathcal{M}}, z^{(i)}) || p(\tilde{\mathcal{M}}, z^{(i)}) \big] = \\
&\quad = \mathbb{E}_{\tilde{\mathcal{M}}, z \sim q(\tilde{\mathcal{M}}, z)} \left[\log \frac{q(z|u_0, \ldots, u_T) \prod_{k=0}^{T} \Big(\prod_{j=1}^{d} q_R(r_{kj}) \Big) \Big(\prod_{l=1}^{h} q_U(u_{kl}) \Big) q_V(v_k)}{p(z|\beta_0, \ldots, \beta_T) \prod_{k=0}^{T} \Big(\prod_{j=1}^{d} p(r_{kj}|\gamma) \Big) \Big(\prod_{l=1}^{h} p(u_{kl}) \Big) p(v_k|\alpha)} \right] \\
&\quad = \sum_{k=0}^{T} \sum_{l=1}^{h} D_{KL}\big(q(u_{kl}) || \mathcal{N}(0,1) \big) + \sum_{k=0}^{T} \sum_{j=1}^{d} D_{KL}\big(q(r_{kj}) || \text{Bern}(\gamma) \big) \\
&\quad + \sum_{k=0}^{T} D_{KL}\big(q(v_k) || p_\theta(v_k|\alpha) \big) + \mathbb{E}_{\tilde{\mathcal{M}}, z \sim q(\tilde{\mathcal{M}}, z)} \left[\log \frac{q(z|u_0, \ldots, u_T)}{p(z|\beta_0, \ldots, \beta_T)} \right].
\end{aligned} \tag{3.13}$$

The closed form expressions, for each of the resulting Kullback–Leibler divergences is contained in Table 3.1. Derivations are contained in Appendix A.

$D_{KL}\big(q(v_k) \|\| p_\theta(v_k\|\alpha) \big)$	$\log\Big(\frac{\mathrm{B}\big(\rho_k w_k, (1-\rho_k) w_k\big)}{\mathrm{B}(1, \alpha)} \Big) + (1 - \rho_k w_k)\psi(1) + \big(\alpha - (1-\rho_k) w_k\big)\psi(\alpha) + (-1 + w_k - \alpha)\psi(1+\alpha)$
$D_{KL}\big(q(u_{kl}) \|\| \mathcal{N}(0,1) \big)$	$\frac{\sigma_{kl}^2 + \mu_{kl}^2 - 1}{2} - \log \sigma_{kl}$
$D_{KL}\big(q(r_{kj}) \|\| \text{Bernoulli}(\gamma) \big)$	$\pi_{kj}\Big(\text{logit}(\pi_{kj}) - \text{logit}(\gamma) \Big) + \log \frac{1 - \pi_{kj}}{1 - \gamma}$
$\mathbb{E}_{\tilde{\mathcal{M}} \sim q(\tilde{\mathcal{M}})} \Big[D_{KL}\big(q(z) \|\| p_\theta(z\|\beta_0, \beta_1, \ldots, \beta_T) \big) \Big]$	$\sum_{k=0}^{K} \mathbb{E}_{u_k \sim q(u_k)} \Big[q(z_k) \big(\log q(z_k) - \mathbb{E}_{\mathcal{V}^k \sim q(\mathcal{V}^k)} [\log \beta_k] \big) \Big]$

Table 3.1: On the left, Kullback–Leibler divergences between the prior distribution of the latent variables and the corresponding variational posteriors. On the right the corresponding close-form expression. The function $\mathrm{B}(a, b)$ is the beta function, ψ is the digamma function, and $\mathbb{E}_{\mathcal{V}^k \sim q(\mathcal{V}^k)} [\log \beta_k]$ is given in closed form in A.10.

Algorithm 1 Intervention recovery algorithm based on amortized variational inference.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
repeat
  $x^{(1)} \dots x^{(B)} \leftarrow$ Random minibatch of B datapoints. (drawn from $\mathcal{D}$)
  for each $x^{(i)} \in x^{(1)} \dots x^{(B)}$ **do**
    $q(u^{(i)}), q(r^{(i)}), q(v^{(i)}), q(z^{(i)}) \leftarrow$ Compute the variational posteriors
    $u_0^{*(i)}, \dots, u_T^{*(i)}, r_0^{*(i)}, \dots, r_T^{*(i)}, z^{*(i)} \leftarrow$ Sample via the reparameterization trick.
    $\mathbf{g^{(i)}} \leftarrow \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, u_0^{*(i)}, \dots, u_T^{*(i)}, r_0^{*(i)}, \dots, r_T^{*(i)}, z^{*(i)}; x^{(i)}, A^{\mathcal{G}})$
  end for
  $\mathbf{g} \leftarrow \frac{1}{B} \sum_{i=1}^{B} \mathbf{g^{(i)}}$ (Average gradients)
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Update parameters using gradients g (with SGD or Adam)
until convergence of $(\boldsymbol{\theta}, \boldsymbol{\phi})$
return $\boldsymbol{\theta}, \boldsymbol{\phi}$

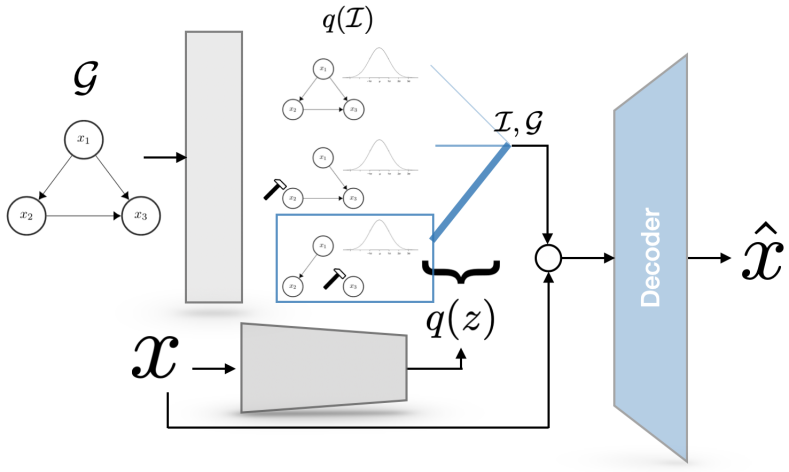

Figure 3.2: A diagram of the variational model we propose for the intervention identification problem.

3.3 Inference Algorithm

We use mini-batch gradient descent to maximize the ELBO w.r.t to the model parameters θ and the variational parameters ϕ. We estimate the gradients using path-wise gradient estimation method described in Sub-Section 2.2.2. This amounts to sampling $z^{*(i)}, \tilde{\mathcal{M}}^*$ from the variational posterior using the reparametrization trick, and minimizing the following objective function:

$$\mathcal{L}(\theta, \phi, z^{*(i)}, \tilde{\mathcal{M}}^*; x^{(i)}, \mathcal{G}) = -\log p(x^{(i)}, z^{*(i)}, \tilde{\mathcal{M}}^*, \mathcal{G}; \theta) + \Omega(\phi) \tag{3.14}$$

where $\Omega(\phi)$ is a representation of the sum of the Kullback–Leibler divergence terms present in Table 3.1. To employ the reparametrization trick with distributions of discrete variables we use the Gumbel-softmax continuous relaxation, also presented in 2.2.2. Having estimated the gradients, for each sample, we average them and feed it to a first order stochastic optimization algorithm Adam [85].

The proposed learning algorithm is described in Algorithm 1. We implemented the Algorithm with the PyTorch deep learning framework [2]. With PyTorch we define our model and calculate the gradient estimates using reverse-mode automatic differentiation. Figure 3.2 contains a diagram of the proposed model.

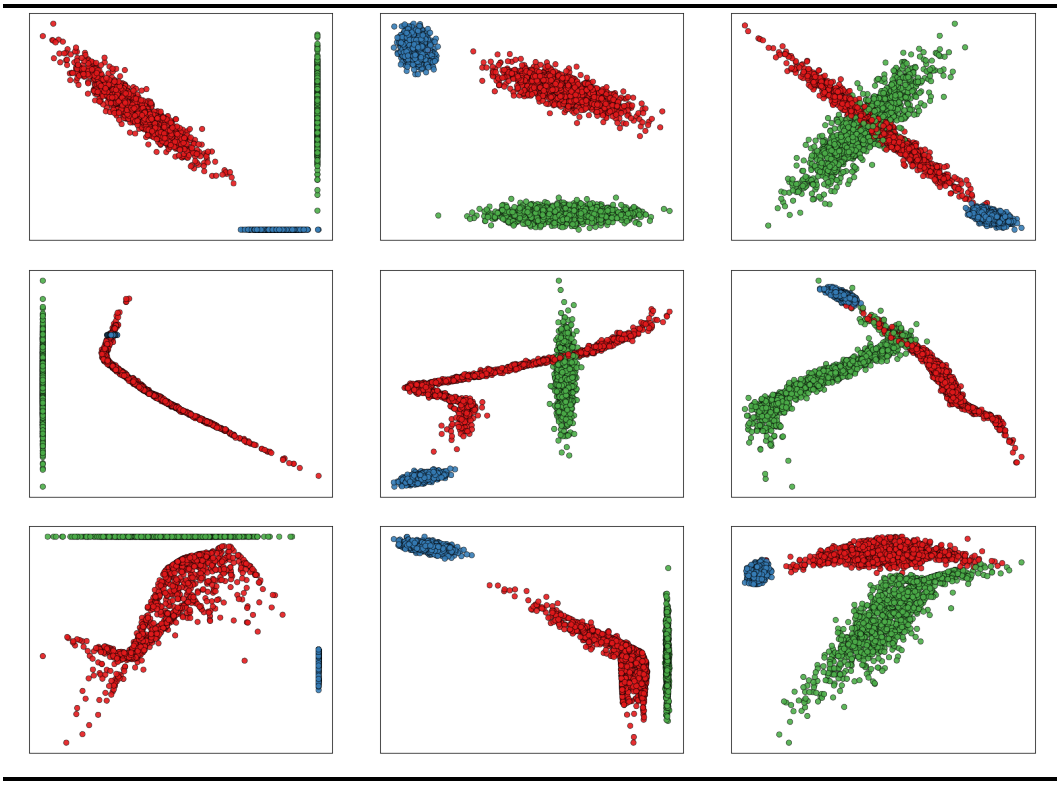

Figure 3.3: The underlying graph is has two variables and one edge. The cause is in the horizontal axis, the effect is in the vertical axis. The rows correspond to different models, particularly, Linear Gaussian, non-linear Gaussian and non-linear non-Gaussian. The columns correspond to distinct types of interventions. Particularly, atomic, stochastic and imperfect. Red observational. Green intervention on the effect variable. blue intervention on the causal variables.

3.4 Experimental Setup

We tested our method on several synthetic data. The synthetic datasets allow us to do a systematic, controlled comparison of different methods under different scenarios (graph size and density, intervention and assignment types). In order to generate SCMs, perform interventions on them and sample the corresponding entailed distributions, we created an open source Python package named PyCausal, which is available on GitHub[1].

We evaluate our method using the rand index measure. The rand index [86] is a measure of similarity between two data clustering. Let $\mathcal{Z}^{(1)}$ and $\mathcal{Z}^{(2)}$ be two partitions of the set X, a be the number of pairs of elements in X that are in the same subset in $\mathcal{Z}^{(1)}$ and the same subset in $\mathcal{Z}^{(2)}$, and b be the number of pairs of elements in X that in different subsets in $\mathcal{Z}^{(1)}$ and in different subset $\mathcal{Z}^{(2)}$, rand index measure is defined as follows:

$$R \doteq \frac{a+b}{\binom{|X|}{2}} \tag{3.15}$$

Single-node interventions We generated SCMs with $d \in \{10, 20\}$ variables with a Erdős-Rényi scheme with expected number of edges per node $e \in \{1, 4\}$. We provide a detailed description of how we generate SCMs in Appendix D. We generated 10 SCMs for each combination of e, conditional

[1] https://github.com/goncalorafaria/PyCausal

Model Type	d = 10		d = 20	
	ER1	ER4	ER1	ER4
	Stochastic Interventions:			
Linear Gaussian	0.9816 ± 0.019	0.9492 ± 0.007	0.9815 ± 0.005	0.9785 ± 0.006
Non-Linear Gaussian	0.9643 ± 0.015	0.9242 ± 0.006	0.9655 ± 0.005	0.9431 ± 0.003
Non-Linear Non-Gaussian	0.8314 ± 0.062	0.7812 ± 0.1074	0.8929 ± 0.027	0.8412 ± 0.087
	Imperfect Interventions:			
Linear Gaussian	0.9262 ± 0.030	0.8802 ± 0.040	0.8072 ± 0.098	0.7819 ± 0.054
Non-Linear Gaussian	0.9541 ± 0.012	0.9205 ± 0.005	0.9670 ± 0.007	0.9434 ± 0.004
Non-Linear Non-Gaussian	0.9103 ± 0.004	0.9090 ± 0.004	0.9345 ± 0.005	0.9196 ± 0.005
	Atomic Interventions:			
Linear Gaussian	0.9141 ± 0.0018	0.9148 ± 0.0021	0.9507 ± 0.0059	0.9461 ± 0.0030
Non-Linear Gaussian	0.9138 ± 0.0053	0.9195 ± 0.0041	0.9506 ± 0.0056	0.9435 ± 0.0032
Non-Linear Non-Gaussian	0.9100 ± 0.0056	0.9034 ± 0.0115	0.9314 ± 0.0104	0.9306 ± 0.0041

Table 3.2: Results for the single-node intervention experiment. The metric used is the average rand index.

density (linear Gaussian, non-linear Gaussian, and normalizing flow), and intervention type (stochastic, atomic and imperfect). In each of the SCMs, we performed 1 interventions for every variable. For each SCM within each configuration, we explored the following hyperparameter range: $\gamma \in \{-.1, -.01\}$. For the remaining hyperparameters, we set $\alpha = 9$, $h = 248$ and the truncation level $K = 11$—the hyperparameter configuration that achieved the best log-likelihood on a validation set. From the generated SCM, we produced a dataset with $n = 10000$ samples, where each intervention has $\lfloor \frac{n}{d+1} \rfloor$ elements. This dataset was split into training (80%) and validation (20%). Figure D.1 contains 9 scatter plots off all the settings we consider in this experiment, however for visualization proposes we consider SCMs with $d = 2$. The results for this experiment can be found in Table 3.2. The results show that we can consistently recover a significant portion of the correspondences, using our variational inference algorithm, when intervention affect a single variable. When the graphs are denser the problem appears to become harder.

Multi-Node interventions We generated SCMs with $d = 20$ variables with a Erdős-Rényi scheme with expected number of edges per node $e = 1$. We generated 10 SCMs for each combination of e, conditional density (linear Gaussian, non-linear Gaussian, and normalizing flow), and intervention type (stochastic, imperfect and atomic). In each of the SCMs, we performed a random number of interventions. The number of interventions was sampled from a maximum entropy discrete distribution on the set $\{1, 2, \ldots, 7\}$. For each SCM within each configuration, we explored the following hyperparameter range: $\gamma \in \{-.1, -.01, -.001, -.0001\}$. From the generated SCM, we produced a dataset with $n = 10000$ samples, where each intervention has $\lfloor \frac{n}{d+1} \rfloor$ elements. The results for this experiment can be found in Table

Model Type	d = 20, ER1	
	Fixed single node	Random multi node
Stochastic Interventions:		
Linear Gaussian	0.9815 ± 0.005	0.9618 ± 0.0002
Non-Linear Gaussian	0.9655 ± 0.005	0.9617 ± 0.0006
Non-Linear Non-Gaussian	0.8929 ± 0.027	0.9543 ± 0.0049
Imperfect Interventions:		
Linear Gaussian	0.8072 ± 0.098	0.9322 ± 0.0096
Non-Linear Gaussian	0.9670 ± 0.007	0.9617 ± 0.0001
Non-Linear Non-Gaussian	0.9345 ± 0.005	0.9591 ± 0.0040
Atomic Interventions:		
Linear Gaussian	0.9507 ± 0.0059	0.9685 ± 0.0022
Non-Linear Gaussian	0.9506 ± 0.0056	0.9657 ± 0.0020
Non-Linear Non-Gaussian	0.9314 ± 0.0104	0.9527 ± 0.0101

Table 3.3: Results for the multi-node intervention experiment. The metric used is the average rand index.

3.3. The results show that we can also consistently recover a significant portion of the correspondences when intervention affect multiple variables.

Model assumptions ablation We compared the results of our method when the selected model does not match the SCM that generated the data. We follow the same dataset generation procedure from the single-node interventions experiment. We considered only SCMs with $d = 10$, $e = 1$, and imperfect interventions. We additionally, compared our method with the clustering algorithm *kmeans++* [87]. The results for these settings can be found in Table 3.4. The results show that our method is generally robust, regardless of the conditional density estimator used and problem type. Furthermore, the experimental results show that, as expected, our method outperforms *kmeans++* in our scenario.

Model Type	Underlying SCM Type		
	Linear Gaussian	Non-Linear Gaussian	Non-Linear Non-Gaussian
Linear Gaussian	$\mathbf{0.9262 \pm 0.030}$	0.9068 ± 0.0201	$\mathbf{0.9367 \pm 0.0287}$
Non-Linear Gaussian	0.9133 ± 0.0009	$\mathbf{0.9541 \pm 0.012}$	0.9133 ± 0.0011
Non-Linear Non-Gaussian	0.9136 ± 0.0051	0.9093 ± 0.0044	0.9103 ± 0.004
Kmeans++	0.8860 ± 0.0265	0.9049 ± 0.0188	0.8807 ± 0.0243

Table 3.4: Results for the model assumptions ablation experiment. The metric used is the average rand index.

3.5 Discussion

We introduced an intervention recovery algorithm based on neural networks and variational inference. Experiments with synthetic data show that our approach can recover latent interventions and their associated correspondences to samples.

The proposed method is not simply a new clustering algorithm. Our model adds a few degrees of freedom for each new ”cluster” (intervention distribution), particularly when considering a sparse prior for the intervention targets ($\gamma < 0$), most of the parameters (the assignments in the underlying SCM) between the observation and intervention distributions are shared, this can potentially make the problem easier to identify.

A limitation of our method is that interventions on causes tend to be more challenging to identify. We believe this is because, given a marginal distribution with sufficient variance, the data samples coming from interventions on causes overlap with the ones from the observational distribution(and with a prior distribution that encourages sparse intervention targets can lead to not identifying this scenario).

Chapter 4

Differentiable causal discovery under latent interventions

In this chapter, we propose a data-driven causal discovery method that assumes latent interventions in the dataset. We achieve this by augmenting the statistical model presented in Chapter 3 and constructing an approximate score function that we maximize using a continuous constrained formulation. We experimentally validate our algorithm using a range of synthetic data sets and show that latent interventions improve the structure learning task.

4.1 A model for causal discovery under latent interventions

We assume that the dataset $\mathcal{D}$ is produced by a mixture of SCMs, each resulting from an intervention applied to a base SCM $\mathcal{M}$. More specifically, $\mathcal{D}$ is partitioned into $K+1$ exchangeable groups, with group k containing i.i.d. samples from the intervention SCM $\tilde{\mathcal{M}}^{(k)}$ resulting from applying the k^{th} intervention to the base SCM $\mathcal{M}$. Our goal is to discover, the causal structure $\mathcal{G}$, that is shared between the intervention SCMs, more concretely the causal graph of source SCM $\mathcal{M}$. We reuse the full statistical model proposed in Chapter 3.

We represent the causal graph $\mathcal{G}$ via the adjacency matrix $A^{\mathcal{G}} \in \{0,1\}^{d\times d}$. Following previous work, our prior $p(\mathcal{G})$ models each entry $A^{\mathcal{G}}_{ij}$, corresponding to edge $x_i \to x_j$, as a Bernoulli variable independent of all the others,

$$p(\mathcal{G}) = \prod_{i,j=1}^{d} \sigma(\lambda_{ij})^{A^{\mathcal{G}}_{ij}} \big(1-\sigma(\lambda_{ij})\big)^{1-A^{\mathcal{G}}_{ij}}, \tag{4.1}$$

where $\sigma(u) = e^u/(1+e^u)$ is the usual logistic transformation (sigmoid) and the λ_{ij} are hyper-parameters. This prior over graphs is simplistic since it does not encode that $\mathcal{G}$ has to be a DAG. In this paper, we set $\lambda_{ij} = \lambda_{\mathcal{G}}$, for all i, j; however, in practice, a domain expert using the proposed method can embed prior knowledge in these hyper-parameters (our method can be straightforwardly adapted to that case). Figure 4.1 contains the graphical model representation, of the augmented statistical model for causal

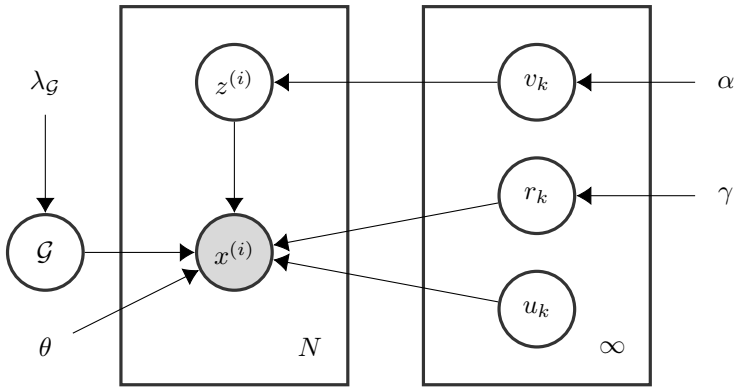

Figure 4.1: Graphical model representation of the Dirichlet process mixture model augmented for the causal discovery problem.

discovery under latent interventions.

4.2 A score for latent interventions

We adopt a score-based formulation of the causal discovery problem (see Section 2.1.5) using the MAP criterion, as presented in Equation 2.5, i.e.:

$$\mathcal{S}(\mathcal{G}) := \max_{\theta} \log p(\mathcal{D}|\mathcal{G}, \theta) + \log p(\mathcal{G}).$$

The log-likelihood $\log p(\mathcal{D}|\mathcal{G}, \theta)$, under our statistical model can be written as:

$$\log p(\mathcal{D}|\mathcal{G}; \theta) = \sum_{i=1}^{N} \log p(x^{(i)}|\mathcal{G}; \theta) = \sum_{i=1}^{N} \log \mathbb{E}_{z^{(i)}, \tilde{\mathcal{M}} \sim p(z^{(i)}, \tilde{\mathcal{M}})} \big[p(x^{(i)}|z^{(i)}, \tilde{\mathcal{M}}, \mathcal{G}; \theta) \big], \tag{4.2}$$

the logarithm of the marginalization of the joint distribution. Alternatively, it can also be written, via the evidence decomposition (see Section 2.16), as:

$$\log p(\mathcal{D}|\mathcal{G}; \theta) = \sum_{i=1}^{N} \mathbb{E}_{z^{(i)}, \tilde{\mathcal{M}} \sim p(z^{(i)}, \tilde{\mathcal{M}}|x^{(i)}, \mathcal{G}; \theta)} \big[\log p(x^{(i)}|z^{(i)}, \tilde{\mathcal{M}}, \mathcal{G}; \theta) \big]. \tag{4.3}$$

both of these ways of writing the log-likelihood make it intractable the evaluate the score on a particular graph let alone search over a the space of DAGs. Exact maximization of this marginal log-likelihood (which involves a product of Gaussians, Beta distributions, and complex conditional distributions generated by neural networks), is also intractable to obtain in closed form. Therefore, we resort to *approximate variational inference* [84].

Fortunately, in Section 3.2, we already designed a tractable variational family $\mathcal{Q}$, that we can use to approximate the true posterior $p(z^{(i)}, \tilde{\mathcal{M}}|x^{(i)}, \mathcal{G}; \theta)$, i. e.:

$$q(z^{(i)}, \tilde{\mathcal{M}}|x^{(i)}) = q(z^{(i)}|\tilde{\mathcal{M}}, x^{(i)}) \prod_{k=0}^{K} \Big(\prod_{j=1}^{d} q_R(r_{kj}) \Big) \Big(\prod_{l=1}^{h} q_U(u_{kl}) \Big) q_V(v_k), \tag{4.4}$$

where the particular form of these variational posteriors is described in Section 3.2, and the associated Kullback–Leibler divergences are contained in Table 3.1. Using this approximate posterior, we can lower-bound to the marginal log-likelihood from Equation 4.3. Giving us

$$\log p(\mathcal{D}|\mathcal{G};\theta) \geq \sum_{i=1}^{N} \underbrace{\mathbb{E}_{z^{(i)},\tilde{\mathcal{M}}\sim q(z^{(i)},\tilde{\mathcal{M}};\phi)}\left[\log p(x^{(i)}|z^{(i)},\tilde{\mathcal{M}},\mathcal{G};\theta)\right] - D_{KL}\left[q(z^{(i)},\tilde{\mathcal{M}})||p(z^{(i)},\tilde{\mathcal{M}})\right]}_{\mathsf{ELBO}_{q(z^{(i)},\tilde{\mathcal{M}};\phi)}(x^{(i)},\mathcal{G};\theta)}.$$

Using this lower-bound, associated with the variational family $\mathcal{Q}$, we can approximate the score $\mathcal{S}(\mathcal{G})$ with the surrogate score $\mathcal{S}_{\mathcal{Q}}(\mathcal{G};\phi)$, for any ϕ, as follows:

$$\mathcal{S}(\mathcal{G}) \geq \underbrace{\max_{\theta} \sum_{i=1}^{N} \mathsf{ELBO}_{q(z^{(i)},\tilde{\mathcal{M}};\phi)}(x^{(i)},\mathcal{G};\theta) + \log p(\mathcal{G})}_{\mathcal{S}_{\mathcal{Q}}(\mathcal{G};\phi)}. \tag{4.5}$$

Observing equation

$$\mathcal{S}_{\mathcal{Q}}(\mathcal{G};\phi) = \mathcal{S}(\mathcal{G}) - \ D_{KL}\left[q(z^{(i)},\tilde{\mathcal{M}};\phi)||p(z^{(i)},\tilde{\mathcal{M}}|x^{(i)},\mathcal{G};\theta^*)\right], \tag{4.6}$$

we notice that, by maximizing the $\mathcal{S}_{\mathcal{Q}}(\mathcal{G};\phi)$ w.r.t. ϕ we minimize the approximation gap between $\mathcal{S}(\mathcal{G})$ and $\mathcal{S}_{\mathcal{Q}}(\mathcal{G};\phi)$, which equals the Kullback–Leibler divergence between the approximate and the true posterior. Given this insight, we propose the surrogate score $\mathcal{S}_{\mathcal{Q}}(\mathcal{G})$, where the following inequality holds for all ϕ:

$$\mathcal{S}(\mathcal{G}) \geq \underbrace{\max_{\theta,\phi} \sum_{i=1}^{N} \mathsf{ELBO}_{q(z^{(i)},\tilde{\mathcal{M}};\phi)}(x^{(i)},\mathcal{G};\theta) + \log p(\mathcal{G})}_{\mathcal{S}_{\mathcal{Q}}(\mathcal{G})} \geq \mathcal{S}_{\mathcal{Q}}(\mathcal{G};\phi). \tag{4.7}$$

As many score-based causal discovery methods do, we can relax the surrogate score from Equation 4.7, yielding

$$\mathcal{S}^*_{\mathcal{Q}}(\Lambda) = \mathbb{E}_{\mathcal{G}\sim\mathsf{Bern}\left(\mathcal{G};\sigma(\Lambda)\right)}\left[\max_{\theta,\phi} \sum_{i=1}^{N} \mathsf{ELBO}_{q(z^{(i)},\tilde{\mathcal{M}};\phi)}(x^{(i)},\mathcal{G};\theta) + \log p(\mathcal{G})\right]. \tag{4.8}$$

4.3 Inference algorithm

The surrogate score, coupled with the acyclicity constraint from [51], enables us to formulate causal discovery under latent interventions as the following optimization problem:

$$\Lambda^* = \arg\max_{\Lambda} \mathcal{S}^*_{\mathcal{Q}}(\Lambda) \quad \text{s.t.} \quad \underbrace{\mathsf{Tr}\left(e^{\sigma(\Lambda)}\right) - d}_{h(\Lambda)} = 0, \tag{4.9}$$

Following Zheng et al. [51], we use the augmented Lagrangian procedure [60–62] (see Section 2.1.5) to transform the problem in Equation 4.9 into a sequence of unconstrained optimization subproblems. When we estimate the gradients using the path-wise gradient estimator [68, 69], each unconstrained

optimization subproblem reduces to sampling the graph and the latent variables from the variational posteriors using the reparametrization trick, minimizing the following objective:

$$\mathcal{L}(\theta, \phi, \Lambda, z, \tilde{\mathcal{M}}, A^{\mathcal{G}}; x, \mu_t, \varphi_t) = -\log p(x|z, \tilde{\mathcal{M}}, A^{\mathcal{G}}; \theta) + \Omega(\phi) - \lambda_{\mathcal{G}}||\Lambda||_1 + \varphi_t h(\Lambda) + \frac{\mu_t}{2} h(\Lambda)^2,$$

where $\Omega(\phi)$ denotes the sum of the Kullback–Leibler divergences, and μ_t and φ_t are the parameters of the augmented Lagrangian at the t^{th} iteration. For estimating the gradients of Λ, $q(z)$ and $q(r_{kj})$ we used a Gumbel-softmax continuous relaxation [71, 73] which, for the causal graph's distribution, was combined with the straight-through gradient estimator [78], to make sure the graph samples actually represented the hard dependencies of the SCM, instead of fractional ones. Having estimated the gradients, for each sample, we average them and feed them to the first order stochastic optimization algorithm Adam [85].

The proposed differentiable causal structure learning algorithm is described in Algorithm 2. We implemented our method with the PyTorch framework [2].

Algorithm 2 Amortized Variational Inference version of the inference algorithm. We set $B = 248, \eta = 2$ and $\delta = 0.9$ in our experiments.

$\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\Lambda} \leftarrow$ Initialize model parameters
$\boldsymbol{\mu_0}, \boldsymbol{\varphi_0} \leftarrow 10^{-6}, 0$
$t \leftarrow 0$
while $h(\boldsymbol{\Lambda}) > 10^{-8}$ **do**
  $\boldsymbol{\Lambda_t} \leftarrow \boldsymbol{\Lambda}$
  repeat
    $x^{(1)} \dots x^{(B)} \leftarrow$ Random mini-batch of B data points. (drawn from $\mathcal{D}$)
    for each $x^{(i)} \in x^{(1)} \dots x^{(B)}$ **do**
      $q(z^{(i)}, \tilde{\mathcal{M}}) \leftarrow$ Compute the variational posteriors
      $\tilde{\mathcal{M}}^{(i)}, A^{\mathcal{G}^{(i)}}, z^{(i)} \leftarrow$ Sample via the reparameterization trick.
      $\mathbf{g^{(i)}} \leftarrow \nabla_{\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\Lambda}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\Lambda}, z^{(i)}, \tilde{\mathcal{M}}^{(i)}, A^{\mathcal{G}^{(i)}}; x^{(i)}, \boldsymbol{\mu_t}, \boldsymbol{\varphi_t})$
    end for
    $\mathbf{g} \leftarrow \frac{1}{B} \sum_{i=1}^{B} \mathbf{g^{(i)}}$ (Average gradients)
    $\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\Lambda} \leftarrow$ Update parameters using Adam.
  until convergence of $(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\Lambda})$
  if $h(\boldsymbol{\Lambda}) > \delta h(\boldsymbol{\Lambda_t})$ **then**
    $\boldsymbol{\mu_{t+1}} \leftarrow \eta \boldsymbol{\mu_t}$
  end if
  $\boldsymbol{\varphi_{t+1}} \leftarrow \boldsymbol{\varphi_t} + \boldsymbol{\mu_{t+1}} h(\boldsymbol{\Lambda})$
  $t \leftarrow t + 1$
end while
return $\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\Lambda}$

4.4 Semi-supervised extensions

The causal discovery algorithm can be extended to cases where we have information about the correspondences $z^{(i)}$ and/or the intervention targets $I^{(k)}$. To achieve this, we only have to ignore the corresponding variational posteriors and use the observed values $z^{(i)}$ and $r^{(i)}$ as constants. We designate the original version of our model the **latent** variant, the one with observed $z^{(i)}$ the **unknown** variant,

and the one with observed $z^{(i)}$ and $r^{(i)}$ the **known** variant. It is straightforward to extend our method to a semi-supervised setting—a scenario where we only observe the correspondence z for a fraction of the samples. Under this scenario, we still use the samples from the variational posterior to predict the unobserved values of z and use the observed ones to improve the variational posterior. In essence, with the semi-supervised variant, we obtain a model that interpolates between the latent variant and the unknown variant. We follow the approach from [88] and extend the ELBO objective (in our case, the lower bound of the distribution $p(\mathcal{D}|\mathcal{G};\theta)$). Let $\mathcal{O}$ be the set of the indices for which the intervention assignment z is known, and with $\hat{z}^{(i)}$ its particular value. We rewrite the bound on the log-likelihood as :

$$\log p(\mathcal{D}|\mathcal{G};\theta) \geq \sum_{i=1}^{N} \mathsf{ELBO}_{\theta,q(\hat{z}^{(i)},\tilde{\mathcal{M}})}(x^{(i)},\mathcal{G}) + \kappa\, \mathbb{I}(i \in \mathcal{O})\, \mathbb{E}_{\tilde{\mathcal{M}}\sim q(\tilde{\mathcal{M}})}\big[\log q(\hat{z}^{(i)}|x^{(i)},\tilde{\mathcal{M}};\phi_z)\big],$$

where in the first term, when $i \in \mathcal{O}$, we do not sample from $q(z^{(i)}|x^{(i)},\tilde{\mathcal{M}};\phi_z)$, using the observed value $\hat{z}^{(i)}$ instead. The variable $\kappa \in]0,1[$ is an hyper-parameter that controls the relative importance of supervised component of the objective. Using this bound, we can write a new relaxed surrogate score $\mathcal{S}^{*}_{\mathsf{SLL}}(\Lambda)$ as:

$$\mathcal{S}^{*}_{\mathsf{SSL}}(\Lambda) = \mathbb{E}_{\mathcal{G}\sim\mathsf{Bern}\left(\mathcal{G};\sigma(\Lambda)\right)}\Big[\sum_{i=1}^{N} \mathsf{ELBO}_{\theta,q(\hat{z}^{(i)},\tilde{\mathcal{M}})}(x^{(i)},\mathcal{G}) + \kappa\, \mathbb{I}(i \in \mathcal{O})\, \mathbb{E}_{\tilde{\mathcal{M}}\sim q(\tilde{\mathcal{M}})}\big[\log q(\hat{z}^{(i)}|x^{(i)},\tilde{\mathcal{M}};\phi_z)\big]\Big].$$

4.5 Experiments

We tested our method on synthetic and real data. The synthetic datasets allow us to do a systematic, controlled comparison of different methods in different scenarios (graph size and density, intervention and assignment types). In order to generate SCMs, perform interventions on them and sample the corresponding entailed distributions, we created an open source Python package named PyCausal, which is available on GitHub[1].

We evaluate our method using the structural hamming distance (SHD) metric. The SHD between two graphs is the ℓ_1 norm between their adjacency matrices, as described by the following expression:

$$\mathsf{SHD}(A^{\mathcal{G}^*}, A^{\mathcal{G}}) = \sum_{i,j=1}^{d} |A^{\mathcal{G}^*}_{ij} - A^{\mathcal{G}}_{ij}|. \tag{4.10}$$

Single-node interventions. We generated SCMs with $d = 10$ variables with a Erdős-Rényi scheme with expected number of edges per node $e \in \{1, 4\}$. We provide a detailed description of how we generate SCMs in Appendix D. We generated 10 SCMs for each combination of e, conditional density (linear Gaussian, non-linear Gaussian, and normalizing flow), and intervention type (stochastic and imperfect). In each of the SCMs, we performed 1 interventions for every variable. For each SCM within each configuration, we explored the following hyperparameter range: $\lambda_{\mathcal{G}} \in \{-.1, -.01, 0, .01, .1\}$, and $\gamma \in \{-.1, -.01\}$. For the remaining hyperparameters, we set $\alpha = 9$, $h = 248$ and the truncation level

[1] https://github.com/goncalorafaria/PyCausal

$K = 11$—the hyperparameter configuration that achieved the best log-likelihood on a validation set. From the generated SCM, we produced a dataset with $n = 10000$ samples, where each intervention has $\lfloor \frac{n}{d+1} \rfloor$ elements. This dataset was split into training (80%) and validation (20%). The models were trained for 500 epochs for the first iteration of the augmented Lagrangian and 50 epochs for the remaining ones, with a full batch($B = 8000$) until $h(\Lambda) < 10^{-8}$, and learning rate of $10^{-2.5}$.

The results from these experiments are shown in Table 4.1. The metric we use is SHD between the adjacency matrices of the ground-truth graph and the estimated one. The columns from Table 4.1 correspond to the different variants we present in Section 4.4: **latent** refers to our model, **unknown** assumes that the correspondences are known (as Brouillard et al. [52]), and **known** assumes that both correspondences and intervention targets are known. We additionally include a "naive" observational model (a baseline which ignores the existence of intervention data, *i.e.*, that assumes $K = 1$ as if all data was generated by the observational model). The experimental results preponderantly show that our method (the latent variant) consistently outperforms the observational baseline and is worst than the unknown variant, as expected, but only slightly. This indicates that taking account latent interventions, when these are present, improves the recovery of the causal graph.

Model Type	e	**latent**	**unknown**	**known**	**observational**
			Stochastic Interventions:		
Linear Gaussian		5.9 ± 6.2	3.4 ± 3.2	0.5 ± 1.3	10.3 ± 7.8
Non-Linear Gaussian	1	12.2 ± 3.9	10.3 ± 2.5	7.0 ± 3.6	13.7 ± 3.8
Non-Linear Non-Gaussian		8.7 ± 6.6	8.0 ± 2.7	6.6 ± 2.2	11.3 ± 5.0
Linear Gaussian		27.2 ± 6.2	24.1 ± 5.8	15.6 ± 6.0	39.6 ± 5.0
Non-Linear Gaussian	4	35.8 ± 3.8	30.3 ± 5.3	27.7 ± 4.3	37.5 ± 5.2
Non-Linear Non-Gaussian		36.1 ± 4.4	35.5 ± 8.1	31.5 ± 5.6	40.2 ± 6.9
			Imperfect Interventions:		
Linear Gaussian		5.8 ± 4.2	6.2 ± 3.06	4.7 ± 3.6	10.4 ± 2.9
Non-Linear Gaussian	1	9.3 ± 2.4	8.9 ± 2.5	7.8 ± 3.9	10.5 ± 2.8
Non-Linear Non-Gaussian		8.8 ± 3.0	9.1 ± 3.5	7.9 ± 1.4	11.5 ± 5.4
Linear Gaussian		35.9 ± 8.3	29.7 ± 5.6	17.7 ± 7.9	39.1 ± 9.1
Non-Linear Gaussian	4	32.1 ± 6.0	32.6 ± 5.8	32.8 ± 5.4	39.8 ± 9.3
Non-Linear Non-Gaussian		30.4 ± 12.2	30.2 ± 11.2	25.8 ± 3.9	36.7 ± 9.8

Table 4.1: Hamming distances on synthetic 10 variable SCMs.

Single-node interventions on cause-effect pairs. We generated two-variable linear Gaussian SCMs (cause-effect pairs) with edge probability $e = \frac{2}{3}$ (equal probability for each of the three possible graphs). The SCMs were generated as described in Appendix D. We sampled 60 SCMs, and generate a dataset of $n = 999$ samples, where each intervention has 333 elements. The sampled graph $\mathcal{G}$, with variables A

and B is one of 3 possible graphs, A causes B, B causes A, or A and B have no cause-effect relation. All of the possible graphs share the MEC. We compare the variants presented in Section 4.4 in addition to the naive observational baseline. Figure 4.2 contains an histogram of Hamming distances for each of the variants. Edges in the anti-causal direction cost $\mathrm{SHD} = 2$, missing edge or wrong edge is $\mathrm{SHD} = 1$, and correct graph is $\mathrm{SHD} = 0$. The results show for this simple problem that cannot be identified using observational data alone, that our method correctly identifies a significant majority of the cause-effect pairs, even without information about the intervention assignments.

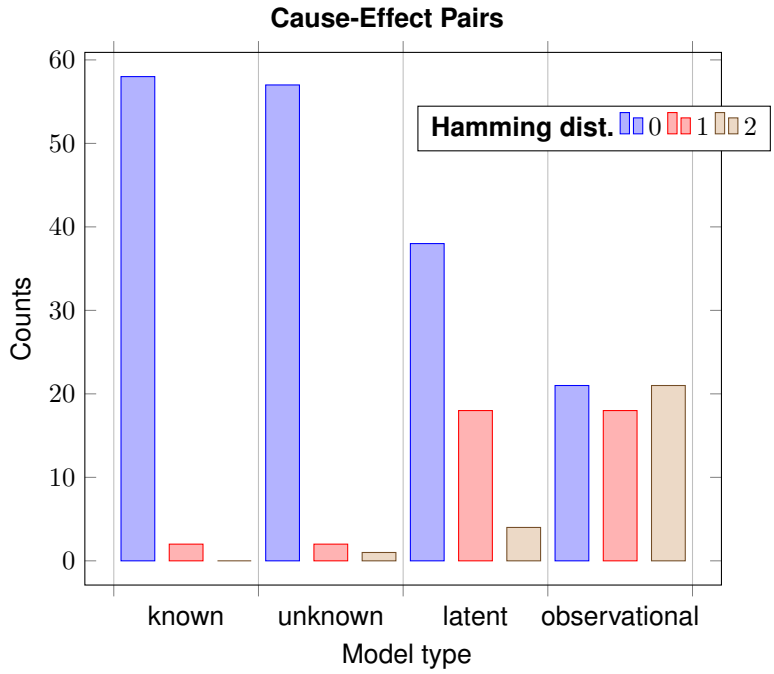

Figure 4.2: Histogram of Hamming distance in the experiments with cause-effect pairs.

Real-world data set Finally, we tested our method on the flow cytometry data set of Sachs et al. [89]. The measurements are the level of expression of phosphoproteins and phospholipids in humans cells. Interventions were performed by using reagents to activate or inhibit the measured proteins. The dataset comprises 7466 items with 11 variables each. Figure 4.3 contains an illustration of the consensus graph from Sachs et al. [89]. This graph contains 11 edges. Table 4.2 compares the estimated graph, under different conditional density assumptions but assuming imperfect interventions, to the consensus graph from Sachs et al. [89]. The results from the real-world data set show that our method outperforms several baselines, even with methods that use information regarding intervention correspondences and targets. The reasons that might justify relatively good results on this standard problem is that i) the causal sufficiency assumption may not hold, ii) the interventions may not be as specific as stated, and iii) the ground truth network is possibly not a DAG since feedback loops are common in cellular signaling networks as noted by Brouillard et al. [52]. These reasons can potentially be detrimental to the other methods and our method appears to be robust to them.

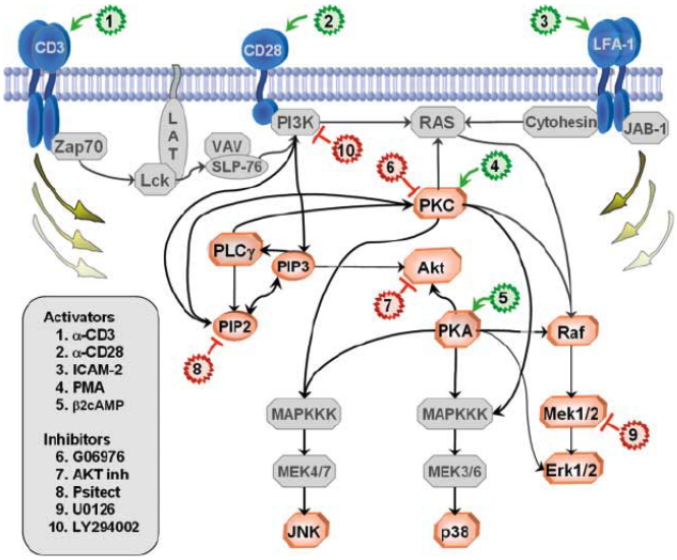

Figure 4.3: Classic signaling network and points of intervention. This is a graphical illustration of the conventionally accepted signaling molecule interactions, the events measured, and the points of intervention by small-molecule inhibitors. This illustration was obtained from Sachs et al. [89].

	SHD	tp	fn	fp	rev	F_1 score
IGSP [53]	18	4	6	5	7	0.42
GIES [14]	38	10	0	41	7	0.33
CAM [50]	35	12	1	30	4	0.51
DCDI-G [52]	36	6	2	25	9	0.31
DCDI-DSF [52]	33	6	2	22	9	0.33
Linear Gaussian (imperfect) **(ours)**	33	7	11	22	3	0.30
Non-Linear Gaussian (imperfect) **(ours)**	19	7	11	8	0	0.42
Normalizing Flow (imperfect) **(ours)**	30	9	9	21	1	0.38

Table 4.2: Results for the flow cytometry dataset. The results for the baselines are reproduced from [52].

Semi-supervised learning experiments We test our semi-supervised method, on 10 variable linear SCMs generated according to Appendix D. We consider stochastic and imperfect interventions. We sampled a 10000 samples dataset from each SCM. For each dataset, we created multiple semi-supervised learning datasets by splitting the original 10000 samples into labelled (with intervention assignments) and unlabelled (without intervention assignments) according to some fraction f. The fraction f goes from 0 to 1, sampled every interval of 0.2. This means that, for each fraction f, we have 10 distinct semi-supervised datasets for both perfect and imperfect interventions. We picked the default hyperparameters, particularly, $\lambda_{\mathcal{G}} = -0.1$, and $\gamma = -0.01$. The results are contained in Figure 4.4.

The results suggest, there is no significant improvement from using a the correspondences for a fraction of the samples under the tested conditions.

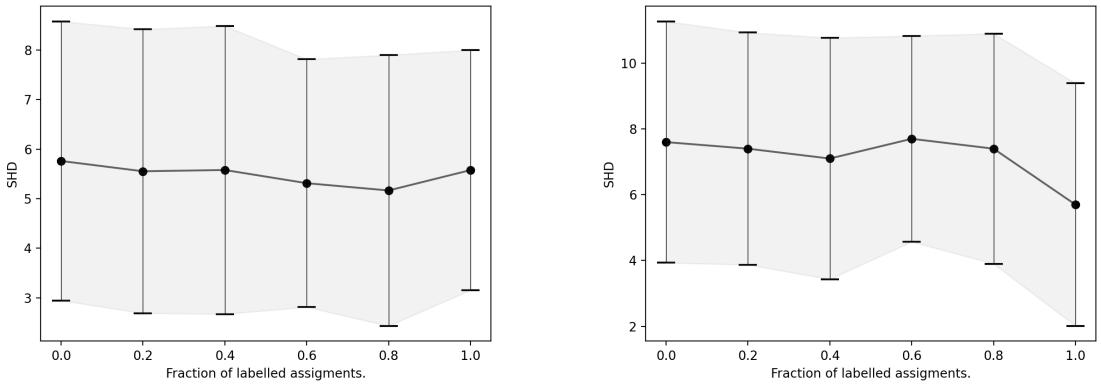

Figure 4.4: Semi supervised experiments on 10 variable Linear SCMs with an expected value of 1 edge per node. On the left, perfect interventions. On the right, imperfect interventions.

4.6 Related work

The previous work that is closest to ours in that by Brouillard et al. [52]. As they do, we assume that the data is clustered in intervention groups, but we relax their assumption that the correspondence between intervention groups and samples is known, opening the door to more realistic scenarios.

The so-called "known" and "unknown" variants of our method share the assumptions made by Brouillard et al. [52]; however, contrary to them, the number of distinct neural networks in our model is not dependent on the number of interventions, which allows scaling well to many interventions. We achieve this by encoding the change in assignments associated with each intervention using a specific latent variable u that we call intervention embedding and conditioning a shared model on it when computing the log-probability.

Our method can be seen as a *variational autoencoder* (VAE) with a Dirichlet process (DP) prior, with a stick-breaking process representatoion. The work by nalisnick2017stickbreaking introduces a VAE where the stick-breaking weights are the latent variables. Our model differs from theirs in that we use the entire DP (including the atoms). We only use the stick-breaking weights in the Kullback–Leibler divergence of the correspondence variable z. This Kullback–Leibler divergence has a closed-form, so we do not sample the stick-breaking weights as they do. van den Oord et al. [90] proposes a VQ-VAE model with discrete latent variables, each represented as a latent embedding vector (an atom). Our approach is similar in that we use atoms (the intervention embeddings and intervention targets) to represent discrete latent variables. However, we do it in a statistically sounder way that more naturally fits our application.

4.7 Discussion

We introduced an efficient variational optimization algorithm for causal structure learning under latent interventions. Our results are competitive with other state-of-the-art algorithms on the flow cytometry data set. Experiments with synthetic data show that our approach can recover causal relations even in

our more challenging scenario, and it can consistently outperform our purely observational alternative.

A limitation of our method is the variational family we use. The proposal for variational posterior considers that the stick-breaking weights, the intervention embeddings, and intervention targets are independent of each other. Since most of these conditional independencies, in general, are not present in the true posterior, there will inevitably be an approximation gap. This approximation gap, in some cases, can potentially create a surrogate score whose maximum structure does not maximize the score $\mathcal{S}(\mathcal{G})$.

Chapter 5

Conclusions

In this chapter, in Section 5.1, we start by summarizing our work. Furthermore, we present some interesting avenues for future research, in Section 5.2. Finally, we discuss the broader impact of our work, in Section 5.3.

5.1 Achievements

We proposed a causal discovery method based on neural networks and variational inference that leverages interventional data with gradient-based methods when interventions and their correspondence to samples are unknown beforehand. Our results are competitive with other state-of-the-art algorithms on the flow cytometry data set. Experiments with synthetic data show that our approach can recover causal relations, even in our more challenging scenario containing latent interventions. It can consistently outperform our purely observational alternative. When given the underlying causal graph, we also evaluated our method on approximating the posterior over the unknown interventions and correspondences, i.e., recovering the latent intervention, on synthetic datasets. It can outperform a clustering baseline on this task.

5.2 Future Work

The research we have carried out opens up interesting possibilities of future work. Our framework is particularly appealing for problems where performing interventions explicitly is expensive or unethical, but where interventions occur naturally in the data without being explicitly observed. We consider the following methodological improvements as promising research directions.

Variational approximation Experimenting with more flexible variational families seems appealing, albeit this may come at the cost of the closed-form expressions for the Kullback–Leibler divergences. Another direction is focusing on particular families of sampling models, *i.e.*, linear models, and using algorithms with more guarantees such as the expectation-maximization algorithm.

Acyclicity constraint Methods that avoid computing the acyclicity constraint, but that can still preserve the acyclicity of the underlying graph—for instance, exploiting the fact that, given the topological ordering of DAG, one can represent the adjacency matrix as a triangular matrix with zeros in the main diagonal. Many continuous relaxation can use the path-wise-gradient estimators to estimate expectations of distributions over orderings.

5.3 Broader Impact

High-quality and reliable causal discovery algorithms are general tools for understanding complex systems. The proposed method combats limitations, of data-driven causal discovery, and can recover causal relations in a scenario where the system of interest was subject to fully latent interventions. We envision a positive impact of our work in areas like astrophysics, where we are too far to intervene, history research, where the system of interest is no longer available, and in science more generally. A concrete example of an application of our method is the flow cytometry data set from Section 4.5. Some researchers claimed that this dataset was problematic because the interventions may not be as specific as stated. As a result, this is difficult for causal discovery methods that do not assume interventions as latent. However, as we have shown, our method compares favorably in this setting. Applications are likely to extend beyond these examples, which seem straightforward from our current position.

Like any general tool, our work can have unwanted applications that could have negative impacts. Moreover, it is crucial for anyone that uses our method to understand its underlying assumptions:

- Causal sufficiency.
- Presence of latent interventions.
- Faithfulness.
- Intervention faithfulness.
- Causal relations form acyclic graphs.
- For a specific intervention distribution, the data is assumed independently and identically distributed.
- Within the proposed variational family we can find a sufficiently approximate distribution to approximate the latent interventions' posterior.

Therefore, it is imperative to analyze the results and deeply consider their predictions before making decisions.

Bibliography

[1] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

[2] S. Li, Y. Zhao, R. Varma, O. Salpekar, P. Noordhuis, T. Li, A. Paszke, J. Smith, B. Vaughan, P. Damania, and S. Chintala. Pytorch distributed: Experiences on accelerating data parallel training. *CoRR*, abs/2006.15704, 2020. URL https://arxiv.org/abs/2006.15704.

[3] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[4] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3): 90–95, 2007. doi: 10.1109/MCSE.2007.55.

[5] M. L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60): 3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

[6] J. Pearl. *Causal inference in statistics : a primer*. Wiley, Chichester, West Sussex, 2016 - 2016. ISBN 9781119186847.

[7] A. Dixit, O. Parnas, B. Li, J. Chen, C. P. Fulco, L. Jerby-Arnon, N. D. Marjanovic, D. Dionne, T. Burks, R. Raychowdhury, B. Adamson, T. M. Norman, E. S. Lander, J. S. Weissman, N. Friedman, and A. Regev. Perturb-seq: Dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *Cell*, 167(7):1853–1866.e17, 2016. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2016.11.038. URL https://www.sciencedirect.com/science/article/pii/S0092867416316105.

[8] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, USA, 2000. ISBN 0521773628.

[9] J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference - Foundations and Learning Algorithms*. Adaptive Computation and Machine Learning Series. The MIT Press, Cambridge, MA, USA, 2017.

[10] J. Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995. ISSN 00063444. URL http://www.jstor.org/stable/2337329.

[11] J. Pearl. *Causality*. Cambridge University Press, 2 edition, 2009. doi: 10.1017/CBO9780511803161.

[12] E. Bareinboim and J. Pearl. Causal inference and the data-fusion problem. *Proceedings of the National Academy of Sciences*, 113(27):7345–7352, 2016. ISSN 0027-8424. doi: 10.1073/pnas.1510507113. URL https://www.pnas.org/content/113/27/7345.

[13] T. S. Verma and J. Pearl. On the equivalence of causal models. *CoRR*, abs/1304.1108, 2013. URL http://arxiv.org/abs/1304.1108.

[14] A. Hauser and P. Bühlmann. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *J. Mach. Learn. Res.*, 13(1):2409–2464, Aug. 2012. ISSN 1532-4435.

[15] N. Shajarisales, D. Janzing, B. Schoelkopf, and M. Besserve. Telling cause from effect in deterministic linear dynamical systems. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 285–294, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/shajarisales15.html.

[16] P. Daniusis, D. Janzing, J. M. Mooij, J. Zscheischler, B. Steudel, K. Zhang, and B. Schölkopf. Inferring deterministic causal relations. *CoRR*, abs/1203.3475, 2012. URL http://arxiv.org/abs/1203.3475.

[17] K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008. URL https://proceedings.neurips.cc/paper/2007/file/3a0772443a0739141292a5429b952fe6-Paper.pdf.

[18] F. Eberhardt and R. Scheines. Interventions and causal inference. *Philosophy of Science*, 74 (5):981–995, 2007. ISSN 00318248, 1539767X. URL http://www.jstor.org/stable/10.1086/525638.

[19] A. Hauser and P. Bühlmann. Two optimal strategies for active learning of causal models from interventional data. *International Journal of Approximate Reasoning*, 55(4):926–939, Jun 2014. ISSN 0888-613X. doi: 10.1016/j.ijar.2013.11.007. URL http://dx.doi.org/10.1016/j.ijar.2013.11.007.

[20] K. Shanmugam, M. Kocaoglu, A. G. Dimakis, and S. Vishwanath. Learning causal graphs with small interventions. *CoRR*, abs/1511.00041, 2015. URL http://arxiv.org/abs/1511.00041.

[21] S. Shimizu, T. Inazumi, Y. Sogawa, A. Hyvärinen, Y. Kawahara, T. Washio, P. O. Hoyer, and K. Bollen. Directlingam: A direct method for learning a linear non-gaussian structural equation model. *J. Mach. Learn. Res.*, 12(null):1225–1248, July 2011. ISSN 1532-4435.

[22] P. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2009. URL https://proceedings.neurips.cc/paper/2008/file/f7664060cc52bc6f3d620bcedc94a4b6-Paper.pdf.

[23] J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15(58):2009–2053, 2014. URL http://jmlr.org/papers/v15/peters14a.html.

[24] J. Peters and P. Bühlmann. Identifiability of gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, Nov 2013. ISSN 0006-3444. doi: 10.1093/biomet/ast043. URL http://dx.doi.org/10.1093/biomet/ast043.

[25] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*, volume 81. 01 1993. ISBN 978-1-4612-7650-0. doi: 10.1007/978-1-4612-2748-9.

[26] J. Pearl. Causality: Models, reasoning, and inference, second edition. *Causality*, 29, 01 2000. doi: 10.1017/CBO9780511803161.

[27] P. Spirtes, C. Meek, and T. S. Richardson. Causal inference in the presence of latent variables and selection bias. *CoRR*, abs/1302.4983, 2013. URL http://arxiv.org/abs/1302.4983.

[28] J. Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16):1873–1896, 2008. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2008.08.001. URL https://www.sciencedirect.com/science/article/pii/S0004370208001008.

[29] S. Triantafillou and I. Tsamardinos. Constraint-based causal discovery from multiple interventions over overlapping variable sets, 2014.

[30] A. Hyttinen, F. Eberhardt, and M. Järvisalo. Constraint-based causal discovery: Conflict resolution with answer set programming. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, page 340–349, Arlington, Virginia, USA, 2014. AUAI Press. ISBN 9780974903910.

[31] M. Kocaoglu, A. Jaber, K. Shanmugam, and E. Bareinboim. Characterization and learning of causal graphs with latent variables from soft interventions. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/c3d96fbd5b1b45096ff04c04038fff5d-Paper.pdf.

[32] J. Peters, P. Bühlmann, and N. Meinshausen. Causal inference using invariant prediction: identification and confidence intervals, 2015.

[33] C. Heinze-Deml, J. Peters, and N. Meinshausen. Invariant causal prediction for nonlinear models, 2018.

[34] D. M. Chickering and D. Heckerman. Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. In *Machine Learning*, pages 29–181, 1997.

[35] R. R. Bouckaert. Probabilistic network construction using the minimum description length principle. In M. Clarke, R. Kruse, and S. Moral, editors, *Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 41–48, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg. ISBN 978-3-540-48130-0.

[36] D. Heckerman, D. Geiger, and D. M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. In R. L. de Mantaras and D. Poole, editors, *Uncertainty Proceedings 1994*, pages 293–301. Morgan Kaufmann, San Francisco (CA), 1994. ISBN 978-1-55860-332-5. doi: https://doi.org/10.1016/B978-1-55860-332-5.50042-0. URL https://www.sciencedirect.com/science/article/pii/B9781558603325500420.

[37] J. Kuipers, G. Moffa, and D. Heckerman. Addendum on the scoring of gaussian directed acyclic graphical models, 2021.

[38] J. Cussens. Bayesian network learning with cutting planes. *CoRR*, abs/1202.3713, 2012. URL http://arxiv.org/abs/1202.3713.

[39] J. Cussens, D. Haws, and M. Studeny. Polyhedral aspects of score equivalence in bayesian network structure learning, 2015.

[40] S. Ott and S. Miyano. Finding optimal gene networks using biological constraints. *Genome informatics. International Conference on Genome Informatics*, 14:124–33, 02 2003.

[41] T. Silander and P. Myllymäki. A simple approach for finding the globally optimal bayesian network structure. *CoRR*, abs/1206.6875, 2012. URL http://arxiv.org/abs/1206.6875.

[42] A. P. Singh and A. W. Moore. Finding optimal bayesian networks by dynamic programming, Jun 2018. URL https://kilthub.cmu.edu/articles/journal_contribution/Finding_optimal_Bayesian_networks_by_dynamic_programming/6605669/1.

[43] J. Xiang and S. Kim. A-star lasso for learning a sparse bayesian network structure for continuous variables. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/8ce6790cc6a94e65f17f908f462fae85-Paper.pdf.

[44] M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning bayesian networks. *CoRR*, abs/1207.1429, 2012. URL http://arxiv.org/abs/1207.1429.

[45] D. M. Chickering. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3 (null):507–554, Mar. 2003. ISSN 1532-4435. doi: 10.1162/153244303321897717. URL https://doi.org/10.1162/153244303321897717.

[46] J. Ramsey, M. Glymour, R. sanchez romero, and C. Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International Journal of Data Science and Analytics*, 3, 03 2017. doi: 10.1007/s41060-016-0032-z.

[47] B. Aragam and Q. Zhou. Concave penalized estimation of sparse gaussian bayesian networks. *Journal of Machine Learning Research*, 16(69):2273–2328, 2015. URL http://jmlr.org/papers/v16/aragam15a.html.

[48] F. Fu and Q. Zhou. Learning sparse causal gaussian networks with experimental intervention: Regularization and coordinate descent. *Journal of the American Statistical Association*, 108(501):288–300, 2013. doi: 10.1080/01621459.2012.754359. URL https://doi.org/10.1080/01621459.2012.754359.

[49] J. Gu, F. Fu, and Q. Zhou. Penalized estimation of directed acyclic graphs from discrete data. *Statistics and Computing*, 29(1):161–176, Feb 2018. ISSN 1573-1375. doi: 10.1007/s11222-018-9801-y. URL http://dx.doi.org/10.1007/s11222-018-9801-y.

[50] P. Bühlmann, J. Peters, and J. Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6):2526 – 2556, 2014. doi: 10.1214/14-AOS1260. URL https://doi.org/10.1214/14-AOS1260.

[51] X. Zheng, B. Aragam, P. Ravikumar, and E. P. Xing. Dags with no tears: Continuous optimization for structure learning, 2018.

[52] P. Brouillard, S. Lachapelle, A. Lacoste, S. Lacoste-Julien, and A. Drouin. Differentiable causal discovery from interventional data. *CoRR*, abs/2007.01754, 2020. URL https://arxiv.org/abs/2007.01754.

[53] Y. Wang, L. Solus, K. D. Yang, and C. Uhler. Permutation-based causal inference algorithms with interventions, 2017.

[54] K. D. Yang, A. Katcoff, and C. Uhler. Characterizing and learning equivalence classes of causal dags under interventions, 2019.

[55] C. Squires, Y. Wang, and C. Uhler. Permutation-based causal structure learning with unknown intervention targets, 2020.

[56] S. Lachapelle, P. Brouillard, T. Deleu, and S. Lacoste-Julien. Gradient-based neural DAG learning. *CoRR*, abs/1906.02226, 2019. URL http://arxiv.org/abs/1906.02226.

[57] X. Zheng, C. Dan, B. Aragam, P. Ravikumar, and E. P. Xing. Learning sparse nonparametric dags, 2020.

[58] I. Ng, Z. Fang, S. Zhu, Z. Chen, and J. Wang. Masked gradient-based causal structure learning, 2020.

[59] D. Kalainathan, O. Goudet, I. Guyon, D. Lopez-Paz, and M. Sebag. Structural agnostic modeling: Adversarial learning of causal graphs, 2020.

[60] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5):303–320, 1969. doi: 10.1007/bf00927673. URL https://app.dimensions.ai/details/publication/pub.1018609722.

[61] M. Powell. A method for nonlinear constraints in minimization problems. *In: Fletcher, R., Ed., Optimization, Academic Press*, pages 283–298, 1969.

[62] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. 1975.

[63] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

[64] R. Sutton, D. A. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, 1999.

[65] P. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Commun. ACM*, 33:75–84, 1990.

[66] J. Wilson. Variance reduction techniques for digital simulation. *American Journal of Mathematical and Management Sciences*, 4:277–312, 1984.

[67] P. L'Ecuyer. Efficiency improvement and variance reduction. *Proceedings of Winter Simulation Conference*, pages 122–132, 1994.

[68] D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2014.

[69] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014.

[70] Y. Kim, S. Wiseman, and A. M. Rush. A tutorial on deep latent variable models of natural language. *CoRR*, abs/1812.06834, 2018. URL http://arxiv.org/abs/1812.06834.

[71] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax, 2017.

[72] J. Bastings, W. Aziz, and I. Titov. Interpretable neural predictions with differentiable binary variables. In *Proc. ACL*, 2019.

[73] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables, 2017.

[74] G. Mena, D. Belanger, S. Linderman, and J. Snoek. Learning latent permutations with gumbel-sinkhorn networks, 2018.

[75] G. M. Correia, V. Niculae, W. Aziz, and A. F. T. Martins. Efficient marginalization of discrete and structured latent variables via sparsity, 2020.

[76] E. Gumbel. Statistical theory of extreme values and some practical applications : A series of lectures. 1954.

[77] C. J. Maddison, D. Tarlow, and T. Minka. A* sampling. In *NIPS*, 2014.

[78] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013.

[79] T. S. Ferguson. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209 – 230, 1973. doi: 10.1214/aos/1176342360. URL https://doi.org/10.1214/aos/1176342360.

[80] D. J. Aldous. Exchangeability and related topics. In P. L. Hennequin, editor, *École d'Été de Probabilités de Saint-Flour XIII — 1983*, pages 1–198, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg. ISBN 978-3-540-39316-0.

[81] J. Sethuraman. A constructive definition of dirichlet priors. 1991.

[82] D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, 2015.

[83] C. Huang, D. Krueger, A. Lacoste, and A. C. Courville. Neural autoregressive flows. *CoRR*, abs/1804.00779, 2018. URL http://arxiv.org/abs/1804.00779.

[84] D. Blei, A. Kucukelbirb, and J. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[85] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[86] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. doi: 10.1080/01621459.1971.10482356. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1971.10482356.

[87] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 9780898716245.

[88] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. *CoRR*, abs/1406.5298, 2014. URL http://arxiv.org/abs/1406.5298.

[89] K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005. doi: 10.1126/science.1105809.

[90] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. *CoRR*, abs/1711.00937, 2017. URL http://arxiv.org/abs/1711.00937.

[91] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56): 1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

[92] S. Elfwing, E. Uchibe, and K. Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *CoRR*, abs/1702.03118, 2017. URL http://arxiv.org/abs/1702.03118.

[93] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization, 2016.

[94] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

[95] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Representations by Back-Propagating Errors*, page 696–699. MIT Press, Cambridge, MA, USA, 1988. ISBN 0262010976.

Appendix A

Variational Inference proofs

A.1 Gradient of the log evidence

$$
\begin{aligned}
\nabla_\theta \mathcal{L}(x,\theta) &= \nabla_\theta \sum_{i=1}^{N} \log p(x^{(i)};\theta) \\
&= \nabla_\theta \sum_{i=1}^{N} \log \sum_z p(x^{(i)},z;\theta) \\
&= \sum_{i=1}^{N} \frac{\nabla_\theta \sum_z p(x^{(i)},z;\theta)}{p(x^{(i)};\theta)} \\
&= \sum_{i=1}^{N} \sum_z \frac{p(x^{(i)},z;\theta)}{p(x^{(i)};\theta)} \nabla_\theta \log p(x^{(i)},z;\theta) \\
&= \sum_{i=1}^{N} \mathbb{E}_{z\sim p(z|x^{(i)};\theta)} \left[\nabla_\theta \log p(x^{(i)},z;\theta) \right]
\end{aligned}
\tag{A.1}
$$

A.2 Evidence decomposition

$$
\begin{aligned}
\log p(x|\theta) &= \sum_z q(z) \log p(x|\theta) \\
&= \sum_z q(z) \log \frac{p(x,z|\theta)}{p(z|x,\theta)} \\
&= \sum_z q(z) \log \frac{p(x,z|\theta) q(z)}{p(z|x,\theta) q(z)} \\
&= \sum_z q(z) \log \frac{p(x,z|\theta)}{q(z)} - \sum_z q(z) \log \frac{p(z|x,\theta)}{q(z)} \\
&= \mathbb{E}_{z\sim q}[\log p(z,x|\theta)] + D_{KL}(q(z)||p(z|x,\theta)) + H[q(z)] \\
&= \mathbb{E}_{z\sim q}[\log p(x|z,\theta)] + D_{KL}(q(z)||p(z|x,\theta)) - D_{KL}(q(z)||p(z))
\end{aligned}
\tag{A.2}
$$

A.3 Score-Function Estimator

$$\begin{aligned}\nabla_\phi \mathbb{E}_{z\sim q(z|x;\phi)}[f(z)] &= \nabla_\phi \sum_z q(z|x;\phi) f(z) \\ &= \sum_z \nabla_\phi q(z|x;\phi) f(z) \\ &= \sum_z q(z|x;\phi) \nabla_\phi \log q(z|x;\phi) f(z) \\ &= \mathbb{E}_{z\sim q(z|x;\phi)} \nabla_\phi \left[f(z) \log q(z|x;\phi) \right]\end{aligned} \tag{A.3}$$

A.4 Beta KL

$$\begin{aligned}D_{\mathrm{KL}}(X_1||X_2) &= \int_0^1 f(x;\alpha,\beta) \ln\left(\frac{f(x;\alpha,\beta)}{f(x;\alpha',\beta')}\right) dx \\ &= \left(\int_0^1 f(x;\alpha,\beta)\ln(f(x;\alpha,\beta))\,dx\right) - \left(\int_0^1 f(x;\alpha,\beta)\ln(f(x;\alpha',\beta'))\,dx\right) \\ &= -h(X_1) + H(X_1,X_2) \\ &= \ln\left(\frac{\mathrm{B}(\alpha',\beta')}{\mathrm{B}(\alpha,\beta)}\right) + (\alpha-\alpha')\psi(\alpha) + (\beta-\beta')\psi(\beta) + (\alpha'-\alpha+\beta'-\beta)\psi(\alpha+\beta)\end{aligned} \tag{A.4}$$

A.5 Gaussian KL

$$D_{\mathsf{KL}}(p,q) = -\int p(x)\log q(x)dx + \int p(x)\log p(x)dx$$

$$= \frac{1}{2}\log(2\pi\sigma_2^2) + \frac{\sigma_1^2+(\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}(1+\log 2\pi\sigma_1^2)$$

$$= \log\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2+(\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

A.6 Kullback–Leibler from our variational approximation

$$
\begin{aligned}
&D_{KL}\big[q(\tilde{\mathcal{M}}, z^{(i)})||p(\tilde{\mathcal{M}}, z^{(i)})\big] = \\
&\quad= \mathbb{E}_{\tilde{\mathcal{M}},z\sim q(\tilde{\mathcal{M}},z)}\left[\log \frac{q(z|u_0,\dots,u_T)\prod_{k=0}^{T}\Big(\prod_{j=1}^{d} q_R(r_{kj})\Big)\Big(\prod_{l=1}^{h} q_U(u_{kl})\Big)q_V(v_k)}{p(z|\beta_0,\dots,\beta_T)\prod_{k=0}^{T}\Big(\prod_{j=1}^{d} p(r_{kj}|\gamma)\Big)\Big(\prod_{l=1}^{h} p(u_{kl})\Big)p(v_k|\alpha)}\right] \\
&\quad= \sum_{k=0}^{T}\sum_{l=1}^{h} D_{KL}\big(q(u_{kl})||\mathcal{N}(0,1)\big) + \sum_{k=0}^{T}\sum_{j=1}^{d} D_{KL}\big(q(r_{kj})||\mathsf{Bern}(\gamma)\big) \\
&+\sum_{k=0}^{T} D_{KL}\big(q(v_k)||p_\theta(v_k|\alpha)\big) + \mathbb{E}_{\tilde{\mathcal{M}},z\sim q(\tilde{\mathcal{M}},z)}\left[\log \frac{q(z|u_0,\dots,u_T)}{p(z|\beta_0,\dots,\beta_T)}\right]
\end{aligned}
\tag{A.5}
$$

Stick-breaking weights

$$
\begin{aligned}
D_{KL}\big(q(v_k)||p_\theta(v_k|\alpha)\big) = \log\Big(\frac{\mathrm{B}\big(\rho_k w_k, (1-\rho_k)w_k\big)}{\mathrm{B}(1,\alpha)}\Big) + (1-\rho_k w_k)\psi(1) + \\
\big(\alpha - (1-\rho_k)w_k\big)\psi(\alpha) + (-1+w_k-\alpha)\psi(1+\alpha)
\end{aligned}
\tag{A.6}
$$

Intervention embeddings

$$
D_{KL}\big(q(u_k)||\mathcal{N}(\mathbf{0}, I_h)\big) = \sum_{j=1}^{h} \frac{\sigma_{kj}^2 + \mu_{kj}^2 - 1}{2} - \log\sigma_{kj} \tag{A.7}
$$

Intervention targets

$$
D_{KL}\big(q(r_{kj})||\mathsf{Bernoulli}(\gamma)\big) = \pi_{kj}\Big(\mathsf{logit}(\pi_{kj}) - \gamma\Big) + \log\frac{1-\pi_{kj}}{1-\sigma(\gamma)} \tag{A.8}
$$

Correspondence

$$
\begin{aligned}
\mathbb{E}_{\mathcal{V}^k,u_0,\dots,u_K\sim q(\mathcal{V}^k)q(u_0)\dots q(u_K)}\Big[D_{KL}\big(q(z)||p_\theta(z|\beta_0,\beta_1,\dots,\beta_K)\big)\Big] = \\
= \sum_{k=0}^{K}\mathbb{E}_{u_k\sim q(u_k)}\Big[q(z_k)\big(\log q(z_k) - \mathbb{E}_{\mathcal{V}^k\sim q(\mathcal{V}^k)}\left[\log\beta_k\right]\big)\Big]
\end{aligned}
\tag{A.9}
$$

where

$$
\begin{aligned}
\mathbb{E}_{\mathcal{V}^k\sim q(\mathcal{V}^k)}[\log\beta_k] &= \mathbb{E}_{v_k\sim q(v_k)}[\log v_k] + \sum_{k'=0}^{k-1}\mathbb{E}_{v_{k'}\sim q(v_{k'})}[\log 1 - v_{k'}] \\
&= \psi(\rho_k w_k) + \sum_{k'=0}^{k-1}\psi\big((1-\rho_{k'})w_{k'}\big) - \sum_{k'=0}^{k}\psi(w_{k'}),
\end{aligned}
\tag{A.10}
$$

and ψ is the digamma function. We approximate using the Taylor series expansion, where we use the reparametrization trick on u_k to estimate the expectations.

Appendix B

Neural Networks

Neural networks are biologically inspired parametric non-linear mappings. The particular neural network we use in our work is the feed forward neural network. Given an vector $x \in \mathbb{R}^{d_{\text{in}}}$, we parameterize with a vector of "weights" θ a non-linear mapping FFN : $\mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$. This mapping is a composition of linear transformations interchanged with a non-linear element-wise operation.

We use many tricks that have been shown to improve the capabilities of Neural Network models, particularly dropout [91], SiLU activation function [92], layer normalization [93], and residual connections [94]. We group these techniques into a sub-layer that form the building block of our relatively more complex models, as shown in Figure B.1.

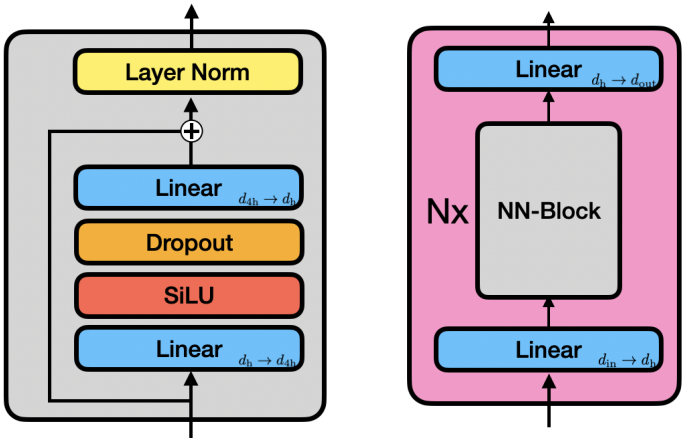

Figure B.1: On the left, diagram of the neural network block sub-layer. On the right, diagram of full neural network we use, where N is ab hyper-parameter.

To enable the use of skip connections, we have to make sure that the inputs and outputs of the sub-layer have the same dimension. For this reason, before applying the stack of sub-layers we apply a linear transformation from $\mathbb{R}^{d_{\text{in}}}$ to $\mathbb{R}^{d_h}$ and after passing the stack from $\mathbb{R}^{d_h}$ to $\mathbb{R}^{d_{\text{out}}}$, as shown in diagram on the right of Figure B.1.

In supervised *machine learning*, neural works are commonly used to parameterize non-linear condi-

tional densities, particularly for distributions of y in the exponential family, we write that

$$p(y|x,\theta) = h(y)g(\phi(x;\theta))\exp\big(\eta(\phi(x;\theta))^\intercal T(y)\big), \tag{B.1}$$

where $h(y)$, $g(\phi(x;\theta))$, $\eta(\phi(x;\theta))$, and $T(y)$ are known functions, such that $p(y|x,\theta)$ is a proper density function of y, and $\phi(x;\theta)$ is a neural network that maps values of x into parameter vectors.

Loss function Given, a finite dataset $\mathcal{D}$ of independent and identically distributed pairs $(x_i, y_i) \in \mathcal{D}$. The parameters θ for the network FFN$(x;\theta)$ are learned based on the optimization of a proposed loss function $\mathcal{L}(\theta;\mathcal{D})$. The loss function is generally based on the *maximum likelihood* or *maximum a posteriori* principles. The optimization problem, considering a loss function that is the negative log-likelihood of the model from Equation B.1, can be written as follows:

$$\theta^* = \arg\min_\theta \underbrace{\sum_{(x_i,y_i)\in\mathcal{D}} -\log h(y_i) - \log g(\phi(x_i;\theta)) - \eta(\phi(x_i;\theta))^\intercal T(y_i)}_{\mathcal{L}(\theta;\mathcal{D})=\log\prod_{(x_i,y_i)\in\mathcal{D}} p(y_i|x_i,\phi)} \tag{B.2}$$

Back-propagation Optimizing Equation B.2 for complex neural network models $\phi(x;\theta)$, can be a difficult. Primarily because the loss $\mathcal{L}(\theta;\mathcal{D})$ generally will not be convex, and the dataset $\mathcal{D}$ might be really big. A usual approach is stochastic gradient descent (SGD), that requires the calculation of the gradient of the loss $\mathcal{L}(\theta;\mathcal{D})$ with respect to the model's parameters θ. This can be efficiently done using the gradient back-propagation algorithm [95], a special case of reverse mode (reverse accumulation) automatic differentiation.

Appendix C

PyCausal: Package for defining large scale Structural Causal Models

C.1 Design Principles

Our goal was to create a package that allowed the creation of SCMs in a declarative using Computational Graphs. There are two types of nodes in this computational graph. Source Nodes which are the source of randomness and Auxiliary nodes which are the result of some operation on already created nodes. The nodes can either be visible or not. The visible nodes are the nodes in the left side of the assignments. Source nodes are marked by default. We can sample the graph or specific variables using ancestor sampling or recursively using a simple graph coloring algorithm.

Having this computational structure, we can easily define functions that perform interventions by slightly altering the underlying computational graph. For sampling from intervention graphs, we have this two stage sampling process. We first sample the distributions associated with the intervention and only then the $i.i.d$ samples from the general SCM.

For simplicity we overloaded the python arithmetic operators and added a few mathematical functions and probability distributions. We can use this package using hardware accelerators like GPUs and TPUs.

C.2 Example

Let's consider the following SCM:

$$\begin{cases} X := N_X, N_X \sim \mathcal{N}(0,1) \\ Z := N_Z, N_Z \sim \text{Beta}(0.5, 0.5) \\ Y := \exp(X^2) + ZN_Y, N_Y \sim \mathcal{N}(0,1) \end{cases} \tag{C.1}$$

Using PyCausal, we can define this SCM in the following way:

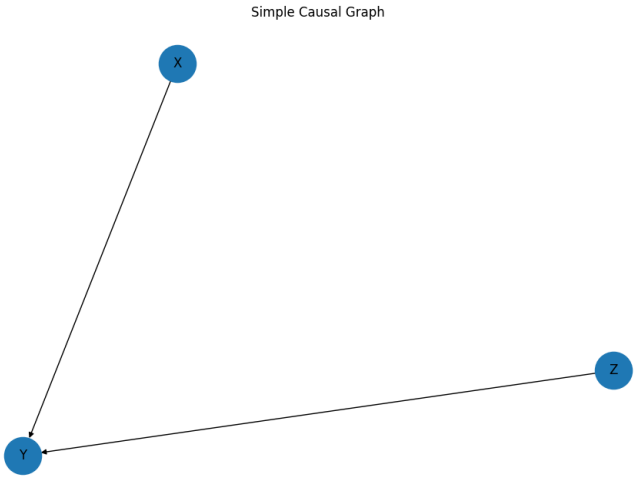

Figure C.1: Illustration of the example causal graph.

```
from pycausal import *

model = SCM("Simple Causal Graph")

X = Variable("X", stats.norm(loc=0,scale=1))
Z = Variable("Z", stats.beta(0.5,0.5))

Ny = HiddenVariable("Ny", stats.norm(loc=0,scale=1))

Y = Ny * Z + exp( X**2 ) << "Y"

model.draw()
```

This gives the graph in Figure C.1.

We can sample from a SCM in the following way:

```
model.sample(2)
```

or

```
(~model)(2)
```

which gives

```
{'Z': array([0.99181398, 0.02439115]),
 'X': array([-0.07538367,  1.69771261]),
 'Y': array([ 2.64181855, 17.87651557])}
```

We can obtain intervention SCMs, in this case an atomic intervention on X, in the following way:

```
imodel = model&{ X: 0 }
```

If we sample this model we obtain:

```
{'X': array([0, 0]),
 'Z': array([0.34692997, 0.16893219]),
 'Y': array([1.42016021, 0.86607793])}
```

We also can assign stochastic interventions, either by giving a distribution as argument or a SCM Variable.(Be careful to not create loops)

```
imodel = model&{ X: stats.norm(loc=0,scale=1) }
```

We can even stack interventions

```
imodel = model&{ X: 0, Ny: 2}
```

or

```
imodel = model&{X: 0}&{Ny: 2}
```

or

```
imodel = model&{X: 0}
jmodel = imodel&{Ny: 2}
```

We can sample individual variables:

```
( ~Y )(2)
```

or

```
Y.sample(2)
```

which gives

```
array([ 2.64181855, 17.87651557])
```

We can even do independence tests, graphical or statistical

```
Y.independent_of(Ny, significance=0.05)
```

equivalently, we can write

```
Y | Ny
```

which gives:

```
False
```

The api contains already many mathematical functions for defining the SCMs. However, we can add new ones in the following way:

```
model = SCM("Matrix assignments model")

new_op = func( torch.nn.softmax, name="softmax")

X = Variable("X", stats.uniform(-2,5), shape=[2,2])
```

```
Ny = HiddenVariable("Ny",stats.beta(0.4,0.1), shape=[1,1])

y2 = -(sin(X)*2)@np.ones(shape=[2,2])
y3 = new_op(y2)

Y = reduce_sum(y3 + Ny, axis=[-1,-2], keepdims=False) << "Y"
```

Appendix D

Description of the Synthetic data sets

To generate the SCMs, we first generated the causal graph. The graphs were generated going in order from node 1 to node N and adding edges from the already visited nodes to the current node i with probability p obtaining the adjacency matrix $A^{\mathcal{G}}$. In order to obtain graphs that weren't topologically ordered we applied a random permutation P to $A^{\mathcal{G}}$. Then, given the causal graph, we sampled the noise distributions and the assignments as described in Table D.1. For every SCM, we generate d interventions, one for each variable. How we sample the new assignment for each intervened variables is described in Table D.2. Before fitting the model, the data is always normalized. We subtract the mean and divide by the standard deviation.

For each setting, we sample $n/(d+1)$ examples. We used $n = 10000$ in the 10 variable data set. Before using our method, the data is always normalized. We subtract the mean and divide by the standard deviation. For stochastic interventions in variable j, we sampled a new Gaussian variable ε_j, as described in Table D.2, and assigned it to x_j. Using imperfect interventions, we go to the last layer of the neural network (in the case of linear model its the weight vectors) representing the assignment, and sample new weights according to Table D.2. Table D.1 contains a description of how we sample assignments. Particularly, the architecture of the neural network, when it is non-linear SCM. Figure D.1 contains scatter plots obtained after sampling the generated two variable SCMs.

Model Name	$p(\mathcal{E})$	weights	activation	layers	#hidden units
Linear Gaussian	$\prod_{j=1}^{d} \mathcal{N}(0, 0.015)$	$\mathcal{N}(0, 2.0)$	identity	1	-
Non-Linear Gaussian	$\prod_{j=1}^{d} \mathcal{N}(0, 0.015)$	$\mathcal{N}(0, 2.0)$	relu	2	5
Non-Linear Non-Gaussian	$\prod_{j=1}^{d} \mathcal{N}(0, 0.015)$	$\mathcal{N}(0, 2.0)$	relu	2	5

Table D.1: Where $k \sim \mathcal{N}(0.1, 0.005)$ and $\ell \sim \mathcal{N}(0, 0.4)$.

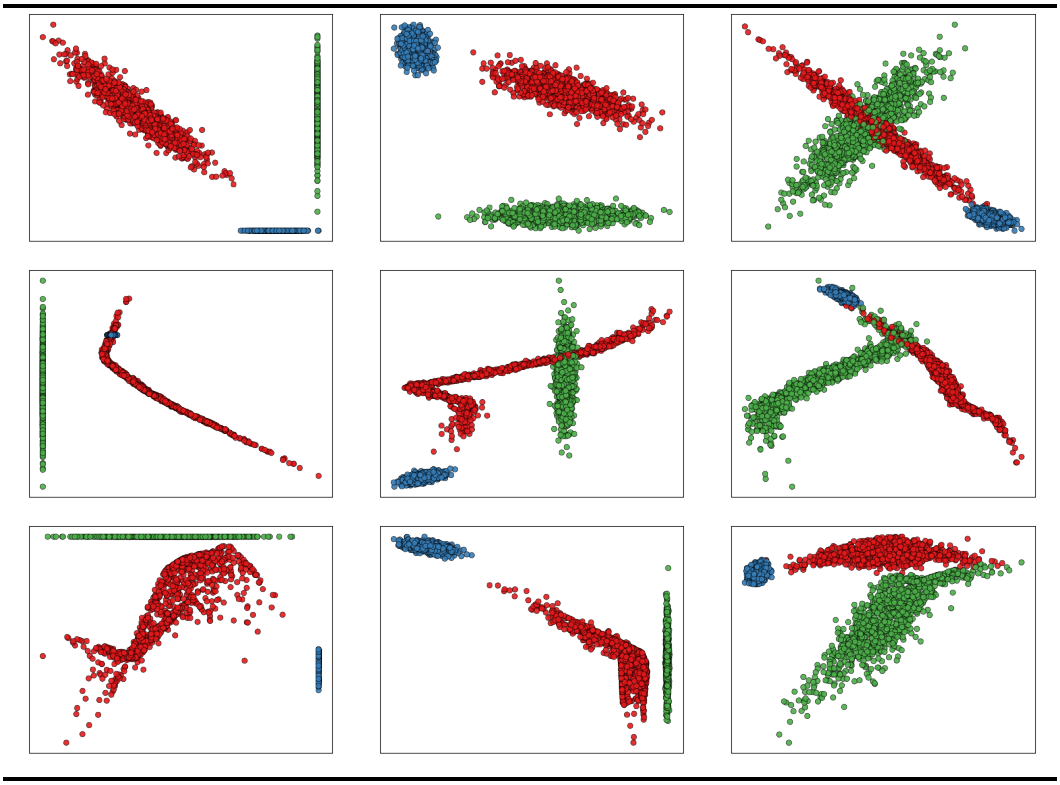

Figure D.1: The underlying graph is has two variables and one edge. The cause is in the horizontal axis, the effect is in the vertical axis. The rows correspond to different models, particularly, Linear Gaussian, Non-Linear Gaussian and Non-Linear Non-Gaussian. The columns correspond to distinct types of interventions. Particularly, atomic, stochastic and imperfect. Red observational. Green intervention on the effect variable. blue intervention on the causal variables.

Intervention type	new $p(\mathcal{E})$	new weights
Atomic	$(1-2b)\cdot u$	0
Stochastic	$\mathcal{N}((1-2b)\cdot u, 0.1)$	0
Imperfect	$\mathcal{N}((1-2b)\cdot u, 0.1)$	$\mathcal{N}(0, 2.0)$

Table D.2: Where $u \sim \mathcal{U}[1.2, 2.2]$ and $b \sim$ Bernoulli(0.5)