CoDiet

COMBATTING DIET RELATED NON-COMMUNICABLE DISEASE THROUGH ENHANCED SURVEILLANCE

D4.2 Extending Probabilistic Graphical Models towards Time Series

Deliverable number D4.2

Work Package WP4	Causally-Informed Machine Learning Models and Personalized Intervention Recommendation
Task 4.2	Forecasting with Causal Models
Task Leader	Czech Technical University
Prepared by	Czech Technical University
Contributors	
Version	1.1
Delivery Date	06/10/2024

Co-funded by the European Union
UK Research and Innovation

Foreword

The work described in this report was developed under the project **CoDiet - Combatting Diet related non-communicable disease through enhanced surveillance** (Grant Agreement number: 101084642; Call: HORIZON-CL6-2022-FARM2FORK-01; Topic: HORIZON-CL6-2022-FARM2FORK-01-10). Any additional information, if needed, should be required to:

Project Coordinator:
Itziar Tueros – itueros@azti.es | AZTI |

WP4 Leader: Technion
Shie Mannor – shie@technion.ac.il | TEC |

Task Leader: Czech Technical University
Jakub Marecek – jakub.marecek@fel.cvut.cz| CTU |

Dissemination Level		
PU	Public, fully open	X
SEN	Sensitive, limited under the conditions of the Grant Agreement	
Classified R-UE/EU-R	EU RESTRICTED under the Commission Decision No2015/444	
Classified C-UE/EU-C	EU CONFIDENTIAL under the Commission Decision No2015/444	
Classified S-UE/EU-S	EU SECRET under the Commission Decision No2015/444	

Co-funded by the European Union

CoDiet is part of the Horizon Europe programme supported by the European Union.

Document History

- (June 30th, 2024) Version 1.0 submitted to the EC and uploaded to CoDiet website.
- (October 10th, 2024) Version 1.1 submitted to the EC and uploaded to CoDiet website. In response to the comment suggesting that "a brief introduction should be provided explaining which methodology is presented in which paper", we have added such summaries on page 4.

1 Executive Summary

Here we describe the general background of Probabilistic Graphical Models (PGMs), in particular the most prominent and relevant, Dynamic Bayesian Networks (DBNs), as well as their relationship to causal learning. An important consideration is the appropriate extension of learning techniques, for which there is extensive literature in the case of static Bayesian Networks (BNs), to apply to temporal data.

This is a methodological deliverable, the intention is to provide technical understanding as well as guide to appropriate tools and the foundation from which to build customized tools to suit the needs of the project. By assisting the researchers in the broader Machine Learning team within CoDiet incorporate DBNs as a fundamental component of the statistical modeling and eventual AI recommender, we hope to provide an accessible and interpretable mechanism to construct and assess various causal models.

In the next Section, we describe the most significant technical features of DBNs, their general capabilities, and, the focus of the research papers that compose the bulk of the deliverable, a description of algorithms for learning DBN models from data. Finally, we present and describe our prototype code, which is hosted at `https://github.com/codiet-eu/d42`.

In the subsequent Section, we describe the CoDiet data that the consortium expects to obtain. We present the overall quantitative and qualitative aspects of the data and the procedures used to obtain it and known statistical properties thereof. Finally we discuss general considerations from the literature as far as the application of (D)BNs towards modeling this form of data, as well as potential approaches that combine this tool with the other consortium methods and skills to perform the eventual causal learning.

These two Sections are followed by five manuscripts:

1. *Learning Dynamic Bayesian Networks from Multiple Trajectory Data: First Principles and Some Numerical Comparisons* (arXiv:2406.17585)
2. *ExDAG: Exact learning of DAGs* (arXiv:2406.15229)
3. *ExDBN: Exact learning of Dynamic Bayesian Networks* (arxiv identifier to be assigned)
4. *Empirical Bayes for Dynamic Bayesian Networks With Generalized Variational Inference* (arXiv:2406.17831)
5. *Causal Learning in Biomedical Applications: A Benchmark* (arXiv:2406.15189)

whose latest versions are also availale in arXiv, the pre-print server, at the links above. These capture the work of the team at CTU (Vyacheslav Kungurtsev, Petr Rysavy, Fadwa Idhlacen, Jakub Marecek, Xiaoyu He, Ales Wodecki, Pavel Rytir). The deliverable has also benefitted from the insights of the teams at NKUA (Dimitrios Gunopulos, Vana Kalogeraki, Kleopatra Markou) and Technion (Shie Mannor, Mark Kozdoba).

Manuscript listed as (1) above presents a broad review of approaches to learning dynamic Bayesian networks presented in the literature, with some experimental comparisons. It is currently under submission to a journal.

Manuscript listed as (2) above presents a novel approach to learning directed acyclic graphs (DAGs) using mixed-integer programming. The results improve upon the state-of-the-art approaches that claim to provide maximum likelihood estimates (or global minimizers of empirical risk, or similar). It is currently under submission to a conference.

Manuscript listed as (3) above specializes the approach of (2) to learning dynamic Bayesian networks using mixed-integer programming. The results improve upon the state-of-the-art approaches in learning dynamic Bayesian networks and have been demonstrated on substantial amounts of data. It is currently under submission to a conference.

Manuscript listed as (4) demonstrates a sampling-approximation based approach to scaling the approaches (2) and (3) to massive datasets. It is still being finalized.

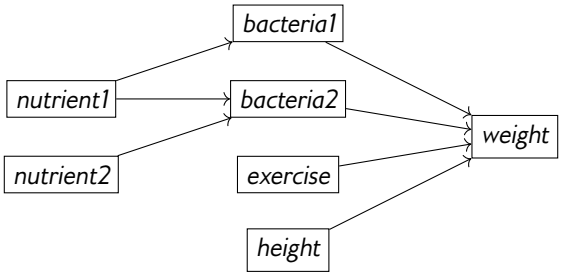
Manuscript listed as (5) above presents a synthetic benchmark we have used in testing (1 and 3 above). In particular, we have sampled concentrations of reactants of the Krebs cycles in various resolutions, various numbers of experiments, etc, to demonstrate that one can trade-off length of the time series for the number of time series, as discussed in the review using DYNOTEARS. We would now like to present trade-off in more detail using the recently finished ExDBN (3 above). This is still still being finalized.

2 Dynamic Bayesian Networks

2.1 Introduction

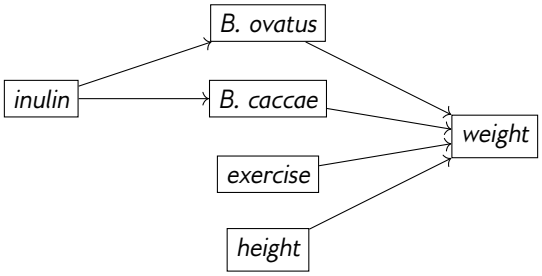
To motivate our work, let us consider an intentionally simplistic example:

Example 1. Bacteria *that live in our gut help us process food including* nutrient1 *and* nutrient1. *What metabolites become available in the blood stream (metabolome), depends on the composition of the population of bacteria (microbiome). In many settings, we could model the metabolome as a high-dimensional unobserved state. If one wishes to study the impact on an easily observable quantity such* weight, *one should like to consider confounders including* height *and the amount of* exercise. *This relationship can be represented by a directed acyclic graph (DAG) below.*

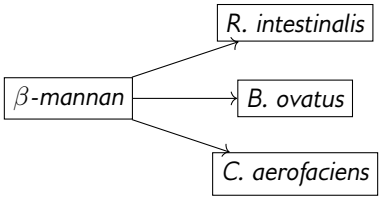


In practice, these relationships are much more complex and our goal is to infer quantitative aspects of such causal relationships from measurements of random variables, often available in the form of high-dimensional time series that are not sampled uniformly. Although some of the random variables are easily observed (e.g., weight), some others (e.g., related to the metabolome) need not be. Consider the following.

Example 2. *Metabolome depends not only on the diet and microbiome, but also the microbiome is affected by the metabolome and the diet. For example, inulin, a polysaccharide that is found in the cell walls of certain plants, promotes the growth of intestinal bacteria, which modulate the intake of energy from food. In the microbiome, bacteria such as* Bacteroides ovatus *and* Bacteroides caccae *compete for inulin and their prevalence depends on both their past prevalence, inulin levels, and concentrations of other dietary fiber, at least up to some level of inulin.*



Example 3. *A similar interaction network can be found in relation to β-mannan. The primary degrader of β-mannan is* Roseburia intestinalis, *together with others such as* Bacteroides ovatus. *For more details, see publication [3]. Therefore, a diet rich in β-mannans positively influences the growth of these two strains. In contrast, a fiber-free diet decreases their levels while promoting other bacteria, such as* Collinsella aerofaciens. *This can be crudely represented in the following causal network.*



Ideally, we would be able to make such causal models quantitative, not least to distinguish that β-mannan-rich diet promotes *Roseburia intestinalis*, while β-mannan-poor diet may promote *Collinsella aerofaciens*.

2.2 Dynamic Bayesian Networks

2.2.1 Modeling and Inference

Simply speaking, DBNs model multidimensional data time-varying $X(t)$ as though each variable $X_i(t+1)$ depends on a particular set of other variables. This will always include Markovian dependence $X_j(t)$, sometimes include lagged componented $X_j(t-\tau)$ for some $\tau > 1$, sometimes include time-independent covariates Z, and also usually include a BN model for dependence on variables at the current time $X_j(t+1)$, by a structure defined through a Directed Acyclic Graph (DAG). For instance, in the example 2 above, we can consider that insulin levels, bacteria concentrations, and weight change over time, while height remains constant. And thus the quantity weight can be defined as a Markov process, that is, a stochastic process that depends on the previous time, together with any time independent components, like,

$$\begin{aligned}
\texttt{weight}(t+1) &= \beta_w \texttt{weight}(t) + \beta_h \texttt{height} + \beta_e \texttt{exercise}(t) + \beta_o \texttt{B.Ovatus}(t) + \beta_c \texttt{B.Caccae}(t) \\
\texttt{B.Ovatus}(t+1) &= \beta_o^o \texttt{B.Ovatus}(t) + \beta_i^o \texttt{insulin}(t+1) \\
\texttt{B.Caccae}(t+1) &= \beta_c^c \texttt{B.Caccae}(t) + \beta_i^c \texttt{insulin}(t) + \beta_{i1}^c{}' \texttt{insulin}(t-1) + \beta_{i2}^c{}' \texttt{insulin}(t-2)
\end{aligned}$$

The graph given by the figure for Example 2 defines the *structure* that is, which variables influence which in the transition, while a statistical procedure is used to estimate the weights β_h, etc., in the model, to ultimately define the procedure quantitatively. We have also included more complexity as by the flexibility the DBN model permits, by including time lagged effects of insulin on the Caccae bacteria concentration.

We can thus witness the two primary modeling opportunities offered by DBNs. For one, through both the DAG structure at instatemporal influence as well as the time-lagged influence, we obtain a qualitative picture of the relationship between variables. The dependence structure as to what variable depends on what and how reveals information about the mechanism by which these variables interact. As such, DBNs are emplematic of what's often referred to as *interpretable artificial intelligence (XAI)*. Practically, this also is related to the fact that the structure is a central element of causal learning, that is causal inference and discovery. The conditional independence structures that define causal influence and the potential of do-calculus as causal interventions can be directed observed from the structure of a DBN.

The quantitative model representation as well as the parameters associated with the model provide degrees of freedom for a versatile framework for modeling various phenomena statistically. Due to its modularity, one can endow domain expertise as far as the functional form of the relationship. For instance, it could be known that there is a nonlinear crowding effect $\beta_c^c \left(\texttt{B.Caccae}(t)\right)^2$ in the third model. The modular representation allows the user to choose the complexity and the form based on available data and domain expertise, adapting the tool to fit the appropriate context.

Learning the structure and parameters of a DBN from data, however, is generally a difficult exercise. The structure presents significance combinatorial explosion, as with each variable an exponential number of potential graphical connections appears. We present the main significant considerations as far as learning DBNs. With appropriate scalable machine learning tools together with embedded domain expertise to learn DBNs at large dimension size, inference on the DBN model can serve to compute the reward function for the reinforcement learning platform.

2.2.2 Learning

Foundations The third paper below, *Learning Dynamic Bayesian Networks from Data: Foundations, First Principles and Numerical Comparisons*, presents an overall comprehensive guide to the basic principles and techniques for learning the structure and parameters of a DBN model. These are important to understand as there are important subtleties as to how to appropriate learn the two, considering the combinatorial-continuous distinction between structure and weights and yet their deep interplay as far as modeling is concerned. The paper describes the statistical considerations, the various models that are available, the criteria used to learn structure, a description of a selection of representative algorithms, and numerical comparisons.

The structure of the DBN, which is represented as a graph, can define a potential causal structure. In the first paper, *ExDAG: Exact learning of DAGs*, integer programming with branch and bound is used to find the exact DAG of a BN learning problem. The work *Causal Learning in Biomedical Applications* presents the use of the dynotears[4] continuous optimization algorithm for learning the structure and weights of a structural equation model of the Krebs cycle.

Large Dimensional Problems It can be observed from the foundations that learning a complete structure defining the relationships between the variables is a challenging combinatorial problem. Furthermore, considering that each edge in a DBN represents a potential relationship that may or may not be statistically significant, adding a requirement that the entire DAG model is statistically significant to assure a degree of certainty in causal discovery and inference, can present insurmountable computational challenges for data with more than a small number of variables.

To this end there are three approaches, that can even be thought of as broadly speaking, towards statistical modeling for large dimensional problems.

First, there are neural network based approaches. Utilizing the universal approximation properties of neural networks, together with favorable scaling of the parameter space with the data volume, neural networks enable the computational power of GPUs to be able to use large numbers of samples, tens of millions, to learn a precise model. This itself can be used for learning a DAG structure, e.g., by reinforcement learning. This approach, however, requires the aforementioned large quantities of data.

Second, and in particular in the finite and especially small sample size regime, Bayesian methods become prominent. In particular, since the sample size does not permit sufficient statistical power for bold significance claims, instead we aim to model *the entire uncertainty* space of the potential DBN. This presents a probabilistic picture as to *given existing background knowledge and the data, what structure and weight of relationships can be reasonable and to what level of confidence*. Under such settings, DAG structures, which may be learned on subsets of variables and data for computational ease, become particles, which is a stadard framework in filtering. The last paper presented below, *Empirical Bayes for Dynamic Bayesian Networks With Generalized Variational Inference*, presents an approach to how one can initialize a set of approximate point solutions and subsequently use Bayesian methods to sample an uncertainty quantified model. Finally, as a less formalized approach, metaheuristics have been applied to learn a DBN under large variable dimension size.

2.3 Code Package

The github repository `https://github.com/codiet-eu/d42` contains a basic DBN modeling class structure for versatile development in Python. The PGM class allows for generic graphical models, and the DynamicBayesianNetwork general DBNs. DBNOpt and DBNBayes are used as the base for optimizers and samplers. In addition, all of the code that generated the experiments in all of the papers appears in the repository.

3 PGMs and CoDiet

3.1 Description of the Data

A variety of data was measured in the CoDiet project. The first group of inputs are obtained in the questionnaires to the participants and describe *demographic and habits*. Those include sex, age, income, education level, smoking habits, drinking habits, or medical history. Most of those are background information. The second group is formed by slow variables that can be named as *anthropometric measurements*, and includes height, weight, waist circumference, blood pressure, and body composition parameters, such as fat mass or muscle mass. This group will be accompanied by *arterial hardening and AGEs* data measured by non-invasive sensors. Variables mentioned in the past paragraph tend to influence health and can undergo slow changes.

Biochemical data include standard biochemistry parameters such as glucose or lipids levels. Other tests, such as CRP and insulin level measurements, will also be done. Those data will be complemented by *metabolomics data*, i.e., both targeted and untargeted measurements of metabolites. Those data are mostly continuous, often changing in the order of hours; however, only a limited number of samples will be available. *Dietary biomarkers* from urine, such as nitrogen, potassium, and sucrose, will be measured.

The *food intake* of the participants will be reported. For such, two sources of data will be collected - manually noted 24-hour dietary recall data, as well as measured by modern technologies using passive cameras. As a result, we will have discrete variables with food intake available over the course of eight weeks.

Background data stem from *genetics*. The show risk of various diseases of the participants by measuring polygenetic risk scores, DNA methylation, and miRNA data. Besides that, health is tightly bound to the microbiome. Shotgun sequencing will be used to obtain gut metagenome.

Physical activity belongs to the group of fast variables, as it changes rapidly. We will have continuous measurements from the GENEActiv wearables. They provide more than 150 continuous variables, including acceleration, posture, and sleep monitoring.

3.2 Formal Representations

We describe a particular annotation of the variables that is associated with the analysis of a clinical trial. We can divide the variables into several instrumental categories.

1. **Treatment variables** - These are fully observed. They may be continuous or discrete, and an impulse in time, or, more commonly, continuous in time. They are expected to influence health outcomes. Confounders, e.g., genetic, for adherence of desired treatment can play a role in causal influence, but would ideally involve a distinct inquiry and analysis as the psychology of adherence can be isolated. We denote these as $\{U^{(i)}(t)\}$ meaning the i intervention (itself possibly a vector) as parametrized by time.

2. **Background Variables** - These are static variables such as a person's genome or demographic variables. They clearly are variables that can only have a causal direction emanating from the variable. We denote these throughout the text as $\{A^{(i)}\}$.

3. **Fast Observations** - These are taken from the clinical trial and are time dependent. The blood glucose level or the heart rate of an individual who has accepted being a subject in a clinical trial can be monitored more or less continuously with little difficulty. Moreover, this is a quantity which is expected to have significant variability across time, both long and very short term. We denote these by $\{O^{(i)}(t)\}$

4. **Slow Observations** These are meant to represent biomarkers that serve as metrics for an aggregate measure of health. These include blood pressure, serum cholesterol, body weight, etc. The critical distinction between Fast and Slow observations is that the slow observations are those that are expected to undergo significant change, that is at the level of sharp transitions in the state of morbidity, over longer time scales than that of the clinical trial. That is, the results of the clinical trial are meant to only create a direction of change, that is a differential with respect to the relevant time scales. This means we can safely enforce that there are no nonlinear effects at the level of proximate causes. We denote these by $\{R^{(i)}(t)\}$. The contrasting scaling to the Fast Observations will be formalized below.

5. **Latent State of Health** The state of health itself is a latent variable, that is, it cannot be observed (perfectly). However, there are clear biophysical processes involved that can be loosely defined as corresponding to long

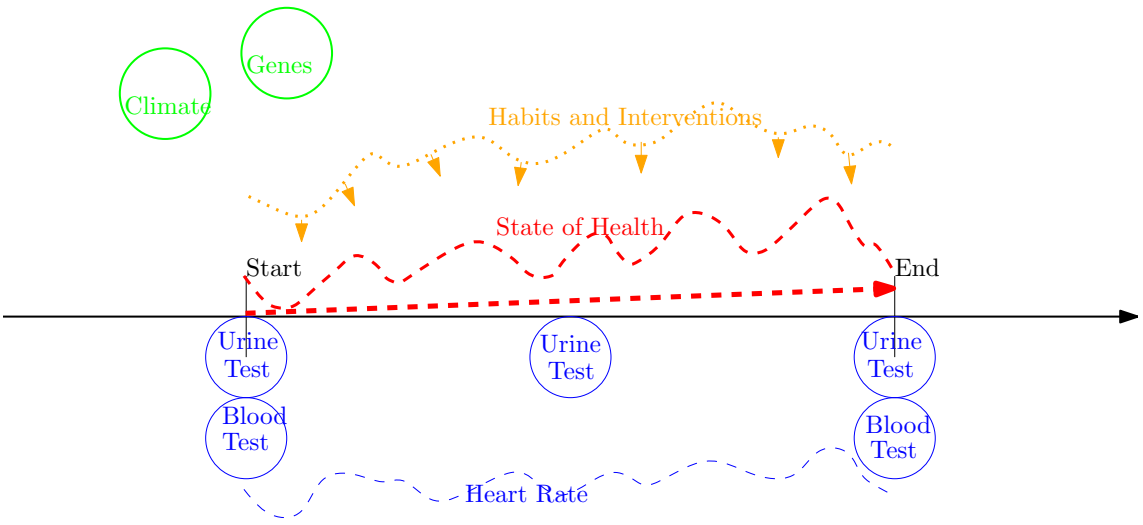


Figure 1: A Generic Model of the Variables Associated to a Clinical Trial.

term aggregate measures of the state of health of a person's organs and cells. This is continuous, and the dimensionality and the time scaling can be flexible as to the availability of data. We furthermore distinguish between fast instrumental latent variables that modulate the effects of interventions $\{V^{(i)}(t)\}$, their higher level representations $\{W^{(i)}(t)\}$ and the slow moving latent generalized states of health $\{X^{(i)}(t)\}$.

This resembles the Hidden Markov Model structure [5], and with the decisions, comes out to a Partially Observable Markov Decision Process [2]. Generically, with the presence of continuous variables, tractable inference and learning cannot be expected. As such, enforcing domain knowledge with structure is essential for the task.

Partially in order to mitigate some of the known disadvantages of such methods [1], we present significant inductive domain bias as to causal ordering (that is, treatments can be subject to **do** calculus and have weak, if any, causes, and observations are not causes in themselves). At the same time, it explicitly encodes multiple time scaling.

3.3 Considerations for Modeling

Structural Equation Models (SEMs) were developed by econometricians to attempt to apply time-dependent ordinary differential equations to fit time series data. (See Figure 1.) This presented an alternative strategy to ascertain causation between variables when completely randomized trials were unavailable. In this paper we present a SEM formulation for the setting described above as follows:

$$\begin{aligned}
\tfrac{dV}{dt} &= C(A, U(t), Z_C), \\
\tfrac{dW}{dt} &= G\left(A, W(t), \{W(t-\tau)\}_{t\in[0,\tau_M]}, \{V(\epsilon s)|_{s=0}^{t/\epsilon}\}, \textstyle\int_0^{t/\epsilon} \epsilon V(\epsilon s) ds\right), \\
\tfrac{dX}{dt} &= F(A, X(t), \{X(t-\tau)\}_{t\in[0,\tau_M]}, W(t), Z_F) \\
O(t) &= Y(A, X(t), Z_O), \\
R(t) &= Q(A, X(t), Z_R),
\end{aligned} \tag{1}$$

where

- Y is generally nonlinear with unstructured noise Z_O,
- We permit autoregressive effects through functional dependence on quantities with a time delay τ up to τ_M

In this formulation, a salient distinction from standard SEMs is the presence of two time scales. Specifically, note that the general state of a person's health, as far as the susceptibility to or even the presence of a chronic illness, is slow to change, as the known effects on nutrition on these matters are expected to take place through chronic

exposure over decades. By contrast, the period of time for any given clinical trial is in a matter of weeks. This suggests that any inputs or interventions decided upon in the clinical trial, as far as time series data, have much faster dynamics that the (latent) state of health.

Formally, we consider that the time scale of the data is of the order ϵ. The fast quantities $\{U(t), V(t), W(t)\}$ can be said to appear in the form for the *semi-flow* of $X(t)$, that is, the operator defining the infinitesimal pushforward of the state trajectory. This does simplify the potential modeling in the sense that there would be nothing potentially gained by considering nonlinear forms of F, as the pushforward along small time is equivalent to the temporal linearization.

For the aggregate modeling there are two possible approaches as far as handling the challenging circumstance of different time sampling for the decisions (nutrition) and outcomes (o-mics and biomarkers)

1. Model Figure (1) as a DBN with time stamps at every observed time, that is, with missing values for all other variables unreported at a particular time.

2. Model (1) as a DBN but with only the beginning, middle, and end as definitive time stamps. We use representation learning to summarize the fast time series as a set of latent variables.

Learning Dynamic Bayesian Networks from Data: Foundations, First Principles and Numerical Comparisons

Vyacheslav Kungurtsev, Petr Ryšavý, Fadwa Idlahcen, Pavel Rytíř, Aleš Wodecki

June 2024

Abstract

In this paper, we present a guide to the foundations of learning Dynamic Bayesian Networks (DBNs) from data in the form of multiple samples of trajectories for some length of time. We present the formalism for a generic as well as a set of common types of DBNs for particular variable distributions. We present the analytical form of the models, with a comprehensive discussion on the interdependence between structure and weights in a DBN model and their implications for learning. Next, we give a broad overview of learning methods and describe and categorize them based on the most important statistical features, and how they treat the interplay between learning structure and weights. We give the analytical form of the likelihood and Bayesian score functions, emphasizing the distinction from the static case. We discuss functions used in optimization to enforce structural requirements. We briefly discuss more complex extensions and representations. Finally we present a set of comparisons in different settings for various distinct but representative algorithms across the variants.

1 Introduction

In this paper we give a comprehensive presentation on the training of Dynamic Bayesian Networks (DBNs), including both structure and parameters, from data. DBNs present a naturally interpretable model when it comes to understanding the precise interaction underlying the relationship between the variables. That is, the conditional independence structure defined by the DBN provides information regarding the mechanistic procedure that defines the model. This is also associated with the field of statistics referred to as causal learning.

There is one general survey article on DBNs we found is [59], which provides a helpful comprehensive resource for references for DBN modeling, inference and learning. In this work rather than seeking to provide a comprehensive literature review, instead we focus on narrating the global landscape of the mathematical understanding of the most important considerations as far as learning a model, including both the structure and parameters, from data.

With this understanding we are able to establish an informative taxonomy regarding methods, providing transparency in regards to the function and intention of each method. There are important subtle distinctions as far as modeling assumptions between some different popular methods, and their awareness is critical for best practices of DBN learning. Finally, we perform comprehensive numerical comparisons, highlighting the particular advantages and disadvantages of each method. We highlight that these comparisons are not meant to be exhaustive or authoritative, but more informative and illustrative in regards to the tradeoffs associated with learning DBNs from data.

We make several assumptions for this work. These are not entirely formal for the sake of precise Theorem proving, but rather a general restriction of the data generating process of interest, so as to highlight the most pedagogical features of DBN learning, as far as it is most commonly done in the literature. These assumptions are restrictive as far as faithful statistical modeling of real world phenomena. However, they tremendously simplify the learning process, and thus allow a more comprehensible presentation of what can hope to achieve with standard simple methods. For transparency, we present them below.

1. *Causal Sufficiency* There are no unobserved confounders, i.e., there is a closed system defining the data generating process wherein all causal sources are observed. This permits for the conditional independence structures to be reliable indicators of edge links in the graph. This is a standard formal assumption in almost all learning algorithms for DBNs and BNs.

2. *Causal Identifiability* we focus on the general case wherein the data regime permits for potential identification of a true causal graph, generally corresponding to the number of data samples (trajectories $\times$ time steps) some exponential factor of mangitude greater than the number of variables (in practice this can mean 5 variables, 100 trajectories of each 50 time steps as a generic example). This permits us to focus on integer programming and other techniques that can obtain statistically significant point estimates for exact structures that recover a ground truth. This assumption is standard in the literature of DBN and SEM learning, as modeling uncertainty in less favorable data regime circumstances presents significant methodological challenges and considerations that require significantly more advanced techniques. However, understanding the nuances of the foundations are essential as far the proper development, implementation, and use of these techniques.

3. *Fully Observed* There are no hidden variables, all quantities of interest are fully observed at every time step. Of course, graphs with hidden (latent) variables and entire structures are instrumental for modeling in many fields. However, the inclusion of latent variables and the required Expectation Maximization modification of the procedures described presents technical complications that add significant additional complexity, and thus would necessity a much greater length while obfuscating the message.

We make a few departures from these assumptions throughout the work, which we explicitly indicate when they occur.

1.1 Contribution of this Work

In this work we present:

1. A thorough explicit analytical description of standard popular DBN representations and statistical models. This includes the structure of potential dependencies of transition functions of the time-dependent random variables on other variables, time independent as well as time-dependent and in-time, Markovian one time step back, and delayed dependence.

2. Extensive commentary and analysis of learning DBNs from data from both the classical PGM/BN perspective as well as the time series perspective. The relationship of learning to the structure of the data, as well as high level intuition on the complex interaction of learning the structure and the parameters is presented.

3. A presentation of the most standard and common criteria for defining the objective or cost of a particular DBN network structure as well as the functional forms of equations that define that the graph satisfies the structural requirements of a DBN, in particular acyclicity

4. Numerical results for examples of popular algorithms for learning DBNs, evaluated on a range of criteria and variety of problems. The set of examples is not meant to be exhaustive, nor the comparisons authoritative, but broadly illustrative of the relative advantages and disadvantages of the different methods available.

1.2 Applications of Dynamic Bayesian Networks

The discovery of dynamic Bayesian networks has found many applications, some of which are medicine [12, 20, 75, 8], economics [43, 44] and aviation [46, 65, 27]. The applications related to aviation are typically related to finding casual structure in a sub-problem related to the dispatch of flights and focused on risk mitigation. The medial applications typically focus on either the discovery of fundamental principles related to chemical reactions, which take place in biological organisms or the extraction of information from clinical data. In economics, it is typically of interest to uncover relationships between the stock market and other selected factors that either influence or are influenced by it. In the following, we give some details about chosen applications as well as specific outcomes that the modelling using DBNs has in practice.

1.2.1 Medical Science

To highlight the importance of DBN discovery in medicine, we detail three separate applications. The first of these focuses on the quantification of disease development [8]. Understanding the progression of diseases is crucial in clinical medicine, as it informs the effectiveness of treatments. Most clinical medicine and pathology textbooks provide detailed descriptions of disease progression and treatment responses. However, there has been limited research quantifying these descriptions in detail. Typically, research examines the temporal aspect by describing treatment outcomes after a certain period. A significant challenge in gaining deeper insights is the relatively small size of clinical datasets, often comprising only a few hundred patients.

In the aforementioned contribution, a heuristic procedure is proposed for exploring and learning non-homogeneous time dynamic Bayesian networks, aiming to balance specificity and simplicity. The approach begins with a fully homogeneous (in time) model parts of which are gradually replaced with sub-models which reflect the expected structure at a given level of time delay. Furthermore, a splitting technique is applied to further improve the predictive behavior of the model, such models are typically termed partitioned DBNs.

A heuristic method was proposed to learn the DBN on synthetic data, which has a structure that should reflect a real data set. The numerical performance in terms of accuracy and solution time reported is hopeful. However, the method has yet to be tested on real world datasets.

The second application is concerned with the mapping of neural pathways [20]. Identifying functional connectivity from simultaneously recorded spike trains is crucial for understanding how the brain processes information and instructs the body to perform complex tasks. The study investigates the applicability of dynamic Bayesian networks (DBNs) to infer the structure of neural circuits from observed spike trains. A probabilistic point process model was employed to assess performance. The results confirm the utility of DBNs in inferring functional connectivity

as well as the directions of signal flow in cortical network models. Additionally, the findings demonstrate that DBNs outperform Granger causality when applied to populations with highly non-linear synaptic integration mechanisms.

The third chosen application focuses on the choice of appropriate treatment regimens for Chronic lymphocytic leukemia (CLL) [75]. This cancer is the most common blood cancer in adults, with a varied course and response to treatment among patients. This variability complicates the selection of the most appropriate treatment regimen and the prediction of disease progression. The aforementioned paper aims to develop and validate dynamic Bayesian networks (DBNs) to predict changes in the health status of patients with CLL and predict the progression of the disease over time. Two DBNs, the Health Status Network (HSN) and the Treatment Effect Network (TEN), were developed and implemented. Relationships linking the most important factors influencing health status and treatment effects in CLL patients were identified based on literature data and expert knowledge. The developed networks, particularly TEN, were able to predict the probability of survival in CLL patients, aligning with survival data collected in large medical registries. The networks can tailor predictions by integrating prior knowledge specific to an individual CLL patient. The proposed approach is a suitable foundation for developing artificial intelligence systems that assist in selecting treatments, thereby positively influencing the chances of survival for CLL patients.

1.2.2 Economics

The relationship between the stock market and national economies deepens as the market matures, highlighting the need to study their dynamic interplay. Economic indicators such as real income and savings rates play crucial roles in influencing stock market capitalization. Macroeconomic fundamentals wield considerable influence over both short and long-term periods. Some researchers argue that finance and economic growth are causally linked, suggesting the stock market's potential to drive economic development. However, not all macroeconomic factors significantly impact stock prices. Understanding the strength of association among these variables offers insights into how the stock market behaves across varying economic landscapes. Research on the Chinese stock market examines how macroeconomic variables shape stock market indices over time, emphasizing the enduring influence of economic fundamentals amid short-term market fluctuations. The aforementioned interplay may be modeled by DBNs and has been detailed in [44].

In the article and analysis of the relationship between the stock market and economic fundamentals using 11 selected factors is modeled using a DBN. Among these, four factors pertain to stock market indicators, while the remaining factors focus on macroeconomic and policy considerations. The first four factors reflect stock market performance, with defensive and cyclical stocks exhibiting varying behaviors during bull and bear markets. The Stock Exchange 50 index, comprising the 50 largest and most liquid stocks in the Shanghai Securities Market, supplements the overall stock market observation. Additionally, the consumer stock index serves as an indicator of societal consumption levels, typically rising during favorable macroeconomic periods and declining during economic downturns.

The results on real data of the described method are mixed. The application to the Shanghai composite market yielded some positive results in terms of the prediction of macroscopic quantities, but only limited success in terms of constituent market price prediction. The modeling of the components of the market in sufficient detail is a difficult problem due to the numerical

tractability being limited as the number of variables increases.

1.2.3 Aviation

The global collection of aircraft and the airspace in which they operate, is a complex system generating a vast amount of data, making it a challenging domain to model mathematically. This system includes critical elements such as aircraft, airports, flight crews, weather events, and routes, each with many subcomponents. For example, aircraft have numerous subsystems and components, each subjected to various stresses and maintenance actions, which influence their time dynamics. Airports have multiple runways and internal logistical processes, which influence the operational capacity. These system components interact in complex ways. For instance, each flight corresponds to an aircraft operated by a flight crew traveling from one airport to another via a route that may need to change due to weather. Multiple flights operate simultaneously, requiring coordination to avoid incidents while maximizing throughput and minimizing delays.

To give an idea about a specific aviation problem that may be tackled we describe the airport operation uncertainty characterisation, which has been developed in [27]. The model outlines aircraft flow through the airport, emphasizing integrated airspace and airside operations. It characterizes various operational milestones based on an aircraft flow's Business Process Model and Airport Collaborative Decision-Making methodology. Probability distributions for factors influencing aircraft processes need to be estimated, along with their conditional probability relationships. This approach results in a dynamic Bayesian network that manages uncertainties in aircraft operating times at the airport. The nodes of the network describe various aspects of the airport and flight operations. They cover meteorological conditions, arrival airspace variables such as timestamps and congestion metrics, airport infrastructure, operator and flight data, airside operational times and flight regulations, and the causes of delays.

The key outcomes of this work include the statistical characterization of processes and uncertainty drivers, and a causal model for uncertainty management using a DBN. Analyzing 34,000 aircraft operations at Madrid Airport revealed that arrival delays accumulate throughout the day due to network effects, while departure delays do not follow this pattern. The major delay drivers identified were the time of day, ASMA congestion, weather conditions, arrival delay amount, process duration, runway configuration, airline business model, handling agent, aircraft type, route origin/destination, and ATFCM regulations. Departure delays are significantly impacted by events of longer duration, which also offer greater potential for recovery.

2 Background - Dynamic Bayesian Networks

We present the general, and then specific forms, of DBN models. Consider that there is an n-dimensional stochastic process $X(t)$. The individual random variables $X_i(t)$ for all $i \in [n_x]$ can be valued as discrete, or as members of some field, such as $\mathbb{R}$. In addition, there can be an n_z-dimensional random variable Z. Let us denote the generic spaces as $\mathcal{X}$ and $\mathcal{Z}$, respectively.

The defining character of DBNs is modeling the dependence of $X_i(t)$ on other quantities, i.e., defining the the evolution of the stochastic process $X(t) \to X(t+1)$. Formally, for the probability kernel defining the iterations of the stochastic process, the dependence must be Markovian, that is

$$p(X_i(t+1) \in A) = f_i(X(t), X_{j\neq i}(t+1), \{X_i(t-\tau)\}_{\tau=1,\dots,p}, Z) \tag{1}$$

where A is some set in the Borel σ-algebra of $\mathcal{X}$. That is, the transition kernel can depend on the current state of the other random variables, the values of the random variables at the previous time, the time-independent variables Z, as well as a possibly autoregressive effect through dependence on $\{X_i(t-\tau)\}_{\tau=1,...,p}$.

In addition, there is the important requirement that no in-time string of dependencies forms a cycle. This presents the necessity in introducing graph theoretical notions to precisely characterize DBNs. Generically, we say a directed graph is a set of vertices $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ and edges $\mathcal{E} = \{e_1, e_2, ..., e_m\}$, where $e_j = (v_{j_1}, v_{j_2})$ denotes the existence of a directed path between the two notes $v_{j_2} \to v_{j_1}$. We also say in this case that j_2 is a parent of j_1, or $j_2 \in dpa(j_1)$.

The DBN model incorporates a directed acyclic graph (DAG) $\bar{\mathcal{G}} = \mathcal{G}(\mathcal{V}(X(t-1), X(t)), \mathcal{E}_d) \cup \mathcal{G}(\mathcal{V}(X(t), Z), \mathcal{E}_z) \cup \mathcal{G}(\mathcal{V}(X(t), X(t-\tau)), \mathcal{E}_\tau)$. The first two define connections in the model between the temporal random variables. That is $e = \{V_1, V_2\} \in \mathcal{E}_d$ with $V_i \in \{\{X_i\}\}$ if $p(X(t+1)_i \in A)$ is a function of $V_2 = X_j$, that is $j \in dpa_d(i)$, and $e = \{V_1, V_2\} \in \mathcal{E}_s$ if $p(X_i(t+1) \in A)$ is a function of $V_2 = X_j(t+1)$, that is $2 \in dpa_s(i)$. Finally, we also have a (non-symmetric) matrix encoding the dependencies on the self-history $\tau \in dpa_\tau(i) \subset \{1, ..., p\}$ and the dependencies on the static variables $dpa_z(i) = \left\{ Z_j : \frac{\partial f_i(\cdot)}{\partial Z_j} \neq 0 \right\}$. These, of course, can be encoded as graphs as well.

This permits us to write (1) as,

$$p(X_i(t+1) \in A) = f(A, \{X_j(t)\}_{j \in dpa_d(i)}, \{X_j(t+1)\}_{j \in dpa_s(i)}, \{X_i(t-\tau)\}_{\tau \in dpa_\tau(i)}, \{Z_j\}_{j \in dpa_z(i)}) \tag{2}$$

Notice that the encoding of the explicit dependencies presents the possibility of using a common f as opposed to one depending on the transition out-node i, in the case wherein all the variables X_i are of the same distributional family. This eases the computation of the likelihood of the data given the parameters and structure, etc.

We will sometimes use, for shorthand:

$$\{V_j(t+1)\}_{j \in dpa(i)} = \{X_j(t)\}_{j \in dpa_d(i)} \cup \{X_j(t+1)\}_{j \in dpa_s(i)} \cup \{X_i(t-\tau)\}_{\tau \in dpa_\tau(i)} \cup \{Z_j\}_{j \in dpa_z(i)} \tag{3}$$

See Figure 1 for an illustration. In this case for $i = 1$, the Markovian transitions are from itself and from X_2, and there are no intra time nodes or static nodes directed to it, and so $V_{dpa(1)}(t) = V_{dpa_d(1)} = \{X_1(t-1), X_2(t-1)\}$. For node 2, there are no Markovian transitions and two intra-node dependencies, thus $V_{dpa(2)}(t) = V_{dpa_s(2)}(t) = \{X_1(t), X_3(t)\}$. Finally, for $X_3(t)$, there are two Markovian dependencies, and a static covariate dependence. Thus $V_{dpa(3)}(t) = V_{dpa_d(3) \cup dpa_z(3)}(t) = \{X_1(t-1), X_3(t-1), Z\}$.

Now that we have established the general form of the DBN, we see that we have a fundamentally still very general problem to solve, in that the function f can encode any sort of dependency on the different variables in the parent set of the note of interest. They can depend as according to various nonlinear interactions, that can themselves embody different conditional independence information In order to complete the model, we need to define the form of the function f.

2.1 Simple Parametric Conditional Probability Dependency

For certain kinds of variables, it becomes both possible and prudent to use certain simple parametric families for defining f. In particular, for binary Bernoulli random variables correspond to Dirichlet distributions for the prior of the weights together with using Conditional Probability

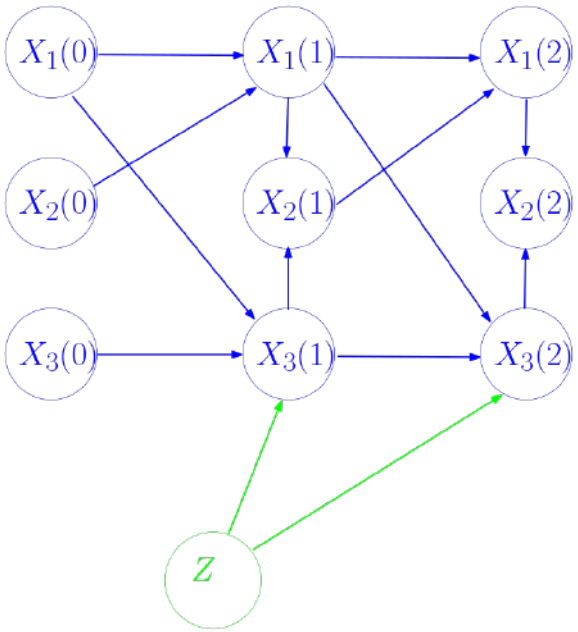


Figure 1: A Possibl DBN Graphical Network defining the transitions of $X(t)$

Tables to define the form. For continuous random variables, Gaussian linear models provide a means of computing the maximum likelihood linear parameters using covariance matrices.

A significant advantage of using parametric families arises from the closed form computation of criteria, which permits closed form computation of the marginal posterior of a structure. This permits structure learning algorithms to be able to score graphs offline, assisting in the search. Many score maximizing procedures such as [26, 14, 3, 2] use this approach. The score is ultimately an integration of the posterior of the parameters in the model given the structure, which also indicates that the sampling of the optimal parameters, once obtaining the maximum a posteriori structure, is straightforward for these models.

In addition, one can use neural models including the Generative Flow Network approaches as given by [17, 2]. These use a Reinforcement Learning iteration to ultimately sample from a high scoring network as according to a defined score. Reinforcement Learning broadly, e.g., [74] is another framework by which the structure search for these standard specific models can be aided by neural networks.

Linear Structural Equation Models (SEMs) present an opportunity to use an adjacency matrix to define both the structure and weights in a computationally advantageous form. This highlights the correspondence between the Dynamic Systems and the graph theoretic developments in causal learning.

2.1.1 Discrete Variables

Binary Variables The case of binary random variables is wherein $X_i(t) \in B(1, p_i^X), Z_j \sim B(1, p_j^Z)$, etc., that is, they are all of Bernoulli type. Empirical samples for all $k \in [K]$, where

k indexes a set of sample trajectories satisfy $X_i^{(k)}(t), Z_j^{(k)} \in \{0,1\}$ for all t, i, and j. With this most simple scenario, the modeling flexibility as well as the nuances of structure learning becomes a natural pedagogical start.

There is a degree of flexibility in the choice of statistical model for defining the transition function. We will explore three options - the noisy or model, the linear logit model, and the complete linear logit model.

The *noisy or model* defines the transitions as

$$\begin{aligned} p(X_i(t+1)=0) &= (1-\lambda_0)\textstyle\prod_{l\in dpa(i)}(1-\lambda_l)^{V_l} \\ p(X_i(t+1)=1) &= 1-(1-\lambda_0)\textstyle\prod_{l\in dpa(i)}(1-\lambda_l)^{V_l} \end{aligned} \tag{4}$$

This model is referred to as noisy or because essentially it calculates a probabilistic perturbation of the binary OR operation. This model presents one implementation of causal independence, wherein the influence of each covariate is independent with respect to the others.

For the linear logit models, define the sigmoid function,

$$\sigma(x) = \frac{e^x}{1+e^x}$$

The reason the models we define next are referred to as linear is that the transition is defined to be,

$$X_i(t+1) = \sum_{j\in dpa_d(i)} \beta_j^d X_j(t) + \sum_{j\in dpa_s(i)} \beta_j^s X_j(t+1) + \sum_{\tau\in dpa_\tau(i)} \beta_\tau^a X_i(t-\tau) + \sum_{j\in dpa_z(i)} \beta_j^z Z_j \tag{5}$$

We shall see that this linear form is broadly common in modeling the transitions of variables in DBN models for other variable types.

The probability kernel given by (5) is

$$p(X_i(t+1)=1) = \sigma\left(\beta_0 + \sum_{j\in dpa_d(i)} \beta_j^d X_j(t) + \sum_{j\in dpa_s(i)} \beta_j^s X(t+1)_j + \sum_{\tau\in dpa_\tau(i)} \beta_\tau^a X_i(t-\tau) + \sum_{j\in dpa_z(i)} \beta_j^z Z_j\right) \tag{6}$$

One alternative that frequently arises in practice is the necessity to accurately model *Conditional Probability Dependencies* (CPDs) as defined by *Conditional Probability Tables* (CPTs). As an example, please see Table 1.

It is clear that the information in Table 1 cannot be modeled with a linear transition function as in (5). In this case, if one wanted to construct such a model, one would instead have to be able to include all of the combinations between the possible parent nodes.

Formally, a transition model could look like, for Table 1,

$$\begin{aligned} X_1(t+1) = \beta_0 + \beta_1 Z_2 + \beta_2 X_3(t+1) + \beta_3 X_3(t+1) Z_2 + \beta_4 X_3(t+1) Z_2 + \beta_5 X_2(t) \\ + \beta_6 X_2(t) Z_2 + \beta_7 X_2(t) X_3(t+1) + \beta_8 X_2(t) X_3(t+1) Z_2 \end{aligned}$$

and in the general case,

$$X_i(t+1) = \prod_{j\in dpa_d(i)} \prod_{k\in dpa_s(i)} \prod_{\tau=1,\ldots,p} \prod_{l\in dpa_z(i)} \textstyle\sum_{\alpha\in\left(\mathbb{Z}_2^+\right)^4} \beta_{i,j,k,l,\tau}^{\alpha} \left(X_j(t) X_k(t+1) X_i(t-\tau) Z_j\right)^{\alpha} \tag{7}$$

Table 1: An example of a CPT

$X_2(t)$	$X_3(t+1)$	Z_2	$X_1(t+1)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

where the parameters are appropriately normalized. With the combinatorial explosion in this model clearly visible, it can be seen that such circumstances present significant difficulties as far as computing hardware expense in both processing and memory, when it comes to modeling high dimensional datasets.

On the other hand, this structure of statistical model presents two structural conditions denoted as *local parameter independence* and *unrestricted multinomial distribution* [61]. These ensure that for every *configuration*, that is, every possible combinations of values of the parents of a node, there is an independent parameter vector. This leads to a corresponding combinatorial explosion of parameter vectors in the statistical model. On the other hand, however, the analytical calculation of parameter likelihoods and posterior distributions become possible, facilitating more straightforward evaluation of scoring metrics quantifying the information quality of an entire (D)BN. That is, the marginal likelihood of a structure can be computed without first computing the likelihood of the weights.

Multinomial Multinomial distributions are over discrete valued random variables that can take on multiple possible values. The distinction between a user friendly linear parameter presentation and the expressiveness at the cost of parametric dimensionality of unrestricted multinomial distributions becomes apparent in the increased complexity of modeling multinomial relative to Bernoulli distributions.

Now, consider that for every i, $X_i(t) \in \{u^1, ..., u^m\}$ some multinomial sample, with Dirichlet sampled initial values, and always with some multinomial distribution $\{\theta_i^m(t)\}$. The set of parameters indicating the probability that $X_i(t+1) = u^k$ given a particular configuration of the parent nodes $V_{dpa(i)}(t+1)$ is denoted $\theta^k_{i,v_{dpa(i)}}$.

First, consider a linear model. Let us simplify the notation,

$$\sum_{j\in dpa_d(i)} \beta_j^d X_j(t) + \sum_{j\in dpa_s(i)} \beta_j^s X_j(t+1) + \sum_{\tau\in dpa_\tau(i)} \beta_\tau^a X_i(t-\tau) + \sum_{j\in dpa_z(i)} \beta_j^z Z_j = \sum_{j\in dpa(i)} \beta_j V_j(t+1)$$

With this, the form of the transition probability is,

$$p(X_i(t+1) = u^l) = \frac{\exp\left(\beta_{i,0} + \sum_{j\in dpa(i)} \sum_{q\in[m]} \beta_{i,j}^{l,q} \mathbf{1}(V_j = u^q)\right)}{\sum_{s\in[m]} \exp\left(\beta_{i,0} + \sum_{j\in dpa(i)} \sum_{q\in[m]} \beta_{i,j}^{s,q} \mathbf{1}(V_j = u^q)\right)} \tag{8}$$

which is a standard linear logit.

On the other hand, with an unrestricted multinomial distribution, we can define the full transition distribution explicitly, meaning for every possible combination of values instantiated by a node's parents in a given network, we define a specific probability. In this case even the already cumbersome notation of (7) is insufficient to present the model representation. On the other hand, we will see that this representation eases the likelihood and Bayesian score computations. Finally, we distinguish $dpa(i) = dpa_t(i) \cup dpa_z(i)$ as the time-dependent and time-independent covariates. We also distinguish the possible values of Z to be $Z_j \in \{1, ..., w^q\}$

We simply denote:

$$p(X_i(t+1) = u^l | V_{dpa(i)}(t+1), \theta_i) := \theta_i^{\xi(V_{dpa(i)}(t+1)}, \xi \in \Xi_i, \Xi_i = \prod_{j \in dpa_t(i)} \mathbb{Z}_m^+ \times \prod_{j \in dpa_z(i)} \mathbb{Z}_q^+ \quad (9)$$

That is, there is an multi-index that enumerates the entries of Ξ_i for each transition i. We can see that this presents a highly parametrized model, which will imply significant parametric uncertainty when there are finite data samples. On the other hand, with this highly precise model, the maximum likelihood becomes much more straightforward to compute, as well as the Bayesian scores. Indeed, this is exactly what local parameter independence facilitates – you can compute the likelihood by counting the instances of each transition and dividing by the count of each predecessor configuration. On the other hand, when the total count of every possible predecessor configuration is low, due to unfavorable sample complexity, this cannot be said to be a high quality estimate of the actual validity of that dependence. On the other hand, by letting these parameters take on distributions, in a Bayesian setting, the computation of a posterior for a structure becomes easier, and the uncertainty is available by sampling the posterior, anyway.

2.1.2 Continuous Variables

Gaussian Variables A Gaussian Bayesian Network can be considered a continuous variable equivalent to binary variables in the sense that the structure permits closed form expressions for computing the likelihood, posterior, etc. In this case, however, the additive linear term is standard. Formally, we assume that $X(t) \sim \mathcal{N}(\mu; \Sigma)$. The transition function becomes:

$$\begin{aligned} p(X_i(t+1) | dpa_d(i) \cup dpa_s(i) \cup dpa_\tau(i) \cup dpa_z(i)) = \mathcal{N} \Bigg(& \beta_0 + \sum_{j \in dpa_d(i)} \beta_j^d X_j(t) \\ & + \sum_{j \in dpa_s(i)} \beta_j^s X_j(t+1) + \sum_{\tau \in dpa_\tau(i) \subset \{1,...,p\}} \beta_\tau^a X_i(t-\tau) + \sum_{j \in dpa_z(i)} \beta_j^z Z_j; \sigma^2 \Bigg) \end{aligned} \quad (10)$$

and the result is that,

$$\begin{aligned} & \mu_{X(t+1)} = \beta_0 + \beta^T \mu, \, \sigma^2_{X(t+1)} = \sigma^2 + \beta^T \Sigma \beta, \\ & Cov\left[\{X(t)\}, \{X(t+1)\}, \{X(t-\tau)\}, Z; X(t+1)\right] = \sum \beta_j \Sigma_{i,j} \end{aligned}$$

Indeed in [36, Theorem 7.3-7.4] it is shown that there is a bidirectional equivalence between such a normal joint distribution and normal transition function.

We shall see that this permits, in the temporal case, a repeated composition of the propagation of the covariance with each time step, when computing the likelihood and performing inference. This is associated with the deep theory of filtering methods, which typically studies Gaussian DBN propagation with a simple state-observable structure.

Exponential Family Functional Form An exponential family is defined with, recalling $\mathcal{X}$ to be an abstract space for which both $X(t), Z \in \mathcal{X}$,

1. A sufficient statistics function $\tau : \mathcal{X} \to \mathbb{R}^K$ for some K
2. A convex set of a parameter space $\Theta \subset \mathbb{R}^m$
3. A natural parameter function $t : \mathbb{R}^m \to \mathbb{R}^K$
4. A measure A over $\mathcal{X}$

The exponential family is a distribution of the form

$$P_\theta(\xi) = \frac{1}{Z(\theta)} A(\xi) \exp\left\{(t(\theta), \tau(\xi)\right\},\ Z(\theta) = \sum_{\xi} A(\xi) \exp\left\{(t(\theta), \tau(\xi))\right\} \tag{11}$$

The case of natural parameters is the most standard, and the one we have been exploring in the formulations above, this corresponds to $(t(\theta), \tau(\xi)) = (\theta, \tau(\xi))$. One has to be careful, however, in constraining the space of parameters θ to ones normalized, i.e.,

$$\Theta = \{\theta \in \mathbb{R}^m : \int \exp((\theta, \tau(\xi)))d\xi < \infty\}$$

Linear Structural Equation Models Consider the general case wherein the function f, as given by (1), is given a linear parametrization with respect to continuous variables $X(t), Z$, as in (5), however for continuous variables. One can then perform learning by minimizing the appropriate least squares fit to the data. This is most common in the approach of Linear Structural Equation Models, in which case a linear parametrization permits greater computational ease.

Linear Structural Equation Models (LSEMs) are the most common non-Gaussian DBN for modeling continuous variables. With LSEMs (see, e.g. [7] for a general reference and [53] for application to causal inference) presume a general linear structure that is associated with a discretization of a dynamical system:

$$\dot{X}(t) = f(X(t), Z)$$

with this generality, there is a degree of ambiguity in the literature, because there are a number of ways to consider a discrete model of this.

An SEM could refer to a purely time-instant (static) model, with dependencies dpa_s and dpa_z only, as in [17, 45]. More recently, DBNs more broadly have become interchangeable with SEMs, for instance, the representation in [50] has all dependencies as described here except, it can be argued for simplicity, Z.

Nonlinear, Nonparametric and Neural Models The structure of f, or even if there is an f at all, is of course flexible like with any statistical modeling. More complex statistical models for the transition introduce significant additional difficulties in training, by adding nonconvexity to the landscape and significantly expanding the degrees of freedom in the model that need to be fit with data. Given the emphasis in this article on simple illustrative DBNs, we will but briefly mention some examples, neither comprehensive nor authoritative.

Broadly speaking, there are a number of popular parametric forms of nonlinear models that can be used from time series literature, e.g. [19]. Neural networks have enabled computationally intensive empirical unsupervised time series models [22, 9]. Nonlinear models in the SEM statistical community have also been studied [42]. The work [58] uses splines to model the nonlinear relationships in the transition distributions. The work [35] uses a kernel nonparametric regression model to learn DBNs for gene regulatory networks.

3 Learning From Multiple Trajectories

Consider that we receive N samples of trajectories $\mathcal{T}^j$, each with a total time of T

$$\mathcal{S} = \cup_{n=1}^{N} \mathcal{T}^n = \{Z^{(n)}, X^{(n)}(0), X^{(n)}(1), X^{(n)}(2), ..., X^{(n)}(T)\}_{n=1,...,N} \tag{12}$$

and we are interested in fitting a DBN model to this data. This amounts to defining the specific form of f in (1). More specifically, it amounts to identifying the parents of each $X_j(t)$ in the graph $\bar{\mathcal{G}}$, as well as specific functional form of the transition function f.

3.1 Maximum Likelihood Calculations

In reviewing the literature on learning DBNs from data, it is typical to disregard the distinction of the trajectory sample $\mathcal{T}^i$ and the time transition samples $\{X^{(i)}(t), X^{(i)}(t+1)\}$. As far as understanding the meta-methodological cause of this, it appears that this can be said to be due to DBNs being considered not uniquely, but as a special kind of Bayesian Network, or as splices of the same time series trajectory.

Consider the two methodological components thereof, time series analysis and PGMs. For the latter, consider two popular works that are effectively extensions of methods developed for BNs extended to DBNs, the continuous reformulation of the problem into one with adjacency matrices as decision variables, called "NOTEARS" in the static case [72] and "dynotears" [50], as well as the use of "Generative Flow Networks", a Reinforcement Learning-motivated sampler, for the static case in [17] and the dynamic case in [2]. It can be seen that in all of these cases, the likelihood is expressed as,

$$\begin{aligned} p\left(\mathcal{S}|\theta_G, \bar{\mathcal{G}}\right) = \prod_{n=1}^{N} \prod_{s=1}^{T} p\Big(X^{(n)}(T-s+1)|\theta, Z^{(n)}, \{X_j^{(n)}(T-s)\}, \\ \{X_j^{(n)}(T-s+1)\}, \{X^{(n)}(T-s-\tau)\}_{\tau=1,...,p}\Big) \end{aligned} \tag{13}$$

and with the standard application of the logarithm, change into a sum, and maximization, or a posteriori maximization through a Bayesian criterion, as the target.

And similarly, in consulting standard texts on time series analysis with detailed derivations of Likelihood computation for various models, e.g. [19, 47], we see that in the derivations of the likelihood, the data is considered to be a sequence of i.i.d. observations, that is, a sequence of observations from a stochastic process $\{\hat{X}(0), \hat{X}(1), \hat{X}(2), ..., \hat{X}(T)\}$, rather than the general form given in (12), and is fit to (13), just with a simpler expression in the sum index.

This presents the natural question as to whether or, in light of this expression's universal use, why, these approaches "commute", that is, whether the equivalent expressions for the likelihood, brought from different perspectives, are appropriately equivalent and true.

We shall see that indeed, arithmetically, the expression for the likelihood is correct for DBNs, and so this makes the calculation of the

3.2 Considerations from Axioms of Causal Learning

DBNs, compared to BNs, contain both time-varying as well as static variables. While the aggregate structure is still a DAG, suggesting that formally many of the same principles regarding inference as well as structure and weight learning in BNs carry over to DBNs, the presence of time, especially when long trajectories are expected, adds significant complications.

Consider having a set of T sampled trajectories. On the one hand each trajectory is i.i.d., but above that, each time point relative to the previous presents an additional sample, with additional information. This presents the question: how can we distinguish the amount, and specific utility, of information gained from an additional trajectory, versus that gained from an additional time point?

This indicates the utility of including both static Z and dynamic X variables in the model. Instead one can consider a new trajectory as a new sample of $\hat{Z}$, which itself samples $X(0) \sim \pi(X(0)|Z)$ then, $X(0), X(1), ..., X(T)$. As such, one has T samples in order to learn $P\left(X(t+1)|X(t), \hat{Z}\right)$. However, what can be said about how informative a marginal trajectory is towards learning $P\left(X(t+1)|X(t)\right)$, that is, the marginal conditional over the population of Z?

It seems intuitive that in some way $P(\hat{Z})$ as well as $\pi(X(0)|Z)$ should weigh the in, where $P(\hat{Z})$ is the population prior of $P(\hat{Z})$, corresponds to the information gained for $P\left(X(t+1)|X(t)\right)$. For continuous variables, the information depends on the cross correlation as the prior evaluation is perturbed. It is clear then that information complexity is actually benefited from low variance, or low cardinality of a discrete space, between trajectories. Thus, the DBN model is particularly suitable for understanding long and complex time evolution of systems that do not change much in different contexts.

Recall that causal sufficiency requires that all confounding variables be present and observed. It is clear that different trajectories represent some distinctions in circumstance of object that the observations are taken from. If this is a latent variable, this presents an insurmountable probably to identification.

As far the required observations for causal identification, there exists at least one Z such that for all trajectories Z is observed, and Z is in the parent of some $X(t)$. We can consider that the classic *Randomized Clinical Trial* is exactly that $Z_H \in \{0, A\}$ and then testing for $p(X(t+1)|X(t), Z_\setminus, Z_H = 0) \neq p(X(t+1)|X(t), Z_\setminus, Z_H = A)$, with a null and alternative hypothesis and $Z_\setminus$ as other covariates, assumed to me completely independent of Z_H.

The less that $Z_\setminus$ mediates the transitions, the more the trajectories can be treated as independent.

3.2.1 Conditional Independence and d(irected)-separation

Let G be a (D)BN. Let $X_1, \ldots, X_n$ be the set of random variables of (D)BN. Let V, W be subsets of $\{1, \ldots, n\}$. We say that the set X_V is conditional independent of X_W given X_Z if the following condition holds:

$$P(X_V|X_W, X_Z) = P(X_V|X_Z).$$

Independence of various sets of variables can be determined by examining d-separation (d means directional) criterion of the (D)BN dag [13].

A (undirected) trail $T = (V_T, E_T)$ (path that does not contain any vertex twice) of G is *blocked* by the set Z if $\forall v \in V(G)$ either (i) $v \in Z \cap V_T$, and in-degree of v is at most 1; (ii) $v \notin Z$ and *children*$(v) \cap Z = \emptyset$, and both arcs of T connected to v are directed to v. The sets V

and W are d-separated if any trail between V and W is blocked by the set Z. If V and W are not d-separated, we say that they are d-connected.

The set X_V is conditional independent of X_W given X_Z, if V and W are d-separated by Z.

3.2.2 Causal Sufficiency

We discuss various definition from [5].

Definition 3.1 *We say that $U = u$ is directly sufficient for $V = v$ if for all $c \in R(V - (X \cup Y)$ and all $u \in R(U)$ it holds that $(M, u) \models [X \leftarrow x, C \leftarrow c]Y = y$.*

Definition 3.2 *We define that $X = x$ is strongly sufficient for $Y = y$ if there is an $N = n$ such that $Y \subseteq N$ and y is a restriction of n to Y and $X = x$ is directly sufficient for $N = n$*

Definition 3.3 *We define $X = x$ is weakly sufficient for $Y = y$ in M if for $u \in R(U)$ it holds that $(M, u) \models [X \leftarrow x]Y = y$*

3.2.3 Causal discovery and Inference

The problem of Causal discovery is to find a true graph G as the best possible explanation of the given data. There are various causal discovery methods. Such as score-based algorithms, which try to recover the true causal graph by finding a graph that maximize a given scoring function. Another example are Constraints based algorithms or continuous optimization algorithms.

3.3 Closed System Graph Causal Identification Model and Likelihood Information

In an effort to establish appropriate first principles by which to study the computational and statistical properties of joint structure-parameter learning in DBNs, we will present two definitions of specific setting and problem. In this first case, we consider the more mathematically convenient circumstance of causal sufficiency, or more broadly, a closed system whereby all of the forces and mechanisms influencing the random variables are either observed, or are ultimately latent variables that are completely determined by observed variables.

Closed System Graph Causal Identification Model: Assume that $\{X(t), Z\}$ are random variables whose interdependencies are fully described y some theoretical DBN defined by a graph $\bar{G}$ and $\tilde{f} \approx f$, there $\tilde{f}$ is defined as the transition function given by

$$p(X_i(t+1) \in A) = f(X(t), X_{j \neq i}(t+1), \{X_i(t-\tau)\}_{\tau=1,\ldots,p}, Z) + \epsilon$$

wherein ϵ is a zero mean error term. This additive noise model formulation has been leveraged to establish results on the identifiability of the structure $\bar{\mathcal{G}}$ [53, 32].

The statistical task is as follows:

- Frequentist: Given $\mathcal{S}$, identify the correct ground truth $\bar{\mathcal{G}}$ and a set of parameters that maximizes the likelihood of the data given the model, $\hat{\theta}$.
- Bayesian: Given $\mathcal{S}$ and some background prior uncertainty knowledge over the structure $\pi_G(\bar{\mathcal{G}})$ and parameters $p(\theta|\bar{\mathcal{G}})$, find the a posteriori distribution over the graphs $p\left(\bar{\mathcal{G}}|\mathcal{S}\right)$ and, hierarchically, the weights $p\left(\theta|\bar{\mathcal{G}}, \mathcal{S}\right)$.

As $|\mathcal{S}| \to \infty$, it is known that standard scoring and likelihood metrics enable recovery of the ground truth structure and parameters $\left(\bar{\mathcal{G}}, \theta^{\bar{\mathcal{G}}}\right)$. However, with the superexponential scaling of possible graph structures

Finally, let us investigate in more detail the validity of the iid-inter-intra-trajectory assumption implicit in the likelihood form given by (13).

Consider that we have two observed trajectories for three time steps, that is,

$$\mathcal{S} = \left\{X^{(1)}(0), X^{(1)}(1), X^{(1)}(2), X^{(1)}(3), X^{(2)}(0), X^{(2)}(1), X^{(2)}(2), X^{(2)}(3)\right\}$$

We know the trajectories themselves are independent, so we can write the likelihood as a product. The critical consideration now is the treatment of the starting value $X^{(i)}(0)$. It can be taken as an exogenous variable, which would place it in the same role as the conditioned parameters θ and $\bar{\mathcal{G}}$. Alternatively, a prior of $p\left(X(0)|\theta^0, \theta, \bar{\mathcal{G}}\right)$ would specify that a particular DBN is associated with certain starting points. However, notice that we must add an additional parameter θ^0, which would functionally play a similar role as simply conditioning on $X^{(i)}(0)$ itself. So, likelihood can be written to be of the form,

$$\begin{aligned}
L\left(\mathcal{S}|\theta, \bar{\mathcal{G}}\right) &= p\left(X^{(1)}(1), X^{(1)}(2), X^{(1)}(3)|X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(1), X^{(2)}(2), X^{(2)}(3)|X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) \\
&= p\left(X^{(1)}(2), X^{(1)}(3)|X^{(1)}(1), X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(1)}(1)|X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) \\
&\quad \times p\left(X^{(2)}(2), X^{(2)}(3)|X^{(2)}(1), X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(1)|X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) \\
&= p\left(X^{(1)}(3)|X^{(1)}(2), X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(1)}(2)|X^{(1)}(1), X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(1)}(1)|X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) \\
&\quad \times p\left(X^{(2)}(3)|X^{(2)}(2), X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(2)|X^{(2)}(1), X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(1)|X^{(2)}(0), \theta, \bar{\mathcal{G}}\right)
\end{aligned}$$

However, the latter transitions are independent given the starting point, suggesting that arithmetically we are indeed back to (13). So this is technically correct.

In order to see why this is still consistent with the intuition that trajectories should have a greater degree of independence, let us continue rewrite the likelihood:

$$\begin{aligned}
L\left(\mathcal{S}|\theta, \bar{\mathcal{G}}\right) &= p\left(X^{(1)}(3)|X^{(1)}(2), X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(1)}(2)|X^{(1)}(1), X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(1)}(1)|X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) \\
&\quad \times p\left(X^{(2)}(3)|X^{(2)}(2), X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(2)|X^{(2)}(1), X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(1)|X^{(2)}(0), \theta, \bar{\mathcal{G}}\right) \\
&= p\left(X^{(1)}(3)|X^{(1)}(2), \theta, \bar{\mathcal{G}}\right) p\left(X^{(1)}(2), X^{(1)}(1)|X^{(1)}(0), \theta, \bar{\mathcal{G}}\right) \\
&\quad \times p\left(X^{(2)}(3)|X^{(2)}(2), \theta, \bar{\mathcal{G}}\right) p\left(X^{(2)}(2), X^{(2)}(1)|X^{(2)}(0), \theta, \bar{\mathcal{G}}\right)
\end{aligned}$$

Continuing this through, we can see that as $T \to \infty$, the expression becomes

$$p\left(X^{(i)}(T)|X^{(i)}(T-1), \theta, \bar{\mathcal{G}}\right) p\left(X^{(i)}(T-1), \cdots, X^{(i)}(1)|X^{(i)}(0), \theta, \bar{\mathcal{G}}\right)$$

from which we can see the intuition of the circumstance. Asymptotically, the second term approaches the stationary distribution, and the independence assumption becomes valid. Otherwise, we can consider that for T much longer than the mixing time, this assumption is also valid for most of the transitions. However, otherwise we can see that:

1. The larger the measure of the support, and the more distinct the starting points $X^{(i)}(0)$ are from each other, the longer it can take for the stochastic process to mix to erase the information from initial conditions.

2. In finite time, the influence of history will depend on the conductance of the Markovian process defined by $(\theta^{\bar{\mathcal{G}}}, \bar{\mathcal{G}})$, that is,

$$\phi(\bar{\mathcal{G}}) = \min_{S,S' \subset \bar{\mathcal{G}}, |S|, |S'| < |\bar{\mathcal{G}}|/2} \left\{ \frac{A(S, S'; \theta, \bar{\mathcal{G}})}{|S|} \right\} \tag{14}$$

where,

$$A(S, S'; \theta, \bar{\mathcal{G}}) := \frac{\sum_{i \in S} \sum_{j \in S'} p(X_j(t+1)|V_i)}{|S|}$$

where V_i could be any predecessor in the graph for $X_i(t+1)$.

This appears in the previous likelihood as follows: we are actually not learning generic trajectories, but those associated with the history of the trajectory, since we are learning conditional distributions. So, in the previous calculation, under the most unfavorable scenario, $(X^{(1)}(1), X^{(1)}(2), X^{(1)}(3))$ and $(X^{(2)}(1), X^{(2)}(2), X^{(2)}(3))$ would correspond to different regions of state space for X, that is $X^{(1)}(t) \geq C_1 + C_2$ and $X^{(2)}(t) \leq C_1 - C_2$, for some large $C_2 > 0$, and we learning completely independent transitions that don't inform each other, and moreover, with low spatial correlations, the information gained in the marginal is proportional to $X^{(i)}(0)$.

3.4 Sample Complexity for Forecasting

Where the intuition described above arises is in recent results in sample complexity. We shall see that while the arithmetic of (13) is still correct for DBNs, there are indeed important distinctions on the sample complexity with respect to the number of different trajectories N and the length of the trajectory T.

Classically, theoretical analyses of time series sample complexity typically assumed that the trajectory is much longer than the mixing time and by cutting the synthetic burn in period, as such obviates any need to analyze historical dependence. (see the review of the previous results in [64])

We shall report on the theoretical small sample complexity results reported in [64], which is yet unpublished but extends and otherwise mentions similar recent results in [68, 67, 73, 16].

They derive the sample complexity results for learning and identifying a dynamic system,

$$\begin{aligned} X(t+1) &= AX(t) + B\epsilon(t), \\ Y(t+1) &= WX(t) + \xi(t) \end{aligned} \tag{15}$$

which can be seen a simple Hidden Markov Model and $\epsilon(t), \xi(t)$ are i.i.d. normal random variables. With a goal of fitting a *test* trajectory of length T' (that is, not necessarily equal to T), i.e.,

$$L(\hat{f}; T', P_x) := \mathbb{E}_{P_x} \left[\frac{1}{T'} \sum_{t=1}^{T'} \left\| \hat{f}(X(t)) - f_W(X(t)) \right\|^2 \right]$$

with a minimax risk, i.e., minimizing, algorithmically, the maximal risk associated with the worst case population subsample $P_x \in \mathcal{P}_x$. They compute the guarantees associated with the least squares solution, as defined by the specification of (13) to the form given in (15), with a least squares loss, i.e.,

$$\hat{W} \in \arg\min_W \sum_{i=1}^{N} \sum_{t=1}^{T} \left\| WX^{(i)}(t) - Y^{(i)}(t) \right\|^2$$

Finally, they require a *trajectory small-ball* assumption, that can be understood as a uniform bound on the covariance matrices associated with the noise in the sequence.

With this, they present three major results, which are restated here in their informal form.

Theorem 3.1 *[64, Theorem 1.1-3]*

1. *If $N \geq n$, $T' \leq T$, and the trajectories are drawn from a trajectory small ball distribution, then the excess prediction risk over horizon length T' is $\Theta\left(n/(NT)\right)$*
2. *If $N \leq n$, $NT \geq n$ and A is marginally unstable and diagonalizable, then the worst case excess prediction risk over horizon length T' is $\Theta\left(n/(NT)\right) \max\left\{nT'/(NT), 1\right\}$*
3. *If $N \geq n$ and covariate trajectories are such that A is marginally unstable and diagonalizable, then the worst-case excess prediction risk over T' is $\Theta\left(n/(NT) \max\left\{T'/T, 1\right\}\right)$*

From this Theorem, we can consider that with enough samples, standard rates of sample complexity treating the trajectory length T and the number of trajectories N apply. However, for large relative dimension size of the variable space, the complexity does not scale as well, but is similarly proportional. Finally, when attempting to fit longer trajectories T', we finally see that there is greater benefit towards obtaining data samples with long trajectory lengths over sampling more trajectories.

We report on the one prominent result as far as learning so as to achieve accurate inference on BNs. The classic work [15] reports on a sample complexity, in VC dimension analysis, of modeling a Bayesian Network to be,

$$\tilde{O}\left(\frac{n^2}{\epsilon^2}\left(n2^k + \log\frac{1}{\delta}\right)\right)$$

where $\tilde{O}$ suppresses multiplicative terms of $\log(n/\epsilon)$, δ and ϵ define the probability of an inference within a small distance of the true outcome, n is the number of variables, and k is the number of potential parents.

3.5 Sample Complexity for Identification

The sample complexity given above is for a measure of forecasting error, i.e., excess prediction risk formally. As noted in the Introduction, DBNs are used for a number of purposes. This includes not just forecasting, but also identifying a graph structure that is an interpretative model of potential causal relationships between variables.

To the best of our knowledge, there are no sample complexity results on graph and causal discovery identification which take separate consideration of trajectories and time steps in the data. Instead we report on a few general recent results on the overall sample complexity for learning a (D)BN as well as a recent result on causal discovery specifically.

In general, identifying the Bayesian Network is NP-Complete with respect to the number of variables [10]. It is noted that the number of possible DAGs for 10 variables is greater than $4{\times}10^{18}$ [52].

There are some additional sample complexity results worth reporting from the literature.

The work [49] presents poly-time identifiability in the case of bounded treewidth or acyclic super-structure, and otherwise confirms NP-Hardness of search with respect to data. A creative

recent work [25] uses models from physiology to argue for $O\left(M^k\right)$ practical complexity, with M the cardinality of a discrete valued network and k is the number of potential parents.

More favorable results are presented for linear SEMs with a recent algorithm that improves the sample complexity to $O\left(n^2 \log k\right)$ in the case of sub-Gaussian errors and $O\left(n^2 k^{2/m}\right)$ for $4m$-bounded moment errors.

Finally, the work [66] considers sample complexity of causal discovery specifically, which runs at the number of samples required being $O\left(n! l^{3n/8}\right)$, where l is the cardinality of the possible random variable values in a discrete network.

3.6 Formalization: Open System Forecasting Model

In practice, in many cases wherein DBNs are employed for modeling, understanding and forecasting, the underlying system is not completely closed, as in a physics experiment, or deliberately marginalized, as in a randomized clinical trial. Instead, it models a complex and often infinite dimensional system, with intricate and impossible-to-know interactions with the environment. With the presence of unknown confounders, causal sufficiency isn't satisfied. Moreover, it can happen that multiple structures and parameters become equally effective at accurately modeling the process, even highly distinct ones suggesting distinct causal mechanisms.

For instance, DBNs are often used for predictive maintenance, as in [1, 69]. By an appropriate representation of the underlying complex engineering system as distilled into some low dimensional latent structure, one can develop DBNs to monitor signals of deterioration or damage in the system as based on the historical transitions over time in performance.

An interesting formalism of this is given in [21]. For some underlying stochastic process $\dot{Y} = g(Y(t), W(t))$ with (e.g., Brownian) noise $W(t)$ where $Y \in \mathcal{Y}$ is very high, if not infinite, dimensional, one can consider a DBN model as a finite dimensional reduced order model of the system, and one that maximizes the information relevance towards maintenance. Formal guarantees are provided as far as probabilistic invariance, that is,

$$p\left(X(t) \in A, \forall\, t \in [T]\right) \approx p\left(Y(t) \in \tilde{A}, \forall 0 \leq t \leq T\right)$$

indicating the potential for DBNs to serve as useful indicators of higher level properties of stochastic processes, regardless of the fundamental impossibility of formal causal structure identification in such cases.

4 Understanding Structure and Parameter Learning Algorithms

A fundamentally unique feature of learning DBNs corresponds to how structure and weights are treated, both in and of themselves and with respect to each other, as far as modeling and training. Theoretical foundations and best practices developed in the mature disciplines of the statistics of graphical models, random graph theory, time series, causal learning, and others, can provide a diverse source of insight for developing efficient and reliable methodologies.

Here we present a number of important points of consideration that can be observed from looking at the literature at successful attempts at representation as far as inference and learning. With the distinctions described below, we are able to properly identify and characterize existing structure-weight learners, as well as suggest and provide straightforward extensions to fill in the natural empty places in the taxonomy.

Structure Learning Structure Learning is the procedure of defining $\bar{\mathcal{G}}$ from data. This is a critical aspect to learning DBNs because this defines different independence structures between the random variables. Furthermore, these graphical conditional independence structures are interpretable as far as implying causal inference and discovery. It also precedes parameter learning - the space and dimensionality of the parameters in the model itself will vary as depending on the structure of the graph connections. Of course, the quality of the resulting fit on the parameter should inform the quality of the fit of the structure, insofar as it is instrumental.

Given both the rapidly exponentially exploding complexity of considering any encoding of structure, the resulting combinatorial optimization can become difficult to solve with large variable dimension. Structure learning provides a rich source of challenging problems for combinatorics, integer programming, and other discrete applied mathematics. However, at the same time, given the relative paucity of circumstances and means by which the curse of dimensionality can be mitigated, there is a degree to which structure learning serves as a significant limitation to the overall modeling procedure. This means that often, in more challenging settings, approximate suboptimal graph structure, or using alternative modeling techniques, are used.

Parameter Learning Recall from the previous Section that there is often flexibility in the choice of the statistical model that corresponds to individual potential structures. This flexibility permits for incorporating off the shelf methods attuned for specific parametric forms.

There are some structure solvers that define and score a structure without defining parameters. These make use of binary or Gaussian models, as defined above, for which the computation of the marginal posterior is tractable. Specifically, the posterior of the graph structure given the data is computed through an integration that treats parameters as nuisance through an integration $\int p(\{X_i^n\}|\theta)p(\theta|\mathcal{G})d\theta$. In this case, a specific set of parameters is not explicitly defined, however, it can be said that parameters are computed implicitly. Indeed, the marginal likelihood of the structure is simply the integration, over the parameter space, of the posterior distribution for the parameters.

One can note this specific phenomenon regarding the interplay of learning structure and weights as unique to DBNs. Indeed it presents a clear tradeoff between computational ease and model faithfulness. One can also consider whether the structure of parameters are more important and significant as far as the overall modeling of the system of interest, and thus choose more or less complex models, and more or less stringent and exhaustive structure search, depending on this choice.

Frequentist and Bayesian We shall use the frequentist versus Bayesian distinction to indicate a point estimate based on the optimization of a loss function or criterion, and a probabilistic model, implemented with sampling, that obtains a posterior distribution of the structure and weights given the data, respectively. A frequentist estimate is given as a complete specific structure encoding and a specific value for the parameters. It is generally expected, or at least sought, that the relationships that the graph identifies between various is statistically significant. This presumption often becomes unrealistic in practice, and obtaining an appropriately scaled statistically significant entire network, that is, with all significant edges, is typically unavailable. However, since DBNs are generative rather than discriminative, this is often not a practical concern, as they are a component in an overall statistical modeling pipeline.

The alternative of Bayesian approaches allows for modeling the full distribution of uncertainty for the model considering the data. This makes the degree of confidence in the model

quantitatively transparent. Thus, for any regime of data and parameters, some density could be sampled. However, the combinatorial burden of structure learning then becomes transferred to a slow mixing time. Moreover, inference will require numerical integration, and a set of samples is less interpretable to a lay user of the model. Thus, the choice between the two is generally instrumental, that is, in accordance with the ultimate modeling goal.

An effective and commonly used technique is to employ mixtures of a finite set of structures, see [24]. This provides flexibility and the transparent uncertainty in the model, without having to mix through the entire combinatorial space defining possible structure.

Considerations Regarding the Relationship between Structure and Parameter Learning It is clear that the two are not independent or orthogonal, but rather the hierarchical structure, and the discrete-continuous distinction, presents a number of possible choices as far as algorithmic options.

For instance, consider a particular point estimate of a structure and set of parameters. However, consider that the set of parameters is close to zero, and moreover, that is so close so as to include zero in a, e.g., 95% confidence interval. In this case, it is clear that this implies that the presence of this edge itself in the graph is suspect, that is, not implied by the data.

Criteria for structure still depend on the weights, even if it's implicitly through integrating the marginal likelihood. Thus, if the weights have a poorly specified prior, or the parametric form for the model is incorrect, then this will curtail the legitimacy of the structure scoring process.

It would be expected that a structure with a low marginal likelihood should have greater uncertainty in the parameters.

These subtle but intuitive considerations suggest that modeling and learning with DBNs is often not an off-the-shelf straightforward use of a black box tool, but requires intuition as to the nature and mechanistic properties of the system of interest.

Hierarchical and One-Shot Methods In general one can consider most learning methods to be hierarchical in the sense of first learning the structure, and with an amortized structure estimating or sampling the weights. The use of SEMs defined by adjacency matrices including both structure and weights simultaneously introduced what can be referred to as a one-shot approach (we remark the interestingly similar recent popularity of one-shot methods for neural architecture search as including parameter learning [29]).

In this case, a point estimate is obtained for both the structure and the weights simultaneously by solving an appropriate optimization problem that fits both of these as decision variables to the data. To this end there are two approaches we see in the literature. In [45] an IP (for BNs, readily adapted to DBNs) is presented that treats the structure as binary variables encoding the activation of edge links in the graph and the parameters as separate variables, and solves the challenging nonlinear mixed IP (relaxation into conic programs was considered in [37]). Alternatively, the recent work DYNOTEARS [2] presented a gradient based method for solving the structure-parameter learning as a purely continuous optimization problem for weight matrices in the graph. Enforcing sparsity is done to encourage proper structure learning.

This presents a straightforward path to solving an optimization problem using existing toolboxes to obtain a fairly accurate point estimate of the structure and parameters. Methodologically, however, we observe that specifically, there is nothing to prevent encoding a binary variable indicating that an edge is present, and a parameter having a low magnitude to the point of zero

being within the margin of error (or even being exactly zero in the IP case). These are clearly contradictory as far as the meaning of the edge.

We make one additional remark going back to hierarchical approaches. Note that one can consider that the frequentist-Bayesian distinction can be applied to present a taxonomy of methods. As a curious example, many Bayesian scoring methods, e.g. the IP method [3], can be considered hybrid frequentist-Bayesian. This is because implicitly the grading is done with a Bayesian parameter model, but a point estimate, that is one unique structure, is returned. Methodologically, we see that the advantages of a hierarchical is an offline calculation of scoring that permits the use of simple and powerful off the shelf commercial grade IP solvers, and the disadvantage is the conceptual contradiction of applying a frequentist mindset to learning structure with Bayesian models as weights. However, one can easily mitigate this in practice by sampling from multiple structures, as weighted in frequency by their respective marginal likelihoods. Regardless, theoretically, in the asymptotic regime, consistency can still be maintained with all approaches and variations thereof, however [36].

5 Learning, Loss Criteria and Constraint Definitions

Now we will proceed to present some of the analytical expressions associated with learning DBNs. Recall that we assume we have a sample of N trajectories over time horizon T, that is, we restate (12),

$$\mathcal{S} = \cup_{n=1}^{N} \mathcal{T}^n = \{Z^{(n)}, X^{(n)}(0), X^{(n)}(1), X^{(n)}(2), ..., X^{(n)}(T)\}_{n=1,...,N}$$

5.1 Criteria

In order to ascertain the performance of different structures, a score function serves as an objective in an optimization process. The score function is meant to evaluate the statistical accuracy of a model. In performing structure learning as guided by a score, we are performing a likelihood, or some maximum a posteriori maximization, in the process of traversing the decision landscape of structures.

Selection criteria for models appears in both the BN/PGM and the time series modeling literature. In [47] a thorough exploration of the evaluation and computation of various criteria is presentented for a range of different time series models. In [19] it is recommended to use a general form for an information criterion to evaluate possible networks is, for N samples and k parameters:

$$\Delta_{k,N} = -2 \log L_k + C_{k,N} \tag{16}$$

where L_k is the likelihood of the data given the model and parameters and $C_{k,N}$ is a parsimony term with the following forms:

- AIC $C_{k,N} = 2k$
- AICc $C_{k,N} = \frac{N+k}{N-k-2}$
- BIC $C_{k,N} = k \log N$

There are a variety of options as far as how to use this criterion to choose a model. We first present a few that are natural but do not appear consistently in the literature, before continuing to discuss the Bayesian structure learning approach.

We write generically the likelihood $L_k(\{X\}|\Theta, \mathcal{G})$, and write,

1. **One Shot Frequentist** Directly maximize L_k with respect to Θ and Ξ simultaneously for a simultaneous frequentist solution, with $C_{k,N}$ defined as a sparsity metric (i.e., $l0$ "norm")

2. **Hierarchical Frequentist** Maximize $L_k(\{X\}|\Theta(\mathcal{G}), \mathcal{G})$ with respect to $\mathcal{G}$. To evaluate the likelihood given $\mathcal{G}$, one must compute $\Theta(\mathcal{G})$. This itself can be the maximum likelihood of the parameters, i.e.,
$$\Theta(\mathcal{G}) = \arg\max_{\theta} L_k(\{X\}|\theta, \mathcal{G})$$
where conditioning on $\mathcal{G}$ enforces certain components of θ to be zero.

A popular alternative is to use Bayesian criteria. In this case, the actual score function is the marginal posterior of the candidate structure given the data, that is $p(\mathcal{G}|\{X\})$. For particular kinds of parametrized DBNs, computing this posterior can be done in closed form. For a classic discussion on the statistical intuition, motivation, and some formulations of Bayesian criteria, see [31]

5.2 Likelihood Calculations

A common assumption made in the literature [36, 26] is that of global parameter independence. That is, it holds that the parameters $(\theta|\bar{\mathcal{G}})$ can be decomposed to be separable across the transitions for each variable X_i, i.e. using the notation $\theta^{\bar{\mathcal{G}},i}$ to indicate parameters associated with the transition step for variable $X_i(t+1)$,

Assumption 5.1 *It holds that,*

$$p\left(\theta^{\bar{\mathcal{G}}}|\bar{\mathcal{G}}\right) = \prod_{i\in[n]} p\left(\theta^{\bar{\mathcal{G}},i}|\bar{\mathcal{G}}\right)$$

and that, for any data sample $\mathcal{S}$,

$$p\left(\mathcal{S}|\theta_G, \bar{\mathcal{G}}\right) = \prod_{i\in[n]} p\left(\mathcal{S}|\theta^{\bar{\mathcal{G}},i}, \bar{\mathcal{G}}\right)$$

Notice that here the separability is with respect to trajectories, and not necessarily time steps. Furthermore, below we shall see that an additional assumption of local independence is needed to furthermore assure independence across the parameters defining the dependence of the transition of X_i on each parent.

Since the likelihood is a separable function of the parameters, maximizing it corresponds to maximizing the set of parameters separately for each , that is, seek to maximize, where we perform the usual condition dependence chain $p(X(t+1), X(t), ..., X(0)|\theta) = p(X(t+1)|X(t), X(t-1), ..., X(1)|\theta) = p(X(t+1)|X(t), \theta)p(X(t)|X(t-1), \theta), ..., p(X(0))$ to facilitate the presentation of the chain of conditioning to facilitate the posterior derivation.

$$\begin{aligned} p\left(\mathcal{S}|\theta^{\bar{\mathcal{G}},i}, \bar{\mathcal{G}}\right) = \prod_{n=1}^{N}\prod_{s=1}^{T} p\Big(X_i^{(n)}(T-s+1)|Z^{(n)}, \{X_j^{(n)}(T-s)\}_{j\in dpa_d(i)}, \\ \{X_j^{(n)}(T-s+1)\}_{j\in dpa_s(i)}, \{X_i^{(n)}(T-s-\tau)\}_{\tau\in dpa_\tau(i)}, \theta^{\bar{\mathcal{G}},i}, \bar{\mathcal{G}}\Big) \end{aligned} \tag{17}$$

5.2.1 Binary Variables

In the case wherein all variables $\{X_i, Z\}$ are valued $\{0, 1\}$ sampled from a Bernoulli distribution, this presents the simplest calculation, recalling the definition of the transition model.

More significantly, here, we shall see that the more complex representation permits for closed form computation of the marginal posterior of the structure.

To begin with, the simple linear model (5). In this case we write,

$$p(X_i(t+1) = 1; \theta^{\bar{\mathcal{G}},i}) = \sigma\left(\theta_0^{\bar{\mathcal{G}},i} + \sum_{j \in dpa(i)} \theta_j^{\bar{\mathcal{G}},i} V_j\right)$$

From this functional form we can obtain, recalling generically $V_{dpa(i)}$ for any parents, by any of the dependencies, of the variable i.

$$\begin{aligned} p\left(\mathcal{S}|\theta^{\bar{\mathcal{G}},i}, \bar{\mathcal{G}}\right) = \prod_{n=1}^{N} \prod_{t=0}^{T-1} \Big[& \mathbf{1}(X_i^{(n)}(t+1) = 1) P(X_i^{(n)}(t+1) = 1 | V_{dpa(i)}^{(n)}; \theta) \\ & + \mathbf{1}(X_i^{(n)}(t+1) = 0) P(X_i^{(n)}(t+1) = 0 | V_{dpa(i)}^{(n)}; \theta) \Big] \end{aligned} \tag{18}$$

Now we take a logarithm of the expression, turning the products into sums,

$$\begin{aligned} \log\left(p\left(\mathcal{S}|\theta^{\bar{\mathcal{G}},i}, \bar{\mathcal{G}}\right)\right) = \sum_{n=1}^{N} \sum_{t=0}^{T-1} \Bigg[& \mathbf{1}(X_i^{(n)}(t+1) = 1) \Bigg[\theta_0^{\bar{\mathcal{G}},i} + \sum_{j \in dpa(i)} \theta_j^{\bar{\mathcal{G}},i} V_j \\ & - \log\left(1 + \exp\left\{\theta_0^{\bar{\mathcal{G}},i} + \sum_{j \in dpa(i)} \theta_j^{\bar{\mathcal{G}},i} V_j\right\}\right)\Bigg] \\ & - \mathbf{1}(X_i^{(n)}(t+1) = 0) \log\left(1 + \exp\left\{\theta_0^{\bar{\mathcal{G}},i} + \sum_{j \in dpa(i)} \theta_j^{\bar{\mathcal{G}},i} V_j\right\}\right)\Bigg]\Bigg] \end{aligned} \tag{19}$$

In this case, the maximum likelihood cannot be computed in closed form, and numerical methods must be used. The similar situation holds for computing a Bayesian score under this model restiction.

Now we consider the full combinatorial representation as defined by (20), which, with binary outcomes, simplifies to:

$$\begin{aligned} & p(X_i(t+1) = 1 | V_{dpa(i)}, \theta_i) := \theta_i^{\xi(V_{dpa(i)})}, \\ & \xi \in \Xi_i,\ \Xi_i := \Xi_i^d \times \Xi_i^s := \textstyle\prod_{j \in dpa_t(i)} \mathbb{Z}_2^+ \times \prod_{j \in dpa_z(i)} \mathbb{Z}_2^+ \end{aligned} \tag{20}$$

Indeed this corresponds to the local parameter independence and the unrestricted multinomial conditions that facilitates the closed form computation for the Bayesian Dirichlet scores. To this end, we now extend the presentation in [31] (see also [30]) to include the contribution of the static Z variables to the model.

First we begin by writing the full expression for the likelihood and computing the likelihood-maximizing parameter values, making use of the modeling representation in (20).

We introduce one more piece of notation, indicating the set of dynamic variables that contribute in the DAG structure to node i,

$$V_{i,d}^{(n)}(t) = \{X_j^{(n)}(t-1)\}_{j \in dpa_d(i)} \cup \{X_j^{(n)}(t)\}_{j \in dpa_s(i)} \cup \{X_i^{(n)}(t-\tau)\}_{\tau \in dpa_\tau(i)}$$

this distinguishes the dynamic variables from the static ones.

$$
\begin{aligned}
p\left(\mathcal{S}|\theta,\bar{\mathcal{G}}\right) &= \prod_{i\in[n_x]}\prod_{n=1}^{N}\prod_{t=0}^{T-1}\Big[p(X_i(t+1)=1|V_{dpa(i)},\theta_i)X_i^{(n)}(t+1) \\
&\qquad +(1-p(X_i(t+1)=1|V_{dpa(i)},\theta_i))(1-X_i^{(n)}(t+1)\Big] \\
&= \prod_{i\in[n_x]}\prod_{\xi^d\in\Xi_i^d}\prod_{\xi^s\in\Xi_i^s}\prod_{n=1}^{N}\prod_{t=0}^{T-1}\Big[p(X_i(t+1)=1|\xi,\theta_i)X_i^{(n)}(t+1)\mathbf{1}(\xi^d=V_{i,d}^{(n)}(t+1))\mathbf{1}(\xi^s=Z^{(n)}) \\
&\qquad +(1-p(X_i(t+1)=1|V_{dpa(i)},\theta_i))(1-X_i^{(n)}(t+1))\mathbf{1}(\xi^d=V_{i,d}^{(n)}(t+1))\mathbf{1}(\xi^s=Z^{(n)})\Big] \\
&= \prod_{i\in[n_x]}\prod_{\xi^d\in\Xi_i^d}\prod_{\xi^s\in\Xi_i^s}\prod_{n=1}^{N}\prod_{t=0}^{T-1}\Big[\theta_i^{\xi(V_{dpa(i)})}X_i^{(n)}(t+1)\mathbf{1}\left(\xi^d=V_{i,d}^{(n)}(t+1)\right)\mathbf{1}\left(\xi^s=Z^{(n)}\right) \\
&\qquad +\left(1-\theta_i^{\xi(V_{dpa(i)})}\right)\left(1-X_i^{(n)}(t+1)\right)\mathbf{1}\left(\xi^d=V_{i,d}^{(n)}(t+1)\right)\mathbf{1}\left(\xi^s=Z^{(n)}\right)\Big]
\end{aligned}
\tag{21}
$$

Let $\mathbf{N}(A;\mathcal{C})$ be the counting operator of the number of elements of $\mathcal{C}$ that satisfy the condition given by A. Now take the logarithm of the likelihood expression and obtain a sum-separable set of terms for the log likelihood of each parameter, and perform generative learning to find the parameters. Specifically,

$$
\begin{aligned}
&\log p\left(\mathcal{S}|\theta_i^{\xi(V_{dpa(i)})},\bar{\mathcal{G}}\right) \\
&\quad=\mathbf{N}\left(\left[(X_i^{(n)}(t+1)=1)\cap\left(V_{i,d}^{(n)}(t+1)\times Z^{(n)}=\xi(V_{dpa(i)})\right)\right];\mathcal{S}\right)\log\left(\theta_i^{\xi(V_{dpa(i)})}\right) \\
&\qquad+\mathbf{N}\left(\left[(X_i^{(n)}(t+1)=0)\cap(V_{i,d}^{(n)}(t+1)\times Z^{(n)}=\xi(V_{dpa(i)})\right];\mathcal{S}\right)\log\left(1-\theta_i^{\xi(V_{dpa(i)})}\right)
\end{aligned}
$$

From which the natural maximum likelihood estimate can be formed:

$$
\hat{\theta}_i^{\xi(V_{dpa(i)})}=\frac{\mathbf{N}\left(\left[(X_i^{(n)}(t+1)=1)\cap\left(V_{i,d}^{(n)}(t+1)\times Z^{(n)}=\xi(V_{dpa(i)})\right)\right];\mathcal{S}\right)}{\mathbf{N}\left(\left[\left(V_{i,d}^{(n)}(t+1)\times Z^{(n)}=\xi(V_{dpa(i)})\right)\right];\mathcal{S}\right)} \tag{22}
$$

Note that in this case, the counts are over both the samples of trajectories and the time points between them. Observe the role of the static variables Z as simply interacting covariates in the form. Thus, when Z is of a mechanistic form that mediates the transitions, its influence is absorbed as simply an added dimension to the parameter space. We can, however, force a distinction between dynamic and static effects if we assume their causal independence. This would correspond to a kernel transition of the form:

$$
p\left(X_i(t+1)=1|V_{dpa(i)},\theta_i\right):=\theta_i^{\xi^d(V_{dpa_t(i)})}\theta_i^{\xi^s(Z_{dpa_z(i)})} \tag{23}
$$

where $V_{dpa_t(i)}$ denotes the full set of time-dependent variables that influence i. It can be seen that we can obtain the maximum likelihood estimates as,

$$
\begin{aligned}
\hat{\theta}_i^{\xi^d(V_{dpa(i)})}&=\frac{\mathbf{N}\left(\left[(X_i^{(n)}(t+1)=1)\cap\left(V_{i,d}^{(n)}(t+1)=\xi(V_{dpa_t(i)})\right)\right];\mathcal{S}\right)}{\mathbf{N}\left(T\left[\left(V_{i,d}^{(n)}(t+1)=\xi(V_{dpa_t(i)})\right)\right];\mathcal{S}\right)} \\
\hat{\theta}_i^{\xi^s(Z_{dpa_z(i)})}&=\frac{\mathbf{N}\left(\left[(X_i^{(n)}(t+1)=1)\cap\left(Z^{(n)}=\xi(Z_{dpa_z(i)})\right)\right];\mathcal{S}\right)}{\mathbf{N}\left(\left[\left(Z^{(n)}=\xi(Z_{dpa_z(i)})\right)\right];\mathcal{S}\right)}=\frac{\mathbf{N}\left(\left[(X_i^{(n)}(t+1)=1)\cap\left(Z^{(n)}=\xi(Z_{dpa_z(i)})\right)\right];\mathcal{S}\right)}{T\mathbf{N}\left(\left[\left(Z^{(n)}=\xi(Z_{dpa_z(i)})\right)\right];n\in[N]\right)}
\end{aligned}
\tag{24}
$$

From this we can see indeed that with independent causal influence, the estimate for the parameters governing the static nodes Z's influence carries more statistical power, with an effective sample size scaled by T.

Now we present the computation of the Bayesian Dirichlet scores. This amounts to computing the marginal posterior of the structure by performing an integration treating parameter as nuisance. This is derived, for instance, in [26], and used in the popular integer BN structure learner GOBNILP [14]. The marginal posterior of the structure is given by:

$$p\left(\bar{\mathcal{G}}|\mathcal{S}\right) = \int_{\theta} p(\mathcal{S}|\theta^{\bar{\mathcal{G}}}, \bar{\mathcal{G}}) p(\theta^{\bar{\mathcal{G}}}|\bar{\mathcal{G}}) d\theta^{\bar{\mathcal{G}}}$$

In order to compute the BDe, we need a prior on the weights, which we write as a Dirichlet distribution,

$$p(\theta^{\bar{\mathcal{G}}}|\bar{\mathcal{G}}) = \prod_{i\in[n]} \prod_{\xi\in\Xi_i} \frac{\Gamma\left(\alpha_0^{i,\xi} + \alpha_1^{i,\xi}\right)}{\Gamma\left(\alpha_0^{i,\xi}\right) + \Gamma\left(\alpha_1^{i,\xi}\right)} \theta_{i,\xi,0}^{\alpha_0^{i,\xi}} \theta_{i,\xi,1}^{\alpha_1^{i,\xi}}$$

Recalling the expression for (21), we can see that the BDe can be computed by,

$$\begin{aligned}
&p\left(\bar{\mathcal{G}}|\mathcal{S}\right) \\
&= \prod_{i\in[n_x]} \prod_{\xi^d\in\Xi_i^d} \prod_{\xi^s\in\Xi_i^s} \prod_{n=1}^{N} \prod_{t=0}^{T-1} \int_{\theta} \Big[\theta_i^{\xi(V_{dpa(i)})} X_i^{(n)}(t+1) \mathbf{1}\left(\xi^d = V_{i,d}^{(n)}(t+1)\right) \mathbf{1}\left(\xi^s = Z^{(n)}\right) \\
&\qquad + \left(1 - \theta_i^{\xi(V_{dpa(i)})}\right)\left(1 - X_i^{(n)}(t+1)\right) \mathbf{1}\left(\xi^d = V_{i,d}^{(n)}(t+1)\right) \mathbf{1}\left(\xi^s = Z^{(n)}\right)\Big] \\
&\qquad \times \frac{\Gamma\left(\alpha_0^{i,\xi}+\alpha_1^{i,\xi}\right)}{\Gamma\left(\alpha_0^{i,\xi}\right)+\Gamma\left(\alpha_1^{i,\xi}\right)} \theta_{i,\xi}^{\alpha_0^{i,\xi}} \theta_{i,\xi}^{\alpha_1^{i,\xi}} d\theta \\
&= \prod_{i\in[n_x]} \prod_{\xi^d\in\Xi_i^d} \prod_{\xi^s\in\Xi_i^s} \frac{\Gamma\left(\alpha_0^{i,\xi}+\alpha_1^{i,\xi}\right)}{\Gamma\left(\alpha_0^{i,\xi}\right)+\Gamma\left(\alpha_1^{i,\xi}\right)} \times \int_{\theta} \theta_{i,\xi,0}^{\alpha_0^{i,\xi}+\mathbf{N}_{i,\xi}-\mathbf{N}_{i,\xi,1}} \theta_{i,\xi,1}^{\alpha_1^{i,\xi}+\mathbf{N}_{i,\xi,1}} d\theta \\
&= \prod_{i\in[n_x]} \prod_{\xi^d\in\Xi_i^d} \prod_{\xi^s\in\Xi_i^s} \frac{\Gamma\left(\alpha_0^{i,\xi}+\alpha_1^{i,\xi}\right)}{\Gamma\left(\alpha_0^{i,\xi}\right)+\Gamma\left(\alpha_1^{i,\xi}\right)} \times \frac{(\alpha_1^{i,\xi}+\mathbf{N}_{i,\xi,1})}{(\mathbf{N}_{i,\xi}+\alpha_1^{i,\xi}+\alpha_0^{i,\xi})}
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{N}_{i,\xi,1} &= \mathbf{N}\left(\left[(X_i^{(n)}(t+1)=1) \cap \left(V_{i,d}^{(n)}(t+1) \times Z^{(n)} = \xi(V_{dpa(i)})\right)\right]; \mathcal{S}\right), \\
\mathbf{N}_{i,\xi} &= \mathbf{N}\left(\left[V_{i,d}^{(n)}(t+1) \times Z^{(n)} = \xi(V_{dpa(i)})\right]; \mathcal{S}\right)
\end{aligned}$$

and with dynamic-static causal influence independence, the score becomes,

$$\begin{aligned}
p\left(\bar{\mathcal{G}}|\mathcal{S}\right) = &\prod_{i\in[n_x]} \prod_{\xi^d\in\Xi_i^d} \frac{\Gamma\left(\alpha_0^{i,\xi^d}+\alpha_1^{i,\xi^d}\right)}{\Gamma\left(\alpha_0^{i,\xi^d}\right)+\Gamma\left(\alpha_1^{i,\xi^d}\right)} \times \frac{(\alpha_1^{i,\xi^d}+\mathbf{N}_{i,\xi^d,1})}{(\mathbf{N}_{i,\xi^d}+\alpha_1^{i,\xi^d}+\alpha_0^{i,\xi^d})} \\
&\times \prod_{\xi^s\in\Xi_i^s} \frac{\Gamma\left(\alpha_0^{i,\xi^s}+\alpha_1^{i,\xi^s}\right)}{\Gamma\left(\alpha_0^{i,\xi^s}\right)+\Gamma\left(\alpha_1^{i,\xi^s}\right)} \times \frac{(\alpha_1^{i,\xi^s}+\mathbf{N}_{i,\xi^s,1})}{(\mathbf{N}_{i,\xi^s}+\alpha_1^{i,\xi^s}+\alpha_0^{i,\xi^s})}
\end{aligned} \tag{25}$$

Thus, for the DBN case, computing the above amounts to evaluating the BD score. We observe, in addition, that this derivation indicates how one can sample from the posterior distribution of the weights given the structure that a learner identifies as maximizing the desired score. Indeed

the posterior of the weights given the structure is shown above, it is the expression under the integral sign, i.e.,

$$p\left(\theta|\bar{\mathcal{G}},\mathcal{S}\right)=\prod_{i\in[n_x]}\prod_{\xi^d\in\Xi_i^d}\prod_{\xi^s\in\Xi_i^s}\frac{\Gamma\left(\alpha_0^{i,\xi}+\alpha_1^{i,\xi}\right)}{\Gamma\left(\alpha_0^{i,\xi}\right)+\Gamma\left(\alpha_1^{i,\xi}\right)}\theta_{i,\xi,0}^{\alpha_0^{i,\xi}+\mathbf{N}_{i,\xi}-\mathbf{N}_{i,\xi,1}}\theta_{i,\xi,1}^{\alpha_1^{i,\xi}+\mathbf{N}_{i,\xi,1}} \tag{26}$$

5.2.2 Gaussian DBNs

Now we present the derivation of the likelihood and Bayesian criterion (BGe) for DBNs with Gaussian models. The development follows [26] and extends their derivation in two ways. First we perform the recursion for computing the entire trajectory time data. Second, we include a specific parametrization and show how one can simultaneously perform the recursion to obtain a posterior of the weights. We, however, simplify our model to only include Markovian influence, and not lagged autoregressive effects.

We apply the model in [26] to (10) to obtain the following transition likelihood function for the first step and prior for both the overall likelihood transition and the parameters themselves:

$$\begin{aligned}
&p(X_i(1)\cup X_{j\in dpa_d}(0)\cup X_{j\in dpa_s}(1)\cup Z_{j\in dpa_z}|\beta,dpa_d(i)\cup dpa_s(i)\cup dpa_z(i))=\mathcal{N}\left(\mu(0),W\right)\\
&\mu(0)=\begin{pmatrix}\mu_i^x(1;0) & \mu^x_{j\in dpa_d(i)}(0) & \mu^x_{j\in dpa_s(i)}(0) & \mu^z_{j\in dpa_z}(0)\end{pmatrix}^T\\
&\mu^x(0)\in\mathbb{R}^{n_x},\,\mu^z(0)\in\mathbb{R}^{n_z}\\
&\mu_i^x(1;0)\sim\beta^0+\sum_{j\in dpa_d(i)}\beta^d_{i,j}X_j(0)+\sum_{j\in dpa_s(i)}\beta^s_{i,j}X_j(1)+\sum_{j\in dpa_z(i)}\beta^z_{i,j}Z_j\\
&(\beta^0,\beta^d,\beta^s,\beta^z)\sim\mathcal{N}(\eta(0),\psi\Upsilon(0))
\end{aligned} \tag{27}$$

Now consider that the variables have corresponding priors marginal:

$$X(0),Z\sim\mathcal{N}\left((\mu_x(0),\mu_z(0)),\{\Sigma(0),\Sigma_z\}\right) \tag{28}$$

this will be also used to derive the corresponding equivalent posterior analysis. Note that the DAG structure is important for the sensibility of these definitions.

Let us define W. The parameter prior introduces a normal-Wishart distribution on the mean with precision matrix T, dropping the i dependence

$$\begin{aligned}
&p\left(\mu(0)|W,\bar{\mathcal{G}}\right)=\mathcal{N}\left(\nu(0),\alpha_\mu W\right)\\
&\nu(0)=\begin{pmatrix}\mu^+(0):=\eta_0+\eta_d\cdot\mu^x_{j\in dpa_d}(0)+\eta_s\cdot\mu^x_{j\in dpa_s}+\eta_z\cdot\mu^z_{j\in dpa_z(i)}(0)\\ \mu^x_{j\in dpa_d}(0)\\ \mu^x_{j\in dpa_s}(0)\\ \mu^z_{j\in dpa_z(i)}(0)\end{pmatrix}\\
&p\left(W|\bar{\mathcal{G}}\right)=c(n_i(1),\alpha_w)|T|^{\alpha_w/2}|W|^{(a_\alpha-n_i(1)-1)/2}e^{-1/2\,tr(TW)}\equiv\text{Wishart}\left(W|\alpha_w,T\right)\\
&c(n_i(1),\alpha_w):=\left(2^{\alpha_w n/2}\pi^{n(n-1)/4}\prod_{i=1}^{n_x}\Gamma\left(\tfrac{\alpha_w+1-i}{2}\right)\right)^{-1}\\
&n(1)=1+|dpa(i)|\\
&\alpha_\mu W=\\
&\psi\begin{pmatrix}(\Upsilon(0)\,/\,\Upsilon_{\setminus 0}(0))(\Upsilon(0)\,/\,\Upsilon_{\setminus 0}(0))^T & \mu^x_{dpa_d(i)}(0)(\Upsilon(0)\,/\,\Upsilon_{\setminus 0}(0))(\Upsilon(0)\,/\,\Upsilon_{\setminus d}(0))^T & .. \\ .. & .. & .. \\ .. & .. & \mu^z_{dpa_z(i)}(0)(\Upsilon(0)\,/\,\Upsilon_{\setminus 0}(0))(\end{pmatrix}
\end{aligned} \tag{29}$$

where A/B denotes the Schur complement of A with respect to B. In [26, Theorem 4 and Theorem 5] it is shown that parameter independence is preserved through the computation of the posterior. Note that the posterior now is with respect to all the data that is present in a transition. The DAG structure ensures that $\nu(0)$ is well defined as a vector, rather than implicitly as a function of $\mu^x_{j\in dpa_s(i)}(0)$. Finally, the last line related the two models together, indicating how the Wishart distribution arises from the parameter distribution, in this case.

With this, we obtain the joint likelihood expression:

$$\begin{aligned}
&p\left(\mathcal{S}|\beta^{\bar{\mathcal{G}}},\bar{\mathcal{G}}\right) = \prod_{n=1}^{N}\prod_{t=0}^{T-1}\prod_{i\in[n_x]} p\left(X_i^{(n)}(t+1)\cup V^{(n)}_{j\in dpa(i)}(t+1)|\theta^{\bar{\mathcal{G}}},\bar{\mathcal{G}}\right)\\
&= \prod_{i\in[n_x]}\prod_{t=0}^{T-1}\prod_{n=1}^{N} p\left(X_i^{(n)}(t+1)\cup\{X_j^{(n)}(t)\}_{j\in dpa_d(i)},\{X_j(t+1)\}_{j\in dpa(i)},\{Z_j\}_{j\in dpa_z(i)}|\mu_x(0),\mu_z(0),\eta(0),\bar{\mathcal{G}}\right)\\
&:= \prod_{i\in[n_x]}\prod_{t=0}^{T-1}\prod_{n=1}^{N} p\left(\mathcal{S}_i^{(n)}(1)|\mu_x(0),\mu_z(0),\eta(0),\bar{\mathcal{G}}\right)
\end{aligned} \tag{30}$$

for mean μ and nowhere singular covariance matrix W.

Now, with this redundant embedding in both prior in the variable space and parameter space, we deduce how to compute the posterior of the distribution distribution of the data μ and W from [26] for $T = 2$, and subsequently, compute the posterior of the parameters in the model, while showing it is equivalent by a straightforward Bayesian posterior propagation. After deriving the base case $T = 2$, we continue with the induction for T to $T+1$, in order to derive the final posterior of the data, from which we can compute the marginal likelihood of the structure, as well as sample the final posterior values.

Now from the original we know that the likelihood of the data can be given by:

$$\begin{aligned}
&p\left(\mu(1)|W,\mathcal{S}_i^{(n)}(1),\bar{\mathcal{G}}\right)\sim\mathcal{N}(\nu(1),(\alpha_w+N)W(1)),\\
&W(1)\sim\text{Wishart}(\alpha_w+N,R(1))\\
&\nu(1)=\begin{pmatrix}\mu^+(1)\\ \mu^x_{j\in dpa_d(i)}(1)\\ \mu^x_{j\in dpa_s(i)}(1)\\ \mu^z_{j\in dpa_z(i)}(1)\end{pmatrix} := \tfrac{1}{\alpha_\mu+N}\left[\alpha_\mu\nu(0)+N\bar{\nu}(1)\right]\\
&R = T + S_N(1) + \tfrac{\alpha_\mu N}{\alpha_\mu+N}(\nu(0)-\bar{\nu}(1))(\nu(0)-\bar{\nu}(1))^T\\
&\bar{\nu}(1)=\begin{pmatrix}\bar{\nu}(1;0)\\ \bar{\nu}^x_{j\in dpa_d(i)}(0)\\ \bar{\nu}^x_{j\in dpa_s(i)}(0)\\ \bar{\nu}^z_{j\in dpa_z(i)}\end{pmatrix} := \begin{pmatrix}\frac{1}{N}\sum_{n=1}^{N}X_i^{(n)}(1)\\ \frac{1}{N}\sum_{n=1}^{N}X^{(n)}_{j\in dpa_d}(0)\\ \frac{1}{N}\sum_{n=1}^{N}X^{(n)}_{j\in dpa_s(i)}(1)\\ \frac{1}{N}\sum_{n=1}^{N}Z^{(n)}_{j\in dpa_z(i)}\end{pmatrix}
\end{aligned} \tag{31}$$

Now, before we proceed with the next time step, let us define the propagation in the hyper-parameters, that is of the model η,Σ.

$$
\begin{aligned}
&\beta(1) := (\beta^0(1), \beta^d(1), \beta^s(1), \beta^z(1)) \sim \mathcal{N}(\eta(1), \Upsilon(1)) \\
&\eta^0(1) = \eta^0(0) + \tfrac{1}{N}\bar{\eta}^0(1) - \Sigma(0,1)\Sigma^{-1}_{dpa(i)}(0,0)\eta^0_{dpa(i)}(0) \\
&\eta^d(1) = \eta^d(0) + \Sigma^{-1}_{\setminus d}(0,0;1)\Sigma_d(0,1) \\
&\eta^s(1) = \eta^s(0) + \Sigma^{-1}_{\setminus s}(0,0;1)\Sigma_s(0,1) \\
&\eta^z(1) = \eta^z(0) + \Sigma^{-1}_{\setminus z}(0,0;1)\Sigma_z(0,1) \\
&\Upsilon(1) = \psi\Upsilon(0) + \Sigma(1,1) - \Sigma(0,1)\Sigma^{-1}(0,0;1)\Sigma(0,1) \\
&\Sigma(1,1) = \tfrac{1}{N^2}(\nu(0) - \bar{\nu}(1))(\nu(0) - \bar{\nu}(1))^T \\
&\Sigma(0,1) = \tfrac{1}{N^2}\left(\nu(0) - \bar{\nu}(1)\right)\left(\begin{pmatrix} \sum_n \bar{X}_i^{(n)}(1) \\ \sum \bar{V}^{(n)}_{j\in dpa(i)}(1) \end{pmatrix} - \bar{\nu}(1)\right)^T \\
&\Sigma(0,0;1) = \tfrac{1}{N^2}\left(\begin{pmatrix} \sum_n \bar{X}_i^{(n)}(1) \\ \sum \bar{V}^{(n)}_{j\in dpa(i)}(1) \end{pmatrix} - \bar{\nu}(1)\right)\left(\begin{pmatrix} \sum_n \bar{X}_i^{(n)}(1) \\ \sum \bar{V}^{(n)}_{j\in dpa(i)}(1) \end{pmatrix} - \bar{\nu}(1)\right)^T
\end{aligned}
\tag{32}
$$

Recalling that, and $W_{ab}, a, b \in \{0, d, s\}$ the block mean vector and covariance matrix components corresponding to the estimates for $\beta_0, \beta_d, \beta_s$, respectively, can be similarly computed through $\Upsilon(1)$.

We now perform the grand inductive step, to obtain the recursion from $T-1$ to T to be as follows, for the posterior of the data:

$$
\begin{aligned}
&p\left(\mu(T)|W, \mathcal{S}^{(n)}, \bar{\mathcal{G}}\right) \sim \mathcal{N}(\nu(T), (\alpha_w + NT)W(T)), \\
&W(T) \sim \text{Wishart}(\alpha_w + NT, R(T)) \\
&\nu(T) = \begin{pmatrix} \mu^+(T) \\ \mu^x_{j\in dpa_d(i)}(T) \\ \mu^x_{j\in dpa_s(i)}(T) \\ \mu^z_{j\in dpa_z(i)}(T) \end{pmatrix} := \tfrac{1}{\alpha_\mu + N}\left[\alpha_\mu \nu(T-1) + N\bar{\nu}(T)\right] \\
&\quad = \tfrac{1}{\alpha_\mu + N}\left[\alpha_\mu \nu(0) + N\textstyle\sum_{t\in[T]}\bar{\nu}(t)\right] \\
&R(T) = R(T-1) + S_N(T) + \tfrac{\alpha_\mu N}{\alpha_\mu + N}(\nu(T-1) - \bar{\nu}(T))(\nu(T-1) - \bar{\nu}(T))^T \\
&\quad = T + \sum_{t=1}^{T} S_N(t) + \tfrac{\alpha_\mu N}{\alpha_\mu + N}\sum_{t=1}^{T}(\nu(t-1) - \bar{\nu}(t))(\nu(t-1) - \bar{\nu}(t))^T \\
&\bar{\nu}(T) = \begin{pmatrix} \bar{\nu}(T; T-1) \\ \bar{\nu}^x_{j\in dpa_d(i)}(T-1) \\ \bar{\nu}^x_{j\in dpa_s(i)}(T-1) \\ \bar{\nu}^z_{j\in dpa_z(i)} \end{pmatrix} := \begin{pmatrix} \frac{1}{N}\sum_{n=1}^{N} X_i^{(n)}(T) \\ \frac{1}{N}\sum_{n=1}^{N} X^{(n)}_{j\in dpa_d}(T-1) \\ \frac{1}{N}\sum_{n=1}^{N} X^{(n)}_{j\in dpa_s(i)}(T) \\ \frac{1}{N}\sum_{n=1}^{N} Z^{(n)}_{j\in dpa_z(i)} \end{pmatrix}
\end{aligned}
\tag{33}
$$

In general terms, the form is broadly preserved. As such, we can reproduce the evaluation for the marginal likelihood, that is the BGe score, directly from [26]

$$
\begin{aligned}
p\left(\mathcal{S}|\bar{\mathcal{G}}\right) = (2\pi)^{n_x - (\tilde{n}_x + \tilde{n}_z)NT/2}\left(\tfrac{\alpha_\mu}{\alpha_\mu + NT}\right)^{(\tilde{n}_x + \tilde{n}_z)/2} \tfrac{c(1+\tilde{n}_x+\tilde{n}_z, \alpha_w + (1+\tilde{n}_x+\tilde{n}_z))}{c(1+\tilde{n}_x+\tilde{n}_z, \alpha_w - 1 + (\tilde{n}_x+\tilde{n}_z) + NT)} \\
\times |R(T-1)|^{\frac{a_w - n_x + \tilde{n}_x + \tilde{n}_z}{2}} |R(T)|^{-\frac{\alpha_w - n_x + \tilde{n}_x + \tilde{n}_z + NT}{2}}
\end{aligned}
\tag{34}
$$

where $\tilde{n}_x \leq n_x, \tilde{n}_z \leq n_z$ are maximal, or the appropriate weighted average, of the sparsity of dependence on covariates on the transition to $X(t+1)$ (that is, the dimension of $V_{dpa(i)}$).

We can also express the parametric form of the posterior of the distribution of the weights, which also follows along the recursion.

$$
\begin{aligned}
&\beta(T) := (\beta^0(T), \beta^d(T), \beta^s(T), \beta^z(T)) \sim \mathcal{N}(\eta(T), \Upsilon(T)) \\
&\eta^0(T) = \eta^0(0) + \tfrac{1}{N}\bar{\eta}^0(T) - \Sigma(T-1,T)\Sigma^{-1}_{dpa(i)}(T-1,T-1)\eta^0_{dpa(i)}(T-1) \\
&\eta^d(T) = \eta^d(T-1) + \Sigma^{-1}_{\setminus d}(T-1,T-1;T)\Sigma_d(T-1,T) \\
&\eta^s(T) = \eta^s(T-1) + \Sigma^{-1}_{\setminus s}(T-1,T-1;T)\Sigma_s(T-1,T) \\
&\eta^z(T) = \eta^z(T-1) + \Sigma^{-1}_{\setminus z}(T-1,T-1;T)\Sigma_z(T-1,T) \\
&\Upsilon(T) = \Upsilon(T-1) + \Sigma(T,T) - \Sigma(T-1,T)\Sigma^{-1}(T-1,T-1;T)\Sigma(T-1,T) \\
&\Sigma(T,T) = \tfrac{1}{N^2}(\nu(T-1) - \bar{\nu}(T))(\nu(T-1) - \bar{\nu}(T))^T \\
&\Sigma(T-1,T) = \tfrac{1}{N^2}\left(\nu(T-1) - \bar{\nu}(T)\right)\left(\begin{pmatrix}\sum_n \bar{X}_i^{(n)}(T) \\ \sum \bar{V}^{(n)}_{j\in dpa(i)}(T)\end{pmatrix} - \bar{\nu}(T)\right)^T \\
&\Sigma(T-1,T-1;T) = \tfrac{1}{N^2}\left(\begin{pmatrix}\sum_n \bar{X}_i^{(n)}(T) \\ \sum \bar{V}^{(n)}_{j\in dpa(i)}(T)\end{pmatrix} - \bar{\nu}(T)\right)\left(\begin{pmatrix}\sum_n \bar{X}_i^{(n)}(T) \\ \sum \bar{V}^{(n)}_{j\in dpa(i)}(T)\end{pmatrix} - \bar{\nu}(T)\right)^T
\end{aligned}
\tag{35}
$$

This defines the distribution of weights. Let us finally consider the reverse transformation, of obtaining the score from the weights. Indeed, this can be done straightforwardly, as then $W(T)$ can be recovered from $\Upsilon(T)$, and then $\mu(T)$ will have mean $\nu(T)$

5.3 Enforcing Acyclicity

For ensuring that the structure that is learned is a proper Directed Acyclic Graph, there are a number of options as far as formulations for the various optimization problems defining the learning. Below we detail how these are enforced both when the structure is defined by integer decision variables as well as the continuous one shot formulation.

Integer Variables The primary challenge in solving optimization problems on DAGs stems from the exponential size of the acyclicity constraint. A well-known method to ensure acyclicity involves using cycle elimination constraints, which were originally introduced in the context of the Traveling Salesman Problem (TSP) in [11]. Supposing that the set of all cycles is denoted by $\mathcal{C}$, these constraints often take the form

$$
\sum_{(i,j)\in C} e_{i,j} \leq |C| - 1, \quad \forall C \in \mathcal{C}, \tag{36}
$$

where $e_{i,j}$ denote binary decision variables that indicate which edges are present in the directed graph. These constraints may be complemented by different score functions to complete the optimization problem leading to dag recovery. This can then lead to different types of problems, some of which are linear [51, 34, 45], some quadratic [56]. Furthermore, this method of cycle elimination is also typically augmented with a cutting plane method [48, 56].

Another method for acyclicity enforcement is derived from a well-known combinatorial optimization problem called linear ordering (LO) [28]. In the LO problem, we aim to find "the best" permutations, which may be further constrained. In the case a directed acyclic graphs,

these permutations correspond to the placements of edges and since the basis has only quadratic cardinality, the number of constraints is limited. The cycles are then excluded by imposing LO constraints. A perceived drawback of this approach is the neccessity for a quadratic cost function [57, 28].

The third method for eliminating cycles involves enforcing constraints to ensure the nodes adhere to a topological order. A topological order is a linear arrangement of the nodes in a graph such that an arc (j, k) exists only if node j precedes node k in this order. The discrete decision variables, indexed by the node and placement in the topological order determine the graph. It has been reported that in some cases this approach can lead to polynomial time learning [60].

Recently, an alternative approach based on layered networks has been proposed [57]. The concept of layering forbids the placement of arcs between layers in a given direction. The problem of finding a layered graph is defined by the number of layers and the minimal number of layers for a given DAG is unique. This contrasts the topological order method described in the previous paragraph, which can have a possitive influence on the construction of the branch-and-bound tree [57].

One Shot Continuous Formulations Recall that in one shot continuous variable adjacency matrix formulations, the variables denote both the structure (as far as their nonzeros) as well as the sign and magnitude of the weights themselves. Thus it is natural to consider that a constraint in the form of an equality of some function to zero could correspond to ensuring the right zero-nonzero structure of the adjacency matrix to establish acyclicity. On the other hand, considering that this must involve considerations of multiple transitions, potentially extensive matrix multiplication could, and we shall see is, involved.

The algorithm NOTEARS [72] and DYNOTEARS[50] uses the following functional constraint in a continuous optimization algorithm to enforce the DAG structure of the graph,

$$tr \exp\{W \odot W\} - d = tr\left(I + W + \frac{W^2}{2} + \frac{W^3}{3!} + ...\right) - d = 0 \tag{37}$$

which is meant to approximate the following (perhaps more easily enforced) set of constraints,

$$\begin{aligned} &tr(I + W \odot W) - d = 0 \\ &tr(I + W \odot W \odot W) - d = 0 \\ &... \\ &tr(I + W \odot^{n_x} W) - d = 0 \end{aligned} \tag{38}$$

In [70] they introduce a different constraint term that also enforces the DAG constraint, but appears to have better numerical stability, for small $\mu > 0$:

$$tr\left((I + \mu W \odot W)^d\right) - d \tag{39}$$

In the procedure NO BEARS [41] the spectral radius is used to define the presence of a DAG constraint on the adjacency graph. Certain numerical approximations make this relatively feasible, despite the high complexity and nondifferentiability of the spectral radius of a matrix.

Finally, [71] present DAGS with NOCURL, which obviates the need for an explicit functional constraint by solving:

$$(U^*, p^*) = \arg_{U \in \mathbb{S}} \min_{p \in \mathbb{R}^d} f(U \odot ReLU(grad(p)))$$

with $\mathbb{S}$ the space of $d \times d$ skew-symmetric matrices and $grad(p)_{ji} = p_i - p_j$ defines the gradient flow on the nodes of the graph.

6 Methods for Learning Structure and Parameters in DBNs

Now we describe the details of several prominent algorithms that are used to train DBNs. These are not meant to be exhaustive, nor are they even intended to be chosen among the best performing in general. Rather, we hope to present a comprehensive variety, that is, we intend that each broad type of method that is commonly used and studied has a representative among the algorithms chosen. These algorithms use very different techniques, and treat all of the aforementioned considerations regarding learning, that is, the correspondence between structure and weights, and the distinction between points and samples and hierarchical and one shot methods. In addition, approximate (or "local") versus exact (or "global") methods will indicate the tradeoffs associated with seeking the best solution or seeking to find a satisficing statistical model.

6.1 Highlighted Existing Methods

6.1.1 Constraint Based

Under the assumptions of causal sufficiency (no hidden confounders) and faithfulness, classical algorithms developed by Spirtes et al. [62] have been proven to estimate the DAG without exhaustive enumeration of possible structures (which is impossible in interesting cases). The Peter-Clark (PC) algorithm is a method to retrieve the skeleton and directions of the edges, relying on an empirical hypothesis test of Conditional Independence (CI) for each pair of variables given a subset of other variables. It starts from a complete undirected graph and deletes sequentially edges based on these CI relations. PCMCI [55] is adapted to time-series datasets and works for lagged links (causes precede effects). It operates in two stages: 1/ PC testing which identifies a potential set of parents with high probabilities for each variable X_j^t. 2/ using these parents as conditions for the momentary conditional independence (MCI) to address the false positives and test all variable pairs. Statistical tests ParCorr, GPDC, and CMI are used in both steps. PCMCI+ extends PCMCI to include contemporaneous links [54].

This is a good representative of a method that clearly prioritizes structure, and is a statistically principled frequentist technique for identifying said structure. As such there are strong asymptotic theoretical results for this method, and it is broadly accepted to be reliable as far as identifying the ground truth. As any method prioritizing structure, however, the necessity of focusing exclusively on a discrete procedure limits the scalability of this approach.

6.1.2 Score Based:

There are a number of methods that attempt to either optimize to obtain or sample from a high score of a Bayesian Criterion. We include a few of these methods due to their significant difference as far as the method of optimization/sampling.

Integer Programming The Integer Programming based [3] uses the local score (BDeu, BGe, DiscreteLL, DiscreteBIC, DiscreteAIC, GaussianLL, GaussianBIC, GaussianAIC, GaussianL0)

to optimize the network, amortizing its evaluation, thus obviating the need to compute parameters to compute the score. This algorithm was later relased as GOBNILP [14] (Globally Optimal Bayesian Network learning using Integer Linear Programming). GOBNILP finds the network with the highest BDeu score under the constraint that the underlying structure can be represented as a DAG. For every node in the graph v and every possible parent set W, binary variable $I(W \to v)$ is created. The optimization criterion is then sum over all possible vertices and all possible parent sets, where the BDeu score for the selected parent set of every node is considered, i.e.,

$$\sum_{v} \sum_{W} I(W \to v) \cdot BDeu(v, W). \tag{40}$$

The constraints are then of two types. First, each vertex needs to have only a single parent set, which for node v formulates as

$$\sum_{W} I(W \to v) = 1. \tag{41}$$

The second constraint requires that there are no cycles in the graph. This is imposed by *cluster constraints*, which require that there must be 1 node with no parents for any set of nodes. As there are exponentially many such sets of nodes, the optimization problem is solved, and if a cycle is in the final solution, the cluster constraint that prohibits the found cycle is added. Such computation is iterated until a DAG is found, which also ensured that the optimal model is found.

This algorithm represents the curious "frequentist-Bayesian" approach to structure-parameter learning. As it is an IP based method, there are also practical limitations in regards to scaling, however, the method is broadly known to be reliable and, for its search space, efficient.

GFlowNets GFlowNet (GFN) for structure learning [17] consists of approximating the posterior instead of finding a single DAG, to reduce uncertainty over models. They construct the sample DAG from the posterior as a sequential decision problem by starting from an empty graph and adding one edge at a time. The GFN environment is similar to Reinforcement Learning where the states are different graphs, each associated with a reward which is the score of that structure. They define a terminal state s_f to which every connected state is called complete. The actions taken are edge adding (no edge reversal or removal). In addition, they define a mask that prevents having cycles in the graphs. GFN's goal is to model the whole distribution proportional to the rewards. It also borrows from Markov chain literature, using forward and backward transition probabilities, $P_\theta(s'/s)$ and $P_B(s/s')$ in the loss function that satisfies the detailed balance condition:

$$\mathcal{L}(\theta) = \sum_{s \to s'} \left[log \frac{R(s')P_B(s/s')P_B(s_f/s)}{R(s)P_\theta(s'/s)P_\theta(s_f/s')} \right]^2 \tag{42}$$

where $R(s)$ is the reward function of state s. Extending GFN to DBN required changing the scoring function BDe and BGe adequately and changing the mask used before to also take into account the stationarity assumption (transitions are invariant in time) and to be a block upper triangular matrix (no edges going from time slice $t+1$ to t).

This has been recently extended in [18] for sampling the structure and weights simultaneously using recent developments of expanding the GFN environment to continuous variables [39].

Monte Carlo Greedy Hill Search Monte Carlo methods are classical for solving difficult statistical problems, and have been a popular choice for learning the structure and parameters of a DBN. There are two prominent Monte Carlo methods in the literature that developed the foundations and have been seminal in the development of structure learning algorithms. These include the work (1128 citations as of this writing) [24] as well as (2344) [63], who developed the popular MMHC, a max-min hill climb (MMHC) procedure.

In the numerical experiments, we use MMHC from the package bnstruct [23].

MCMC We use [38], a more recent development. It uses order based structure sampling and at the same restricts the search space using conditional independence tests. Performance of the method is generally the strongest performer for difficult problems.

6.1.3 One Shot Linear SEMs:

There are two prominent procedures that represent one shot learning of LSEMs. The two are based on integer and continuous based optimization. LSEMs indeed uniquely presents the opportunity for continuous optimization methods, and as such presents the possibility of scaling the estimation procedure, at the cost of theoretical guarantees of global convergence.

Integer Programming The mixed integer-linear program defined in [45] is presented here:

$$\begin{aligned}
\min_{(E_W,E_A,,W,A)} \quad & \mathbf{E}(E_W, E_A, W, A) + \lambda_W \|E_W\|_0 + \lambda_A \|E_A\|_0 \\
& := \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{i=1}^{d} \left([X_{m,t}]_i - \sum_{j=1}^{d} W_{j,i} [X_{m,t}]_j \right. \\
& \left. - \sum_{l=1}^{\max\{p,t\}} \sum_{j=1}^{n} A_{l,j,i} [X_{m,t-l}]_j \right)^2 + \lambda_W \sum_{i,j} [E_W]_{i,j} + \lambda_A \sum_{l,i,j} [E_A]_{l,i,j} \\
\text{s.t.} \quad & W \cdot (1 - E_W) = 0, \\
& A \cdot (1 - E_A) = 0, \\
& DAG(E_W), \\
& (E_W, E_A) \in \left[\{0,1\}^{d^2} \right] \times \left[\{0,1\}^{d^2} \right]^p \\
& W \in \mathbb{R}^{d \times d},\ A \in \mathbb{R}^{p \times d \times d}
\end{aligned} \tag{43}$$

We can see that a linear model is fit with a standard least squares loss to the data. The constraints appear, in order, as enforcing that an absent structure, defined by the binary variable $[E_W]_{i,j} = 0$, corresponds to a zero weight, that is $[W]_{i,j}$, and similarly for A. Next, we enforce a DAG constraint on the integer variables. This was described above in the previous section. Finally, the binary and continuous variables are indicated.

Continuous Optimization We begin by presenting the general algorithm introduced in [50] which followed the well cited [72]. In this paper, they consider the transition dynamics of $X(t)$ can be expressed using the SEM:

$$X_{t+1} = X_t W + \sum_{\tau=1}^{\tau_M} X_{t-\tau} A_\tau + \sigma_t \tag{44}$$

which includes the transition encoding W, whose sparsity pattern reflects the patterns of causation and magnitudes the linear regression coefficients in the transition. A_i are autoregressive matrices in case of lagged effects. Here σ_t is the noise (note that in the original, this is denoted as Z_t, which we avoid for confusion).

They solve the optimization problem,

$$\begin{aligned} \min_{W,A} \quad & \tfrac{1}{2n} \sum_{t,i} \|X_t^i - X_t^i W + \sum_{\tau=1}^{\tau_M} X_{t-\tau}^i A_\tau\|^2 + \lambda_W \|W\|_1 + \lambda_A \|A\|_1 \\ \text{subject to} \quad & Tr\left[\exp\left(W \circ W\right)\right] - d = 0 \end{aligned} \tag{45}$$

wherein the nonlinear constraint function is based on the description in Section 5.3, in particular, see the motivation by (37).

This method is able to impressively identify the ground truth structure for many synthetic examples, while also performing well as far as predictive modeling and forecasting of real world datasets.

6.2 Novel Modifications of Existing Methods

In developing the work for this paper, a few natural developments of existing algorithms, that wouldn't be worthwhile to appear independently, arose. We present each of these methods and describe them

One Shot Structure-Parameter Consistent Frequentist We propose a modified variant of (46). In this case, we introduce positive and negative weights, and require a lower bound for the weights. Thus, if the edge is active, the weights are forced to be bounded away from zero. This is based on two motivations:

1. In principle, a structure being correctly identified should correspond to the weights associated with any active edge to be nonzero. Thus a search for the structure fitting the data well should be expected to have weights that would reject a null hypothesis of zero.

2. In the literature on sparsity ($\|\cdot\|_0$) constrained optimization, e.g. [4], it can be seen that a necessary (but not sufficient) condition for optimality is an L-stationarity condition that implies that, effectively, the indices $\mathcal{I}(\theta^*) = supp(\theta^*)$ are such that $\theta_i, i \in \mathcal{I}(\theta^*)$ must be bounded away from zero a distance corresponding the Lipschitz constant of the gradient and the gradient vector components corresponding to the components of θ^* that are zero.

$$
\begin{aligned}
\min_{\substack{E_{W^+}, E_{W^-}, E_{A^+}, E_{A^-} \\ W^+, W^-, A^+, A^-}} \quad & \mathbf{E}((E_{W^+}, E_{W^-}, E_{A^+}, E_{A^-}, W^+, W^-, A^+, A^-)) \\
& + \lambda_{W^+} \|E_{W^+}\|_0 + \lambda_{W^-} \|E_{W^-}\|_0 + \lambda_{A^+} \|E_{A^+}\|_0 + \lambda_{A^-} \|E_{A^-}\|_0 \\
& := \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{i=1}^{d} \left([X_{m,t}]_i - \sum_{j=1}^{d} [E_{W^+}]_{l,j,i} W^+_{j,i} [X_{m,t}]_j - \sum_{j=1}^{d} [E_{W^-}]_{l,j,i} W^-_{j,i} [X_{m,t}]_j \right. \\
& \left. - \sum_{l=1}^{\min\{p,t\}} \sum_{j=1}^{n} [E_{A^+}]_{l,j,i} A^+_{l,j,i} [X_{m,t-l}]_j - \sum_{l=1}^{\min\{p,t\}} \sum_{j=1}^{n} [E_{A^-}]_{l,j,i} A^-_{l,j,i} [X_{m,t-l}]_j \right)^2 \\
& + \lambda_{W^+} \sum_{i,j} [E_{W^+}]_{i,j} + \lambda_{W^-} \sum_{i,j} [E_{W^-}]_{i,j} \\
& + \lambda_{A^+} \sum_{l,i,j} [E_{A^+}]_{l,i,j} + \lambda_{A^-} \sum_{l,i,j} [E_{A^-}]_{l,i,j} \\
\text{s.t.} \quad & W^+ \geq b_W,\ W^- \leq -b_W \\
& A^+ \geq b_A,\ A^- \leq -b_A \\
& E_{W^+} + E_{W^-} \leq 1,\ E_{A^+} + E_{A^-} \leq 1 \\
& DAG\left(E_{W^+} + E_{W^-}\right), \\
& E_{W^+}, E_{W^-} \in \left[\{0,1\}^{d^2}\right] \\
& E_{A^+}, E_{A^-} \in \left[\{0,1\}^{d^2}\right]^p \\
& W^+, W^- \in \mathbb{R}^{d \times d},\ A^+, A^- \in \mathbb{R}^{p \times d \times d}
\end{aligned}
\tag{46}
$$

where $b_W, b_A > 0$ are lower bounds on the magnitude of these weights.

7 Numerical Results

Note: this is a work on progress, and the numerical results reported here are exploratory

The synthetic datasets were generated following causalLens [40]. We set the maximum lag to 1 and the graph complexity to 30, corresponding to complex causal graphs. The algorithms used are Dynotears with hyperparameters *lambda_w* and *lambda_a* equal to 0.05, and a small *w_threshold* of 0.01. For structure identification in tables 4 and 5, we only compare the structure of the algorithms, so the binary adjacency matrix (rather than the weighted one) is taken from Dynotears. For GOBNILP, the algorithm only supports IID data. To use it, we run it twice on the data in the first time slice to get the prior network, then on the two first slices to get the transition network. For the MCMC, we use iterative MCMC followed by order MCMC from BiDAG package, to sample the MAP DAG. We set *alpha* to , *alphainit* to 0.01 and change *hardlimit*, limit on the size of parent sets , according to the number of variables per experiment. For PCMCI+, we choose an *pc_alpha* of 0.01 and use ParCorr as the conditional independence test (which assumes univariate, continuous variables with linear dependencies and Gaussian noise). We further correct the p-values by False Discovery Rate control with an *alpha_level* of 0.01. And finally we use Max Min Hill Climbing algorithm from bnstruct package using the BIC score. The parameters of the *one shot frequentist ILP approach* (see (46)) were $b_W = b_A = 0.1$. The regularization parameters were $\lambda_{A^+} = \lambda_{A^-} = \lambda_{W^+} = \lambda_{W^-} = 0.05$.

Structure of Numerical Comparisons We can consider three main purposes for which DBNs may be used for, and so we perform tests comparing the learners for these three criteria in an appropriate manner. In addition, we report on the time of execution, and present results across the scale of small and medium covariate dimension problems.

1. **Generative Accuracy** A DBN is a generative model, meaning there are no labels, however, it is still meant to model the relationship between random variables. Thus a natural comparison as to the overall statistical quality of a model would be the classic train-test data split comparison of loss. That is, using a holdout validation set from the data, perform the learning to define a DBN model on the training data, and then perform a set of inference queries on this model, and compare their output to the ground truth output given by the validation set.

2. **Ground Truth Graph Identification** One of the primary goals of using BN and DBN models for fitting various time-varying phenomena is causal discovery and causal inference. This amounts to being able to accurately reconstruct the graph from a noisy realization of the ground truth. Indeed under the causal identifiability assumption given above, the relative success by which a learner is able to compute this ground truth graph is, understandably so, a central for evaluating DBN learners in the literature.

Data Regimes : *Favorable Regime for Identification*: This corresponds to $NT \gg n$, in which case, the more generally well-developed methods are able to identify the ground truth graph. We shall take:

$$(n, N, T) \in \{(3, 30, 10), (5, 50, 50), (10, 100, 200)\} \tag{47}$$

High Dimensional Regime: In this case, causal identification will not be available because the number of trajectories and time steps is insufficient to specify the exact graph that generated the data. However, we can still attempt to train DBN models that fit the data appropriately.

$$(n, N, T) \in \{(3, 5, 10), (5, 10, 20), (10, 20, 40), (20, 40, 50), (30, 60, 100)\} \tag{48}$$

7.1 Model Validation Accuracy

For validation of the accuracy, we split the time series so that the first 70 % are used for training, and the remaining 30 % are used for testing. Then, we use the `dbnR` package to evaluate the log-likelihood of the train data given the predicted model. Results are presented in Tables 2 and 3.

Table 2: Log-likelihood for the favorable regime. TL indicates a setting that did not finish within the time limit, and E indicates a setting that ended in an error.

	(3,30,10)	(5,50,50)	(10,100,200)
GFN	-3.872058	144.9295	TL
Dynotears	-7.249293	-68.31561	-1911.473
Gobnilp	-8.655779	-37.15421	-1474.687
MCMC	-6.983337	-71.15038	148.7103
PCMCI+	-9.333284	-144.704	-1986.396
MMHC	-7.827745	-61.64009	TL
One Shot F. ILP	-1.554613	31.7567	TL

Table 3: Log-likelihood for the high dimensional regime. TL indicates a setting that did not finish within the time limit, and E indicates a setting that ended in an error.

	(3,5,10)	(5,10,20)	(10,20,40)	(20,40,50)	(30,60,100)
GFN	-Inf	192.0897	TL	TL	TL
Dynotears	23.58121	35.43635	126.4152	-876.4649	-2071.534
Gobnilp	24.81038	44.50767	229.5804	TL	TL
MCMC	32.12857	46.05138	208.4295	-457.6892	135.5351
PCMCI+	23.6628	13.21597	36.11931	-1021.048	-2506.6
MMHC	E	E	E	E	E
One Shot F. ILP	29.60606	75.58703	TL	TL	TL

7.2 Structure Identification

To evaluate the qualitative measures of the predicted structure, we compared the predictions with the ground truth adjacency matrix. The comparison was made using the *structural Hamming distance*, which is informally the number of edges that need to be either removed from or added to the predicted structural graph. The second measure is the AUROC, a standard metric that measures the area under the receiver operator characteristic. The results can be found in Tables 4 and 5.

Table 4: Expected SHD and AUROC for favorable dimensional regime for identification. TL indicates a setting that did not finish within the time limit, and E indicates a setting that ended in an error.

	(3,30,10)		(5,50,50)		(10,100,200)	
	SHD	AUROC	SHD	AUROC	SHD	AUROC
GFN	49.0	0.766	1030.0	0.871	16432.0	0.787
Dynotears	13.0	0.658	447.0	0.608	5578.0	0.583
Gobnilp	25.0	0.493	655.0	0.548	5018.0	0.562
MCMC	31.0	0.486	378.0	0.555	4321.0	0.687
PCMCI+	19.0	0.5	242.0	0.610	5016.0	0.541
MMHC	20.0	0.650	516.0	0.552	TL	TL
One Shot F. ILP	46.0	0.770	620.0	0.824	TL	TL

Table 5: Expected SHD and AUROC for high dimensional regime. TL indicates a setting that did not finish within the time limit, and E indicates a setting that ended in an error.

	(3,5,10)		(5,10,20)		(10,20,40)		(20,40,50)		(30,6(
	SHD	AUROC	SHD	AUROC	SHD	AUROC	SHD	AUROC	SHD
GFN	57.212	0.797	442.0	0.728	3636.0	0.850	19228.0	0.863	90471
Dynotears	13.0	0.658	174.0	0.603	1126.0	0.558	2863.0	0.525	9384.0
Gobnilp	33.0	0.634	175.0	0.712	1022.0	0.663	TL	TL	TL
MCMC	43.0	0.647	228.0	0.545	904.0	0.664	2753.0	0.615	9801.0
PCMCI+	19.0	0.5	121.0	0.5	822.0	0.519	2408.0	0.508	7639.0
MMHC	E	E	E	E	E	E	E	E	E
One Shot F. ILP	44.0	0.470	322.0	0.692	TL	TL	TL	TL	TL

8 Discussion and Conclusion

We hope this paper has provided a useful guide to the main principles behind learning the structure and parameters of a DBN. We focused on the fundamentals for the most simple cases, while targeting breadth in the scope of the various methodological approaches to learning these models from data.

There is an important aspect to DBNs that we did not discuss, as for the simple cases of learning it can be considered an orthogonal topic. This would be inference. DBNs are a generative model, so by themselves they do not accomplish any particular statistical decision test. However, one can perform various inference inqueries, such as the probability an instance of $X(2)$ with $X(1) = 3.2$ and $Z = 3$ be greater than 2.1. One natural one for DBNs is a forward time forecast. Causal inference can also be performed through queries DBN models. Inference and approximate inference have a number of different procedures available, as far as efficiently and effectively sampling from the network.

Furthermore, inference algorithms are required in order to further extend DBN modeling to many real world datasets. For one, they become necessary for the expectation step in an Expectation-Maximization algorithm to learn structure with hidden variables. Often, with sys-

tems wherein the mechanism of action isn't observed, a latent variable structure is able to model a rough set of possible dependencies that fits the observed data directed to and from in the graph.

For larger dimensions, IP approaches become computationally infeasible. In such a circumstance, given the Sample Complexity discussed in Section 3.

When data is plentiful, that is, millions and possibly easily available streaming samples, then neural network approaches can be effective. This suggests, for instance, the potential scalability of Generative Flow Networks [2] for instance. Reinforcement Learning is another common approach [74]. Otherwise, in the high-dimensional regime, wherein samples are finite but there are many covariates, Bayesian methods [6] or meta-heuristics are typically applied [33].

Acknowledgements

The authors would like to thank and Ondřej Kuželka for his suggestions and discussion on this work. This work has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101084642.

References

[1] Md Tanjin Amin, Faisal Khan, and Syed Imtiaz. Fault detection and pathway analysis using a dynamic bayesian network. *Chemical Engineering Science*, 195:777–790, 2019.

[2] Lazar Atanackovic, Alexander Tong, Bo Wang, Leo J Lee, Yoshua Bengio, and Jason S Hartford. Dyngfn: Towards bayesian inference of gene regulatory networks with gflownets. *Advances in Neural Information Processing Systems*, 36, 2024.

[3] Mark Bartlett and James Cussens. Integer linear programming for the bayesian network structure learning problem. *Artificial Intelligence*, 244:258–271, 2017. Combining Constraint Solving with Mining and Learning.

[4] Amir Beck and Nadav Hallak. On the minimization over sparse symmetric sets: projections, optimality conditions, and algorithms. *Mathematics of Operations Research*, 41(1):196–223, 2016.

[5] Sander Beckers. Causal Sufficiency and Actual Causation. *Journal of Philosophical Logic*, 50(6):1341–1374, December 2021.

[6] Eva Besada-Portas, Sergey M Plis, Jesus M de la Cruz, and Terran Lane. Parallel subspace sampling for particle filtering in dynamic bayesian networks. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2009, Bled, Slovenia, September 7-11, 2009, Proceedings, Part I 20*, pages 131–146. Springer, 2009.

[7] Natasha K Bowen and Shenyang Guo. *Structural equation modeling*. Oxford University Press, 2011.

[8] Marcos L.P. Bueno, Arjen Hommersom, Peter J. Lucas, Gerald Anne Martijn Lappenschaar, and Joost Janzing. Understanding disease processes by partitioned dynamic bayesian networks. *Journal of Biomedical Informatics*, 61, 05 2016.

[9] Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11141–11149, 2024.

[10] David Maxwell Chickering. Learning bayesian networks is np-complete. *Learning from data: Artificial intelligence and statistics V*, pages 121–130, 1996.

[11] Vašek Chvátal, William Cook, George Dantzig, Delbert Fulkerson, and Selmer Johnson. *Solution of a Large-Scale Traveling-Salesman Problem*, volume 2, pages 7–28. 11 2010.

[12] David Collett. *Modelling Survival Data in Medical Research*. 05 2023.

[13] R.G. Cowell, Alexander Dawid, Steffen Lauritzen, and David Spiegelhalter. *Probabilistic Networks and Expert Systems*, volume 43. 01 2001.

[14] James Cussens. Gobnilp: Learning bayesian network structure with integer programming. In *International Conference on Probabilistic Graphical Models*, pages 605–608. PMLR, 2020.

[15] Sanjoy Dasgupta. The sample complexity of learning fixed-structure bayesian networks. *Machine Learning*, 29:165–180, 1997.

[16] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, 20(4):633–679, 2020.

[17] Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 518–528. PMLR, 01–05 Aug 2022.

[18] Tristan Deleu, Mizu Nishikawa-Toomey, Jithendaraa Subramanian, Nikolay Malkin, Laurent Charlin, and Yoshua Bengio. Joint bayesian inference of graphical structure and parameters with a single generative flow network. *Advances in Neural Information Processing Systems*, 36, 2024.

[19] Randal Douc, Eric Moulines, and David Stoffer. *Nonlinear time series: Theory, methods and applications with R examples*. CRC press, 2014.

[20] Seif Eldawlatly, Yang Zhou, Rong Jin, and Karim Oweiss. Reconstructing functional neuronal circuits using dynamic bayesian networks. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 2008:5531–4, 02 2008.

[21] Sadegh Esmaeil Zadeh Soudjani, Alessandro Abate, and Rupak Majumdar. Dynamic bayesian networks as formal abstractions of structured stochastic processes. In *26th International Conference on Concurrency Theory (CONCUR 2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

[22] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

[23] Alberto Franzin, Francesco Sambo, and Barbara Di Camillo. bnstruct: an R package for Bayesian Network structure learning in the presence of missing data. *Bioinformatics*, 33(8):1250–1252, 12 2016.

[24] Nir Friedman and Daphne Koller. Being bayesian about network structure. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 201–210, 2000.

[25] Salih Geduk and İlkay Ulusoy. A practical analysis of sample complexity for structure learning of discrete dynamic bayesian networks. *Optimization*, 71(10):2935–2962, 2022.

[26] Dan Geiger and David Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. *The Annals of Statistics*, 30(5):1412–1440, 2002.

[27] Victor Gomez Comendador, Álvaro Sanz, Rosa Valdés, and Javier Pérez Castán. Characterization and prediction of the airport operational saturation. *Journal of Air Transport Management*, 69:147–172, 06 2018.

[28] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. On the acyclic subgraph polytope. *Mathematical Programming*, 33:28–42, 09 1985.

[29] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 544–560. Springer, 2020.

[30] David Heckerman. A tutorial on learning with bayesian networks. *Innovations in Bayesian networks: Theory and applications*, pages 33–82, 2008.

[31] David Heckerman, Dan Geiger, and David M Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20:197–243, 1995.

[32] Patrik Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. *Advances in neural information processing systems*, 21, 2008.

[33] Jinqiu Hu, Laibin Zhang, Lin Ma, and Wei Liang. An integrated safety prognosis model for complex system based on dynamic bayesian network and ant colony algorithm. *Expert Systems with Applications*, 38(3):1431–1446, 2011.

[34] Tommi Jaakkola, David Sontag, Amir Globerson, and Marina Meila. Learning bayesian network structure using lp relaxations. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 358–365. JMLR Workshop and Conference Proceedings, 2010.

[35] Sunyong Kim, Seiya Imoto, and Satoru Miyano. Dynamic bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. *Biosystems*, 75(1-3):57–65, 2004.

[36] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[37] Simge Kucukyavuz, Ali Shojaie, Hasan Manzour, Linchuan Wei, and Hao-Hsiang Wu. Consistent second-order conic integer programming for learning bayesian networks. *Journal of Machine Learning Research*, 24(322):1–38, 2023.

[38] Jack Kuipers, Polina Suter, and Giusi Moffa. Efficient sampling and structure learning of bayesian networks. *Journal of Computational and Graphical Statistics*, 31(3):639–650, 2022.

[39] Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-Garcıa, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks. In *International Conference on Machine Learning*, pages 18269–18300. PMLR, 2023.

[40] Andrew Lawrence, Marcus Kaiser, Rui Sampaio, and Maksim Sipos. Data generating process to evaluate causal discovery techniques for time series data, 04 2021.

[41] Hao-Chih Lee, Matteo Danieletto, Riccardo Miotto, Sarah T Cherng, and Joel T Dudley. Scaling structural learning with no-bears to infer causal transcriptome networks. In *Pacific Symposium on Biocomputing 2020*, pages 391–402. World Scientific, 2019.

[42] Sik-Yum Lee and Hong-Tu Zhu. Statistical analysis of nonlinear structural equation models with continuous and polytomous data. *British Journal of Mathematical and Statistical Psychology*, 53(2):209–232, 2000.

[43] Shiqing Ling, Michael McAleer, and Howell Tong. Frontiers in time series and financial econometrics: An overview. *Journal of Econometrics*, 189, 03 2015.

[44] Yue Liu, Yijing Wang, Guihuan Zheng, Jue Wang, and Kun Guo. The dynamical relationship between capital market and macroeconomy: based on dynamic bayesian network. *Procedia Computer Science*, 162:46–52, 2019. 7th International Conference on Information Technology and Quantitative Management (ITQM 2019): Information technology and quantitative management based on Artificial Intelligence.

[45] Hasan Manzour, Simge Küçükyavuz, Hao-Hsiang Wu, and Ali Shojaie. Integer programming for learning directed acyclic graphs from continuous data. *INFORMS journal on optimization*, 3(1):46–73, 2021.

[46] Bryan Matthews, Santanu Das, Kanishka Bhaduri, Kamalika Das, Rodney Martin, and Nikunj Oza. Discovering anomalous aviation safety events using scalable data mining algorithms. *Journal of Aerospace Information Systems*, 10:467–475, 10 2013.

[47] Allan DR McQuarrie and Chih-Ling Tsai. *Regression and time series model selection*. World Scientific, 1998.

[48] George Nemhauser, Martin Savelsbergh, and Gabriele Sigismondi. Constraint classification for mixed integer programming formulations. *IEEE Transactions on Software Engineering - TSE*, 20, 01 1991.

[49] Sebastian Ordyniak and Stefan Szeider. Parameterized complexity results for exact bayesian network structure learning. *Journal of Artificial Intelligence Research*, 46:263–302, 2013.

[50] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, pages 1595–1605. Pmlr, 2020.

[51] Young Woong Park and Diego Klabjan. Bayesian network learning via topological order. *Journal of Machine Learning Research*, 18:1–32, 10 2017.

[52] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

[53] Jonas Peters, Joris M Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. 2014.

[54] Jakob Runge. Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets, 2022.

[55] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances*, 5(11):eaau4996, 2019.

[56] Pavel Rytíř, Aleš Wodecki, and Jakub Mareček. Exdag: Exact learning of dags, 2024.

[57] Ronald Seoh. Solving bayesian network structure learning problem with integer linear programming. *arXiv preprint arXiv:2007.02829*, 2020.

[58] Charupriya Sharma and Peter van Beek. Scalable bayesian network structure learning with splines. In *International Conference on Probabilistic Graphical Models*, pages 181–192. PMLR, 2022.

[59] Pedro Shiguihara, Alneu De Andrade Lopes, and David Mauricio. Dynamic bayesian network modeling, learning, and inference: a survey. *IEEE Access*, 9:117639–117648, 2021.

[60] Ali Shojaie and George Michailidis. Penalized likelihood methods for estimation of sparse high dimensional directed acyclic graphs. *Biometrika*, 97:519–538, 09 2010.

[61] David J Spiegelhalter and Steffen L Lauritzen. Sequential updating of conditional probabilities on directed graphical structures. *Networks*, 20(5):579–605, 1990.

[62] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. The MIT Press, 2000.

[63] Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65:31–78, 2006.

[64] Stephen Tu, Roy Frostig, and Mahdi Soltanolkotabi. Learning from many trajectories. *arXiv preprint arXiv:2203.17193*, 2022.

[65] Rosa Valdés, Victor Gomez Comendador, Álvaro Sanz, Eduardo Ayra, Javier Pérez Castán, and Luis Sanz. *Bayesian Networks for Decision-Making and Causal Analysis under Uncertainty in Aviation*. 11 2018.

[66] Samir Wadhwa and Roy Dong. On the sample complexity of causal discovery and the value of domain expertise. *arXiv preprint arXiv:2102.03274*, 2021.

[67] Lei Xin, George Chiu, and Shreyas Sundaram. Learning the dynamics of autonomous linear systems from multiple trajectories. In *2022 American Control Conference (ACC)*, pages 3955–3960. IEEE, 2022.

[68] Yu Xing, Benjamin Gravell, Xingkang He, Karl Henrik Johansson, and Tyler H Summers. Identification of linear systems with multiplicative noise from multiple trajectory data. *Automatica*, 144:110486, 2022.

[69] Jie Yu and Mudassir M Rashid. A novel dynamic bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis. *AIChE Journal*, 59(7):2348–2365, 2013.

[70] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. Dag-gnn: Dag structure learning with graph neural networks. In *International Conference on Machine Learning*, pages 7154–7163. PMLR, 2019.

[71] Yue Yu, Tian Gao, Naiyu Yin, and Qiang Ji. Dags with no curl: An efficient dag structure learning approach. In *International Conference on Machine Learning*, pages 12156–12166. Pmlr, 2021.

[72] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

[73] Yang Zheng and Na Li. Non-asymptotic identification of linear dynamical systems using multiple trajectories. *IEEE Control Systems Letters*, 5(5):1693–1698, 2020.

[74] Zuowu Zheng, Chao Wang, Xiaofeng Gao, and Guihai Chen. Rbnets: A reinforcement learning approach for learning bayesian network structure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 193–208. Springer, 2023.

[75] Piotr Ładyżyński, Maria Molik, and Piotr Foltynski. Dynamic bayesian networks for prediction of health status and treatment effect in patients with chronic lymphocytic leukemia. *Scientific Reports*, 12:1811, 02 2022.

ExDAG: Exact learning of DAGs

Anonymous Author
Anonymous Institution

Abstract

There has been a growing interest in causal learning in recent years. Commonly used representations of causal structures, including Bayesian networks and structural equation models (SEM), take the form of directed acyclic graphs (DAGs). We provide a novel mixed-integer quadratic programming formulation and an associated algorithm that identifies DAGs on up to 25 vertices with high precision, where identifiability conditions hold. We call this method ExDAG, which stands for exact learning of DAGs. The exact learning is guaranteed by the global convergence of the branch-and-bound-and-cut algorithm, which is utilized. Although there is a super-exponential number of constraints that prevent the formation of cycles, the algorithm adds constraints violated by solutions found, rather than imposing all constraints in each continuous-valued relaxation. Our empirical results show that ExDAG outperforms state-of-the-art solvers in terms of precision when considering Gaussian noise on medium-sized graphs.

1 Introduction

Learning of causal representations and causal inference has received significant attention recently (Peters et al., 2017; Schölkopf et al., 2021; Ahuja et al., 2023, e.g.). With the aim of tackling a variety of challenges in a wide range of applications, many models and methodologies (Pearl, 2009; Park et al., 2023; Buchholz et al., 2024; Lorch et al., 2024; Yu et al., 2021; Zhang et al., 2024; Chen et al., 2021, e.g.) have been introduced. In machine-learning literature, probabilistic graphical models (Koller and Friedman, 2009), in general, and Bayesian networks, in particular, are often used. In statistics and biomedical applications, structural equation models (Yuan and Bentler, 2006; Duncan, 2014) and additive noise models (Peters et al., 2017) are very popular. All can be seen as learning of (edge-weighted) directed acyclic graphs (DAGs).

Learning of DAGs, where the vertices correspond to the random variables and the oriented edges represent their dependencies, underlies the learning of both Bayesian networks and structural equation models, where algebraic manipulations can be interpreted as interventions on the causal system (Bottou et al., 2013). The identification of such a structure is usually mediated by a score function, whose minimization identifies a class of graphs. Alternatively, one may employ selective model averaging (Madigan and Raftery, 1994).

In the present article, we focus on the learning of a DAG using a polynomial score function under the assumption of identifiability, which is given by persistent excitation (Willems et al., 2005, Section 2), or equivalently, rank of the Henkel matrix (Willems et al., 2005, Theorem 1). Depending on the construction of the score function (Heckerman, 2022), score-optimal DAG maximizes likelihood for Gaussian and non-Gaussian noise. Our main contribution is not restricted to a particular noise distribution, though (see Section 3).

The paper is structured as follows. Section 2 contains a brief overview of the state-of-the-art, which is most relevant to the presented work along with motivation that lead to the development of ExDAG. The following two sections (Sections 3 and 4) detail the model used for identification, identifiability conditions and a description of the algorithmic treatment, which follows from the formulation of the problem as a mixed integer quadratic optimization. Lastly, Section 5 contains a detailed benchmarks that contextualize the performance of ExDAG with respect to the state-of-the-art.

1.1 Main Contributions

Our contributions within the learning of a DAG, such as in the learning of a Bayesian network, comprise the

Preliminary work. Under review by AISTATS 2024. Do not distribute.

following.

- We consider the identification of static Directed Acyclic Graph (DAGs) that leads to a mixed-integer quadratic problem, which is a natural choice since the form of the cost function guarantees that the global solution of the problem is a maximum likelihood estimator (see Section 3).
- We propose ExDAG, a branch-and-bound-and-cut algorithm for solving the formulation, which avoids the use of exponentially many constraints at the root node and avoids exponential-time preprocessing steps, while making use of mixed-integer quadratic programming techniques, which lead to a guarantee of global convergence.
- We perform a head-to-head comparison of ExDAG with state-of-the-art solvers of Andrews et al. (2024); Zheng et al. (2018); Waxman et al. (2024), which have appeared in major venues. The comparison shows favorable precision figures for graphs up to 50 vertices.

2 Related Work

Focusing on the identification of Bayesian networks representing causal inference, we may delineate between two major strategies that lead to identification. The first one of these focuses on global optimality, which, due to the presence of cycle-exclusion constraints, leads to an exponential runtime (Cussens, 2011; Bartlett and Cussens, 2017; Cussens et al., 2017; Studený et al., 2021; Kitson et al., 2023, e.g.). This naturally leads to scaling issues, which are typically overcome by imposing additional assumptions such as a maximal degree or certain structural properties of the graph (e.g., existence of a decomposition from structural graph theory, cf. Hliněný et al. (2008)). This greatly limits the applicability of global methods in practice as many problems of interest do not conform to these assumptions.

Recent locally convergent methods (Zheng et al., 2018; Andrews et al., 2024; Waxman et al., 2024) address the aforementioned shortcoming by formulating the problem in continuous optimization. Cycle exclusion is then enforced by means of a continuous function of the adjacency matrix, which allows for scaling to hundreds of vertices. This scaling, however, is bought at a price as the convergence to global optimum is no longer guaranteed (due to non-convex constraints) and results may vary based on problem set. For certain problem sets, these solvers enjoy reasonable precision, for others not so much. The aforementioned sensitivity to instance selection is further documented in Section 5.

In this paper, we present a novel method, which comes with global optimality guarantees, while still allowing for scaling. The basis for the algorithmic solution is the structural equation model (SEM) described in the following section.

3 Problem Formulation and Identification Results

In general, the problem of score-based Bayesian network learning can be seen the identification of a SEM (Hoover and Demiralp, 2003; Kilian, 2011). Assuming

- linearity,
- no backwards in time dependence between the d random variables (autoregressive order zero) and
- n available samples per random variable,

the model reads

$$X = XW + Z, \tag{1}$$

where $W \in \mathbb{R}^{d,d}$ is the weighted adjacency matrix of the network, $Z \in \mathbb{R}^{n,d}$ is the additive noise vector, $X \in \mathbb{R}^{n,d}$ is the data matrix in which each column corresponds to a random variable.

Now a cost function, which is the maximum likelihood estimator for sufficiently small regularization and a wide array of noise types is defined as

$$J(W) = \|X - XW\|_F^2 + \lambda \|W\|, \tag{2}$$

where $\lambda > 0$ is a sufficiently small regularization coefficient, $\|\cdot\|$ denotes an arbitrary norm, which is usually chosen to be the $l1$-norm and $\|\cdot\|_F$ denotes denotes the Frobenius norm. The problem of score based DAG learning can then be cast as

$$\begin{aligned} &\min_W J(W), \\ &G(W) \in \Gamma_{DAG}. \end{aligned} \tag{3}$$

Under a variety of identifiability assumptions, it has been shown that the solution of (3) recovers a DAG from a given equivalence class with high probability in the static case under Gaussian (van de Geer and Bühlmann, 2012; Aragam et al., 2017) and non-Gaussian noise vectors (Shimizu et al., 2006; Loh and Bühlmann, 2013). We refer to Willems et al. (2005); Ahuja et al. (2023) further discussion on identifiability.

4 Identification as a Mixed Integer Quadratic Problem

Let us present an intentionally simplistic formulation for the identification of static DAGs. The construction is such that the cycle-exclusion constraints can be

added using a callback at runtime, which is key when scaling to larger instances. Another key feature is that this callback makes use of a simple separation routine that has only quadratic complexity in the number of graph vertices in the worst case.

Suppose that the description of the directed weighted graph is given by the following set of variables

$e_{i,j} \in \{0,1\}$ is decision variable that is 1 if and only if there exists an edge from vertex i to j $(i \neq j)$

$w_{i,j} \in \mathbb{R}$ is the decision variable that represents the weight of edge $e_{i,j}$.

The objective function utilizes a penalized l_p norm:

$$J_p = \sum_{i=1}^{n} \sum_{j=1}^{d} \left| X_{i,j} - \sum_{k} X_{i,k} w_{k,j} \right|^p + \lambda \sum_{e \in E} e, \quad (4)$$

which avoids the use of a bilinear term, while being equivalent as long as we utilize the additional constraints

$$w_{k,j} \leq c e_{k,j}, \quad w_{k,j} \geq -c e_{k,j} \text{ for all } k, j \in \{1, 2, \ldots, d\}, \quad (5)$$

where $c > 0$ denotes a constant that corresponds to the biggest weight magnitude allowed. The regularization constant $\lambda > 0$ in (4) is discussed in Section 5. The exponent $p \in \mathbb{N}$ is $p = 1, 2$. Although the objective (4) gives rise to a mixed-integer linear formulation when $p = 1$, the global minimum would no longer represent the maximum likelihood estimator. (Cf. Section 3.) For this reason, we set $p = 2$ and deal with a mixed integer quadratic problem in the following sections.

Finally, let C be the set of cycles of a graph on d vertices, where each cycle $c \in C$ of length k is described by a set of edges, i.e., $c = \{(i_1, i_2), (i_2, i_3), \ldots, (i_{k-1}, i_1)\}$. A constraint excluding one cycle $c \in C$ from a solution in terms of e reads

$$\sum_{(i,j) \in c} e_{i,j} \leq k - 1. \quad (6)$$

A key challenge is the number of cycles, and thus the number of constraints (6).

4.1 The Branch-and-Bound-and-Cut Algorithm

A key contribution of ours is a branch-and-bound-and-cut algorithm for solving the formulation above. We utilize the usual branch-and-bound algorithm (Achterberg, 2007, e.g.), but implement cycle exclusion (6) using so-called "lazy" constraints. Lazy constraints are only checked when an integer-feasible solution candidate has been identified. When a lazy constraint is violated, it is included across all nodes of the branch-and-bound tree. In summary, at the root node, we utilize only $O(|E|)$ constraints (5). Subsequently, one introduces cycle-exclusion constraints (6), but numerical practice has shown that the number of cycle constraints needed is far lower than the "full" super-exponential amount.

Notice that once a new mixed-integer feasible solution candidate is found, it is easy to detect cycles therein using depth-first search (DFS). If a cycle is found, we add the corresponding lazy constraint (6). The DFS algorithm has a worst-case quadratic runtime in the number of vertices of the graph, in constrast to algorithms separating related inequalities from a continuous-valued relaxation (Borndörfer et al., 2020; Cook et al., 2011), such as the quadratic program in our case. In particular, we have tried three variants of the addition of lazy constraints:

1. Adding only the lazy constraint for the first cycle found.
2. Adding only the lazy constraint for the shortest cycle found.
3. Adding multiple lazy constrains for all cycles found in the current integer-feasible solution candidate.

We use Variant 3 throughout our numerical experiments, despite going contrary to the received wisdom (Achterberg, 2007, Chapter 8.9) suggesting that one needs to add only a subset of cuts and utilize a carefully crafted selection criterion to identify "good" cuts.

5 Numerical Results

In the first comparison, detailed in Section 5.2, ExDAG is compared to the state of the art solvers NOTEARS, DAGMA and BOSS (Andrews et al., 2024; Zheng et al., 2018; Waxman et al., 2024). Experiments under the assumption of Gaussian noise are performed for different generation methods and average edge degrees. Second, it is shown, how the convergence to the global minimum allows us to further improve the results of one of the experiments from the preceding section by granting more computational time. Lastly, we show a simple application to a dataset, which was featured at last years NeurIPS competition.

5.1 Setup Common to all the Benchmarking Experiments and Comparison Metrics

We have implemented a branch-and-bound-and-cut algorithm utilizing Gurobi Optimizer 11, which has been

configured to use the simplex algorithm and to expect lazy constraints (lazyConstraints = 1). These parameter settings are important for three reasons. The simplex algorithm produces corner points of the polyhedra given by (5) and any of the lazy constraints. Corner points of the continuous-valued relaxation can be cut off by the constraints (6), in contrast to points in the interior of the optimal face, which would be obtained by a barrier solver (Gondzio, 2012). Second, when Gurobi expects lazy constraints, it avoids pruning the branch-and-bound-and-cut tree prematurely, which would have impacted the global convergence properties (Sahinidis and Grossmann, 1991) otherwise. Third, lazy constraints are added directly to the LP relaxation, without going through the cut filtering process (Achterberg, 2007, Chapter 8.9). The Python source code is provided in the Supplementary Material and will be open-sourced upon acceptance. In the following, we refer to the implementation as ExDAG.

In each of the benchmark experiments, we consider an initial graph, represented by a weighed adjacency matrix W_{init}, which is to be learnt. Next, inputs are generated from W_{init} under Gaussian noise as in Zheng et al. (2018). Lastly, the inputs are used to estimate the structure of the DAG using the relevant method, where we denote the adjacency matrices generated by a method $\cdot$ by $W_{\cdot}$. The structure of the output adjacency matrix often captures spurious relationships, which can result in an edge with a negligible weight in the solution $W_{\cdot}$ (Zhou, 2009; Wang et al., 2016). This effect is negated by setting a near-zero threshold parameter $\delta > 0$, using which we eliminate the edges of weight smaller than δ from $W_{\cdot}$. To level the playing field between solvers, we select the optimal parameter $\delta > 0$, whenever a ground truth is available and compare the resulting matrices w.r.t. to this optimal parameter selection.

Next, the metrics that are used to assess the quality of the identification are defined. Let V and W be two adjacency matrices (one can think of one being the ground truth W_{init} and the other being the identified adjacency matrix $W_{\cdot}$). Then define the structural Hamming distance (SHD) as

$$\rho(V, W) = \sum_{i,j=1}^{d} r_{ij}(V, W), \tag{7}$$

where

$$r_{ij}(V, W) = \begin{cases} 0 & \text{if } V_{ij} \neq 0 \neq W_{ij} \text{ or } V_{ij} = 0 = W_{ij} \\ \frac{1}{2} & \text{if } V_{ij} \neq 0 \text{ and } W_{ji} \neq 0 \\ 1 & \text{otherwise.} \end{cases} \tag{8}$$

SHD is used as a score describing the similarity of the two DAGs in terms of edge placement and is commonly used to assess the quality of solutions (Zheng et al., 2018; Cussens, 2011; Pamfil et al., 2020).

In addition to SHD, the F1 score is used to evaluate the results, it reads

$$F_1 = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}}, \tag{9}$$

where

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}}, \tag{10}$$

$$\text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}}. \tag{11}$$

Note that the dependence of the quantities on V and W in (10) and (11) is suppressed.

We use two well-known ensembles of random graphs: the Erdős–Rényi model (ER) of Erdős et al. (1960) and the scale-free network model (SF) of Barabási and Albert (1999). In particular, the SF and ER generators used in Zheng et al. (2018) were utilized to match the experiments of Zheng et al. (2018) closely. As is often the case in the causal learning community, we report the average and worst case (min for F1 and max for SHD) for a sample statistic for 10 seeds. The worst-case scenarios then trivially give a bound on the sample variance. All the experiments in the following section were performed on a computing cluster with AMD EPYC 7543 cpus. We set a limit of 32 GB RAM and two cores per task.

5.2 Comparison of Identification Methods Under Gaussian Noise

We consider graphs with an average degree between 2 and 5 with data skewed by Gaussian noise. Note that several different regularizations were chosen in the experiments presented. These needed to be chosen per sample count n since the number of samples affects the ratio of the loss minimizing and regularizing part of the cost function (4). These were made using the following considerations. The first step was a selection of a wide enough set of hyper-parameters $\lambda \in \{4, 3, 2, 1, 0.1, 0.05, 0.01\}$. Then, a multitude of experiments with a short time limit (15 minutes) were performed in which the rate of the decrease of the dual gap was observed. The ones with the fastest rates of decrease were chosen for the comparison in which the computation time limit was set to 2 hours. The aforementioned strategy may be replicated for cases in which the ground truth is not known.

The F1 score and SHD of the identified graphs for the generation methods ER2, SF2, SF3, and SF4 are depicted in Figures 5.2, 5.2, 5.2, and 5.2, respectively.

Summarizing the results, one can see that ExDAG is able to hold its own in the lower sample regimes, while soundly outperforming the other solvers in the high sample regime, across the two generation methods and three edge densities considered.

5.3 Finding the Optimal Running Time

The global convergence of the solver, guaranteed by the use of the use of the branch-and-bound-and-cut (B&C) algorithm (Mitchell, 2002), needs to also be studied numerically. The progress towards global optimality is monotone, but may stall at times. The information about this progress suggests optimal running times for different problems. In Figure 5.3, we show that the first plateau was reached relatively quickly in one of the experiments of the previous section. Thus, a time limit of 2000 seconds would have sufficed to arrive at the SHD reported.

5.4 Identifying DAGs from Datasets with Real Interpretation using ExDAG

To test the capabilities of ExDAG further, we use it to learn a DAG from `alarm.csv`, the only publicly available dataset from a competition held at NeurIPS 2023[1]. ExDAG obtains a best SHD of 55 with a Gscore 0.6258 with $\lambda = 0.5$, which improves upon NOTEARS substantially, where NOTEARS identifies DAG with the best SHD score of 65 over 100 different seeds, with the best Gscore of 0.5578. Notice that the identifiability in this case is not well understood. Indeed, depending on the spectral properties of the system, a sufficient number of samples may or may not be sufficient (Simchowitz et al., 2018) for identifiability. Furthermore, the maximum likelihood estimator is not well understood, when the data points are from $\{0, 1\}$ and the range of the noise is also $\{0, 1\}$.

6 Conclusion

A novel, cycle-based formulation for identifying static Bayesian networks based on the structural vector autoregressive model was proposed. This formulation leads to a mixed-integer quadratic program (MIQP), whose solution is the maximum likelihood estimator for a variety of noise distributions. The cycle-based formulation of the problem allows us to add valid cycle-exclusion constraints only upon violation. Although the separation of cycle-based inequalities from continuous-valued relaxations is NP-Hard in some settings (Borndörfer et al., 2020), and only heuristics are known (Cook et al., 2011; Vo et al., 2023) in other

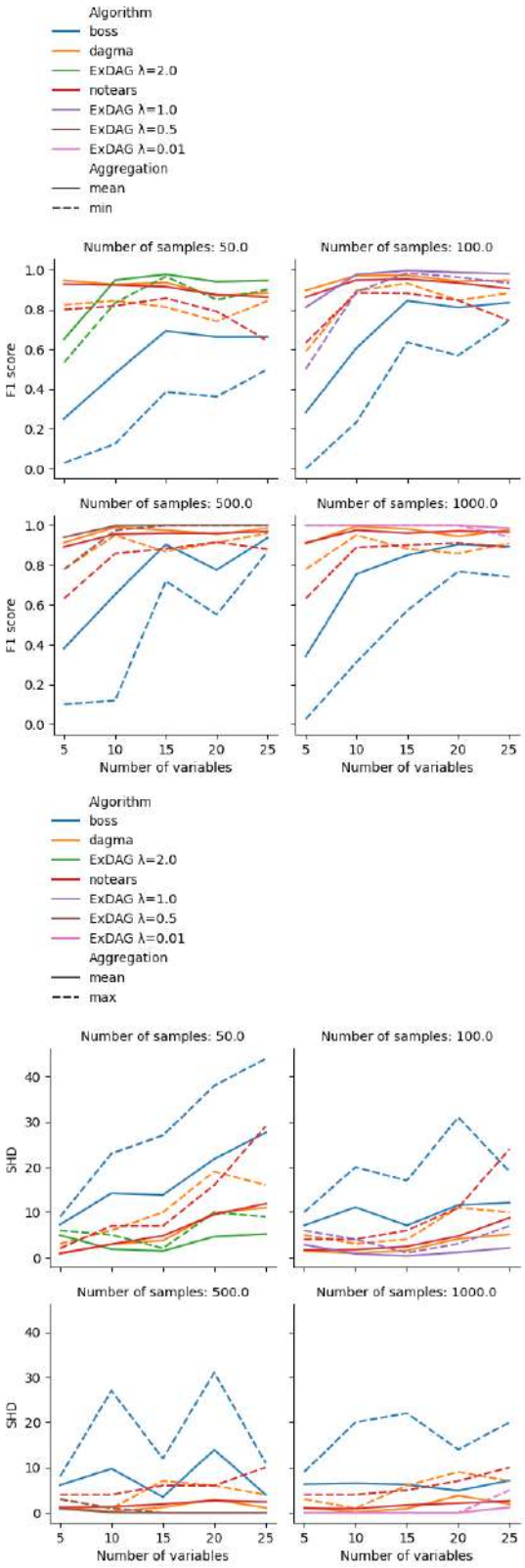


Figure 1: SHD and F1 score for a test case using ER2 random ensemble; mean and maximum across 10 runs.

[1]Cf. `https://codalab.lisn.upsaclay.fr/forums/13855/2071/`

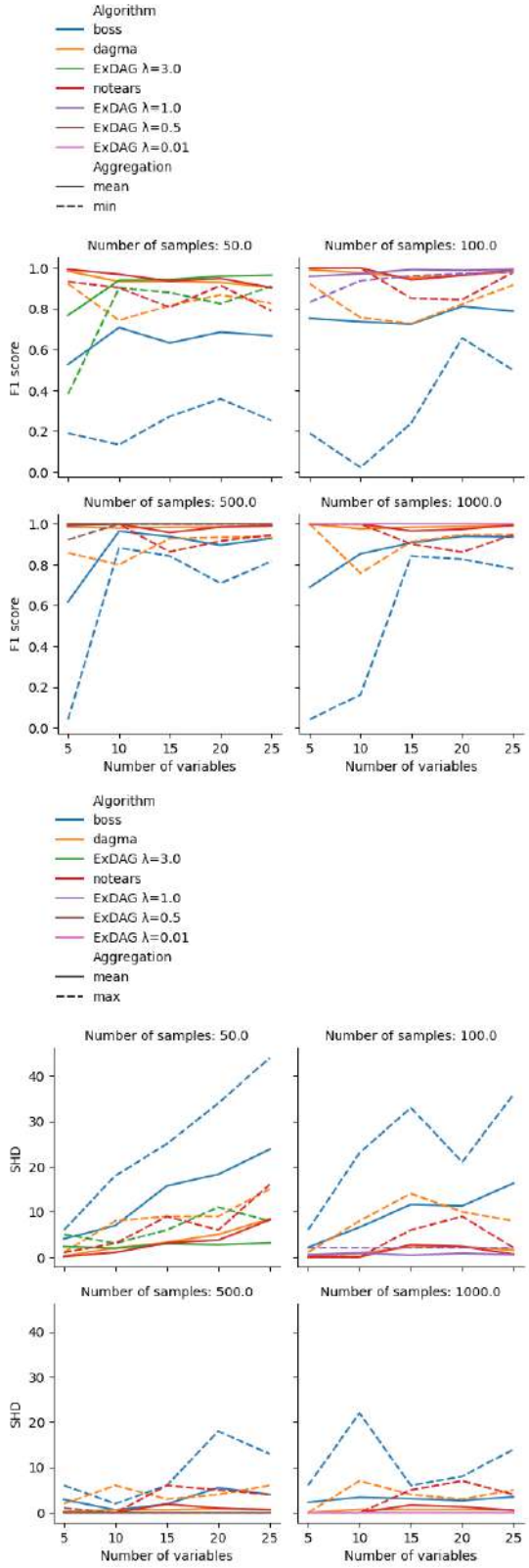


Figure 2: SHD and F1 score for a test case using SF2 random ensemble; mean and maximum across 10 runs.

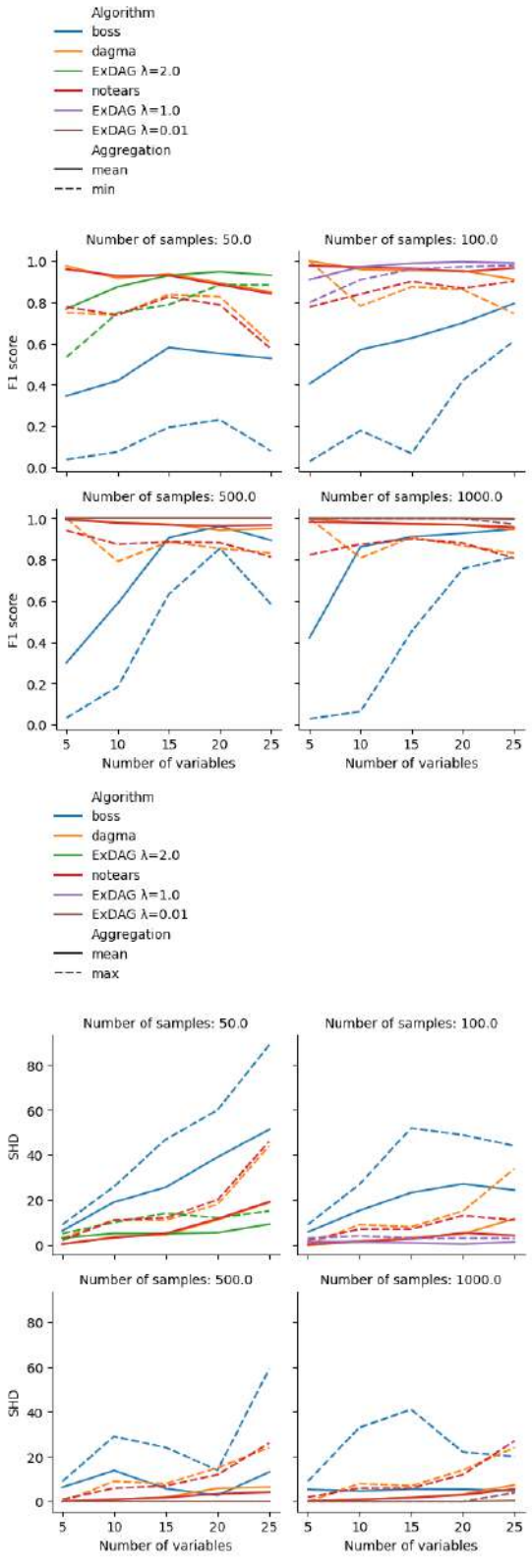


Figure 3: SHD and F1 score for a test case using SF3 random ensemble; mean and maximum across 10 runs.

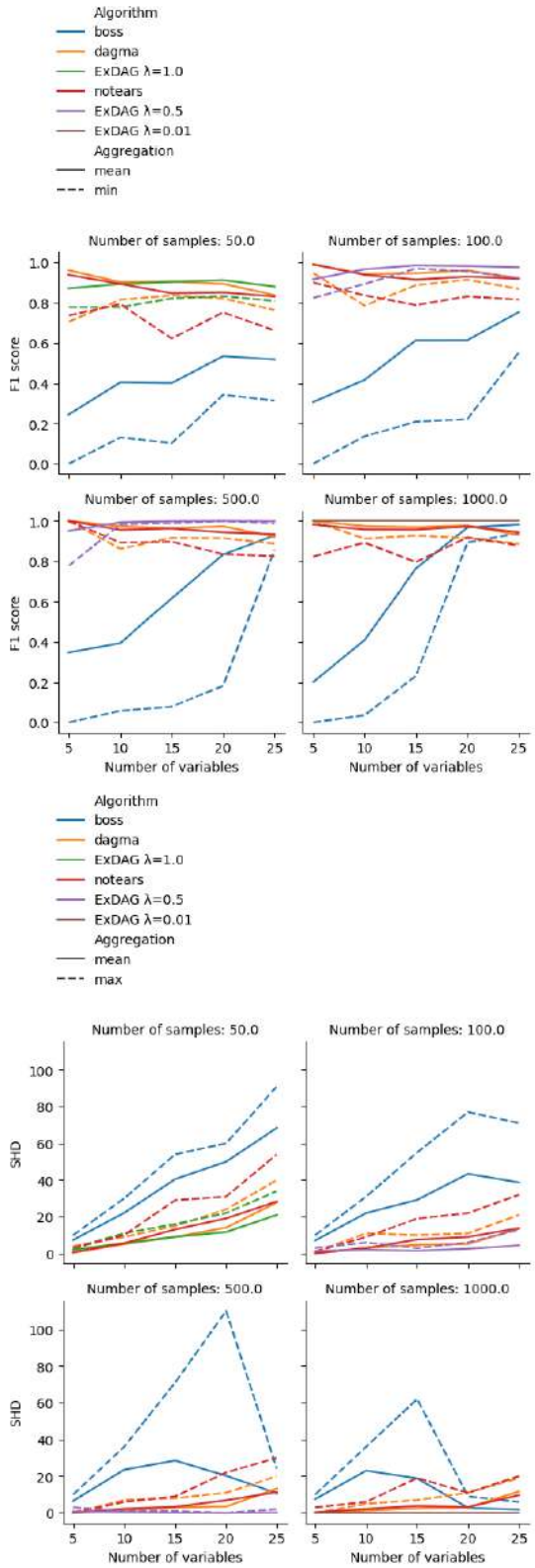


Figure 4: SHD and F1 score for a test case using SF4 random ensemble; mean and maximum across 10 runs.

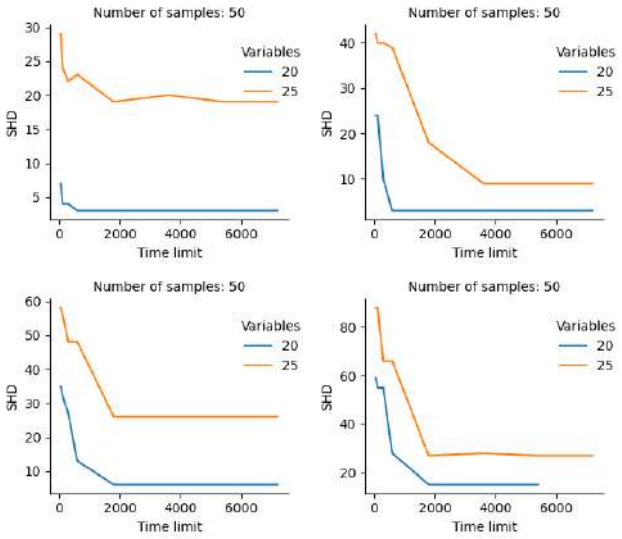


Figure 5: Convergence progress for the generation methods SF2, ER2, SF3, SF5

settings, our approach is inspired by generalized Benders decomposition for MIQP (Geoffrion, 1972) and the generation of subtour elimination constraints from integer solutions (Aguayo et al., 2018) for the travelling salesman problem (Cook et al., 2011), which makes it possible to have a separation method with quadratic complexity in the number of vertices of the DAG, i.e., random variables.

This provides a robust and near-exact reconstruction of DAGs up to 25 vertices, when a sufficient number of samples is available, which surpasses the state-of-the-art (see Section 5.2). We also demonstrate how to utilize the primal-dual nature of the solver to formulate termination criteria, which could be a challenge in some applications.

As an important step for further work, one could consider decompositions from structural graph theory (Hliněný et al., 2008), utilized similarly to their use in *a priori* enumeration of cycles in (Studený et al., 2021). Similarly, one could consider certain pre-processing of the instances utilizing conditional independence.

References

Achterberg, T. (2007). *Constraint Integer Programming*. PhD thesis, Technische Universitaet Berlin.

Aguayo, M. M., Sarin, S. C., and Sherali, H. D. (2018). Solving the single and multiple asymmetric traveling salesmen problems by generating subtour elimination constraints from integer solutions. *IISE Transactions*, 50(1):45–53.

Ahuja, K., Mahajan, D., Wang, Y., and Bengio, Y. (2023). Interventional causal representation learning. In *International conference on machine learning*, pages 372–407. PMLR.

Andrews, B., Ramsey, J., sanchez romero, R., Camchong, J., and Kummerfeld, E. (2024). Fast scalable and accurate discovery of dags using the best order score search and grow-shrink trees. *Advances in neural information processing systems*, 36:63945–63956.

Aragam, B., Amini, A. A., and Zhou, Q. (2017). Learning directed acyclic graphs with penalized neighbourhood regression.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

Bartlett, M. and Cussens, J. (2017). Integer linear programming for the bayesian network structure learning problem. *Artificial Intelligence*, 244:258–271.

Borndörfer, R., Hoppmann, H., Karbstein, M., and Lindner, N. (2020). Separation of cycle inequalities in periodic timetabling. *Discrete Optimization*, 35:100552.

Bottou, L., Peters, J., Quiñonero-Candela, J., Charles, D. X., Chickering, D. M., Portugaly, E., Ray, D., Simard, P., and Snelson, E. (2013). Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14(11).

Buchholz, S., Rajendran, G., Rosenfeld, E., Aragam, B., Schölkopf, B., and Ravikumar, P. (2024). Learning linear causal representations from interventions under general nonlinear mixing. *Advances in Neural Information Processing Systems*, 36.

Chen, R., Dash, S., and Gao, T. (2021). Integer programming for causal structure learning in the presence of latent variables. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1550–1560. PMLR.

Cook, W. J., Applegate, D. L., Bixby, R. E., and Chvátal, V. (2011). *The traveling salesman problem: a computational study*. Princeton university press.

Cussens, J. (2011). Bayesian network learning with cutting planes. In *27th Conference on Uncertainty in Artificial Intelligence*. arXiv:1202.3713.

Cussens, J., Haws, D., and Studený, M. (2017). Polyhedral aspects of score equivalence in bayesian network structure learning. *Mathematical Programming*, 164:285–324.

Duncan, O. D. (2014). *Introduction to structural equation models*. Elsevier.

Erdős, P., Rényi, A., et al. (1960). On the evolution of random graphs. *Publ. math. inst. hung. acad. sci*, 5(1):17–60.

Geoffrion, A. M. (1972). Generalized benders decomposition. *Journal of optimization theory and applications*, 10:237–260.

Gondzio, J. (2012). Interior point methods 25 years later. *European Journal of Operational Research*, 218(3):587–601.

Heckerman, D. (2022). A tutorial on learning with bayesian networks.

Hliněný, P., Oum, S.-i., Seese, D., and Gottlob, G. (2008). Width parameters beyond tree-width and their applications. *The computer journal*, 51(3):326–362.

Hoover, K. and Demiralp, S. (2003). Searching for the causal structure of a vector autoregression. *SSRN Electronic Journal*.

Kilian, L. (2011). Structural Vector Autoregressions. CEPR Discussion Papers 8515, C.E.P.R. Discussion Papers.

Kitson, N., Constantinou, A., Zhigao, G., Liu, Y., and Chobtham, K. (2023). A survey of bayesian network structure learning. *Artificial Intelligence Review*, 56:1–94.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*.

Loh, P.-L. and Bühlmann, P. (2013). High-dimensional learning of linear causal networks via inverse covariance estimation. *Journal of Machine Learning Research*, 15.

Lorch, L., Krause, A., and Schölkopf, B. (2024). Causal modeling with stationary diffusions. In *International Conference on Artificial Intelligence and Statistics*, pages 1927–1935. PMLR.

Madigan, D. and Raftery, A. E. (1994). Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89(428):1535–1546.

Mitchell, J. E. (2002). Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, 1(1):65–77.

Pamfil, R., Sriwattanaworachai, N., Desai, S., Pilgerstorfer, P., Beaumont, P., Georgatzis, K., and Aragam, B. (2020). Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*.

Park, J., Buchholz, S., Schölkopf, B., and Muandet, K. (2023). A measure-theoretic axiomatisation of causality. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 28510–28540. Curran Associates, Inc.

Pearl, J. (2009). *Causality*. Cambridge University Press, 2 edition.

Peters, J., Janzing, D., and Schölkopf, B. (2017). *Elements of causal inference: foundations and learning algorithms*. The MIT Press.

Sahinidis, N. and Grossmann, I. (1991). Convergence properties of generalized benders decomposition. *Computers & Chemical Engineering*, 15(7):481–491.

Schölkopf, B., Locatello, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., and Bengio, Y. (2021). Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634.

Shimizu, S., Hoyer, P. O., Hyvärinen, A., and Kerminen, A. J. (2006). A linear non-Gaussian acyclic model for causal discovery. *J. Mach. Learn. Res.*, 7:2003–2030.

Simchowitz, M., Mania, H., Tu, S., Jordan, M. I., and Recht, B. (2018). Learning without mixing: Towards a sharp analysis of linear system identification. In *Conference On Learning Theory*, pages 439–473. PMLR.

Studený, M., Cussens, J., and Kratochvíl, V. (2021). The dual polyhedron to the chordal graph polytope and the rebuttal of the chordal graph conjecture. *International Journal of Approximate Reasoning*, 138:188–203.

van de Geer, S. and Bühlmann, P. (2012). ℓ_0-penalized maximum likelihood for sparse directed acyclic graphs. *The Annals of Statistics*, 41.

Vo, T. Q. T., Baiou, M., Nguyen, V. H., and Weng, P. (2023). Improving subtour elimination constraint generation in branch-and-cut algorithms for the TSP with machine learning. In *International Conference on Learning and Intelligent Optimization*, pages 537–551. Springer.

Wang, X., Dunson, D., and Leng, C. (2016). No penalty no tears: Least squares in high-dimensional linear models. In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1814–1822, New York, New York, USA. PMLR.

Waxman, D., Butler, K., and Djurić, P. M. (2024). Dagma-dce: Interpretable, non-parametric differentiable causal discovery. *IEEE Open Journal of Signal Processing*, 5:393–401.

Willems, J. C., Rapisarda, P., Markovsky, I., and De Moor, B. L. (2005). A note on persistency of excitation. *Systems & Control Letters*, 54(4):325–329.

Yu, Y., Gao, T., Yin, N., and Ji, Q. (2021). Dags with no curl: An efficient dag structure learning approach.

Yuan, K.-H. and Bentler, P. M. (2006). Structural equation modeling. *Handbook of statistics*, 26:297–358.

Zhang, Z., Ng, I., Gong, D., Liu, Y., Gong, M., Huang, B., Zhang, K., van den Hengel, A., and Shi, J. Q. (2024). Analytic DAG constraints for differentiable DAG learning. https://openreview.net/forum?id=Z8RPghUs3W.

Zheng, X., Aragam, B., Ravikumar, P. K., and Xing, E. P. (2018). Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31.

Zhou, S. (2009). Thresholding procedures for high dimensional variable selection and statistical estimation. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., and Culotta, A., editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Not Applicable
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. Yes

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. Not Applicable
   (b) Complete proofs of all theoretical results. Not Applicable
   (c) Clear explanations of any assumptions. Yes

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). Yes
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. Yes
   (b) The license information of the assets, if applicable. Yes
   (c) New assets either in the supplemental material or as a URL, if applicable. Not Applicable
   (d) Information about consent from data providers/curators. Not Applicable
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. Not Applicable
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable

EXDBN: EXACT LEARNING OF DYNAMIC BAYESIAN NETWORKS

Pavel Rytíř
Faculty of Electrical Engineering
Czech Technical University in Prague

Aleš Wodecki
Faculty of Electrical Engineering
Czech Technical University in Prague

Georgios Korpas
HSBC Lab, Innovation and Ventures
HSBC
London

Jakub Mareček
Faculty of Electrical Engineering
Czech Technical University in Prague

ABSTRACT

Causal learning from data has received much attention in recent years. One way of capturing causal relationships is by utilizing Bayesian networks. There, one recovers a weighted directed acyclic graph, in which random variables are represented by vertices, and the weights associated with each edge represent the strengths of the causal relationships between them. This concept is extended to capture dynamic effects by introducing a dependency on past data, which may be captured by the structural equation model, which is utilized in the present contribution to formulate a score-based learning approach. A mixed-integer quadratic program is formulated and an algorithmic solution proposed, in which the pre-generation of exponentially many acyclicity constraints is avoided by utilizing the so-called branch-and-cut ("lazy constraint") method. Comparing the novel approach to the state of the art, we show that the proposed approach turns out to produce excellent results when applied to small and medium-sized synthetic instances of up to 25 time-series. Lastly, two interesting applications in bio-science and finance, to which the method is directly applied, further stress the opportunities in developing highly accurate, globally convergent solvers that can handle modest instances.

1 INTRODUCTION

The problem of causal learning using graphical structures has received considerable attention from a wide range of communities in recent years. This attention comes from the wide range of applications including, but not limited to, medicine (Rajapakse & Zhou, 2007), machine learning (Koller & Friedman, 2009), econometrics (Luetkepohl, 2005; Demiralp & Hoover, 2003; Malinsky & Spirtes, 2018) and others (Guo et al., 2020; Assaad et al., 2022). One key reason for this is that in many applications data is abundant, but modeling using first principles may be difficult due to the complexity of the problem at hand (Guo et al., 2020). Some of this complexity may arise due to an abundance of non-linear effects, only a partial ability to observe the system, or unexpected stochastic effects influencing the system. A key benefit of learning via graphical structures is the full explainability of the output; the network may be either used to compute outputs for different situations or the learned graph structure may be inspected and dependencies of particular interest analyzed.

In this contribution, we revisit the score-based learning of dynamic Bayesian networks utilizing a directed acyclic graph (DAG) structure augmented by additional time dependencies from data (Murphy, 2002; Dean & Kanazawa, 1989; Assaad et al., 2022). This approach to learning causality has been successfully applied to a variety of problems, many of which are related to applications in medicine (Zandonà et al., 2019; van Gerven et al., 2008; Michoel & Zhang, 2023; Zhong et al., 2023). Besides medical applications, the dynamic Bayesian network approach representations are widely used in econometrics (Hoover & Demiralp, 2003b) and financial risk modeling (Ballester et al., 2023). This broad scope of applications has spawned a large number of excellent solvers, which under different assumptions are able to discover the underlying causal structure of a system.

The use of various assumptions is key to ensure the tractability of a solver, since the the number of constraints that is needed to impose to acyclicity of the representing graph is super-exponential in the number of random variables.

One of the possible assumptions is to separate observational and interventional data (Gao et al., 2022), which reduces the number of dependencies that need to be found. Another is the assumption of underlying continuous dynamics represented by stochastic differential equations Bellot et al. (2021). One can also assume a-priori knowledge about the time-lagged data and incorporate this knowledge into the solver Sun et al. (2021). One can also deal with the general problem and propose local methods (Pamfil et al., 2020; Gao et al., 2022), which can scale further at the cost of some loss of accuracy. Note that many of the previous works also combine several of these approaches to arrive at solvers that are tractable and applicable to a wide range of applications.

We utilize mixed-integer programming in learning dynamic Bayesian networks. All of the previous works mentioned above focus mostly on solving the curse of dimensionality and scaling with adequate precision. On the other hand, we focus on leveraging fundamental principles that apply to quadratic mixed-integer programs to find global solutions to the score-based DAG learning problem, which results in a high quality reconstruction of the DAG. Furthermore, we tackle the curse of dimensionality by avoiding the pre-generation of the acyclic constraints. It is shown, that given sufficient data, only a small amount of these constraints are actually needed to ensure the acyclicity of the resulting graph, which leads to the runtime generation of these constraints granting a large speedup. The formulation and its implementation are easily reproducible, making it accessible to a wide range of potential practitioners.

2 Problem Formulation

Before formulating the problem of score-based Bayesian network learning as a mixed-integer program, let us describe the state space using a structural vector autoregressive model (Hoover & Demiralp, 2003a; Kilian, 2011). Let $d, T \in \mathbb{N}$ and assume that $X_{i,t}$ is a set of stochastic processes, where $i \in \{1, 2, \dots, d\}$ and $t \in \{1, 2, \dots, T\}$. Let the underlying DAG to be learned be characterized by the set of vertices and edges organised in a pair (V, E), where the vertices are indexed by the set of integers $\{1, 2, \dots, d\}$ and $E \subset V \times V$. Denote the auto-regressive order by $p \in \mathbb{N}$ and let

$$W \in \mathbb{R}^{d,d}, \quad A_i \in \mathbb{R}^{d,d}, \quad i \in \{1, 2, \dots, p\}, \tag{1}$$

be the weighed adjacency matrix of (V, E) and A_i be the matrices encoding the time regressive dependencies. The intra-slice interactions defined at the present time are expressed by the weight matrix W and the inter-slice interactions are expressed by A_i. For the sake of simplicity, the matrices A_i are assumed to be constant. Let $X_t \in \mathbb{R}^{n,d}$ be the data matrix at time t, then the linear auto-regressive model of order p reads

$$X_t = X_t W + X_{t-1} A_1 + X_{t-2} A_2 + \ldots + X_{t-p} A_p + Z, \tag{2}$$

where $Z \in \mathbb{R}^{n,d}$ is the error vector, which is not assumed to be Gaussian. Note that non-linear auto-regressive models can also be formulated in an analogous way. The problem may be written in a simplified manor as

$$X_t = X_t W + Y_t A + Z, \tag{3}$$

where

$$A = A_1 \,|A_2 \,\dots|A_p\,, \quad Y_t = X_{t-1}\,|X_{t-2}\,\dots|X_{t-p} \tag{4}$$

To maximize the fit of the data over the model, a score function, which reads may be formulated

$$J\left(W, A\right) = \left\|X - XW - YA\right\|_F^2 + \lambda \left\|W\right\| + \eta \left\|A\right\|, \tag{5}$$

where $\|\cdot\|$ denotes an arbitrary matrix norm and $\lambda, \eta > 0$ are sufficiently small regularization coefficients. The problem of interest then reads

$$\begin{aligned} &\min_{W,A} J\left(W, A\right), \\ &G\left(W\right) \in \Gamma_{DAG}, \end{aligned} \tag{6}$$

where A need not be constrained, since cycles are excluded by construction; $\|\cdot\|$ denotes an arbitrary norm, which is usually chosen to be the L1-norm and $\|\cdot\|_{\mathbb{F}}$ denotes the Frobenius norm.

Remark 1 *The identifiability of W and A using 6 has been studied for both Gaussian and non-Gaussian noise. Irrespective of the noise, the identifiability of A is a consequence of the basic theory of autoregressive models (Kilian, 2011). The identifiability of W is a bit more involved and must be separated into the Gaussian and non-Gaussian case. In either case, however, identifiability is possible under mild conditions (Hyvärinen et al., 2010; Peters & Bühlmann, 2012).*

3 Mixed Integer Quadratic Programming Formulation

Formulating the learning problem as a mixed-integer quadratic problem sets things up so that a globally convergent algorithm may be used. This is fundamental in order for high-precision learning to be possible.

Let $e_{i,j} \in \{0,1\}$ and $e^s_{i,j} \in \{0,1\}$ be decision variables that govern the placement of edges between random variables at time level t and between time levels t and $t-s$, respectively, and let $w_{i,j} \in \mathbb{R}$ and $a^t_{i,j} \in \mathbb{R}$ be the associated edge weights. Using these variables, the scoring function of problem equation 6 becomes

$$J_p = \sum_{i=1}^{n}\sum_{j=1}^{d}\left| X_{i,j} - \sum_{k=1}^{d} X_{i,k} w_{k,j} - \sum_{s=1}^{p}\sum_{k=1}^{d} X^s_{i,k} a^s_{k,j} \right|^2 + \text{REG}, \tag{7}$$

which avoids the use of a bi-linear term if the additional constraints

$$w_{k,j} \leq c e_{k,j}, \quad w_{k,j} \geq -c e_{k,j} \text{ for all } k,j \in \{1,2,\ldots,d\}. \tag{8}$$

and

$$a^s_{k,j} \leq c e^s_{k,j}, \quad a^s_{k,j} \geq -c e^s_{k,j} \text{ for all } k,j \in \{1,2,\ldots,d\}, s \in \{1,2,\ldots,p\} \tag{9}$$

are imposed, where $c > 0$ is the maximal admissible magnitude of any weight and $\lambda > 0$ is a regularization constant.

Where REG is a regularization expression equals either: (L1)

$$\text{REG} = \lambda \sum_{i=1}^{n}\sum_{j=1}^{n} e_{i,j} + \eta \sum_{s=1}^{p}\sum_{i=1}^{n}\sum_{j=1}^{n} e^s_{i,j}. \tag{10}$$

or (L2)

$$\text{REG} = \lambda \sum_{i=1}^{n}\sum_{j=1}^{n} e_{i,j} + \eta \sum_{s=1}^{p}\sum_{i=1}^{n}\sum_{j=1}^{n} a^s_{i,j}. \tag{11}$$

Lastly, the acyclicity constraints are described. Let C denote the set of all cycles in a graph with d vertices, where each cycle $c \in C$ of length k is represented as a set of edges: $c = \{(i_1,i_2),(i_2,i_3),\ldots,(i_{k-1},i_1)\}$. The constraint excluding a cycle $c \in C$ from a solution then reads

$$\sum_{(i,j)\in c} e_{i,j} \leq k-1. \tag{12}$$

The algorithmic treatment of constraint equation 12 is key in the following section, in which the algorithmic treatment is discussed as implementing the branch-and-bound-and-cut algorithm without a reduction mechanism for this constraint is doomed to fail due to the super-exponential number of such constraints.

4 Algorithmic Implementation using Branch-and-Bound-and-Cut

One of our main contributions is the development of a branch-and-bound-and-cut algorithm to solve the aforementioned formulation. Since the acyclic constraints 12 needs to only be imposed for the edges of the graph representing the intra-slice level, all of what follows only applied to the intra-slice picture. While we leverage the traditional branch-and-bound approach as described in (Achterberg, 2007, e.g.), we incorporate cycle exclusion constraints equation 12 using "lazy" constraints. These

are only enforced once an integer-feasible solution candidate is found. If a violation of a lazy constraint occurs, the constraint is added across all nodes in the branch-and-bound tree. At the root node, only $O(|E|)$ constraints 8 and 9 are initially used. Cycle-exclusion constraints equation 12 are added later. Note that this method is not a heuristic and does not lead to a possibly harmful reduction (or extension) of the solution space leading to omitting possible solutions or returning solutions which are not DAGs. Furthermore, it is shown that the number of constraints that are actually needed in a computation are many orders of magnitude less than the number of all possible constraints.

Once a new mixed-integer feasible solution candidate is identified, detecting cycles becomes straightforward using depth-first search (DFS). If a cycle is detected, the corresponding lazy constraint equation 12 is added to the problem. The DFS algorithm solves the problem of cycle detection in a worst-case quadratic runtime relative to the number of vertices in the graph, which contrasts with algorithms that separate related inequalities from a continuous relaxation (Borndörfer et al., 2020; Cook et al., 2011), such as the quadratic program in our case. Three variants of adding lazy constraints for the problem were tested.

- Adding lazy constraint only for the first cycle found.
- Adding lazy constraint only for the shortest cycle found.
- Adding multiple lazy constrains for all cycles found in the current iteration in which an integer-feasible solution candidate is available.

The third mentioned variant was found to consistently deliver the best results, despite (Achterberg, 2007, Chapter 8.9). Therefore, it is applied in all of the numerical tests that follow.

5 Data Generation

We generated data in a manner similar to that described in Zheng et al. (2018) and Pamfil et al. (2020). The evaluation of ExDBN was performed on synthetic data generated as follows. First, a random intra-slice DAG was created using either the Erdős-Rény (ER) model or the scale-free Barabási–Albert (SF) model. The DAG weights were sampled uniformly from the intervals $[-2.0, -0.5] \cup [0.5, 2.0]$.

Next, inter-slice graphs were generated using the ER model. For each inter-slice graph, weights were sampled from the interval $[-0.5\alpha, -0.2\alpha] \cup [0.2\alpha, 0.5\alpha]$, where $\alpha = 1/\eta^{t-1}$, $\eta \geq 1$ is the decay parameter, and t is the time of the slice. $t = 0$ corresponds to the intra-slice, while $t \in \{1, \ldots, p\}$ represents the inter-slices.

Specifically, we have adapted ER and SF generators from Zheng et al. (2018) for dynamic networks. Notice that this results in a slightly different generator than in Pamfil et al. (2020), which may explain some of the differences in performance of DYNOTEARS, compared to the original article.

6 Numerical Experiments

In recent years, many solvers have been developed to facilitate the graphical learning of Bayesian networks that represent causality (Pamfil et al., 2020; Hyvärinen et al., 2010; Malinsky & Spirtes, 2018; Gao et al., 2022; Dallakyan, 2023; Lorch et al., 2021). Each of these solvers (including the one presented) face the curse of dimensionality, which somewhat restricts the applicability of each solver and thus through testing needs to be provided. It is impossible to test the proposed solution w.r.t. every solver developed. There is, however, a significant branch of development that allows for direct comparison and by transitivity of results the comparison with many previous solvers follows.

In 2020, Pamfil et al. (2020) have developed a locally convergent method, called DYNOTEARS, that learns causality as a Bayesian network that supersedes the solution methods previously developed (Hyvärinen et al., 2010; Malinsky & Spirtes, 2018; Zheng et al., 2018). Further developments based on previous publications include formulating the problem in the frequency domain or defining differentiable Bayesian structures (Dallakyan, 2023; Lorch et al., 2021). In the following, we provide a head-to-head comparison with DYNOTEARS and thus by transitivity with the methods documented by Hyvärinen et al. (2010); Malinsky & Spirtes (2018).

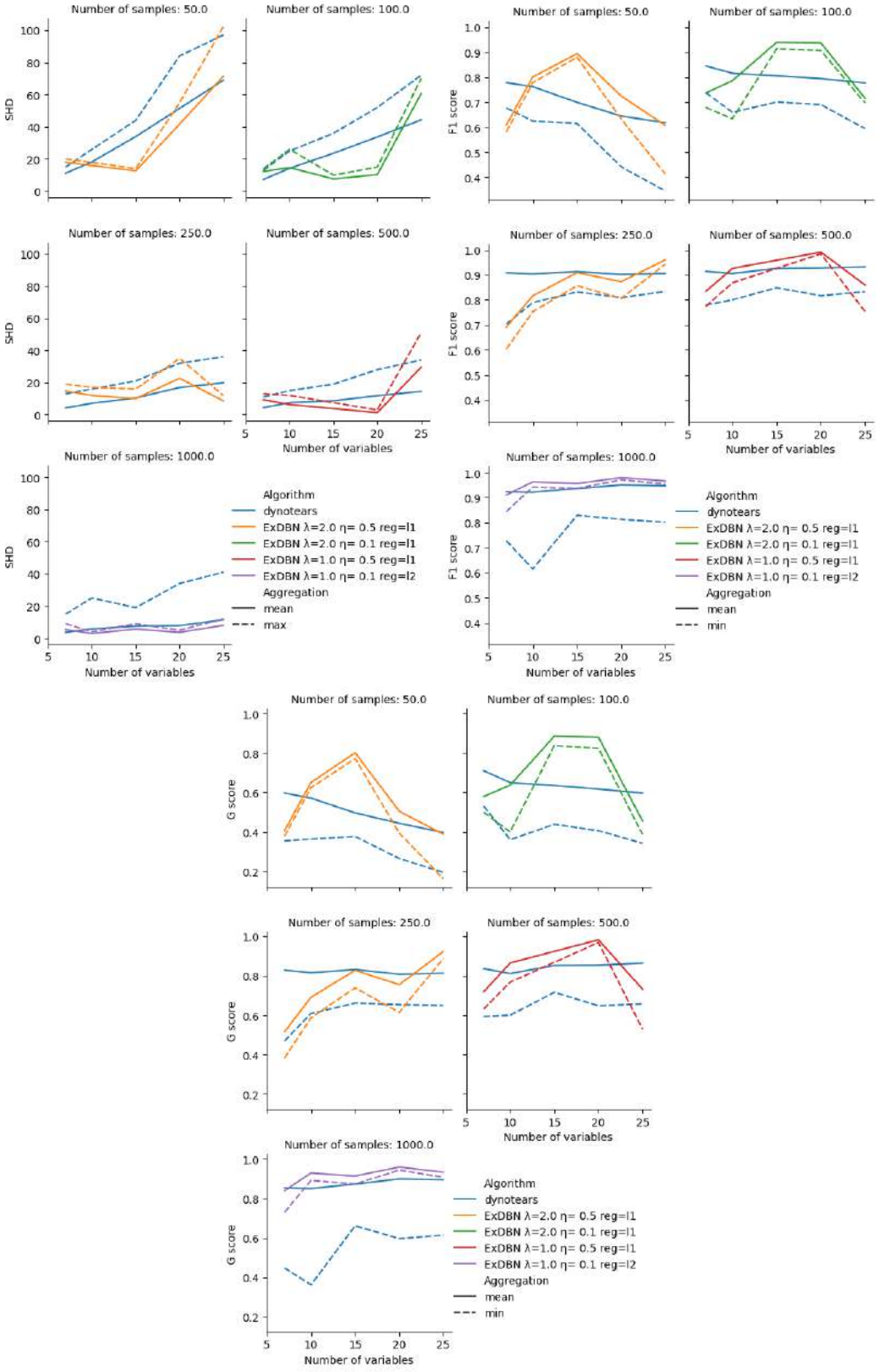


Figure 1: SHD, F1 score, and G score for a test case using ER3-1 random ensemble, i.e., edge-vertex ratio 3 on intra graph, e-v ratio 1 on inter graph, recursion depth 1.

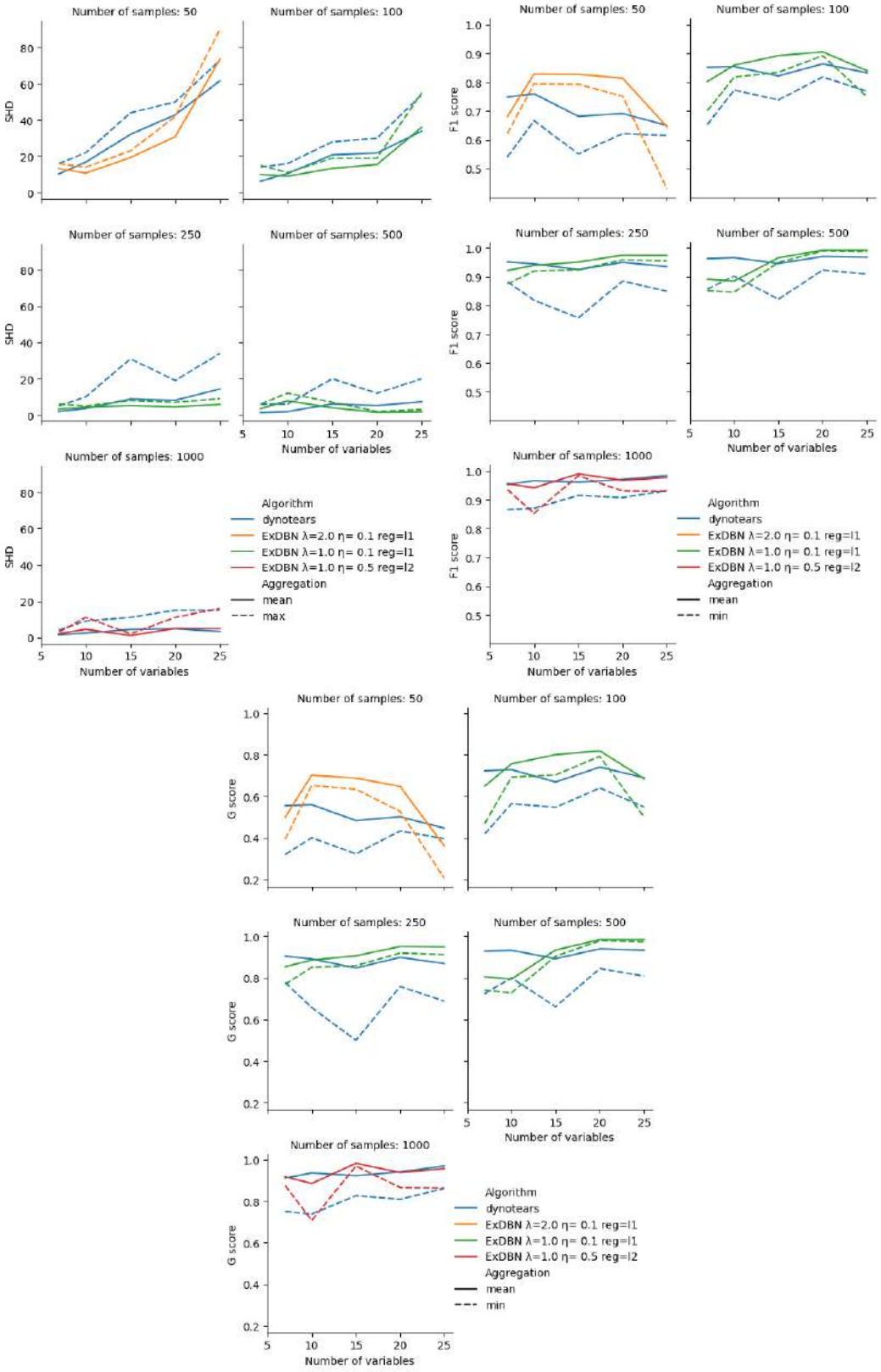


Figure 2: SHD, F1 score, and G score for a test case using SF3-1 random ensemble, i.e., edge-vertex ratio 3 on intra graph, e-v ratio 1 on inter graph, recursion depth 1.

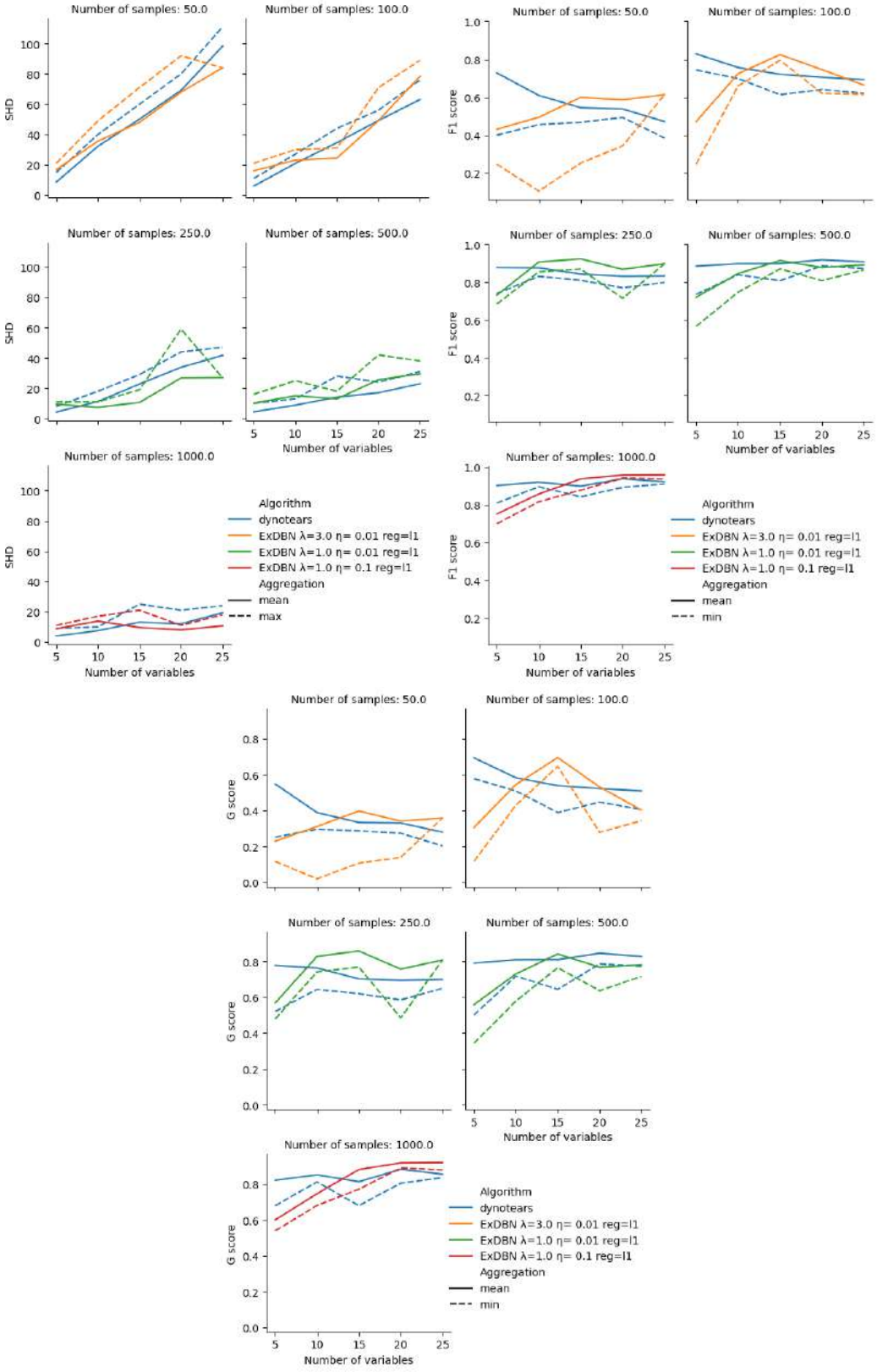


Figure 3: SHD, F1 score, and G score for a test case using ER2-1-1 random ensemble, i.e., edge-vertex ratio 2 on intra graph, e-v ratio 1 on inter graphs, recursion depth 2.

6.1 Benchmark Setup and Quantities of Interest

In Section 6.2, W_{init} denotes the adjacency matrix representing the intra-slice dependencies and A_{init} denotes the inter-slice dependencies of the ground truth, where A_{init} is used to denote a p-tuple as in equation 4. W_{init} and A_{init} are then used to generate data while applying noise of Gaussian distribution. Following the data generation process, a matrices W and A are identified and compared with W_{init} and A_{init}. Because the noisy data inevitably leads to some falsely identified edges, typically with negligible weights, one may remove edges with weight less than $\delta > 0$ from the W and A, resulting in a graph W^{δ} and A^{δ}, respectively. To compare methods for known ground truth W_{init} and A_{init}, we choose the best possible $\delta > 0$ for each method. This $\delta > 0$ may then be used as a reference for learning from data, where a ground truth is not known. Next, we introduce the relevant metrics used to evaluate the quality of reconstruction, when W_{init} is available.

In the following, we suppose that a DBN represented by an inter-slice matrix V and inter-slice matrix A is denoted by an ordered pair (V, A). Let (V, A) and (V, A) be two such pairs, then one defines the structural Hamming distance (SHD) as

$$\rho\left(V, A; W, B\right) = \sum_{i,j=1}^{d} r_{ij}\left(V, W\right) + \sum_{k=1}^{p} \sum_{i,j=1}^{d} r_{ij}\left(A_k, B_k\right), \tag{13}$$

where

$$r_{ij}\left(C, D\right) = \begin{cases} 0 & \text{if } C_{ij} \neq 0 \text{ and } D_{ij} \neq 0 \text{ or } C_{ij} = 0 \text{ and } D_{ij} = 0 \\ \frac{1}{2} & \text{if } C_{ij} \neq 0 \text{ and } D_{ji} \neq 0 \\ 1 & \text{otherwise.} \end{cases} \tag{14}$$

SHD is used as a score that describes the structural similarity of two DAGs in terms of edge placement and is commonly used to asses the quality of solutions (Zheng et al., 2018; Pamfil et al., 2020). Besides SHD

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \text{ and recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}}, \tag{15}$$

are used Andrews et al. (2024) to evaluate the quality of structural recovery. It is important to note that precision and recall isolate the false positives and negatives, respectively, opposed to SHD, where these quantities are both accounted for simultaneously. The last metric that can be used to evaluate structural similarity is the F1 score and reads

$$F_1 = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}}. \tag{16}$$

Note that all of the quantities evaluated in equation 15 and equation 16 are a result of summing up all of the differences over both inter and intra slice dependencies between a given pair (V, W) and a grownd truth.

Although structural similarity is a key concern, merely comparing structural properties does not tell the full story, as the weights play a crucial role in the resulting statistical behavior of the found DAG. This motivates us to additionally utilize a cost function based metric, which reads

$$\sigma_p\left(V, W\right) = \left|J_p\left(V\right) - J_p\left(W\right)\right|, \tag{17}$$

where $\lambda = 0$ and typically $p = 2$. We may also evaluate the differences in adjacency matrices by considering

$$\left\|V - W\right\|_{\mathbb{F}}, \tag{18}$$

where $\left\|\cdot\right\|_{\mathbb{F}}$ denotes the Frobenius norm.

6.2 Synthetic Benchmark Results

In the following benchmark, the generation methods described in Section 5 are used to compare ExDBN with DYNOTEARS Pamfil et al. (2020) under the assumption of Gaussian noise. Even though the cost function is a maximum likelihood estimator (see Section 1) for non-Gaussian noise also, we leave this evaluation for a future publication. The scaling is studied for different numbers of variables, samples and graph generation methods with the relevant metrics; SHD, F1 score and G

score recorded in Figures 1, 2 and 3. A statistical ensemble with 10 different seeds was used for each of the experiments and the mean and worst possible case value is used in the plots. It should be noted, that naturally the worst possible value and the mean can together be used to bound the variance with respect. The solution time is capped for ExDBN at 7200 seconds and the regularization applied in ExDBN needs to be scaled appropriately with the number of samples as it is assumed that the optimal choice of regularization constant is a decreasing function of the number of samples. We use the aforementioned as a guide (in a non-strict way) to find the right regularization for a given sample size. This follows from the fact that the regularization is to be kept proportionally small to the main objective expressed by equation 7. Furthermore, it was found that changing the regularization from L1 to L2 is beneficial for identification when the number of samples is large. Furthermore, if we don't know the ground truth graph. We can try to run the algorithm for multiple values of λ and η and then use the one which yields better MIP GAP. For smaller number of samples the regularization L1 works better. For bigger number of samples L2 yields good results and it is usually faster.

The tests results may be divided into two categories by the average number of edges. Figures 1 and 2 show higher degree graphs (average degree 3) and Figure 3 depicts the reconstruction of a lower degree graph. In the case of the lower degree graph, it is clear that both DYNOTEARS and ExDBN perform similarly with ExDAG beating out DYNOTEARS some of the time with the converse being true equally often. In the case of the identification of higher degree graphs, however, one can notice that the worst possible performance and the mean performance are much closer in the case of ExDBN, where we can point out for instance the G score in the case with 1000 samples. In these instances the differences between the worst possible G score difference is between 0.3 and 0.5 in the case of DYNOTEARS but stays well under 0.1 in the case of ExDBN. The aforementioned can be interpreted as superior reliability of solution as the worst possible reconstruction is consistently better. Focusing on the 1000 sample case still, while somewhat taking into account the previous ones also, we see that the performance gap between the solvers increases in favor of ExDBN as we increase the number of samples. In the lower sample cases, one may also observe that ExDBN outperforms DYNOTEARS for many graph sizes in the mean and consistently outperforms DYNOTEARS in the worst possible case (min/max depending on metric). Note that the global convergence of the method, which is rooted in the fundamentals of mixed integer quadratic programming, allows us to increase computation time, which leads to improving the metrics reported further. While some time-sensitive applications like short term stock evaluation might not be able to benefit from this, others like biomedical applications might benefit as a computation lasting several days, in which the accuracy in measurably improved (by monitoring the duality gap), is desirable.

6.3 Application in Biomedical Sciences

In biomedical sciences, there is a keen interest in learning dynamic Bayesian networks with the view of estimating causal effects (Tennant et al., 2020) and identifying confounding variables that require conditioning. A recent metaanalysis (Tennant et al., 2020) of 234 articles on learning DAGs in biomedical sciences found that the averaged DAG had 12 nodes (range: 3–28) and 29 arcs (range: 3–99). Interestingly, none of the DAGs were as sparse as the commonly considered random ensembles; median saturation was 46%, i.e., each of all possible arcs appeared with probability 46% and does not converge to a global minimum of the problem.

As an example, we consider a recently proposed benchmark of Ryšavý et al. (2024), where Krebs cycle is to be reconstructed from time series of reactant concentrations of varying lengths. There, DYNOTEARS cannot reach (Ryšavý et al., 2024) F1 score of 0.5 even with very long time series. In contrast, our method can solve instances equation 7 to global optimality. Using ExDBN, however, the global minimization is ensured given sufficient time and thus the maximum likelihood estimator is found. It should be noted, however, that depending on the number of samples and noise, it may be that even the maximum likelihood estimator is not sufficiently accurate. This does not however reflect poorly on the method itself, but it rather a matter of the modification of data collection methods associated with the experiment. In one hour time limit, ExDBN can find a solution with 38% duality gap.

6.4 Application in Finance

In financial services, there is also a number of important applications. The original DYNOTEARS paper considered a model of diversifying investments in stocks based on dynamic Bayesian networks. Independently, Ballester et al. (2023) consider systemic credit risk, which is one of the most important concerns within the financial system, using dynamic Bayesian networks. They found that the transport and manufacturing sectors transmit risk to many other sectors, while the energy sector and banking receive risk from most other sectors. To a lesser extent, there is risk transmission present between approximately 25% of the pairs of sectors, and these network relationships explain between 5 % and 40 % of single systemic risks. Notice that these instances are much denser than the commonly used random ensembles.

We elaborate upon the example of Ballester et al. (2023), where 10 time-series capture the spreads of 10 European credit default swaps (CDS). Considering the strict licensing terms of Refinitiv, the data from Ballester et al. (2023) are not available from the authors, but we have downloaded 16 time-series capturing the spreads of 16 European CDS with RED6 codes 05ABBF, 06DABK, 0H99B7, 2H6677, 2H66B7, 48DGFE, 6A516F, 8A87AG, 8B69AP, 8D8575, DD359M, EFAGG9, FF667M, FH49GG, GG6EBT, NN2A8G, from January 1st, 2007, to September 25th, 2024. This amounts to over 11 MB of time-series data, when stored as comma-delimited values in plain text. While the procedure for learning the dynamic Bayesian network in Ballester et al. (2023) is rather heuristic, we can solve the mixed-integer programming (MIP) instance for the 16 European CDS within 2 minutes. In the heuristic of Ballester et al. (2023), they first perform unconditional independence tests on each set of two time series containing an original series and a lagged time series, to prune the subsequent number of unconditional independence tests performed. There are 45 unconditional and conditional independence tests performed first, to suggest further 200 conditional independence tests. We stress that the procedure of Ballester et al. (2023) does not come with any guarantees, while our instance equation 7 is solved to global optimality. The run-time to global optimum of 2 minutes (using L2 regularization) validates the scalability of mixed-integer programming solvers.

7 Conclusion

Dynamic Bayesian networks have wide-ranging applications, including those in biomedical sciences and computational finance, illustrated above. Unfortunately, their use has been somewhat limited by the lack of well-performing methods for learning those. Our method, ExDBN, provides the best possible estimate of the DBN, in the sense of minimizing empirical risk equation 7. Significantly, our method does not suffer much from the curse of dimensionality even for real-world, dense instances, which are typically challenging for other solvers. This is demonstrated most clearly in the systemic-risk transmission use-case detailed in Section 6.4, in which the global minimizer is found within 2 minutes. Additionally, the use of the guarantees on the distance to the global minimizer (so-called MIP gap, available ahead of the convergence to the global minimizer) provides a significant tool for fine-tuning the parameters of the solver in the case of real-world application, where the ground truth is not available. Combined with global convergence guarantees of the maximum likelihood estimator, this provides a robust method, with state-of-the-art statistical performance.

References

Tobias Achterberg. *Constraint Integer Programming*. PhD thesis, Technische Universitaet Berlin, 2007.

Bryan Andrews, Joseph Ramsey, Ruben sanchez romero, Jazmin Camchong, and Erich Kummerfeld. Fast scalable and accurate discovery of dags using the best order score search and grow-shrink trees. *Advances in neural information processing systems*, 36:63945–63956, 05 2024.

Charles Assaad, Emilie Devijver, and Eric Gaussier. Survey and evaluation of causal discovery methods for time series. *Journal of Artificial Intelligence Research*, 73:767–819, 02 2022. doi: 10.1613/jair.1.13428.

Laura Ballester, Jesúa López, and Jose M. Pavía. European systemic credit risk transmission using bayesian networks. *Research in International Business and Finance*, 65:101914, 2023.

ISSN 0275-5319. doi: https://doi.org/10.1016/j.ribaf.2023.101914. URL https://www.sciencedirect.com/science/article/pii/S0275531923000405.

Alexis Bellot, Kim Branson, and Mihaela van der Schaar. Neural graphical modelling in continuous-time: consistency guarantees and algorithms. In *International Conference on Learning Representations*, 2021. URL https://api.semanticscholar.org/CorpusID:246485884.

Ralf Borndörfer, Heide Hoppmann, Marika Karbstein, and Niels Lindner. Separation of cycle inequalities in periodic timetabling. *Discrete Optimization*, 35:100552, 2020.

William J Cook, David L Applegate, Robert E Bixby, and Vasek Chvátal. *The traveling salesman problem: a computational study*. Princeton university press, 2011.

Aramayis Dallakyan. On learning time series summary dags: A frequency domain approach. 2023. URL https://api.semanticscholar.org/CorpusID:258179448.

Thomas L. Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5, 1989. URL https://api.semanticscholar.org/CorpusID:57798167.

Selva Demiralp and Kevin Hoover. Searching for the causal structure of a vector autoregression*. *Oxford Bulletin of Economics and Statistics*, 65:745 – 767, 12 2003. doi: 10.1046/j.0305-9049.2003.00087.x.

Tian Gao, Debarun Bhattacharjya, Elliot Nelson, Miaoyuan Liu, and Yue Yu. Idyno: Learnmalinskying nonparametric dags from interventional dynamic data. In *International Conference on Machine Learning*, 2022. URL https://api.semanticscholar.org/CorpusID:250340690.

Ruocheng Guo, Lu Cheng, Jundong Li, P. Richard Hahn, and Huan Liu. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys*, 53(4):1–37, July 2020. ISSN 1557-7341. doi: 10.1145/3397269. URL http://dx.doi.org/10.1145/3397269.

Kevin Hoover and Selva Demiralp. Searching for the causal structure of a vector autoregression. *SSRN Electronic Journal*, 04 2003a. doi: 10.2139/ssrn.388840.

Kevin D. Hoover and Selva Demiralp. Searching for the causal structure of a vector autoregression. *Macroeconomics eJournal*, 2003b. URL https://api.semanticscholar.org/CorpusID:16111786.

Aapo Hyvärinen, Kun Zhang, Shohei Shimizu, and Patrik Hoyer. Estimation of a structural vector autoregression model using non-gaussianity. *Journal of Machine Learning Research*, 11:1709–1731, 07 2010.

Lutz Kilian. Structural Vector Autoregressions. CEPR Discussion Papers 8515, C.E.P.R. Discussion Papers, August 2011. URL https://ideas.repec.org/p/cpr/ceprdp/8515.html.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. 01 2009. ISBN 978-0-262-01319-2.

Lars Lorch, Jonas Rothfuss, Bernhard Scholkopf, and Andreas Krause. Dibs: Differentiable bayesian structure learning. *ArXiv*, abs/2105.11839, 2021. URL https://api.semanticscholar.org/CorpusID:235187432.

Helmut Luetkepohl. *The New Introduction to Multiple Time Series Analysis*. 01 2005. ISBN 978-3-540-40172-8. doi: 10.1007/978-3-540-27752-1.

Daniel Malinsky and Peter Spirtes. Causal structure learning from multivariate time series in settings with unmeasured confounding. In Thuc Duy Le, Kun Zhang, Emre Kıcıman, Aapo Hyvärinen, and Lin Liu (eds.), *Proceedings of 2018 ACM SIGKDD Workshop on Causal Disocvery*, volume 92 of *Proceedings of Machine Learning Research*, pp. 23–47. PMLR, 20 Aug 2018. URL https://proceedings.mlr.press/v92/malinsky18a.html.

Tom Michoel and Jitao David Zhang. Causal inference in drug discovery and development. *Drug Discovery Today*, 28(10):103737, 2023. ISSN 1359-6446. doi: https://doi.org/10.1016/j.drudis.2023.103737. URL https://www.sciencedirect.com/science/article/pii/S1359644623002532.

Kevin Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, 01 2002.

Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Paul Beaumont, Konstantinos Georgatzis, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, 2020. URL https://api.semanticscholar.org/CorpusID:211010514.

Jonas Peters and Peter Bühlmann. Identifiability of gaussian structural equation models with equal error variances. *Biometrika*, 101, 05 2012. doi: 10.1093/biomet/ast043.

Jagath Rajapakse and Juan Zhou. Learning effective brain connectivity with dynamic bayesian networks. *NeuroImage*, 37:749–60, 10 2007. doi: 10.1016/j.neuroimage.2007.06.003.

Petr Ryšavý, Xiaoyu He, and Jakub Mareček. Causal learning in biomedical applications: A benchmark, 2024. URL https://arxiv.org/abs/2406.15189.

Xiangyu Sun, Guiliang Liu, Pascal Poupart, and Oliver Schulte. Nts-notears: Learning nonparametric temporal dags with time-series data and prior knowledge. *ArXiv*, abs/2109.04286, 2021. URL https://api.semanticscholar.org/CorpusID:237454655.

Peter W G Tennant, Eleanor J Murray, Kellyn F Arnold, Laurie Berrie, Matthew P Fox, Sarah C Gadd, Wendy J Harrison, Claire Keeble, Lynsie R Ranker, Johannes Textor, Georgia D Tomova, Mark S Gilthorpe, and George T H Ellison. Use of directed acyclic graphs (DAGs) to identify confounders in applied health research: review and recommendations. *International Journal of Epidemiology*, 50(2):620–632, 12 2020. ISSN 0300-5771. doi: 10.1093/ije/dyaa213. URL https://doi.org/10.1093/ije/dyaa213.

Marcel A.J. van Gerven, Babs G. Taal, and Peter J.F. Lucas. Dynamic bayesian networks as prognostic models for clinical patient management. *Journal of Biomedical Informatics*, 41(4):515–529, 2008. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2008.01.006. URL https://www.sciencedirect.com/science/article/pii/S1532046408000154.

Alessandro Zandonà, Rosario Vasta, Adriano Chió, and Barbara Di Camillo. A dynamic bayesian network model for the simulation of amyotrophic lateral sclerosis progression. *BMC Bioinformatics*, 20, 04 2019. doi: 10.1186/s12859-019-2692-x.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

Q. Zhong, Y. Cheng, Z. Li, D. Wang, C. Rao, Y. Jiang, L. Li, Z. Wang, P. Liu, Y. Zhao, P. Li, J. Suo, Q. Dai, and K. He. Ultra-efficient causal learning for dynamic csa-aki detection using minimal variables. In *medRxiv*, 2023. URL https://api.semanticscholar.org/CorpusID:265657431.

Empirical Bayes for Dynamic Bayesian Networks Using Generalized Variational Inference

Vyacheslav Kungurtsev, Apaar*, Aarya Khandelwal*, Parth Sandeep Rastogi**,
Bapi Chatterjee, Jakub Mareček

April 2024

1 Introduction

Dynamic Bayesian Networks (DBNs) are a class of Probabilistic Graphical Models that enable the modeling of a Markovian dynamic process through defining the kernel transition by the DAG structure of the graph found to fit a dataset. There are a number of structure learners than enable one to find the structure of a DBN to fit data, each of which with its own set of particular advantages and disadvantages.

The structure of a DBN itself presents transparent criteria in order to identify causal discovery between variables. However, without the presence of large quantities of data, identifying a ground truth causal structure becomes unrealistic in practice. However, one can consider a procedure by which a set of graphs identifying structure are computed as approximate noisy solutions, and subsequently amortized in a broader statistical procedure fitting a mixture of DBNs. Each component of the mixture presents an alternative hypothesis on the causal structure. From the mixture weights, one can also compute the Bayes Factors comparing the preponderance of evidence between different models.

This presents a natural opportunity for the development of *Empirical Bayesian* methods. See, e.g. [12] for a classic overview. Empirical Bayes attempts to combine the frequentist and Bayesian modeling approaches by using a frequentist point estimate as a prior for a subsequent Bayesian model. In this case, we perform this as a mixed integer programming problem [1], solving for the optimal structure and weights on a subsample of the data. This provides us with a point estimate of an optimal (and consistent, see [9]) structure. Subsequently we repeat the procedure with a different subsample of the data while encouraging model heterogeneity. After obtaining M models with least-squares optimized weights $\{\Theta_m\}_{m=1,\dots,M}$, now we solve a measure valued problem wherein we relax the structure component into learning a mixture across these networks, corresponding to optimization over an M dimensional simplex and at the same time, solving for the measure defining the posterior distribution of the weights using a prior that centers on the MIP solution and enforces an f-divergence from this prior as a constraint.

There are a number of fundamental advantages to this procedure. As described in [5], in the common scenario of interest for learning (D)BNs, with high dimensionality of the feature space and a paucity of data samples, overconfidence in regards to certainty of structure is often deleterious to appropriate certification of model accuracy and variance. On the other hand, appropriately constructing a mixture model a priori is methodologically challenging as far as the poor scaling of any appropriate encoding with the problem dimension. With an uninformative prior, mixing times can be unreasonably long. However, by initially finding a set of distinct high quality models to serve as the basis of the mixture, we manage to perform Bayesian learning, that is uncertainty quantification for non-asymptotic sample size circumstances, without the potential computational expense of encoding and sampling the entire DBN representation at once.

This formulation leads to a class of problems of contemporary interest in Bayesian Statistics described as Generalized Variational Inference (GVI), e.g. [7]. GVI presents a novel probabilistic approach to considering Bayesian statistical models that generalizes classical Variational Inference to consider

*These authors contributed equally

the use of alternative score functions as well as a generic set of probability distances and divergences. GVI was found to outperform classical Bayesian methods for circumstances of incorrect or imprecise model and prior specification.

As this statistical method is new while at the same time quite general, few algorithms and procedures have been developed for it. Recently the work [6] appeared which presented a comprehensive method for a very similar Coherent GVI (CGVI) problem, with theoretical convergence guarantees and promising illustrative performance across a variety of problems and choice of divergences. The procedure transforms the measure valued optimization into a low dimensional convex (however, non-Lipschitz) problem and high dimensional sampling.

In order to simplify the computational task of sampling from high dimensional but low sample settings, we make the simplifying assumption of a linear Structural Equation Model (LSEM). Formally, we have a state vector X_t which evolves as,

$$X_t = X_t W + X_{t-1} A_1 + ... + X_{t-p} A_p + Z \tag{1}$$

where W is the adjacency matrix of an acyclic graph for intra-slice edges and A_i for inter-slice edges, respectively, and Z is the noise. Consider the availability of data in the form $\{X_{m,t}\}$ with $m = 1, ..., M$ independent samples of trajectory length $t = 1, ..., T$ of dimensionality $X_{m,t} \in \mathbb{R}^d$. This is a popular representation for a DBN, with the sparsity (zero and nonzero) structure of the adjacency matrices corresponding to the presence or lack of edges, thus defining the graph structure of the DBN transitions.

We now proceed to describe the major components of the approach and details of the procedure.

2 Empirical Bayesian Model and Prior Estimate with MIP

Learning BNs has a natural association to Integer Programming (IP) and other combinatorial solvers, given the extensive rich combinatorial structure [3]. However, in many cases of interest, the data are high-dimensional and yet we are limited as far as quantities of data samples. In such circumstances, enforcing sparsity becomes necessary to establish meaningful models. At the same time, the statistical regime is not particularly favorable for recovery of an exact sparse solution [14]. Thus, a discrete programming solution could be too rigid in its ability to fit a wide range of problem instances.

Empirical Bayes presents an approach that attempts to incorporate the advantages of uncertainty quantification of estimates that comes with Bayesian data analysis while mitigating the risks associated with misspecified priors. The procedure is simple: perform standard frequentist likelihood estimation to obtain a point estimate for the model, then subsequently design a prior that is centered on the estimate and includes a ball of uncertainty defined by probability distance. See, e.g. [2] and [4] for comprehensive monographs.

In standard Bayesian approaches, one would present an explicit prior for (Θ, Ξ) as arising from some external parametrically defined distribution, that is $(\Theta, \Xi) \sim p(\Theta, \Xi; \eta)$ for some parameter η. A posterior estimate is computed with $p\left((\Theta, \Xi)|\{X_{n,t}\}, \eta\right) \propto p\left(\{X_{n,t}\}|(\Theta, \Xi)p(\Theta, \Xi; \eta)\right.$. Typically, however, there is no reason to suspect that η is known. In a fully Bayesian approach we would assign a hyperprior distribution $p(\eta)$ and require integration, i.e. treating η as a "nuisance parameter", by $p\left((\Theta, \Xi)|\{X_{n,t}\}\right) \propto \int p\left(\{X_{n,t}\}|(\Theta, \Xi)p(\Theta, \Xi; \eta)\right) p(\eta) d\eta$. This presents the need for significant computation. Furthermore, with a poorly specified hyperprior $p(\eta)$, especially in the small-sample case we are interested in here, could have an outsized negative influence on the accuracy of the posterior for (Θ, Ξ). With empirical Bayes, one instead computes an estimate of $\hat{\eta}$ from the data, and uses this to subsequently develop the full model. Note that it is also possible to keep the prior $p((\Theta, \Xi))$ non-parametric, however, in [2] it is noted that parametric estimates of the hyperprior perform better, especially in the far from asymptotic case. For instance one can use the maximum posterior estimate $\hat{\eta}$ given the data, and use $\delta_{\hat{\eta}}$ as the hyperprior distribution for η, obviating the need for computing a complicated integral.

We specify a *data driven robust prior* (also referred to as a Parzen window), as inspired from distributionally robust optimization, which has been observed to yield favorable generalization properties for machine learning models [13]. To this end we consider that that true Bayesian model corresponds to a mixture of structures together with an uncertainty ball centered on a particular parameter set.

To this end, we present some important notation:

$$
\begin{aligned}
&X_{n,t} \in \mathbb{R}^d, \text{ Data sample of } n'th \text{ trajectory and } t \text{ time instance} \\
&\Theta_m \in \mathbb{R}^{d(\Xi_m)},\ \Xi_m = (\Xi_m^W, \Xi_m^A) \in \{0,1\}^{d\times d}, \{0,1\}^{d\times d\times p},\ m\text{'th model decision variables} \\
&\mathbf{S}(\{X_{t,n}\}_{N,T}, S) = \mathcal{S} := \{X_{i,t}\}_{i=n^1,\ldots,n^S, t\in[T]},\ \{n^1,\ldots,n^S\} \sim \mathcal{U}\left\{ \begin{array}{c} N \\ S \end{array} \right\} \\
&(\Theta, \Xi) = MIP(\{X_{n',t}\}),\ \text{the MIP solution operation on the data } \{X_{n',t}\}_{n'\in\mathcal{S}}
\end{aligned}
\tag{2}
$$

Here $\mathbf{S}$ represents the uniform sub-sampling operator. We can now present the generic form of the source of our initial frequentist estimate.

2.1 Integer Programming Estimates

In this case, we perform hierarchical inference based on the mixed nonlinear integer programming formulation defined in [11] (see also [9], which presents a relaxation technique using SOCPs for solving large instances).

To this end the DBN model incorporates a directed acyclic graph (DAG) $\bar{\mathcal{G}} = \mathcal{G}(\mathcal{V}, \mathcal{E}_W) \cup \mathcal{G}(\mathcal{V}, \mathcal{E}_A)$ which defines the present connections in the model. That is $e = \{X_1, X_2\} \in \mathcal{E}_W$ if $W_{1,2} \neq 0$, that is 1 is a parent of 2 in $\mathcal{E}_W$.

We present the IP problem for solving $(E_W, E_A) \in \left[\{0,1\}^d \times \{0,1\}^d\right] \times \left[\{0,1\}^d \times \{0,1\}^d\right]^p$, below. Note that with integer variables, one can induce a regularization of, effectively $\|W\|_0$, rather than the one norm as previously defined in the one-shot formulation. See the discussion on [9, pg 6] indicating that for DAGs, the relaxed $l1$ formulation is often inadequate to enforce sparsity.

$$
\begin{aligned}
\min_{(E_W, E_A, W, A)} \quad & \mathbf{E}(E_W, E_A, W, A; \tilde{\mathcal{S}}) + \lambda_W \|E_W\|_0 + \lambda_A \|E_A\|_0 \\
& := \sum_{n=1}^{\tilde{N}} \sum_{t=1}^{T} \sum_{i=1}^{d} \left([X_{n,t}]_i - \sum_{j=1}^{d} W_{j,i} [X_{n,t}]_j \right. \\
& \left. - \sum_{l=1}^{p} \sum_{j=1}^{n} A_{l,j,i} [X_{n,t-l}]_j \right)^2 + \lambda_W \sum_{i,j} [E_W]_{i,j} + \lambda_A \sum_{l,i,j} [E_A]_{l,i,j} \\
\text{s.t.} \quad & W \cdot (1 - E_W) = 0, \\
& DAG(E_W), \\
& (E_W, E_{\tilde{W}}) \in \left[\{0,1\}^{d^2}\right] \times \left[\{0,1\}^{d^2}\right] \\
& (E_A, E_{\tilde{A}}) \in \left[\{0,1\}^{d^2}\right]^p \times \left[\{0,1\}^{d^2}\right]^p \\
& W \in \mathbb{R}^{d\times d},\ A \in \mathbb{R}^{p\times d\times d}
\end{aligned}
\tag{3}
$$

Thus, we compute the initial set of frequentist point estimates for the structure and parameters $\{\Theta_m, \Xi_m\}$ by:

Algorithm 1: Initial Integer Programming Solutions

for $m = 1, \ldots, M$ **do**
  Sample $\mathcal{S}^m := \{X_{n_i^m,\cdot}\} \sim \mathbf{S}(\{X_{n,\cdot}\}, S)$, $n_i^m \in [N]$
  Solve (3) with $\tilde{\mathcal{S}} \to \mathcal{S}^m$ to obtain $(\Theta_m, \Xi_m) := ((W^m, A^m), (E_W^m, E_A^m))$, representing the optimal parameters and structure the IP solver found for subsample m.
end for

There are other approaches that can also generate a set of, rather than one, model structure, e.g. [10]. We aimed to explicitly limit the number of structures while encouraging variety in the strategy proposed here.

3 Generalized Variational Inference Optimization Problem

Consider now that we have obtained a set of M candidate Dynamic Bayesian Network structures and their associated parameters $\{\Theta_m\}_{m=1,\ldots,M}$. In order to define our empirical Bayes approach, we define

a *Coherent Generalized Variational Inference* (CGVI) problem, which takes the form [6],

$$\min_{P\in\mathcal{D}} \mathbb{E}_P[\mathbf{E}_N(\Theta)] \tag{4}$$

wherein P is a probability measure on a set of paremters Θ, $\mathcal{U}$ is a set of admissible measures. Finally $\mathbf{E}_N$ is a loss function on the N data samples with a model parametrized by Θ, that is

$$\mathbf{E}_N(\Theta,\Xi) = \mathbf{E}(\Xi_W,\Xi_A,\Theta_W,\Theta_A;\mathcal{S}) \tag{5}$$

where we shall sometimes suppress the structure in the case of clear context.

3.1 Background

The determination of $\mathcal{D}$ is defined using ϕ-divergences, a natural measure for probability distances often used for risk measures. Defining a utility functional $\mathcal{U}$ on a space of random variables $\mathcal{M}$, with the disutility functional $\mathcal{V}(X) = -\mathcal{U}(-X)$ a divergence is $\Phi(p_\Pi) = \mathcal{V}^*(p_\Pi)$ where p_Π is the density of the prior measure $\Pi(\Theta)$ and $\mathcal{V}^*$ refers to the Fenchel conjugate of $\mathcal{V}$.

The application of duality yields the two-dimensional convex problem:

$$\min_{\lambda\geq 0,\mu\in\mathbb{R}} \{\mu + \epsilon\lambda + (\lambda\Phi)^*(E_N - \mu)\} \tag{6}$$

with $(\lambda\Phi)^*$ being the Fenchel conjugate of $\lambda\Phi$.

Here E_N is computed to be the loss value corresponding to an i.i.d. sample, and in the original there is a summation $\sum_s (\lambda\Phi)^*(E_N^s - \mu)$ over such samples indexed by s. However, in our case of Empirical Bayes, the distribution Π will be the MLE arising from the solution of the IPs above.

In our case we will use the Rényi divergence, defined as,

$$D_\alpha(p||\pi) \triangleq \frac{1}{\alpha-1}\log\mathbb{E}\left[p^\alpha\right]$$

for some $\alpha > 0$. Note that when $\alpha = 1$ this recovers the KL divergence. Letting $\beta = \frac{\alpha}{\alpha-1}$, this divergence is associated, through the Orlitz conjugate, with the isoelastic disutility function,

$$v(x) = \left(1 + \frac{x}{\beta}\right)^\beta - 1$$

With a solution $\bar{\lambda}$, $\bar{\mu}$ to (6), the CGVI density associated with the solution is given by:

$$p(\theta) = \left(1 + \frac{\mathbf{E}_N(\theta) - \bar{\mu}}{\beta\bar{\lambda}}\right)^{\beta-1} \tag{7}$$

Which we can re-write in Gibbs form, to facilitate sampling with Langevin and Hamilton Monte Carlo,

$$p(\theta) = \exp\left[\log\left\{\left(1 + \frac{\mathbf{E}_N(\theta) - \bar{\mu}}{\beta\bar{\lambda}}\right)^{\beta-1}\right\}\right] = \exp\left\{(\beta-1)\log\left[1 + \frac{\mathbf{E}_N(\theta) - \bar{\mu}}{\beta\bar{\lambda}}\right]\right\} \tag{8}$$

3.2 CGVI Formulation of Dynamic Bayesian Network Learning

Now that we have introduced the general CGVI problem, we can proceed to describe how we intend to incorporate it into our general Empirical Bayes procedure.

Recall that we have obtained frequentist estimates for the structure and parameters $\{\Theta_m, \Xi_m\}$ to serve as a pivot for the Empirical Bayes. To this end, we consider that we seek a solution to a hierarchical Bayesian problem. In particular we define a mixture $\mathbf{m} \in \Delta$ that samples from the M structures. Then, we shall see that we can treat the rest as a set of M uncoupled CGVI problems that sample the distribution of weights Θ for the m'th model

Define the loss function over the entire dataset, as a function of a vector of parameters:

$$\mathbf{E}_N(\phi^m(\theta)) := \mathbf{E}_N(\Xi_m, W(\phi_W^m(\theta)), A(\phi_A^m(\theta))) = \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{i=1}^{d}\left([X_{n,t}]_i - \sum_{j=1}^{d} W_{j,i}[X_{n,t}]_j - \sum_{l=1}^{p}\sum_{j=1}^{d} A_{l,j,i}[X_{n,t-l}]_j\right)^2 \tag{9}$$

See that the structure Ξ_m significantly restricts the search space of (W, A). In order to set up the appropriate definitions below for sampling, we introduce the following maps:

$$\begin{aligned}
&\phi^m : \mathbb{R}^{s_m} \to (\mathbb{R}^{d\times d}, \mathbb{R}^{d\times d\times p}),\\
&\mathcal{I}_m = \text{supp}(\Xi_m) := \{(i,j) \in [E_A^m]_{i,j} = 1\} \cup \{(i,j) : [E_W^m]_{i,j} = 1\}\}\\
&s_m = |\text{supp}(\Xi_m)|,\\
&\phi^m(\theta) = (W, A), \text{ where,}\\
&\quad [W]_{i,j} = 0 \text{ when } [E_W]_{i,j} = 0,\ [W]_{i,j} = [\theta]_k, \text{ for some } k \leq s_m, \text{ otherwise}\\
&\quad [A]_{i,j} = 0 \text{ when } [E_A]_{i,j} = 0,\ [A]_{i,j} = [\theta]_k, \text{ for some } k \leq s_m, \text{ otherwise, uniquely}
\end{aligned} \tag{10}$$

The details (lexicographical ordering, etc.) we leave out. Importantly, we have that $\mathbf{E}(\phi^m(\theta))$ presents a loss function for $\theta \in \mathbb{R}^{s_m}$.

With all the notation in place, we present the augmented CGVI with mixture weights by:

$$\min_{P\in\mathcal{D},\mathbf{m}\in\Delta} \sum_{m=1}^{M} \mathbf{m}_m \mathbb{E}_{P_m(\theta^m)} \left[\mathbf{E}_N(\phi^m(\theta^m)\right] \tag{11}$$

where,

$$P = \prod_{m=1}^{M} P_m(\theta^m),\ \theta^m \in \mathcal{M}(\mathbb{R}^{s_m}) \tag{12}$$

that is, a distribution for $\{\theta^i\}_{i\in[M]}$ with each P_m independent. Here $\mathcal{M}(\cdot)$ denotes a distribution over the argument space.

The measure constraint is defined as $\mathcal{D} = \prod_{m\in[M]}^{\otimes} \mathcal{D}_m$ with,

$$\mathcal{D}_m = \{p(\theta) \in \mathcal{M}(\mathbb{R}^{s_m}) | D_\alpha\left(p(\theta) || \delta_{\Theta_m}\right) \leq \epsilon\} \tag{13}$$

Where δ_{Θ_m} is a delta distribution on $[\Theta_m]_{\mathcal{I}_m} = \{[W]_{ij} : [E_W]_{ij} \neq 0\} \cup \{[A]_{ij} : [E_A]_{ij} \neq 0\}$. Thus $\mathcal{D}_m$ is meant to ensure that θ, as sampled from distributions of posterior-maximizing weights for the potential function $\mathbf{E}_N$, is within a probability distance, the specific metric being defined by D_α, of full measure at the original empirical solution Θ_m, restricted to the dimensionality of s_m.

Due to the linearity of the expectation operator, we can see that we can perform the weight optimization and sampling offline for each structure, and then subsequently optimize the weight mixture. Formally:

$$\begin{aligned}
\tilde{\Theta}^m \in \arg \min_{P(\theta^m)\in\mathcal{D}_m} \quad & \mathbb{E}_{P(\theta^m)} \left[\mathbf{E}_N(\phi^m(\theta^m))\right]\\
\longrightarrow \mathbf{m} \sim \quad & \mathbf{m}_i = \frac{\exp\left\{-\mathbb{E}_{\theta^i\sim P(\tilde{\Theta}^i)}\left[\left(\mathbf{E}_N(\phi^i(\theta^i))\right)\right]\right\}}{\sum_{m=1}^{M} \exp\left\{-\mathbb{E}_{\theta^m\sim P(\tilde{\Theta}^m)}[\mathbf{E}_N(\phi^m(\theta^m))]\right\}}
\end{aligned} \tag{14}$$

Thus, we can perform the entire procedure, integer programming, sampling, and convex optimization, for every model $m \in [M]$, entirely independently and in parallel. Then, subsequently, we sample from the mixture as weighted by the marginal posterior for that mixture. Note that this would correspond to a noisy selection by the standard Bayesian Score Criterion as is standard for evaluating DBN structures [8]

4 Algorithms

4.1 Optimization

Now we discuss the specific procedures we use in order to solve the measure valued optimization problem,

$$\tilde{\Theta}^m \in \arg \min_{P(\theta^m)\in\mathcal{D}_m} \mathbb{E}_{P(\theta^m)} \left[\mathbf{E}_N(\phi^m(\theta^m))\right]$$

for each model m. For this, we directly apply the procedures introduced for CGVI in [6]. To begin with, taking the specific form of (6) in our case yields the dual problem

$$\min_{\lambda\geq 0,\mu\in\mathbb{R}} \left\{\mu + \epsilon\lambda + \lambda\left(1 + \frac{\mathbb{E}_{P(\tilde{\theta}^m)}\left[\mathbf{E}_N(\phi^m(\theta^m))\right] - \mu}{\lambda\beta}\right)^{\beta} - \lambda\right\} \tag{15}$$

Where $\beta < 0$ is some parameter. This is an optimization problem convex in two variables λ and μ. However, it is not locally Lipschitz, thus care must be taken as far as optimization. Precise well-conditioned methods must be emphasized, which is possible with the low dimension.

The definition of the objective in the CGVI problem involves an expectation overΠ, the prior distribution from which the divergence should be bounded, which in our case is δ_{Θ_m}, that is, the delta distribution centered at the IP parameter solution Θ_m. And so the optimization problem to solve in our case is, for all $m \in [M]$

$$\min_{\lambda \geq 0, \mu \in \mathbb{R}} \left\{ \mu + \epsilon\lambda + \lambda \left(1 + \frac{\mathbf{E}_N^m - \mu}{\lambda\beta} \right)^{\beta} - \lambda \right\} \tag{16}$$

where $\mathbf{E}_N^m = \mathbf{E}(\Xi_m, \Theta_m; \mathcal{S})$

4.2 Sampling

The posterior, given optimal $\bar{\lambda}$ and $\bar{\mu}$ can be sampled from the distribution:

$$p(\theta^m) = \left(1 + \frac{\mathbf{E}_N(\phi^m(\theta^m)) - \bar{\mu}}{\beta\bar{\lambda}} \right)^{\beta - 1} \tag{17}$$

and alternatively,

$$\begin{aligned} p(\theta) &= \exp\left[\log\left\{ \left(1 + \tfrac{\mathbf{E}_N(\phi^m(\theta^m)) - \bar{\mu}}{\beta\bar{\lambda}} \right)^{\beta-1} \right\} \right] \\ &= \exp\left\{ (\beta - 1) \log\left[1 + \tfrac{\mathbf{E}_N(\phi^m(\theta^m)) - \bar{\mu}}{\beta\bar{\lambda}} \right] \right\} \end{aligned} \tag{18}$$

Recall again that

$$\begin{aligned} &\mathbf{E}\left(\phi^m(\theta^m)\right) = \mathbf{E}\left(E_W^m(\theta^m), E_A^m(\theta^m), W^m(\theta^m), A^m(\theta^m); \{X^{n,t}\}_{n=1,\ldots,N, t=1,\ldots,T}\right) \\ &= \left(\sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{i=1}^{d} \left([X_{n,t}]_i - \sum_{j=1}^{d} [E_W^m(\theta^m)]_{j,i} W_{j,i}^m(\theta^m) [X_{n,t}]_j - \sum_{l=1}^{p} \sum_{j=1}^{d} [E_A^m(\theta^m)]_{l,j,i} A_{l,j,i}(\theta^m) [X_{m,t-l}]_j \right)^2 \right) \end{aligned}$$

where the multiplication of the binary and continuous variables $E_W \cdot W$, etc. is redundant, and just noted for presentation.

4.3 Complete Algorithm

For completeness, now we describe the full procedure, incorporating all of the components of the algorithm described above. This is defined as Algorithm 2.

The evaluation sample (19) is a multinomial with coefficients given by the components of $\mathbf{m}$. It can be understood as the following simple operation: $\rho \sim \mathcal{U}[0, 1]$, that is, a uniform random number between 0 and 1, then $m = \arg\min\{i : \sum_{j=1}^{i} [\mathbf{m}]_j \leq \rho\}$, and finally θ^m is one sample chosen uniformly at random from $\{\Theta_q^m\}_{q \in [Q]}$, the set of samples generated by model m at the optimal $\bar{\lambda}, \bar{\mu}$).

Acknowledgments

The authors would like to thank and Ondřej Kuželka for his suggestions and discussion on this work. This work has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101084642.

References

[1] Mark Bartlett and James Cussens. Integer linear programming for the bayesian network structure learning problem. *Artificial Intelligence*, 244:258–271, 2017.

[2] Bradley P Carlin and Thomas A Louis. *Bayesian methods for data analysis*. CRC press, 2008.

Algorithm 2: Empirical Bayes for Dynamic Bayesian Network Learning Algorithm

Input: Data with N trajectories of T time steps each $\{X_{n,t}\}$. Number of total subsamples M, initial values μ_0, λ_0, divergence parameter $\beta < 0$, distance bound ϵ

for $m = 1, ..., M$ **do**

Select data sub-samples as by (2) and Algorithm 1, that is,

$$\mathbf{S}(\{X_{n,t}\}_N, S) = \mathcal{S}^m = \{X_{i,t}\}_{i=n_m^1,...,n_m^S},\ \{n_m^1, ..., n_m^S\} \sim \mathcal{U}\left\{ \begin{array}{c} N \\ S \end{array} \right\}$$

Solve (3) with the data samples $\mathcal{S}^m = \{X_{i,t}\}_{i=n_m^1,...,n_m^S}$ to obtain the mixed-integer solution $\{\Theta^m, \Xi^m\}$

Let $\lambda = \lambda_0$ and $\mu = \mu_0$ and set `Converged` $\leftarrow$ *FALSE*

Compute $\mathbf{E}_N^m = \mathbf{E}(\Xi_m, \Theta_m; \mathcal{S})$

Solve the optimization problem,

$$\min_{\mu \in \mathbb{R}, \lambda \geq 0} \left\{ \mu + \epsilon\lambda + \lambda \left(1 + \frac{\mathbf{E}_N^m - \mu}{\lambda\beta} \right)^{\beta} - \lambda \right\}$$

to obtain (μ_m^*, λ_m^*)

Sample (17), using the entire dataset $\{X_{i,t}\}_N$ and given $\lambda, \mu = (\mu_m^*, \lambda_m^*)$. After a burn in period, return a set of Q samples $\{\Theta_1, \Theta_2, ..., \Theta_Q\}$

end for and return :

1. Set of Structures $\{\Xi_m\}$ and weights $\{\Theta_m\}$
2. Set of CGVI weight samples for each mixture $\{\Theta_q^m\}_{q\in[Q], m\in[M]}$
3. the mixture vector $\mathbf{m}^*$ as by (14)

Evaluate:

for $r = 1, ..., R$ **do**

Using a validation dataset $\mathcal{V} = \{X_{n,t}\}_{N',T}$, sample from:

$$\begin{array}{l} E^r = \mathbf{E}_V(\phi^m(\theta^m)) := \mathbf{E}(\Xi_m, \phi_W^m(\theta^m), \phi_A^m(\theta^m); \mathcal{V}) \\ ,\ (m, \theta^m) \sim \mathbf{m}^*,\ \{\Theta_q^m\}_{q\in[Q]} \end{array} \tag{19}$$

with $\mathbf{E}_V$ denoting the loss on the validation dataset $\mathcal{V}$ and $\{\Theta_q^m\}_{q\in[Q]}$ is the sample chain at the optimal (μ_m^*, λ_m^*) solution

end for

return : Evaluate the model based on the following values:

$$\left\{ \{E^r\}_{r\in[R]} \cup \{\mathbf{E}_V(\phi^m(\Theta_m))\}_{m\in[M]} \right\} \tag{20}$$

where recall that Θ_m is the m'th integer programming solution for weights.

[3] James Cussens, Matti Järvisalo, Janne H Korhonen, and Mark Bartlett. Bayesian network structure learning with integer programming: Polytopes, facets and complexity. *Journal of Artificial Intelligence Research*, 58:185–229, 2017.

[4] Bradley Efron. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press, 2012.

[5] Nir Friedman and Daphne Koller. Being bayesian about network structure. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 201–210, 2000.

[6] Aurya S Javeed, Drew P Kouri, and Thomas M Surowiec. A risk management perspective on statistical estimation and generalized variational inference. *arXiv preprint arXiv:2310.17376*, 2023.

[7] Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. Generalized variational inference. *stat*, 1050:21, 2019.

[8] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[9] Simge Kucukyavuz, Ali Shojaie, Hasan Manzour, Linchuan Wei, and Hao-Hsiang Wu. Consistent second-order conic integer programming for learning bayesian networks. *Journal of Machine Learning Research*, 24(322):1–38, 2023.

[10] Zhenyu A Liao, Charupriya Sharma, James Cussens, and Peter van Beek. Finding all bayesian network structures within a factor of optimal. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7892–7899, 2019.

[11] Hasan Manzour, Simge Küçükyavuz, Hao-Hsiang Wu, and Ali Shojaie. Integer programming for learning directed acyclic graphs from continuous data. *INFORMS journal on optimization*, 3(1):46–73, 2021.

[12] Herbert E Robbins. An empirical bayes approach to statistics. In *Breakthroughs in Statistics: Foundations and basic theory*, pages 388–394. Springer, 1992.

[13] Matthew Staib and Stefanie Jegelka. Distributionally robust optimization and generalization in kernel methods. *Advances in Neural Information Processing Systems*, 32, 2019.

[14] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

Causal Learning in Biomedical Applications: A Benchmark

Petr Ryšavý
Department of Computer Science
Czech Technical University in Prague
Karlovo namesti 13, Prague 12135
petr.rysavy@fel.cvut.cz
& $XiaoyuHe, JakubMareek$
AI Center
Czech Technical University in Prague
Karlovo namesti 13, Prague 12135
jakub.marecek@fel.cvut.cz

October 6, 2024

Abstract

Learning causal relationships between a set of variables is a challenging problem in computer science. Many existing artificial benchmark datasets are based on sampling from causal models and thus contain residual information that the R^2-sortability can identify. Here, we present a benchmark for methods in causal learning using time series. The presented dataset is not R^2-sortable and is based on a real-world scenario of the Krebs cycle that is used in cells to release energy. We provide four scenarios of learning, including short and long time series, and provide guidance so that testing is unified between possible users.

1 Introduction

Understanding causal models is important in a number of fields, from healthcare to economics, as it allows for precise forecasting and training of reinforcement learning algorithms. Learning causal models involves extracting potential non-linear relationships and dependencies between variables from sampled time series. For example, the modeling of biomarkers of non-communicable disease as a function of diet and action monitoring has shown the potential of being a powerful tool to guide the recommendations for a healthy diet.

Many researchers agree that there is a need for better synthetic datasets to test causal learning algorithms. This is supported by works as [30, 24, 31].

Many synthetic dataset benchmarks suffer from residual information in the data that the R^2-sortability can identify. In the case of real-world datasets, we often cannot be sure what the ground-truth causal relationships are. Often, datasets for causal discovery are too large, and as a result, they are sampled without any standardized sampling approach, thus making different papers using the datasets incomparable.

In this paper, we aim to fill this gap and provide a standardized synthetic dataset that does not suffer from the problems mentioned above. The dataset is based on simulating a set of chemical reactions describing the Krebs cycle, and for that, it uses a publicly available generator at [20]. The randomness in the data is caused by simulating the molecules in a box and providing the molecules with locations and velocities. Whenever molecules forming a left-hand side of a reaction meet, they are replaced with reactants as given by the equation.

First, we provide a brief review of a variety of methods that can be used in causal learning. Later, we provide a list of requirements that we can expect from such causal learning methods to illustrate their expressiveness. A discussion of which criteria are supported by the existing methods follows. Section 4 explains the dataset in detail and shows a possible evaluation of a method on the dataset. As an example, a state-of-the-art method named DyNoTears [25] is used. Next, we compare the presented dataset with other causal learning datasets. In conclusion, we give preference to public repositories where the dataset, as well as the source code for the evaluation of the method, can be found.

2 Preliminaries and Related Work

Learning most causal models involves solving NP-hard non-convex optimization problems. Just as there is "one" convex optimization and "many" non-convex optimization problems, there are many causal models and methods for learning them. Perhaps the most elegant approach to causal learning utilizes techniques from system identification.

System Identification and Linear Dynamic Systems (LDS) Let m be the hidden state dimension and n be the observational dimension. A linear dynamic system (LDS) $\mathbf{L}$ is defined as a quadruple $(\mathbf{F}, \mathbf{G}, \mathbf{\Sigma}, \mathbf{V})$, where $\mathbf{F}$ and $\mathbf{G}$ are *system matrices* of dimension $m \times m$ and $n \times m$, respectively. $\mathbf{\Sigma} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{m \times m}$ are covariance matrices [35]. A single realization of the LDS of length T, denoted $\mathbf{X} = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^{n \times s \times T}$ is defined by *initial conditions* ϕ_0, and realization of noises υ_t and ω_t as

$$\phi_t = \mathbf{F}\phi_{t-1} + \omega_t, \tag{1}$$

$$x_t = \mathbf{G}'\phi_t + \upsilon_t, \tag{2}$$

where $\phi_t \in \mathbb{R}^{m \times s}$ is the vector autoregressive processes with hidden components and $\{\omega_t, \upsilon_t\}_{t \in \{1,2,\ldots,T\}}$ are normally distributed process and observation noises with zero mean and covariance of $\mathbf{\Sigma}$ and $\mathbf{V}$ respectively, i.e., $\omega_t \sim N(0, \mathbf{\Sigma}) \in$

$\mathbb{R}^{m\times s}$ and $\upsilon_t \sim N(0, \mathbf{V}) \in \mathbb{R}^{n\times s}$. The transpose of $\mathbf{G}$ is denoted as $\mathbf{G}'$. Vector $x_t \in \mathbb{R}^{n\times s}$ is the observed output of the system. In non-linear dynamical systems, one replaces the multiplication $\mathbf{F}\phi_{t-1}$ with a function $f(\phi_{t-1})$. It is well known [36] that there are multiple, equivalent conditions for the identifiability of $\mathbf{F}, \mathbf{G}$, given by so-called Hankel matrices, conditions on the transfer function, or frequency-domain conditions, among others. The is also a recent understanding [33] of sample complexity of the problem.

Linear Additive Noise Models Throughout causal modeling, one wishes to learn a function f, which is known as the structural assignment map and is closely related to the f above. Under the assumption that the structural assignments are linear, noises $N_j, j = 1, \ldots, N$ are independently identically distributed (i.i.d.) and follow the same Gaussian distribution, or alternatively, noises $N_j, j = 1, \ldots, N$ are jointly independent, non-Gaussian with strictly positive density, one obtains linear additive noise models (ANM). In studying ANM, one may benefit from a long tradition of work on linear system identification. In particular, the identifiability of linear ANM can be reduced to the identifiability of linear dynamical systems (cf. Proposition 7.5 & Theorem 7.6 in [28]).

Bayesian networks Another classic example in causal learning are *Bayesian networks*, first introduced by Pearl in 1985 [26]. Bayesian networks are formed by a directed acyclic graph (DAG), where each vertex j represents a variable X_j, with edges going from one variable to another representing causal relationships. It is assumed that each variable X_j is independent of other variables but for its parents, PA_j in the DAG, thus allowing a compressed representation of the joint probability as

$$P(X_1, X_2, \ldots, X_N) = \prod_{j=1}^{N} P(X_j \mid \mathbf{PA}_j). \tag{3}$$

The most common approach to exact inference in Bayesian networks is the variable elimination algorithm [27]. Approximate inference algorithms are also often applied. The most common one is the Markov Chain Monte-Carlo (MCMC) algorithm that repeatedly samples from each variable conditioned on the values of its parents. The MCMC algorithm predates Bayesian networks and is often referred to as Gibbs sampling.

In relation to temporal data, the Dynamic Bayesian Networks (DBN) are a well-known extension [8, 19]. DBNs are defined by two Bayesian networks. The first defines the initial state, and the second is the transition model between t and $t + 1$, where nodes in layer t are assumed to be independent. The network can then be unrolled into length T so that each of the time slices for $t \geq 1$ is defined by the transition model.

Counterfactual Framework The counterfactual framework can be used to derive causality. This approach focuses on the question of which input variable

needs to change in order to change the output of a model. The counterfactual framework is connected with the calculation of interventions, i.e., assessing the change of output variables after a hypothetical change of an input variable. In counterfactuals, we ask which inputs need to change to observe a change in the output, while in intervention, we change the inputs to see the change in the output. The counterfactuals were introduced into Bayesian networks by Pearl [21]. Nowadays, their usage is broad, and they find usage in explainable machine learning models [18].

Granger Causality The goal of the Granger Causality [10] is to detect a causal effect of a time series on another time series. The Granger causality measures correlations between the effect series and shifted cause series, thus detecting a lag that represents the time needed for the cause to take shape. The method uses various statistical tests to detect whether adding a cause into a predictive model significantly improves the prediction capabilities of the model. The original paper [10] used linear regression as the testing predictor. Further modifications of the original paper followed and included non-linearity [37], learning from multiple time series [5], applications on spectral data, i.e., in the frequency domain [13], model-free modifications [7], and nonstationarity [32].

Instrumental Variables Instrumental variables can be used to infer causal effects when we cannot control the experimental setting. Suppose that we want to assess the causal effect of the explanatory variable E on the dependent variable D. Normally, we would try to do statistical tests on whether variable D changes when E changes. However, in many applications in medicine, economics, and others, this is not possible, as both E and D can have a common cause and be, therefore, correlated. This introduces bias in many statistical tests. To overcome the issue, we include a third variable, instrument variable I, which we can control and which has influence on D only through E. Then, we observe changes of D on I. When the applied predictor is linear regression, the predictor is a special case of a linear dynamic system [34]. The existence of a hidden state then allows the removal of the correlations stemming from a common, unobserved cause [34].

Instrumental variables are, however, concepts that can be used well beyond linear regression. Non-linear [22] and non-smooth [3] modifications exist. Sometimes, there is a requirement that instrumental variables might have common cofounders. This multilevel modeling is implemented in the instrumental variable toolkit by [15]. Similarly, [9] allows for a latent (hidden) variable.

Tractable Probabilistic Models The tractable probabilistic models (TPMs) are a large group of methods that can be used to model probabilistic distributions compactly in the spirit of neural networks approximating functions. A prime example of TPMs is sum-product networks (SPNs) [29], which represent the probability distribution as a DAG, where "input" random variables are assigned to leaves. Each non-leaf node corresponds to one of two operations, either sum or product. The weights of the edges are then used to learn the probability

distribution. The original paper [29] also proposed an algorithm to learn the structure using backpropagation and expectation maximization. The SPNs are only a subgroup in the broad class of probabilistic circuits [6]. The unified formalism allows using different types of nodes besides the sum and product nodes. Dynamic versions [17, e.g.] are able to work with temporal data.

See also Table 1 in the next section for an overview.

3 The Challenge

As suggested in the Introduction, we would like to learn causal models that are more expressive than many traditional models. In our view, the expressivity of the causal model entails:

- **quantitative** aspects of causality, also in order to simulate from the causal model
- **non-linear** aspects of causality
- **hidden states** (latent variables) of an *a priori* unknown dimension.
- At the same time, one would like to preserve as much explainability as possible, perhaps through targeted reduction [14].
- **cycles** in causal relationships
- **time-series** aspects, such as nonanticipativity and delays: clearly, causal relationships should be established between the cause in the past and the effect in the future, with some delay between the two.
- **mixture-model** aspects: clearly, there are variations between the metabolism in various individuals, perhaps due to genomic differences. One should explore joint problems [23], where multiple causal models are learned without the assignment of individuals to subgroups represented by the causal models given *a priori*.

The ability to simulate from the model entails:

- **quantitative** aspects of causality, in order to simulate from the causal model
- time required to simulate from the model scaling modestly (with the number of random variables and numbers of samples).

The ability to learn the model entails:

- **sample complexity**: number of samples required to build the model. Even simple models such as HMM comprise learning Gaussian mixture models, which are known to have high sample complexity.

Tool	Quantitative	Non-linear	Hidden st.	Cycles	Temporal	Mixture-models	Multiple trajectories	Structure learning in $\mathcal{P}$	Likelihood calculation in $\mathcal{P}$	Marginalization in $\mathcal{P}$	Simulation from the model
Causal Bayesian networks	✓	✓	✓	✗[1]	✓	✓	✓	✗	✓	✗	✓
Structural Equation Modeling	✓	✓	✓	✓	✓	✓	✓	✗	-	-	-
Counterfactual Framework	✓	✓	✓	✓[2]	✓	✓	✓	✗[3]	-	-	-
Granger Causality	✓	✓	✗	✗	✓	✗	✓	✓[4]	-[5]	-	-
Bayesian Structural Time Series Models	✓	✓	✓	✓	✓	✓	✓	✗[6]	-	-	-
Instrumental Variables	✓	✓	✓	✗	✓	✓	✓	✗	-	-	-
Tractable Probabilistic Models	✓	✗	✗	✗	✓[7]	✓	✓	✗	(✓)	(✓)	(✓)

Table 1: Summary of features of selected methods and frameworks.

- **time complexity**: time required to learn the model. Again, even HMM are [2, 16] cryptographically hard to learn in the setting where one has access to i.i.d. samples of observation sequences.

Let us discuss some of these in more detail.

Cycles Standard Bayesian networks do not normally support cycles between the variables. The causal relationships need to form a directed acyclic graph (DAG). Likewise, Granger causality does not support cyclic dependencies by definition. The Granger causality aims to find out whether one variable can be helpful in predicting the future of a second variable. As a result, we are detecting some time lag, that the second variable correlated with the first variable shifted to the future. To obtain a cyclic relationship, we would need a sequence of positive time lags that sum together to zero, which is not possible.

Under some circumstances, we can model cyclical relationships with Dynamic Bayesian networks (DBNs). For each variable, we have its realizations for time $t = 1, 2, \ldots, T$. As a result, DBNs can then be used to model situations, such as the one when X_t causes Y_{t+1}, Y_{t+1} causes Z_{t+2}, which in turn causes X_{t+3}. The overall graph is still a DAG, as there cannot be a cycle within a one-time slice, and neither can a variable have an effect on the past.

Hidden state and Mixture-models Modeling a hidden state in the model and sampling from the mixture of models are tightly connected, as the second can be reduced to the first. Suppose that we want to model a mixture of two

distributions. We can build two separate models for each of the distributions. Then, we introduce a hidden state that models a binary decision, whether we sample from the first or the second distribution.

Model learning When we are interested in the time complexity of model learning, the time requirements differ based on the techniques used. The Bayesian networks do not generally have exact polynomial-time learning. In Granger causality, the complexity of mining causal relationships depends on the algorithms and methods used. In the simplest scenarios, we can base the causal relationships on the F-test, which can be calculated in linear time, assuming that the cumulative distribution function of the Fisher–Snedecor distribution (F-distribution) is precomputed.

Non-linear dependencies In many cases, the possibility of having non-linear models is part of extensions of the original methods. A prominent example of such a method is Granger's causality. The original method was developed with linear dependencies between the features. But further extensions were developed to include nonlinearities, for example, [37]. In Bayesian networks, the original version [26] considered only propositional variables, but subsequent versions [12, e.g.] considered also continuous variables and non-linear dependencies.

4 The Benchmark

As the causal learning community matures, one would like to learn models that are more expressive (see above) and to learn them from datasets that go beyond toy examples.

In this paper, we present a simulated dataset based on the Krebs cycle. The Krebs cycle, also known as the citric acid cycle, is one of the fundamental pathways of biochemistry. The cycle, as illustrated in Figure 1, allows organisms that breathe to convert the energy stored in food to a key energy source (ATP) in muscle cells, for example. Such a cycle presents a natural example of time series that can be used to infer causal relationships between concentrations of the reactants.

4.1 The Data

Depending on the modeling of the time series, each of the reactions can be represented by one or more causal relationships. Our benchmark is based on a simulator in the GitHub repository at [20]. The simulator creates a virtual box with desired particles. The particles move inside the box, following the Boltzmann distribution. Once particles get close to each other, a pre-defined list of reactions is scanned to determine whether a reaction occurs, and if so, reactants are replaced with a product. The simulation continues, and concentrations of different particles are used as test time series. As a result, the time series contains

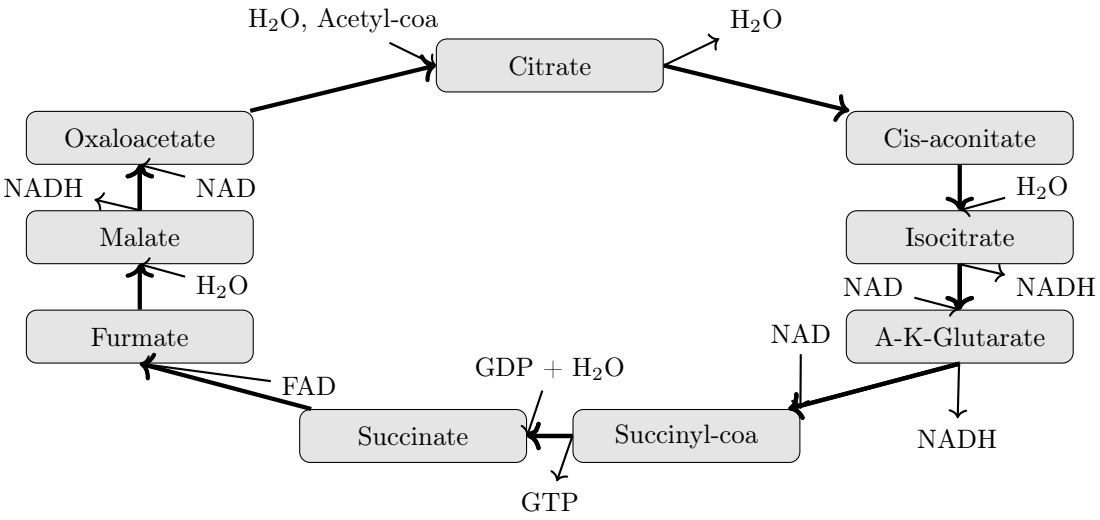


Figure 1: Illustration of the Kreb's cycle reactions used to simulate the concentrations.

noise (caused by the random location of particles), which is added to the locally linear behavior of the system.

In this way, we have generated four datasets, consisting of a time series with 5 to 5000 time steps and 16 features for the reactants, including 10 in the main cycle and 6 additional ones (incl. water). Each of the following datasets is based on simulating approximately 2500 molecules in the bounding box:

KrebsN contains 100 series with normally distributed prior distributions and absolute concentrations.

Krebs3 contains 120 series with relative concentrations, where for each triplet of the 10 main cycle reactants, we used uniform priors, and the remaining 7 particles were set to zero. Such a distribution is motivated by allowing the tested approaches to trace how the higher concentration of the three selected compounds move forward in the cycle.

KrebsL focuses on learning from a few long time series. In this case, we have 10 series with 5000 time steps. We use

KrebsS considers 10000 time series with only 5 time steps each, a complementary scenario to *KrebsL*.

The datasets are summarized in Table 2, showing the dimensions of the time series, the number of molecules used in the simulation, as well as other important features of the data.

Dataset	N. features	Lenght	N. series	Initialization	Concentrations
KrebsN	16	500	100	Normal distribution	Absolute
Krebs3	16	500	120	Excitation of three	Relative
KrebsL	16	5000	10	Normal distribution	Absolute
KrebsS	16	5	10000	Normal distribution	Absolute

Table 2: Summary of the datasets in the Krebs cycle.

4.2 Evaluation Criteria

For comparison, the dataset includes the ground-truth causal matrix as defined by the equations. Also, please note diagonal in Fig. 2a, which is there as well because the presence of a substance at time t implies presence of the same substance at time $t+1$. A single run of an algorithm produces a causal matrix that can be compared to the ground truth one. One of the simplest measures to compare the matrices is the structural hamming distance (SDH), that counts the number of edges that need to be added and the number of edges that need to be removed to convert the predicted causal graph into the ground-truth causal graph.

We propose as the main measure of the quality of the causal matrix to use the $F1$-score, which is the harmonic mean of the precision and recall measures. Let TP, TN, FP, FN be the true/false positive/negative measures as in a classification task. Then, the $F1$-score is defined

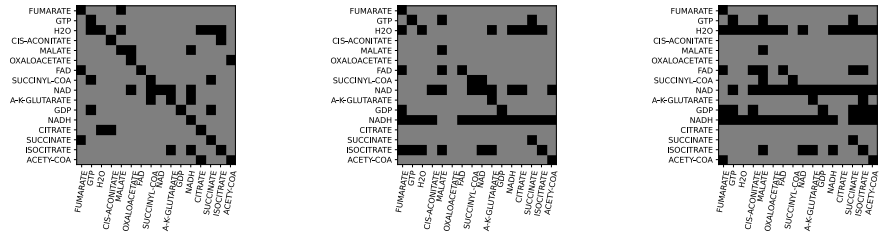
$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \tag{4}$$

$$\text{precision} = \frac{TP}{TP + FP}, \tag{5}$$

$$\text{recall} = \frac{TP}{TP + FN}. \tag{6}$$

Please note, that this measure can be easily extended to the case when the predicted causal matrix is stochastic. In that case, for example, an edge predicted with weight 0.3 when there is no ground-truth edge, contributes 0.3 to false-positive and 0.7 to true-negative.

To assess the stability of the method, we recommend to average the results over at least 10 runs of the method, whenever the tested method is randomized. The standard mean should then be calculated. In the case of deterministic methods, the stability of the $F1$-score cannot be evaluated by simple repeated evaluations followed by standard deviation calculation. Therefore, we recommend using an approach similar to cross-validation to show the stability of the results. In each evaluation, instead of plain restart, we can keep 10 % of the dataset aside to randomize data instead of the method. As a result, by doing repeated evaluations, it is possible to obtain the results' standard deviations and confidence intervals.



(a) Ground-truth (b) DyNoTears on *KrebsN* (c) DyNoTears on *Krebs3*

Figure 2: An illustration of the adjacency matrix produced by various methods and the ground truth matrix representing the set of reactions. Black squares represent 1 an edge in the adjacency matrix, grey 0.

4.3 A Baseline

To illustrate the dataset, we include results of the DyNoTears [25], a state-of-the-art method for causal discovery, implemented in the CausalNex [1] package. DyNoTears is provided with information that forbids edges within the same time slice, and the regularization parameter λ_a is selected from the list $10^{-6}, 10^{-5}, \ldots, 10^{6}$, so that the maximum F1-score is reached. Besides the F1 score, we also measured the time needed for structure learning. Figure 2 shows the adjacency matrix for various methods. We can see that as the F1-score is low, both datasets are challenging for causal discovery.

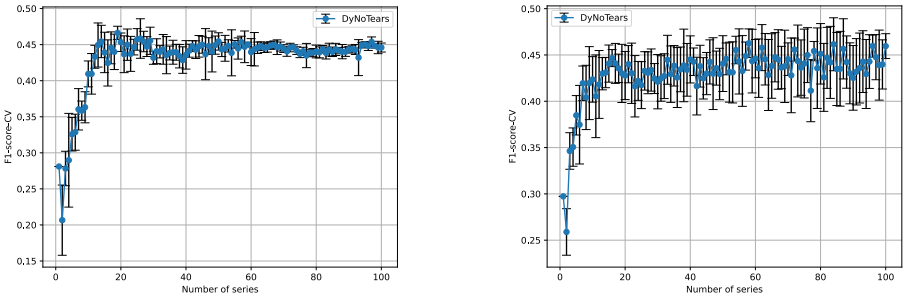
Figure 3 shows how the F1-score improves with the number of time series included in the evaluation. Similarly, Fig. 4 shows how the time requirements of the methods change with the number of time series.

From the results, we can see that the dataset is a major challenge for state-of-the-art identification methods, considering their F1-score is close to 0.5. Therefore, there is room for methods to improve the results further.

5 Discussion

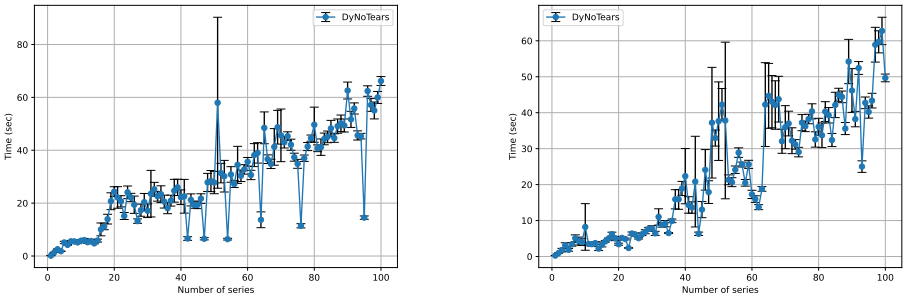
Once the dataset is presented, we are ready to compare it with other existing possibilities and show how it improves upon the other choices in [25, 11, 4]. We will point out the important advantages that the Krebs dataset has against other datasets.

Our method does not assume any ground truth structural model. Instead, our method uses an independent method of simulation from a real-world setting. The dataset is generated by following the chemical reactions in the Krebs cycle. This makes it possible to generate multiple variants (*KrebsN*, *Krebs3*, *KrebsS*, *KrebsS*) consistently. These consist of a time series with 5 to 5000 time steps and 16 features for the reactants, including 10 in the main cycle and 6 additional ones (incl. water).



(a) *KrebsN* (b) *Krebs3*

Figure 3: An illustration of the F1-score of various methods on the Krebs dataset. Please note that the implementation of DyNoTears in CausalNex is deterministic, thus providing the same result each time. To calculate the error bars, randomly selected 10 % of the data were put aside, and then results were averaged over 10 repeats of this procedure.



(a) *KrebsN* (b) *Krebs3*

Figure 4: An illustration of the time requirements of various methods on the Krebs dataset. The error bars show the standard deviation of the measurements calculated from 10 repeats.

Dataset	R^2-sortability	Standard variance	Note
krebsN	0.486	0.008	
krebs3	0.501	0.011	
krebsS	0.497	0.035	(please, see caption)
krebsL	0.492	0.005	

Table 3: The evaluation of R^2-sortability for individual time series. For each of the time series in each dataset, we calculated the R^2-sortability using the `CausalDisco` Python package [30, 31]. To obtain the results, the R^2-sortability values were then averaged over each of the datasets. Please note that for 11 time series in the *krebsS* dataset, the R^2-sortability method did not produce a numeric result. Since this is much less than 1 % of the dataset (and R^2-sortability is bounded by 0 and 1), the average won't be influenced substantially.

The presented dataset is not R^2-sortable. Our method does not suffer from the R^2-sortability issues other synthetic benchmarks suffer from, as explained by [24] and [31]. Indeed, Ormaniec et al. [24] argue that there are usually patterns left by the simulation from structural models that are easy to exploit. This can be quantified by the R^2-sortability [31]. To illustrate how the Krebs dataset stands compared to the R^2-sortability, we implemented a code evaluating the R^2-sortability for our dataset, the results of which can be seen in Table 3. Reference [31] then explains that *"0.5 means that ordering the variables by R^2 amounts to a random guess of the causal ordering"*, meaning that our dataset is not R^2-sortable. Thus, the fact that we do not assume any underlying framework makes our dataset more universal.

The ground truth causal relationships are known. At the same time, our method comes with widely accepted ground-truth data. The advantage can be seen when compared to datasets such as S&P100 (stock returns for 100 top US companies), used in DyNoTears paper [25]. S&P100 is a real-world dataset that suffers from an unclear ground truth causal matrix. Moreover, the authors had to ensure that the data were stationary, as concept drift is likely to happen in stock trading.

A similar situation is connected with the SACHS dataset [4]. This dataset contains single-cell measurements of levels of 11 proteins in immune cells. With 853 samples, the dataset is of a similar size to ours. However, we cannot be sure what the true causal relationships between the variables representing individual genes are in the case of expression data.

Prospect of perfect reconstruction At the same time, our method allows for the prospect of perfect reconstruction. Our dataset is much smaller than another commonly used causality dataset, the DREAM dataset [11]. This is desirable in connection with the fact that most of the problems in causal learning are NP-hard. Because of that, perfect recovery with many variables is computationally infeasible. *Causal discovery algorithms should be tested on*

smaller, easy-to-explain datasets first before proceeding to larger and more complex datasets. The use of larger datasets also brings another reproducibility problem – sampling, often done in an ad hoc, paper-specific fashion – which is not needed with our data.

6 Conclusion

We publish all source files used to generate the data and the figures in this paper in the following GitHub repository `https://github.com/petrrysavy/krebsdynotears`. The repository also contains numeric results that were generated as input to the plots. The generator of the data can be found at `https://github.com/petrrysavy/krebsgenerator`, including a description of how to generate the benchmarking data. The generator is based on a simulator at [20]. The dataset is available at `https://huggingface.co/datasets/petrrysavy/krebs/tree/main`.

References

[1] Paul Beaumont, Ben Horsburgh, Philip Pilgerstorfer, Angel Droth, Richard Oentaryo, Steven Ler, Hiep Nguyen, Gabriel Azevedo Ferreira, Zain Patel, and Wesley Leong. CausalNex, October 2021.

[2] Avrim Blum, Merrick Furst, Michael Kearns, and Richard J Lipton. Cryptographic primitives based on hard learning problems. In *Annual International Cryptology Conference*, pages 278–291. Springer, 1993.

[3] Mehmet Caner and Bruce E Hansen. Instrumental variable estimation of a threshold model. *Econometric theory*, 20(5):813–843, 2004.

[4] Bertrand Charpentier, Simon Kibler, and Stephan Günnemann. Differentiable DAG sampling. In *International Conference on Learning Representations*, 2022.

[5] Yonghong Chen, Govindan Rangarajan, Jianfeng Feng, and Mingzhou Ding. Analyzing multiple nonlinear time series with extended granger causality. *Physics Letters A*, 324(1):26–35, 2004.

[6] Y Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. *UCLA. URL: http://starai. cs. ucla. edu/papers/ProbCirc20. pdf*, page 6, 2020.

[7] Richard A Davis, Pengfei Zang, and Tian Zheng. Sparse vector autoregressive modeling. *Journal of Computational and Graphical Statistics*, 25(4):1077–1096, 2016.

[8] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(2):142–150, 1989.

[9] Peter Ebbes, Michel Wedel, Ulf Böckenholt, and Ton Steerneman. Solving and testing for regressor-error (in) dependence when no instrumental variables are available: With new evidence for the effect of education on income. *Quantitative Marketing and Economics*, 3:365–392, 2005.

[10] C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

[11] Alex Greenfield, Aviv Madar, Harry Ostrer, and Richard Bonneau. Dream4: Combining genetic and dynamic information to identify biological networks and dynamical models. *PLoS ONE*, 5, 2010.

[12] Reimar Hofmann and Volker Tresp. Discovering structure in continuous variables using bayesian networks. In *Proceedings of the 8th International Conference on Neural Information Processing Systems*, NIPS'95, page 500–506, Cambridge, MA, USA, 1995. MIT Press.

[13] Maciej Kamiński, Mingzhou Ding, Wilson A. Truccolo, and Steven L. Bressler. Evaluating causal relations in neural systems: Granger causality, directed transfer function and statistical assessment of significance. *Biological Cybernetics*, 85(2):145–157, Aug 2001.

[14] Armin Kekić, Bernhard Schölkopf, and Michel Besserve. Targeted reduction of causal models. *arXiv preprint arXiv:2311.18639*, 2023.

[15] Jee-Seon Kim and Edward W. Frees. Multilevel modeling with correlated effects. *Psychometrika*, 72(4):505–533, Dec 2007.

[16] Gaurav Mahajan, Sham Kakade, Akshay Krishnamurthy, and Cyril Zhang. Learning hidden markov models using conditional samples. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 2014–2066. PMLR, 2023.

[17] Mazen Melibari, Pascal Poupart, Prashant Doshi, and George Trimponias. Dynamic sum product networks for tractable inference on sequence data. In *Conference on Probabilistic Graphical Models*, pages 345–355. PMLR, 2016.

[18] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 607–617, New York, NY, USA, 2020. Association for Computing Machinery.

[19] Kevin Patrick Murphy. *Dynamic bayesian networks: representation, inference and learning*. University of California, Berkeley, 2002.

[20] August Nagro. Chemistry-engine. `https://github.com/AugustNagro/Chemistry-Engine`, 2015.

[21] Leland Gerson Neuberg. Causality: Models, reasoning, and inference, by judea pearl, cambridge university press, 2000. *Econometric Theory*, 19(4):675–685, 2003.

[22] Whitney K. Newey. Efficient instrumental variables estimation of nonlinear models. *Econometrica*, 58(4):809–837, 1990.

[23] Mengjia Niu, Xiaoyu He, Petr Rysavy, Quan Zhou, and Jakub Marecek. Joint problems in learning multiple dynamical systems. *arXiv preprint arXiv:2311.02181*, 2023.

[24] Weronika Ormaniec, Scott Sussex, Lars Lorch, Bernhard Schölkopf, and Andreas Krause. Standardizing structural causal models, 2024.

[25] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, pages 1595–1605. Pmlr, 2020.

[26] Judea Pearl. Bayesian netwcrks: A model cf self-activated memory for evidential reasoning. In *Proceedings of the 7th conference of the Cognitive Science Society, University of California, Irvine, CA, USA*, pages 15–17, 1985.

[27] Judea Pearl. Chapter 2 - bayesian inference. In Judea Pearl, editor, *Probabilistic Reasoning in Intelligent Systems*, pages 29–75. Morgan Kaufmann, San Francisco (CA), 1988.

[28] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

[29] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.

[30] Alexander Reisach, Christof Seiler, and Sebastian Weichwald. Beware of the simulated dag! causal discovery benchmarks may be easy to game. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27772–27784. Curran Associates, Inc., 2021.

[31] Alexander G. Reisach, Myriam Tami, Christof Seiler, Antoine Chambaz, and Sebastian Weichwald. A scale-invariant sorting criterion to find a causal order in additive noise models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2024. Curran Associates Inc.

[32] Ali Shojaie and George Michailidis. Discovering graphical Granger causality using the truncating lasso penalty. *Bioinformatics*, 26(18):i517–i523, 09 2010.

[33] Anastasios Tsiamis, Ingvar Ziemann, Nikolai Matni, and George J Pappas. Statistical learning theory for control: A finite-sample perspective. *IEEE Control Systems Magazine*, 43(6):67–97, 2023.

[34] Arun Venkatraman, Wen Sun, Martial Hebert, J. Bagnell, and Byron Boots. Online instrumental variable regression with applications to online linear system identification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016.

[35] Mike West and Jeff Harrison. *Bayesian forecasting and dynamic models*. Springer Science & Business Media, 2006.

[36] Jan C Willems, Paolo Rapisarda, Ivan Markovsky, and Bart LM De Moor. A note on persistency of excitation. *Systems & Control Letters*, 54(4):325–329, 2005.

[37] Axel Wismüller, Adora M. Dsouza, M. Ali Vosoughi, and Anas Abidin. Large-scale nonlinear granger causality for inferring directed dependence from short multivariate time-series data. *Scientific Reports*, 11(1):7817, Apr 2021.

References

[1] Marcus Kaiser and Maksim Sipos. Unsuitability of notears for causal graph discovery when dealing with dimensional quantities. *Neural Processing Letters*, 54(3):1587–1595, 2022.

[2] Vikram Krishnamurthy. *Partially observed Markov decision processes*. Cambridge university press, 2016.

[3] Sabina Leanti La Rosa, Maria Louise Leth, Leszek Michalak, Morten Ejby Hansen, Nicholas A. Pudlo, Robert Glowacki, Gabriel Pereira, Christopher T. Workman, Magnus Ø. Arntzen, Phillip B. Pope, Eric C. Martens, Maher Abou Hachem, and Bjørge Westereng. The human gut firmicute roseburia intestinalis is a primary degrader of dietary β-mannans. *Nature Communications*, 10(1):905, Feb 2019.

[4] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, pages 1595–1605. Pmlr, 2020.

[5] John Van der Hoek and Robert J Elliott. *Introduction to Hidden Semi-Markov Models*, volume 445. Cambridge University Press, 2018.