DISS. ETH NO. 19158

Effective Causal Analysis

Methods for Structure Learning and Explanations

A dissertation submitted to

ETH ZURICH

for the degree of
Doctor of Sciences

presented by

JEAN-PHILIPPE PELLET

Ing. info. dipl. EPF,
École polytechnique fédérale de Lausanne

born November 12th, 1982

citizen of St-Livres, VD

accepted on the recommendation of

Prof. Dr. Joachim M. Buhmann
Prof. Dr. Peter Widmayer
Dr. André Elisseeff

2010

Effective Causal Analysis
Methods for Structure Learning and Explanations

Doctoral Thesis

Jean-Philippe Pellet

March 12th, 2010 *(submitted)*
July 2nd, 2010 *(defended)*

EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE ZÜRICH
Pattern Analysis and Machine Learning Group

IBM ZURICH RESEARCH LABORATORY
Data Analytics and Business Optimization Groups

Supervised by
Dr. André Elisseeff
Prof. Dr. Joachim M. Buhmann

Co-referent
Prof. Dr. Peter Widmayer

The only immediate utility of all sciences, is to teach us how to control and regulate future events by their causes.

—DAVID HUME (1748), *An Enquiry concerning Human Understanding*

Development of Western science is based on two great achievements: the invention of the formal logical system (in Euclidian geometry) by Greek philosophers, and the discovery of the possibility to find out causal relationships by systematic experiment (during the Renaissance).

—ALBERT EINSTEIN (1953), Letter to J. S. Switzer

The three rules of the Librarians of Time and Space are: 1) Silence; 2) Books must be returned no later than the date last shown; and 3) Do not interfere with the nature of causality.

—TERRY PRATCHETT (1989), *Guards! Guards!*

ACKNOWLEDGMENTS

The work presented in this thesis would not have been possible without a number of favorable circumstances and supporting people close to me, which I want to acknowledge.

I would like to express my gratitude to my supervisor and friend, André Elisseeff, who convinced me to write a thesis in the first place and then provided continuous support and expert advice on many topics. My academic supervisor, Joachim Buhmann, deserves special thanks for his supervision of an external Ph.D. student, with the associated administrative and mentoring concerns of dealing with a free electron.

My life companion, Cécile Moret, deserves special gratitude and thanks for her continuous support, repeated encouragements, and absolute confidence; she never failed to provide me with welcome optimism when times were tough.

I would also like to thank my close colleagues in the Business Optimization group at IBM Research, in particular my good friends Richard Bödi and Ulrich Schimpel, who created an extraordinarily fun and motivating work environment, and made those years very enjoyable.

Finally, I am greatly indebted to my proofreaders for their very careful work, which considerably increased the readability of my text: Bradley Becker, André Elisseeff, and Jean-Daniel Pellet. Thank you!

ABSTRACT

This study is concerned with causal analysis, an approach to multivariate-data analysis that aims to identify the structure of the data-generating process to understand how external interventions influence the system. In particular, causal knowledge can help understand the data better by preventing erroneous interpretations (e.g., Simpson's paradox), providing causal diagnostics to explain observed values, and evaluating alternative scenarios and policies that affect the data distribution.

We study two key complementary aspects of causal analysis in depth: causal-structure learning and explanatory causal reasoning. The former deals with the computationally hard task of building a causal model; the latter uses the causal model to extract targeted contextual causal information for analysts.

Our contributions to the structure-learning task are novel algorithms which improve upon current state of the art in accuracy and time complexity. We deal with the two main causal contexts: when all potential confounders are observed and when hidden confounders may exist. We argue that the presented algorithms, thanks to their reduced complexity, can tackle a new range of large causal problems.

We then examine the causal-diagnostics task and present a new approach to evidence explanation: causal-explanation trees. Given some observed variables and a fully specified causal model, we show how to extract qualitative and quantitative information about how and why the system ended up in such a state.

In addition to continuous empirical evaluation of the presented algorithms on standard benchmarks, we present in a final chapter two real-world applications of (subsets of) the introduced approaches. We discuss how validation of causal methods is difficult without artificial data, and show pragmatic alternatives to causal-model specification in adverse conditions.

Keywords: causality, causal analysis, causal networks, structure learning, hidden variables, causal explanations.

RÉSUMÉ

Cette étude traite de l'analyse causale, une approche d'analyse de données multivariées dont le but est l'identification de la structure du processus générateur de données pour comprendre comment des interventions influencent le système. Plus spécifiquement, les informations causales aident à mieux comprendre les données en empêchant des interprétations erronées (par exemple, le paradoxe de Simpson), en établissant des diagnostiques causaux pour expliquer des valeurs observées, et en étant capable d'évaluer des politiques et scénarios alternatifs qui modifient la distribution des données.

Nous étudions en profondeur deux aspects complémentaires de l'analyse causale: l'apprentissage de la structure causale, et le raisonnement causal explicatif. Le premier traite de la tâche NP-difficile de construction d'un modèle causal; le second utilise le modèle causal pour extraire de manière ciblée des informations contextuelles dans le cadre d'une analyse causale.

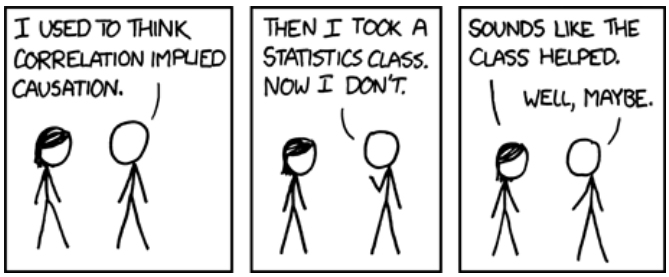
Nos contributions en matière d'apprentissage de structure sont des algorithmes novateurs qui dépassent l'état de l'art en précision et en complexité. Nous examinons deux principaux contextes causaux: lorsque toutes les variables causalement pertinentes sont observées, et lorsque des variables cachées parasites peuvent exister. Nous argumentons que les algorithmes présentés peuvent s'attaquer à une nouvelle dimension de problèmes causaux grâce à leur complexité réduite.

Ensuite, nous nous intéressons aux diagnostics causaux et présentons une nouvelle approche pour l'explication d'observations: les arbres d'explications causales. Étant donné certaines variables observées et un modèle causal complet, nous montrons comment extraire des informations qualitatives et quantitatives pour savoir pourquoi et comment le système a généré ces observations.

En plus de l'évaluation empirique continuelle des algorithmes sur des tests de performance standards, nous présentons dans un chapitre final deux applications réelles de l'analyse causale de données. Nous discutons de la difficulté de la validation des méthodes causales sans données artificielles et proposons des solutions pragmatiques pour la caractérisation de modèles causaux dans ces circonstances délicates.

Mots clés: causalité, analyse causale, réseaux causaux, apprentissage de structure, variables cachées, explications causales.

To André,
causal adept, functional enthusiast,
and master of suspicious persuasion.



NOTATION

X	Random variable, in this document simply called "variable"
$\mathcal{X}$	Domain (set of possible values) of variable X
x	Specific instantiation of variable X; $x \in \mathcal{X}$
X_i	Variable indexed for some purpose
$\mathbf{V}$	Set of variables *or* set of vertices
$\lvert\mathbf{V}\rvert$	Cardinality of set $\mathbf{V}$
$\mathbf{v}$	Specific instantiation of the set of variables $\mathbf{V}$
VARIABLE	Variable with a longer name *or* named node in a graph
NETWORK	Network, graph, or model name

$\mathrm{P}(X)$	Probability distribution of a single variable X
$\mathrm{P}(\mathbf{V})$	Joint probability distribution of a set of variables $\mathbf{V}$
$\mathrm{P}(X \mid \mathbf{Z})$	Conditional probability distribution of X given $\mathbf{Z}$

$\mathbf{X}$	Matrix
$\mathbf{X}^T$	Transposed matrix of matrix $\mathbf{X}$
$\mathbf{X}^{-1}$	Inverse matrix of matrix $\mathbf{X}$
(x_{ij})	Matrix where the (i,j)th entry is x_{ij}
x_{ij}	(i,j)th entry of matrix $\mathbf{X}$
$\mathbf{I}$	Identity matrix

$\mathbf{w}$	Column vector
$\mathbf{w}_i$	Column vector indexed for some purpose
$\mathbf{w}^T$	Transposed (row) vector of column vector $\mathbf{w}$
w_i	ith element of vector $\mathbf{w}$
a	Scalar

Usual meaning of some common variables in this document:

$\mathbf{V}$	Set of all variables in a given context
d	Number of variables (dimensionality) in a given context
n	Number of samples in a given context
$\mathcal{G}$	Graph representing a dataset

Table of Contents

CHAPTER

1

Introduction

THE FIELD OF MULTIVARIATE-DATA ANALYSIS can be seen as comprising two very different kinds of problems: on the one hand, problems of *diagnosis and classification*; on the other hand, problems of *causal inference and prediction* (Glymour & Cooper, 1999).

Let us explain this front-line distinction with a simple example. Suppose we are measuring the atmospheric pressure X with a barometer whose hand indicates a value Y. X and Y are closely related—we may even expect them to coincide if the barometer is perfect. In this case, let us assume that the barometer has a systematic bias b:

$$Y = X + b. \tag{1.1}$$

Knowing this, we can easily infer the actual pressure X from a reading Y:

$$X = Y - b. \tag{1.2}$$

Algebraically, these two equations are equivalent. Are there any reasons why we might give preference to one over the other? It turns out that (1.1) carries much more information than (1.2) once we know that it describes a *cause–effect relationship*. The property of such a relationship is that it is *invariant under manipulation* (Woodward, 2003): If we could manipulate the atmospheric pressure X, (1.1) would still be valid and accurately describe the response of the barometer Y. We find that (1.2), on the other hand, does not describe how the atmospheric pressure X would change if we forced the hand of the barometer to indicate another value Y.

In its simplest form, a deterministic cause–effect relationship from a cause X to an effect Y can be written as follows:

$$Y := f(X), \tag{1.3}$$

where the function f characterizes the process in Nature that generates Y from X, such as the innards of a barometer. We deliberately use the assignment operator ":=" to denote the asymmetry between X and Y. From a causal point of view, f is not a reversible function, and writing

$$X = f^{-1}(Y) \tag{1.4}$$

makes no sense. The function f can be seen as representing "Nature's thousands of little ears that constantly listen to X" (Pearl, 2008) and determine the value of Y—which is not reversible.

We agree that given certain conditions, it *is* relevant to study f^{-1}: the reason why we have barometers in the first place is to determine what the real pressure is, which we do by means of f^{-1} in (1.2). Such a reasoning holds as long as we are in a setting where there are *no interventions on the system*.

Actually, this is the setting of most traditional statistics and machine-learning problems. These are problems of regression, classification, clustering, density estimation, etc. These techniques all examine properties of the joint probability distribution of the variables. Conversely, if the joint probability distribution were fully known, most of these traditional problems could be solved optimally. (1.1) and (1.2) both describe the same joint distribution for X and Y. But statistical properties such as correlation, mean, or dependence are affected by the same problem as (1.2): they are not invariant under manipulation. If, in the middle of the data-collection process, an external experimenter decides to block variable Y at a certain value, regardless of its natural cause X, then statistical properties can change, relationships can be reversed, dependencies can be created or canceled. Relations like (1.2) are not valid any more, but causal relationships like (1.1) still are.[1]

Whether or not one believes that causal relationships are fundamental in Nature, it is hard to deny that our understanding of the world is linked to the cause–effect relationships we discover in it. The identification of the causes of an effect enables us, by favorable manipulation of those causes, to reach the desired effect, and similarly to prevent unwanted effects. As causal knowledge is valid under manipulation, it is portable across very different situations. Nearly everything we do each day reflects how we consciously or unconsciously use our knowledge of cause–effect relationships to deal with our environment. Early on in life, experimenting with these relationships is an everyday activity (Glymour, 2001)—crying long enough may get the parents' attention; putting a hand on a red stove leads to pain; hitting the cat repeatedly makes it flee or scratch.

Causal data analysis can thus be defined as the domain concerned with the identification and exploitation of such manipulation-invariant relationships.

⋆ ⋆ ⋆

As a rule, causal relationships cannot be inferred from a joint probability distribution alone. Actually, a joint probability distribution is just one possible "view" on the data-generating system; manipulating the system leads to another view: another joint probability distribution. Causal knowledge allows us to predict how this distribution changes. Today, the only widely accepted way to learn about causal relationships in science is to conduct a *randomized controlled experiment* (RCE). RCEs are commonly used to evaluate the effect of a given intervention (for example, a new treatment) on patients. RCEs are widely considered to be the most reliable way of finding scientific evidence because of their ability to get rid of spurious associations that are not due to direct causation (Lachin et al., 1988).

A typical RCE is conducted as follows: in a random sample of a population of interest, each individual is randomly assigned to the *test group* or to the *control group*. The test group gets

[1] Of course, if we force the hand of the barometer to move, then (1.1) is not valid any more either, as we are directly changing the data-generating mechanism. A cause–effect relationship is thus invariant under manipulation *of the cause*, not of the effect.

the treatment X while the control group does not (or receives a placebo instead). If the size of the sample is large enough, the randomization ensures that individuals in the test group have on average the same statistical characteristics as the ones in the control group so that the inter-group difference in any measured response variable Y can be causally attributed to X. It so avoids *confounding factors* (Bradford Hill, 1965): potentially unobserved variables which, influencing both assignment to the treatment X and the response variable Y, bias the estimate of the causal effect of X on Y.

Sometimes, however, depending on the variables involved, conducting RCEs can be costly, unethical, or plainly impossible. It is, for instance, hardly possible to provoke solar winds, yet we may still wish to determine their effect on the atmospheres and surfaces of planets. Is it then possible to infer causal relationships from non-experimental data? Can we, without making experiments, predict what an experiment would lead to? If we were all intelligent trees (Dummett, 1964), what could we learn about the world, being able to observe all phenomena around us but not to intervene on any of them? Or, provided we have identified a certain confounding factor for X and Y, can we correct our estimate of the effect of X on Y so as to remove the confounding effect?

This is not unrelated to traditional classification or regression, but is still fundamentally different. Figure 1.1 shows schematically the principles behind a traditional data-analysis task: given some data, the analyst postulates the existence of a joint probability distribution that generated the data, and tries to recover various properties of this distribution by inference (like mean, dependence, conditional distribution, etc.). In general, statistical assumptions are required, such as the shape of the distribution or independence of variables. Figure 1.2 shows the postulate behind causal inference: there is a data-generating system that, if let to evolve according to its own dynamics, has a "natural" distribution that generates *observational data*. If we intervene on the system in an experiment, other data with another distribution may be obtained. What we are looking for now is not the properties of the distributions, but the properties of the model: the manipulation-invariant causal relationships, which will be valid for all distributions. In order to legitimate the causal claims, in addition to statistical assumptions we need *causal assumptions* (Cartwright, 1994), which we discuss shortly.

Pearl (2008) proposes a clear-cut separation between statistical and causal concepts, arguing that those that cannot be learned from the joint probability distribution are by definition causal. Following this, we can classify some common concepts as shown in Table 1.1. Among the statistical concepts, we find *association*: mutual information and correlation, for instance, can both be computed from the joint distribution. It is equally possible to determine whether two variables are independent, or to compute the conditional distribution of a certain variable, given some observation. On the contrary, it is impossible to know whether a detected correlation between two variables is *spurious*, i.e., whether it is due to a causal connection between them or to a confounding factor. The joint distribution is equally unable to give us information about the effect

Causal	**Statistical**
Spurious correlation	Association
Randomization	Regression
Confounding	Independence
Effect	Conditioning
Explanatory variable	Predictive variable

Table 1.1: The causal-vs.-statistical classification of some concepts, following Pearl (2008).

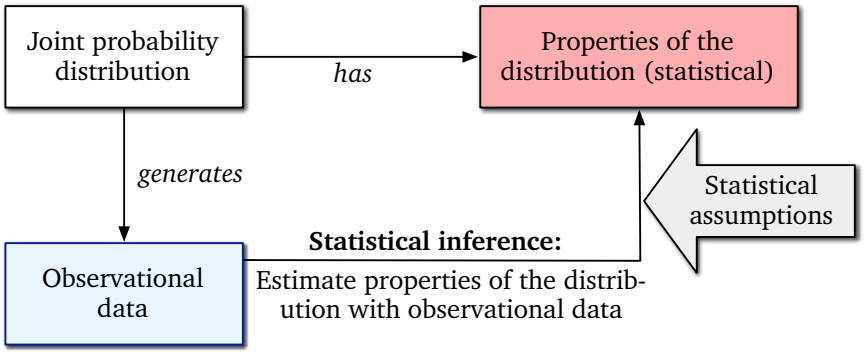


Figure 1.1: The traditional statistical-inference task postulates the existence of a hidden joint probability distribution, certain properties of which we try to recover from the data with certain statistical assumptions.

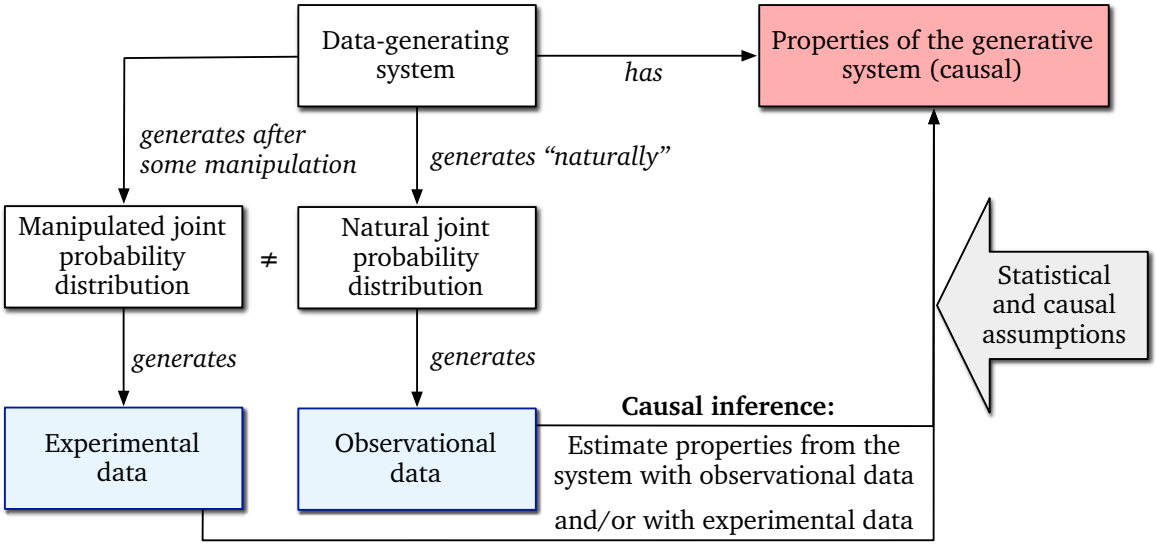


Figure 1.2: The causal-inference task postulates the existence of a hidden data-generating model, whose properties are manipulation invariant and can be investigated using both observational ("natural") data and experimental data (after a manipulation), with both statistical and causal assumptions.

of randomizing a certain variable, or to point to the best explanatory variable, given some observation.[2] Occasionally, statistical and causal concepts may coincide: a given regression coefficient may represent the causal effect of the associated predictor X on Y, or conditioning on a certain variable may get rid of a confounder. But characterizing exactly when two concepts coincide cannot be done with the joint distribution alone, and is thus itself a causal question, by definition.

Given the importance of causal concepts–both in our daily experience of how we interact with the world, and in many areas of science—it may be surprising to realize that until recently, there was no widespread tools or methods to systematically represent causal knowledge or use it in causal-reasoning tasks. The machine-learning community did investigate a related topic known as the *covariate-shift problem* (Shimodaira, 2000). In short, covariate-shift methods deal with the issue of predicting the value of a target variable Y given a set of predictors $\mathbf{X}$, where the model can only be trained from a joint distribution $\mathrm{P}(Y, \mathbf{X})$ but must predict Y in circumstances where the distribution $\mathrm{P}'(\mathbf{X})$ differs from the training-time distribution $\mathrm{P}(\mathbf{X})$. Notably, selection bias in experiments can be seen as a covariate-shift problem (Heckman, 1979). Although this looks similar to causal inference as described in Figure 1.2, covariate-shift methods usually assume that the conditional distribution $\mathrm{P}(Y \mid \mathbf{X})$ remains invariant, and thus cannot apply to cases where the dependencies between Y and any $X \in \mathbf{X}$ change.

⋆ ⋆ ⋆

A significant turning point in methods and tools for dealing with causal problems was the work by Pearl (2000); Spirtes, Glymour, & Scheines (2001); and Glymour & Cooper (1999). These three books describe principles and algorithms for causal-model specification all relying on a common *graphical notation* of the causal knowledge. In short, they visualize the causal structure of the multivariate problem by representing variables as *nodes* in a directed graph, and direct cause–effect relationships by *arcs* between the nodes.

While representing causal knowledge in a causal graph might seem like a simple idea, it has important ramifications. The graph can be given precise semantics that allows researchers to evaluate the effect of interventions on the data-generating system (Pearl, 1995; Valorta & Huang, 2006). The joint distribution together with the additional causal knowledge embedded in the graph are known as the *causal model*, and allow us to tell how the system will react to interventions—or to tell *why* the effect cannot be computed when the causal structure is partially known, by means of causal-identifiability criteria (Galles & Pearl, 1995; Shpitser, 2008).

The causal model is of central interest in the causal analysis. Algorithms and methods have been developed both to help to specify a full causal model from data, and to use an already-specified causal model to draw causal conclusions. The former domain is known as *causal-model selection*; we refer to the latter as *causal reasoning*. We show this schematically in Figure 1.3. Causal-model selection can further be seen as comprising two parts: causal-structure learning, and parameter learning.

The goal of *structure learning* is to construct the structure of the causal graph, which represents the data-generating system. This graph directly determines the causes and effects of each variable. A particularly interesting situation is when only observational data is available and experiments are impossible or too costly to perform. Thanks to the theoretical bases presented in studies such as Pearl (2000) and Spirtes et al. (2001), there are algorithms that can recover part of the causal

[2] We may disagree on the definition of "explanatory variable." We deal with this issue more extensively in Chapter 5.

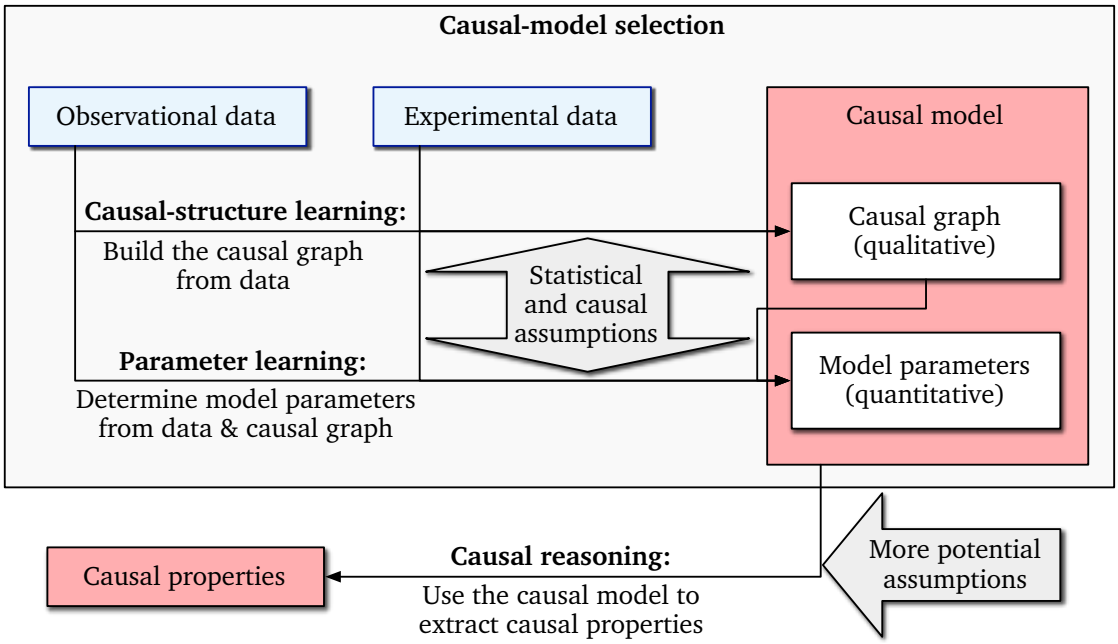


Figure 1.3: Data-based causal analysis seen schematically. First, a causal model is built, specified by a quantitative part (its graphical structure), and a qualitative part (the model parameters), obtained, respectively, with causal-structure-learning and parameter-learning methods. Then, causal reasoning uses the model to extract causal properties.

structure from observational data. Using independence relationships in the data, these algorithms usually return a graph pattern that matches several causal graphs compatible with the statistical evidence found in the data. However, it is in general impossible to recover the full causal structure from observational data alone.

Once a structure has been built, the role of *parameter learning* is to obtain the quantitative specification of the model. Given that each variable X is assigned a set of direct causes $\mathbf{Pa}(X)$ (the graphical parents of X) by the structure-learning step, there are two common possibilities to specify the parameters. The first one determines, for each node X, a conditional probability distribution

$$\mathrm{P}(X \mid \mathbf{Pa}(X)) \tag{1.5}$$

—this effectively turns the causal model into a *Bayesian network* (Pearl, 1988) with a causal interpretation of the arcs, if the causal graph is acyclic, and defines a joint probability distribution. The second possibility, much in the vein of our introductory example (1.1), finds a functional relationship

$$X := f_X(\mathbf{Pa}(X), U_X) \tag{1.6}$$

between X and its causes, allowing for a stochastic noise factor U_X. A set of functions of the type (1.6) is known as a *structural-equation model* (first formulated by Wright, 1921, and given formal causal interpretation by Pearl, 2000); given a distribution on the noise factors, it also defines a joint probability distribution. (Whether preference should be given to causal Bayesian networks or structural-equation models is often a matter of taste and personal experience of the analysts and of what kind of method will be applied to the full model.)

A full causal model, with structure and parameters, can then be used in various ways. Its primary purpose is to evaluate the effect of interventions (which can be seen as a "primitive" of causal models), but it can also be used for evaluating *counterfactual statements* (Lewis, 1973, 2000; Morgan & Winship, 2007). These are queries of the type "what is the probability that Y would be y if we had manipulated X to be x, given that we have observed $X = x'$ and $Y = y'$?" The causal-effect-evaluation primitive can also be used for more complex tasks, such as finding explanatory variables for certain observed evidence $Y = y$ (Halpern & Pearl, 2005).

⋆⋆⋆

This study is concerned with both structure learning and causal reasoning, in an effort to make the whole causal analysis more practical. On the one hand, it proposes a series of algorithms that extend the range of problems for which causal structures can be built automatically, and looks for ways to improve the performance of structure-learning algorithms, so that problems that could take days to solve can be tackled in a matter of minutes. On the other hand, it proposes advanced causal-reasoning approaches to extract and assess causal explanations.

This introduction is followed by Chapter 2, where we introduce the reader to the fundamentals of the statistical causality pioneered by Pearl (2000) and Spirtes et al. (2001). This chapter does not aim to be a comprehensive and detailed reference on this topic, but to provide an intuitive and motivated view of what the important underlying assumptions are, how some causal relationships can be detected in data despite the century-old adage that "correlation does not imply causation," and introduces the formalism based on probabilities used throughout the rest of the thesis.

Our contributions to structure learning are detailed in Chapters 3 and 4. They are divided into two chapters to reflect two kinds of approaches, based on whether or not these make the *causal-sufficiency* assumption. Causal sufficiency is a fundamental concept that postulates that the set of variables examined in the causal analysis includes all confounders; i.e., that there are no hidden common causes for two or more variables. Making this assumption allows us to derive more causal relationships from observational data, but leads to erroneous conclusions in the presence of hidden causes: spurious cause–effect relationships may be derived when the associations are actually due to hidden causes. Such mistakes make the model err when used to compute the effect of all interventions on causes or indirect causes of variables in the spurious relationships. Alternatively, not assuming causal sufficiency makes structure-learning algorithms try to detect spots where hidden causes may be—but usually results in graph patterns that match a larger set of possible causal graphs.

Chapter 3 is concerned with structure-learning algorithms that make this causal-sufficiency assumption. Basic principles and traditional algorithms are presented; and we then introduce novel algorithms related to learning *local neighborhoods*. We show how learning local neighborhoods with, for instance, feature-selection techniques dramatically reduces the time complexity of structure learning for sufficiently sparse causal graphs, while reaching accuracy similar to, or better than, traditional algorithms. This chapter is based on the JMLR paper *Using Markov blankets for causal-structure learning* (Pellet & Elisseeff, 2008*b*). This paper itself extends and generalizes techniques presented in the IDA paper, *A partial-correlation-based algorithm for causal-structure discovery with continuous variables* (Pellet & Elisseeff, 2007).

Chapter 4 deals with the more complex situation where we cannot assume causal sufficiency. We first describe the effect of ignoring certain variables according to their causal role in the graph,

explain what traditional algorithms do in these cases, and then discuss how neighborhood-based approaches of the type presented before can be generalized to handle lack of causal sufficiency. This chapter expands on the material presented in the NIPS paper, *Finding latent causes in causal networks: An efficient approach based on Markov blankets* (Pellet & Elisseeff, 2008*a*).

Testing causal structure-learning algorithms with real-world data can prove difficult, because the verification of the claims encoded in the returned result would usually require real experiments, manipulations, and new measurements (Guyon et al., 2008). This is in contrast to a more traditional regression or classification task, where a dataset is usually split into training, validation, and test sets, and can thus be cross-validated like this (Geisser, 1993). As the actual causal structure of most real problems is generally unknown, benchmarks are usually performed with artificial data, generated from a known structure. In these two chapters, we show how the novel algorithms perform with a range of benchmarks and comparisons with the existing state of the art.

We then switch to the topic of causal reasoning, and the particular case of *evidence explanation*. Chapter 5 shows what causal explanations are, why they are relevant, and how to obtain them. We argue that basing an explanatory analysis on causal considerations makes explanations intuitive and easily interpretable. Here we again describe some traditional approaches, present a new technique, and compare them all on a range of benchmarks. This chapter reuses material from the UAI paper, *Explanation trees for causal Bayesian networks* (Nielsen, Pellet, & Elisseeff, 2008).

Chapter 6 will detail two practical applications of causal analyses. These will confront causal approaches to the real world and help identify favorable situations in which causal considerations are beneficial, and point out some remaining shortcomings of these approaches. In particular, we find ourselves in circumstances where we cannot readily use traditional structure-learning algorithms, or where the observed variables are all effects of a hidden network of more abstract causes, and have to apply modified causal-reasoning techniques. One of these applications was published in the IBM Journal of Research and Development: *Causal networks for risk and compliance: Methodology and application* (Elisseeff, Pellet, & Pratsini, 2010).

Finally, Chapter 7 will conclude this thesis by taking a step back, reviewing the introduced material and evaluating how it contributes to a successful causal analysis in practice, and lists current open issues and research topics.

CHAPTER

2

A Statistical Approach to Causality

This chapter exposes the foundations of the statistical causal theory that the rest of this thesis relies upon. First, we motivate the need for causal analyses with an example that leads to apparently contradictory statistical conclusions. As we analyze it, we deal with various causal concepts mentioned in the introduction, such as causal relationships, interventions, and causal graphs. We then provide formal definitions and background, including graphs, Bayesian networks, and d-separation. We then explain what links cause–effect relationships and conditional-independence relationships together, and mention the assumptions and circumstances under which we may derive causal relationships. Finally, we define causal models and causal reasoning via the do-calculus, a set of rules for evaluating the effect of interventions.

2.1 SIMPSON'S PARADOX: OBSERVATION VS. INTERVENTION

We start this chapter with a motivating example that is a puzzling demonstration of the fundamental difference between observation and intervention, and which is often used as a justification of the usefulness of causal analysis.

2.1.1 A Motivating Example

This subsection is inspired from Pearl (2000).

Suppose a new kind of drug is developed for a particular kind of illness, and a study is conducted to determine whether or not the new drug is effective. For 40 males and 40 females (which we distinguish with the variable G for GENDER), we write down whether they were given the actual drug (using the variable D for DRUG; short: $D = d$) or a placebo ($D = \neg d$), and whether they eventually recovered (denoting RECOVERY with variable R: $R = r$) or died ($R = \neg r$). Table 2.1 shows the sum of the results for both males and females, and Table 2.2 shows the results for males and for females separately.

Both genders	Recovery ($R = r$)	Death ($R = \neg r$)	Sum	Recovery rate
Drug ($D = d$)	20	20	40	50%
Placebo ($D = \neg d$)	16	24	40	40%
Sum	36	44	80	

Table 2.1: Results of the example study combined for males and females.

Females ($G = f$)	Recovery ($R = r$)	Death ($R = \neg r$)	Sum	Recovery rate
Drug ($D = d$)	2	8	10	20%
Placebo ($D = \neg d$)	9	21	30	30%
Sum	11	29	40	

Males ($G = m$)	Recovery ($R = r$)	Death ($R = \neg r$)	Sum	Recovery rate
Drug ($D = d$)	18	12	30	60%
Placebo ($D = \neg d$)	7	3	10	70%
Sum	25	15	40	

Table 2.2: Results of the example study for males and females separately.

Let us try to interpret these tables by answering the question, "is the new drug effective?" From the combined table, we read that taking the drug makes the recovery rate go up from 40% to 50%. If we wanted to sell the drug, we could stop here and answer the motivating question by saying "Yes."

But looking at the females, we see that taking the drug actually *lowers* their recovery rate from 30% to 20%. We would then naturally expect the males to compensate for it and to recover drastically better when given the drug, which would be an intuitive explanation for the beneficial effect read from the combined table. Surprisingly, the drug also has a negative effect on the males as well, lowering their recovery rate from 70% to 60%.

There is something wrong in this study, although we chose 40 males and 40 females and gave the drug to 40 people with a control group of 40 other people. Formally, we have (with the shorthand notation where we write, for instance, $\mathrm{P}(r \mid d)$ instead of $\mathrm{P}(R = r \mid D = d)$):

$$\mathrm{P}(r \mid d) > \mathrm{P}(r \mid \neg d), \tag{2.1}$$

namely, the probability of recovery is higher if we take the drug. But conditioning on the gender reverses this relation:

$$\begin{aligned} \mathrm{P}(r \mid d, f) &< \mathrm{P}(r \mid \neg d, f) \\ \mathrm{P}(r \mid d, m) &< \mathrm{P}(r \mid \neg d, m), \end{aligned} \tag{2.2}$$

namely, if we know the gender, the probability of recovery is higher if we do not take the drug. Which of (2.1) and (2.2) shall we take as the correct characterization of the effect of the new drug?

Absurdly, if we use these probabilities to decide whether or not to give the drug to a new patient, we should give them the drug when their gender is unknown—as prescribed by (2.1)—but not give it to them as soon as we know their gender and can condition on it—as shown by (2.2). If we use these results as a tool to evaluate the sole effect of giving the drug, we are indeed in a paradoxical situation. If we want to solve this interpretation dilemma, we have to consider

more than the information offered by these conditional probability distribution or by the joint probability distribution.

One important fact that we can observe in the tables is that the males recover naturally a lot better than the females. The study was biased by giving the drug to more males than females, making the good recovery rate of the males more important when given the drug, and the bad recovery rate of the females less important in the final sum. Said differently, the gender was used as a *causal criterion* for giving the drug or not, so that the natural better recovery rate of males is used to make the use of the drug appear beneficial in the combined table.

Once we have identified this fact and understood this hidden relation between the variables, we feel that we *have* to condition on the gender to read the true effect of the drug, and that this drug is not beneficial for either a male or a female. The conditioning, by holding constant the value of the variable we condition on, inhibits its effect by holding it constant, too.

The problem is that there is no causal criterion that can tell us whether or not we should condition on the gender in the first place. Only our hidden assumptions and causal information we know about the experiment allow us to determine that it is the right thing to do in this circumstance. In order to demonstrate this, let us now conduct a very similar study, and replace the variable GENDER by BLOOD PRESSURE (B). We assume that the recovery also depends on the blood pressure, and that people with a high blood pressure ($B = h$) recover less than people with a normal blood pressure ($B = n$). Moreover, we know that the new drug influences blood pressure. Now suppose we obtain the same results as in Tables 2.2 and 2.1, BLOOD PRESSURE being substituted for GENDER. Is the conclusion of our study the same? Must we condition on B, just as we determined that we conditioned on G?

Our interpretation, this time, is that we must *not* condition on B. Conditioning holds B constant: we thus remove the effect of the blood pressure on the recovery rate. But if we know that the drug does causally influence the blood pressure as well, we will want to include its influence on the recovery rate in our assessment of whether or not the drug is effective: leaving it out would mean omitting all indirect influence "mediated" by our measurement of B and would not accurately describe the drug effect.

Statistically, the two studies are the indistinguishable: only causal considerations differentiate them. In the next subsection, we make this difference more explicit.

2.1.2 Decoding the Paradox: Observing vs. Intervening

The example we have just shown is an instance of what is known as *Simpson's paradox* (or *Yule-Simpson effect*), stated by Simpson (1951) and first described by Yule (1903). Roughly, it says that any statistical relation can be reversed by including more factors in the analysis. In our case, the change in recovery rate is reversed depending on whether we look at the combined tables or two tables separated by gender.

Formally speaking, it is not a paradox, because it does not lead to a contradiction. Nothing says in probability theory that such a statistical relation cannot be reversed. But it does appear counterintuitive to the common understanding, as illustrated by the question related to the example above of whether or not the drug has a beneficial effect. Actually, (2.1) and (2.2) are only contradictory if they are falsely interpreted to somehow express the causal effect of the drug: in this case, it absurdly seems that knowing the gender changes the effect of the drug.

Examining the example above, we have come to the conclusion that we should condition on the gender but not on the blood pressure. Why? Can we find a formal criterion that is able to justify and generalize this conclusion? How can we deal with more complex situations? Can we find a way to express causal effects with conditional probabilities without the risk of falling into apparent contradictions?

Central to this problem is the difference between observation and intervention. What is the difference between *observing* that a variable X has the value x and *intervening* on this variable, forcing it to have the value x? Why do we need this idea of intervention at all?

The problem is that we are trying to extract a causal interpretation (which has to do with interventions) using the calculus of conditional probabilities, which is purely based on observations. The well-known notation $\mathrm{P}(Y \mid X = x)$ gives the probability distribution of Y, given that we already know that $X = x$. It may differ entirely from the probability distribution of Y when some external agent affects variable X and gives it the value x. And this is what we are interested in, in this example: the "pure" effect of the drug on the recovery. Let us (for now, informally) denote the distribution of Y when X is assigned the value x by $\mathrm{P}(Y \mid do(X = x))$.[3] Now, whether $\mathrm{P}(Y \mid X = x) = \mathrm{P}(Y \mid do(X = x))$ holds depends on how X is generated in the first place: notably, if X and Y share a common cause, chances are high that $\mathrm{P}(Y \mid X = x)$ and $\mathrm{P}(Y \mid do(X = x))$ will be different. In our example, GENDER is such a common cause, and acts as *confounding factor*, whereas BLOOD PRESSURE is not. The important difference between evaluating an intervention and an observation is that intervening on X cuts it from the natural influence of its causes, whereas observing $X = x$ and conditioning on this also implicitly changes the distribution of the causes of X, which in turn may affect Y, as per the principle of belief propagation (Pearl, 1982).

If our drug study had been carefully designed as a randomized controlled experiment (RCE, as described in the introduction), patients would have been randomly assigned to the test group or to the control group, independently of their gender. This is exactly the goal of RCEs: randomize external factors so that they do not act as confounders. On average, we would have expected half of the males and half of the females to receive the drug.

In the next subsection, we show how with causal modeling and causal graphs we can correctly estimate the effect of the drug even with data that does not come from an RCE, and how, given the causal graph, we can determine probability distributions that involve *do*-expressions, such as $\mathrm{P}(Y \mid do(X = x))$, which denote interventions.

2.1.3 Graphical Approach of the Paradox

We first need some preliminary intuitive graphical notation to depict causal assumptions. To do so, we first provide the needed usual definitions around the concept of graphs. As we later deal with graphs that contain different kinds of edges, we explicitly model the allowed edge types.

Definition 2.1 (Abstract graph) *An* abstract graph *is an ordered pair* $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}_{\mathcal{E}} \rangle$ *with* $\mathbf{E}_{\mathcal{E}} \subset \mathbf{V} \times \mathcal{E} \times \mathbf{V}$*, where* $\mathbf{V}$ *is the vertices,* $\mathbf{E}$ *the edges, and* $\mathcal{E}$ *the set of allowed edge types of the graph. Concrete graphs provide a specification of what edge types* $\mathcal{E}$ *are allowed, we thus drop the subscript "*$_{\mathcal{E}}$*" when the kind of concrete graph is clear given the context.*

[3] We provide a formal definition for this notation in Subsection 2.3.3.

A type of concrete graph that we will use a lot is the directed graph.

Definition 2.2 (Directed graph) *A* directed graph *is a graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}_{\mathcal{E}} \rangle$, where $\mathcal{E} = \{\rightarrow\}$; i.e., there is only one possible edge type, "$\rightarrow$". We use the notation $X \rightarrow_{\mathcal{G}} Y$ or $X \leftarrow_{\mathcal{G}} Y$ as equivalent to, respectively, $(X, \rightarrow, Y) \in \mathbf{E}_{\mathcal{E}}$ or $(Y, \rightarrow, X) \in \mathbf{E}_{\mathcal{E}}$. We drop the subscript "$_{\mathcal{G}}$" when the context is clear.*

There can be several types of directed graphs. We are particularly interested in the acyclic ones. To define them, we need to know about paths and cycles.

Definition 2.3 (Path & cycle) *In a directed graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, a* path *of length $\ell > 0$ from X to Y is a sequence of $\ell + 1$ nodes $\boldsymbol{\pi} = (V_0, V_1, V_2 \cdots, V_\ell)$, $V_i \in \mathbf{V}$, $V_0 = X$, $V_\ell = Y$, such that:*

- *either: $\forall i : 0 \leq i < \ell : V_i \rightarrow V_{i+1}$ (then $\boldsymbol{\pi}$ is a* directed path*);*
- *or: $\forall i : 0 \leq i < \ell : V_i \rightarrow V_{i+1} \vee V_{i+1} \rightarrow V_i$ (then $\boldsymbol{\pi}$ is a* undirected path*).*

An (un)directed cycle *is an (un)directed path where $X = Y$.*

Definition 2.4 (Directed acyclic graph) *A directed graph $\mathcal{G}$ is said to be* acyclic *and is then also called a* directed acyclic graph *or* DAG *if and only if there is no directed cycle in $\mathcal{G}$.*

Given a graph, we can define useful node sets with respect to a particular node. We will use these definitions extensively to formulate precise graphical criteria or properties of a graph or node.

Definition 2.5 (Node sets) *In a directed graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, we define, for each node $X \in \mathbf{V}$:*

- $\mathbf{Pa}_{\mathcal{G}}(X) = \{Y \mid Y \rightarrow X\}$ *as the* parents *of X;*
- $\mathbf{Ch}_{\mathcal{G}}(X) = \{Y \mid X \rightarrow Y\}$ *as the* children *of X;*
- $\mathbf{Bd}_{\mathcal{G}}(X) = \mathbf{Pa}_{\mathcal{G}}(X) \cup \mathbf{Ch}_{\mathcal{G}}(X)$ *as the* boundary *of X, i.e., direct neighbors of X;*
- $\mathbf{An}_{\mathcal{G}}(X) = \mathbf{Pa}_{\mathcal{G}}(X) \cup \mathbf{An}_{\mathcal{G}}(\mathbf{Pa}_{\mathcal{G}}(X))$*, recursively defined as the* ancestors *of X;*
- $\mathbf{De}_{\mathcal{G}}(X) = \mathbf{Ch}_{\mathcal{G}}(X) \cup \mathbf{De}_{\mathcal{G}}(\mathbf{Ch}_{\mathcal{G}}(X))$*, recursively defined as the* descendants *of X.*

These notations are generalized to sets of variables, such that, for instance, with $\mathbf{X} \subset \mathbf{V}$, $\mathbf{Pa}_{\mathcal{G}}(\mathbf{X}) = \bigcup_{X \in \mathbf{X}} \mathbf{Pa}_{\mathcal{G}}(X)$, and $\mathbf{Pa}_{\mathcal{G}}(\emptyset) = \emptyset$. When the context is clear, we drop the subscript "$_{\mathcal{G}}$."

A causal graph is defined in terms of direct causal effects. It can be challenging to provide a formal definition of the presence of a direct causation between two variables (Woodward, 2003). First, a causal effect is direct only with respect to a given set of variables $\mathbf{V}$. One may see the barometer hand's movement Y as a direct causal effect of the atmospheric pressure X, or one may wish to have a finer-grained model that details the physical mechanisms happening in the innards of the barometer, in which case X does not directly cause Y any more. From now on, we will denote with $\mathbf{V}$ the set of variables considered in the causal analysis, and provide definitions in the context of $\mathbf{V}$. We will denote individual variables taken from $\mathbf{V}$ with X, Y, Z, W, etc.

Definition 2.6 (Direct causation) *In a context $\mathbf{V}$, X is said to* directly cause *Y, denoted by $X \twoheadrightarrow Y$, if there is an intervention on X that changes the probability distribution of Y when all other variables besides X and Y in $\mathbf{V}$ are held fixed at some values by interventions (Woodward, 2003).*

(Note the hollow-arrow notation $\twoheadrightarrow$, as opposed to the graphical arc $\rightarrow$.) By defining direct causation in terms of manipulations, we reiterate our understanding of cause–effect relationships as the manipulation-invariant properties of the data-generating system, as exposed in the introduction. With this definition, we can now define a causal graph.

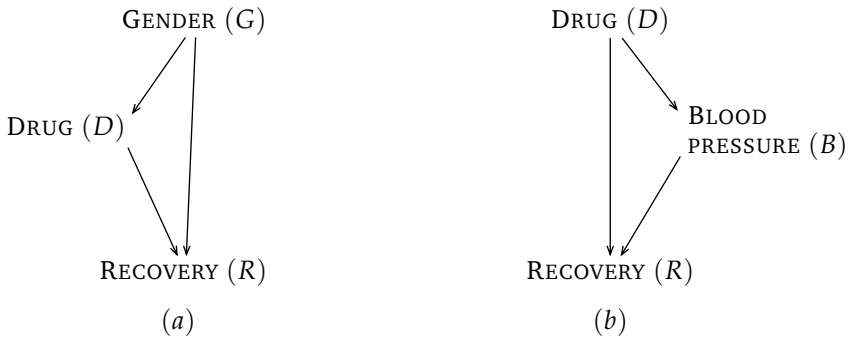


Figure 2.1: (a) Causal graph with node GENDER; (b) Causal graph with node BLOOD PRESSURE.

Definition 2.7 (Causal graph) *A* causal graph *is a DAG $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, where the nodes $\mathbf{V}$ are the variables of interest in a particular study, and where each direct causal effect between two variables X and Y is represented as a directed arc $X \to Y$ in $\mathcal{G}$."*

In a causal graph, we represent direct cause–effect relationships with arrows between the variables. In this definition, the graph is required to be acyclic; i.e., we assume that our system does not contain causal feedback loops, or that a variable cannot influence itself. Causal graphs containing cycles could be very useful in practice, in areas such as biology, where we commonly observe homeostatic phenomena, and the associated feedback mechanisms for regulating the "environment" in order to maintain a stable state (Marieb & Hoehn, 2006), but they vastly remain an open area for research (Richardson, 1996).[4]

Let us now revisit our motivating example with causal graphs. Figure 2.1 (a) shows the causal links in the original example, where DRUG, GENDER, and RECOVERY were measured; and Figure 2.1 (b) is the causal graph for the study where GENDER was replaced by BLOOD PRESSURE.

Figure 2.1 (a) shows an arc from G to D because we determined that in the study, the drug was given to more males than females, and thus that G was used to determine D. Had G been properly randomized, we would not expect the presence of this link.

In a causal graph, it is easy to see which variables are affected by an intervention on a variable X: they are $\mathbf{De}(X)$, all descendants of X. X first propagates its effects to its immediate children (by definition: the direct effects of X), which then do the same with their own children, etc. Now, we can evaluate the total causal effect of a variable X on another variable Y, which is, intuitively, the sum over the effect of every directed path between X and Y. In Figure 2.1 (a), this only includes the arc $D \to R$. In Figure 2.1 (b), we must add the chain $D \to B \to R$.

In this simple example, to answer the question "is G (or B) a confounder for the $D \to R$ causal effect", we can simply see whether or not Z lies on a causal path from D to R. If this is the case, then we should not condition: doing so would "block" the effect of the path on which Z lies. If Z does not lie on a causal path from D to R, then, we can condition on it—and *must* do so whenever Z is a confounding factor for D and R; i.e., a common cause, influencing both. The justification for this can be found in the discussion in Subsection 2.1.1: if we did not condition on the gender G

[4]Algorithms for structure-learning of cycles were introduced very recently (Lacerda et al., 2008), and the approach differs a lot from the approaches presented in this thesis, such that we decided to only deal with acyclic graphs. DAGs also have the important advantage to be able to be used as structure for Bayesian networks, presented in Subsection 2.2.2, which lets us use causal-reasoning techniques such as those presented in Chapter 5.

in Figure 2.1 (a), we would unwittingly include arc $G \rightarrow R$ in our assessment of the effect of D on R, which is not correct. Pearl (2000, p. 79) generalizes the concept of confounder and proposes the *back-door criterion*, a condition that, given a causal graph, characterizes what variables need to be conditioned on in order to obtain an unbiased estimate for a given causal effect.

In this section, we have presented causal graphs as a way to represent causal knowledge, and used them to better explain our example of Simpson's paradox. But we have not touched the topic of how such graphs can be constructed from data. In the next section, we formalize our approach, and show how and under which assumptions direct cause–effect relationships can be characterized from observational data.

2.2 Towards Causal Networks

How do we learn about cause–effect relationships using observational data, without expert knowledge? We remember the old adage that says that "correlation is not causation." There is no such thing as a statistical test of causation, and we have characterized causal relationships as properties that cannot be obtained from the joint probability distribution alone, so how can we say anything about the causal structure of a given system, given observational data?

To formalize our approach in the journey towards causal-structure learning, we need to learn about statistical independence and conditional independence. The reason for this is that although there is no test of causation, there are tests of (conditional) independence. These turn out to be the closest general criterion we can use to learn about causation in an observational study. Next, we also need to learn about Bayesian networks. Bayesian networks are to conditional independence what causal networks are to causation: a graphical representation which is given precise semantics. The concept of Bayesian networks is older than that of causal networks, and understanding their similarities and differences helps isolate the added value that causal networks bring to data analysis.

2.2.1 Independence and Conditional Independence

We recall the concepts of statistical dependence and independence.

Definition 2.8 (Independence) *Two random variables X and Y are said to be* independent*, denoted by $(X \perp\!\!\!\perp Y)$, if and only if:*

$$\mathrm{P}(X, Y) = \mathrm{P}(X)\mathrm{P}(Y). \tag{2.3}$$

X and Y are dependent*, denoted by $(X \not\perp\!\!\!\perp Y)$, if and only if they are not independent. This is straightforwardly generalized to sets of random variables* **X** *and* **Y***, substituting* **X** *and* **Y** *for X and Y in (2.3).*

The independence of two variables is a useful concept: if we know that X and Y are independent, then knowing X tells us nothing about Y, and conversely—independence is a completely symmetric relation. Independence can be generalized as *conditional independence*, defined as follows.

Definition 2.9 (Conditional independence) *Two random variables X and Y are said to be* conditionally independent *given a set of variables $\mathbf{Z}$ (where $X, Y \notin \mathbf{Z}$), denoted by $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$, if and only if:*

$$\mathrm{P}(X, Y \mid \mathbf{Z}) = \mathrm{P}(X \mid \mathbf{Z})\mathrm{P}(Y \mid \mathbf{Z}) \tag{2.4}$$

for all instantiations $\mathbf{z}$ of $\mathbf{Z}$ such that $\mathrm{P}(\mathbf{z}) \neq 0$. X and Y are conditionally dependent *given $\mathbf{Z}$, denoted by $(X \not\perp\!\!\!\perp Y \mid \mathbf{Z})$, if and only if they are not independent. If $\mathbf{Z}$ only contains one variable, $\mathbf{Z} = \{Z\}$, we write $(X \perp\!\!\!\perp Y \mid Z)$ and $(X \not\perp\!\!\!\perp Y \mid Z)$ instead of, respectively, $(X \perp\!\!\!\perp Y \mid \{Z\})$ and $(X \not\perp\!\!\!\perp Y \mid \{Z\})$. Like unconditional independence, conditional independence is straightforwardly generalized to sets of random variables $\mathbf{X}$ and $\mathbf{Y}$, $\mathbf{X}$ and $\mathbf{Y}$ being substituted for X and Y in (2.4).*

Conditional independence of X and Y given $\mathbf{Z}$ means that once we know $\mathbf{Z}$, additionally knowing X tells us nothing about Y, and conversely. It says nothing, however, about the unconditional independence (or: conditional given the empty set) of Y and X. In general, it can be that two variables X and Y are unconditionally independent: $(X \perp\!\!\!\perp Y)$, but that conditioning on Z makes them dependent: $(X \not\perp\!\!\!\perp Y \mid Z)$; or vice versa.

Conditional independence is a complex ternary relation. Conditioning on some variable Z may render X and Y independent when they are unconditionally dependent; it can make X and Y dependent when they are unconditionally independent; or it can have no effect at all. Certain of its properties known as the *graphoid axioms* are described in Lauritzen et al. (1990). Of particular interest to us is its symmetry with respect to X and Y in (2.4), to which we will refer later in the text:

$$(X \perp\!\!\!\perp Y \mid \mathbf{Z}) \iff (Y \perp\!\!\!\perp X \mid \mathbf{Z}). \tag{2.5}$$

Finding conditional independence can give a lot of insight into the structure of the data we are analyzing. However, interpreting a long list of conditional independence relations is difficult. To help with this task, *Bayesian networks* provide a convenient graphical notation.

2.2.2 Bayesian Networks

A Bayesian network (Pearl, 1988) is a graphical and algebraical tool to represent the structure of data with a graph, where an edges between two vertices represents statistical dependencies between the corresponding variables. Bayesian networks can represent a joint distribution compactly, making use of independence in a way that makes the specification of the full joint probability distribution require fewer parameters. In turn, this allows researchers to develop efficient inference algorithms that run on Bayesian networks.

Definition 2.10 (Bayesian network) *A* Bayesian network *$\mathcal{B}$ over a set of variables $\mathbf{V}$ is an ordered pair $\mathcal{B} = \langle \mathcal{G}, \mathbf{P} \rangle$, where $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$ is DAG and $\mathbf{P}$ a set of distributions that describe, for each $X_i \in \mathbf{V}$, the conditional probability $\mathrm{P}(X_i \mid \mathbf{Pa}_{\mathcal{G}}(X_i))$.*

Bayesian networks represent a joint probability distribution $\mathrm{P}(\mathbf{V})$, which can be computed from the local conditional distributions:

$$\mathrm{P}(\mathbf{V}) = \prod_{i=1}^{d} \mathrm{P}(X_i \mid \mathbf{Pa}(X_i)), \tag{2.6}$$

where d is the cardinality of $\mathbf{V}$, or the number of variables in the analysis. Such a network can be used to represent any probability distribution. By the chain rule of probability, any joint probability distribution can be expressed as a product of conditional distributions:

$$\mathrm{P}(\mathbf{V}) = \prod_{i=1}^{d} \mathrm{P}(X_i \mid X_1, X_2, \cdots, X_{i-1}). \tag{2.7}$$

This shows that a Bayesian network in which $\mathbf{Pa}(X_i) = \{X_1, X_2, \cdots, X_{i-1}\}$ (i.e., where the corresponding DAG is fully connected) can represent any joint distribution, including the worst case where all variables are dependent on one another. Such a graph for $d = 6$ is shown in Figure 2.2.

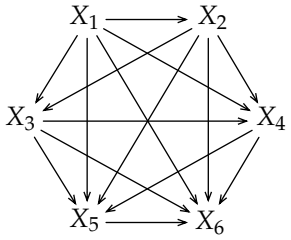


Figure 2.2: A fully connected graph with 6 vertices, able to represent a Bayesian network modeling any possible joint probability distribution.

The case where all variables are dependent on one another is not very interesting; let us see what conditional independence means for the structure of the graph. Suppose for instance that we have $d = 3$ variables and that we find that $(X_3 \perp\!\!\!\perp X_1 \mid X_2)$ holds (how we could find this in practice will be discussed in the next chapter). Then, by Definition 2.9, we know that $\mathrm{P}(X_3 \mid X_2, X_1) = \mathrm{P}(X_3 \mid X_2)$, which can be substituted as the third factor in (2.7), so that

$$\mathrm{P}(X_1, X_2, X_3) = \mathrm{P}(X_1)\mathrm{P}(X_2 \mid X_1)\mathrm{P}(X_3 \mid X_2). \tag{2.8}$$

Using Definition 2.10, we see that X_1 need not be a parent of X_3 in the corresponding graph: (conditional) independence is represented by the absence of links, and (conditional) dependence is represented by the presence of links.

Suppose we are only dealing with binary variables. Assigning a probability to each possible combination of X_1, X_2, X_3 can be done with $2^d - 1 = 7$ independent parameters. Finding $(X_3 \perp\!\!\!\perp X_1 \mid X_2)$ reduces it to a total of 5 necessary parameters. Fewer parameters to estimate makes the estimate more robust given a fixed-size dataset, makes the model less prone to overfitting, and allows inference algorithms to be more efficient (Pearl, 1988).

This begs the question, is it possible to find a graph that can represent exactly all independence and dependence relationships that can be found in the data, thus allowing a perfect reconstruction of the joint distribution with the minimal number of parameters?

If we wish to find a graph that matches the dependence and independence relations in the data, we need a graphical criterion that, given a graph $\mathcal{G}$, tells us the list of dependence and independence relations entailed by $\mathcal{G}$. If we can obtain an isomorphism between random variables and conditional independence on the one side, and vertices and this graphical criterion on the other side, then we can reason on conditional independence using graphs. This is beneficial because graphs and graphical properties are easier to handle and to visualize than conditional-independence properties. Pearl (1988) defined the concept of *d-separation* to fill in the role of this graphical criterion.

Definition 2.11 (*d*-Separation) *In a DAG $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, two nodes X and Y are* d-separated *given a set of nodes $\mathbf{Z}$ ($X, Y \notin \mathbf{Z}$), noted $(X \not\rightleftharpoons Y \mid \mathbf{Z})_{\mathcal{G}}$, if and only if every undirected path from X to Y is blocked by $\mathbf{Z}$. A path of length $\ell = 1$ is never blocked; a path $\boldsymbol{\pi} = (V_0, V_1, V_2, \cdots, V_{\ell-1}, V_\ell)$ (with $V_0 = X$, $V_\ell = Y$) of length ℓ is* blocked *by $\mathbf{Z}$ whenever at least one node V_i, $1 \leq i < \ell$, on $\boldsymbol{\pi}$ is blocked by $\mathbf{Z}$. V_i is blocked by $\mathbf{Z}$ if and only if these two conditions hold:*

- *V_i is serially connected ($\rightarrow V_i \rightarrow$, $\leftarrow V_i \leftarrow$) or is a diverging node ($\leftarrow V_i \rightarrow$) on $\boldsymbol{\pi}$ and $V_i \in \mathbf{Z}$;*
- *V_i is a converging node ($\rightarrow V_i \leftarrow$) on $\boldsymbol{\pi}$ and neither V_i nor any $D \in \mathbf{De}(V_i)$ is in $\mathbf{Z}$.*

X and Y are d-connected *given $\mathbf{Z}$, noted $(X \rightleftharpoons Y \mid \mathbf{Z})_{\mathcal{G}}$ if and only if they are not d-separated given $\mathbf{Z}$. When $\mathbf{Z}$ only contains one variable, $\mathbf{Z} = \{Z\}$, we write $(X \not\rightleftharpoons Y \mid Z)_{\mathcal{G}}$ and $(X \rightleftharpoons Y \mid Z)_{\mathcal{G}}$ instead of, respectively, $(X \not\rightleftharpoons Y \mid \{Z\})_{\mathcal{G}}$ and $(X \rightleftharpoons Y \mid \{Z\})_{\mathcal{G}}$. Similarly, two sets of random variables $\mathbf{X}$ and $\mathbf{Y}$ are* d-*separated or* d-*connected given $\mathbf{Z}$ if and only if for each $X \in \mathbf{X}$ and each $Y \in \mathbf{Y}$ pairwise* d-*separation or, respectively,* d-*connection holds. When the context is clear, we drop the subscript "$_{\mathcal{G}}$."*

The d-separation criterion is rather complex. To determine whether or not $(X \not\rightleftharpoons Y \mid \mathbf{Z})$ holds, we can either find an unblocked path from X to Y to prove that X and Y are d-connected given $\mathbf{Z}$, or show that all possible paths from X to Y are blocked by $\mathbf{Z}$ to prove that they are d-separated. As an example, one can verify in Figure 2.3 that $(X \not\rightleftharpoons Y \mid \mathbf{Z})$, $\mathbf{Z}$ being the set of the nodes marked with bullets (•), holds in case (c) only.

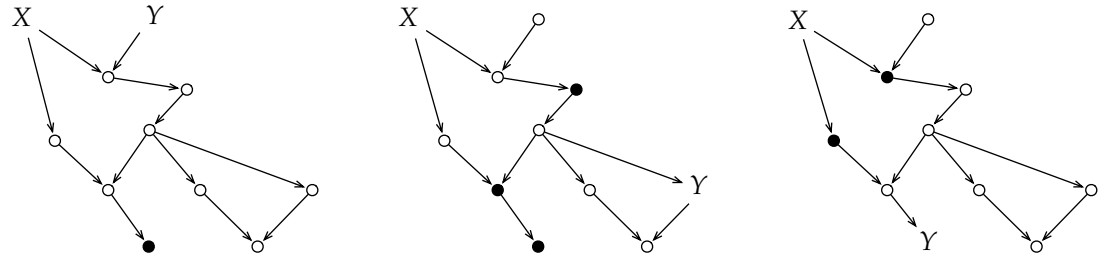


Figure 2.3: Three test cases for the d-separation criterion.

This definition of the d-separation criterion allows us to define what would be a perfect graphical representation of dependence and independence in our data: a graph where conditional independence and d-separation coincide, known as a *perfect map*.

Definition 2.12 (Perfect map) *A DAG $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$ is a* perfect map *of a joint probability distribution $\mathrm{P}(\mathbf{V})$ if and only if d-separation is mapped one-to-one to conditional independence:*

$$\forall \mathbf{X}, \mathbf{Y}, \mathbf{Z} \text{ disjoint subsets of } \mathbf{V}: \quad (\mathbf{X} \not\rightleftharpoons \mathbf{Y} \mid \mathbf{Z})_{\mathcal{G}} \iff (\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}). \tag{2.9}$$

Note the equivalence relationship in the definition. It turns out that not all joint probability distributions contain a set of independence relationships that can be perfectly represented in a DAG; those that can are called *DAG-isomorphic*.[5] Here is an example of a non-DAG-isomorphic

[5] Undirected graphical models, which use undirected graphs to depict the structure of the data, can represent another class of joint probability distributions, some of which DAGs cannot represent; and conversely. See Bishop (2006), pp. 390–393, for more information.

distribution: suppose X, Y, Z are binary variables, and X and Y are drawn uniformly. Then, define Z as $Z = X \oplus Y$, where $\oplus$ is the binary exclusive-OR operator, also known as XOR. Then we have:

$$(X \perp\!\!\!\perp Z) \qquad (X \not\perp\!\!\!\perp Z \mid Y) \tag{2.10}$$

$$(X \perp\!\!\!\perp Y) \qquad (X \not\perp\!\!\!\perp Y \mid Z) \tag{2.11}$$

$$(Y \perp\!\!\!\perp Z) \qquad (Y \not\perp\!\!\!\perp Z \mid X)\,. \tag{2.12}$$

The first column says that any two variables carry no information on each other. Let us examine line 2.10. For instance, let $X = 0$: there is still a 50% probability that $Z = 0$ and a 50% probability that $Z = 1$, depending on the value of Y. The same applies to the case where $X = 1$. Hence, $(X \perp\!\!\!\perp Z)$. But once we know Y, Z is completely determined; hence, $(X \not\perp\!\!\!\perp Z \mid Y)$: in graphical terms, this means that, as per Definition 2.11, Y is a converging node, $X \to Y \leftarrow Z$. But lines 2.11 and 2.12 similarly require that the structure be $X \to Z \leftarrow Y$ and $Y \to X \leftarrow Z$, respectively, for the same reason, which makes it impossible to find a perfect map for this distribution.

There is no finite set of axioms that characterize a distribution for which a perfect map exists (Geiger, 1987). If such axioms existed, they could help the investigation of the expressive power of perfect maps. But even when a distribution is known not to be DAG-isomorphic, we can still represent it in a graph—let us recall that thanks to (2.7), any probability distribution can always be broken down into a series of conditional probabilities. It is thus always possible to represent at least all dependencies with a DAG, called an *independence map*.

Definition 2.13 (Independence map) *A DAG $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$ is an* independence map *of a joint probability distribution $\mathrm{P}(\mathbf{V})$ if and only if* d*-separation implies conditional independence:*

$$\forall \mathbf{X}, \mathbf{Y}, \mathbf{Z} \text{ disjoint subsets of } \mathbf{V}: \quad (\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z})_{\mathcal{G}} \implies (\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z})\,. \tag{2.13}$$

For instance, any of the three structures $X \to Y \leftarrow Z$, $X \to Z \leftarrow Y$, or $Y \to X \leftarrow Z$ is an independence map for the XOR case described above: not all independence relations can be shown by the graph, but every *d*-separation relationship maps to a case of conditional independence.

2.2.3 Independence Equivalence

In DAGs, the *d*-separation criterion lets us read the graph in terms of conditional independence. While it makes use of the presence or absence of arcs in the structure of the graph, it only looks at the direction of the arcs when it comes to converging nodes—the non-converging nodes can be either nodes in a directed chain or diverging nodes (i.e., a common causes for several effects). These converging substructures are so particular that they have the specific name of V-structures.

Definition 2.14 (V-structure) *In a DAG, a* V-structure *is a triplet $X \to Z \leftarrow Y$, where X and Y are nonadjacent in $\mathcal{G}$: $X \notin \mathbf{Bd}(Y)$. Z is then called an* unshielded collider *for X and Y. In a perfect map, its presence implies that X and Y can be made dependent by conditioning on Z:*

$$\exists \mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y, Z\}: \quad (X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY}) \text{ and } (X \not\perp\!\!\!\perp Y \mid \mathbf{S}_{XY} \cup \{Z\})\,. \tag{2.14}$$

The *d*-separation criterion implies that the orientation of the arcs, besides V-structures, does not change the interpretation of the graph in terms of conditional independence. Conversely, then, a

set of conditional-independence relationships cannot fully determine a DAG, because the direction of some arcs can be left free. Two different DAGs can thus represent the same set of conditional-independence relationships. The concept of independence equivalence captures this idea.

Definition 2.15 (Independence equivalence) *Two DAGs $\mathcal{G}_1$, $\mathcal{G}_2$ are* independence equivalent *or* Markov equivalent *if and only if:*

$$\forall \mathbf{X}, \mathbf{Y}, \mathbf{Z} \text{ disjoint subsets of } \mathbf{V}: \quad (\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z})_{\mathcal{G}_1} \iff (\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z})_{\mathcal{G}_2}.$$

Property 2.16 *Two DAGs $\mathcal{G}_1$, $\mathcal{G}_2$ have the property that they are independence equivalent if and only if their undirected version is identical and they have the same set of V-structures (Pearl, 2000).*

This concludes our short presentation of Bayesian networks; for more information, see, e.g., Pearl (1988) or Chapter 8 of Bishop (2006).

2.3 FROM CAUSAL GRAPHS TO CAUSAL MODELS

In this section, we deal with causation, its relation to conditional independence, and the causal interpretation of Bayesian networks.

2.3.1 Causal Networks and Bayesian Networks

To illustrate the interpretation of conditional independence with respect to cause–effect relationships, we consider a simple probabilistic causal example. Suppose we have the following situation: Sometimes, there is a traffic jam on the way from home to work. The traffic jam might delay an employee and he might arrive late at work. Arriving late (depending on how late) can in turn make it more likely that he misses an early meeting. We show that representing this according to our definition of a causal graph (Figure 2.4) yields a valid structure for a Bayesian network, too, and then try to characterize a direct causal relationship in terms of conditional independence.

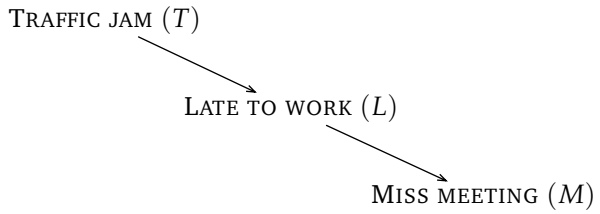


Figure 2.4: Simple causal example.

What this simple description encodes, as per Definition 2.7 is, obviously, the two causal relationships shown graphically with arrows. More importantly, it also says that the traffic jam alone has no influence on whether the employee makes it in time to attend the meeting: the only way for traffic to cause a missed meeting is through the late arrival of the employee. But owing to the causal chain, the traffic jam indeed increases the probability of a missed meeting.

From their probability distribution, all variables are dependent: we have $(T \not\perp\!\!\!\perp L)$ and $(L \not\perp\!\!\!\perp M)$ because of the direct causal effect, and $(T \not\perp\!\!\!\perp M)$ because of the causal chain. Symmetry between

the three variables is broken when conditional independence is introduced. In particular, we find that $(T \perp\!\!\!\perp M \mid L)$. In other terms: once we know how late the employee has arrived, knowing whether there was a traffic jam does not influence the fact that he may miss the meeting anymore. Knowing his arrival time is enough to know the probability of missing the meeting, because the links in Figure 2.4 explicitly encode that T has no effect on M that is not mediated by L.

This is easy to show if Figure 2.4 as interpreted as the perfect-map structure of a Bayesian network: supposing that there is an intrinsic probability of traffic jam $\mathrm{P}(T)$, a probability of arriving late given a traffic jam $\mathrm{P}(L \mid T)$, and a probability of missing the meeting given a late arrival $\mathrm{P}(M \mid L)$, then we can express $\mathrm{P}(T, L, M) = \mathrm{P}(T)\mathrm{P}(L \mid T)\mathrm{P}(M \mid L)$. From here, we obtain:

$$
\begin{aligned}
\mathrm{P}(T, M \mid L) &= \frac{\mathrm{P}(T, L, M)}{\mathrm{P}(L)} && \text{as per the definition of conditional probability} \\
&= \frac{\mathrm{P}(T)\mathrm{P}(L \mid T)\mathrm{P}(M \mid L)}{\mathrm{P}(L)} && \text{by assumption} \\
&= \frac{\mathrm{P}(T, L)}{\mathrm{P}(L)}\mathrm{P}(M \mid L) && \text{as per the def. of conditional probability \& rearrangement} \\
&= \mathrm{P}(T \mid L)\mathrm{P}(M \mid L) && \text{as per the definition of conditional probability again,}
\end{aligned}
$$

which says by Definition 2.9 that $(T \perp\!\!\!\perp M \mid L)$ holds. We arrive at the same conclusion if we read Figure 2.4 with *d*-separation: we find that $(T \not\perp\!\!\!\perp M \mid L)$ holds, and thus $(T \perp\!\!\!\perp M \mid L)$ as per the perfect-map property.

So we see that the Bayesian-network and causal-network interpretations look very close to each other. There are many techniques to learn the structure of Bayesian networks, some of them offering the perfect-map guarantee (Neapolitan, 2003). Could we use them to learn causal networks?

A disturbing fact is that (conditional) independence is symmetrical, as observed in (2.5). If we reverse the two causal relationships, assuming for one moment that missing the meeting can cause a late arrival, which in turn can cause a traffic jam, then we can derive the exact same set of (conditional) independence relationships. What is more, this is also true for the (equally absurd) case where a late arrival would cause both a traffic jam and a missed meeting. A perfect map in terms of conditional independence may be an absurd causal graph.

This is the problem we mentioned in the previous section: several graphs can entail the same set of conditional-independence relations. This means that if we assume that there exists a perfect map that describes the causal structure of a data-generating mechanism, then by definition all graphs in the same equivalence class are perfect maps. Only one, however, is the correct causal graph. This can be seen as the fundamental difference between Bayesian networks, where we only consider conditional independence, and causal networks, where the arcs really denote causation.

Another important issue with causal networks is the problem of hidden variables. Before going on with the discussion on how to relate conditional independence and causation, we need to differentiate two important causal types of datasets, depending on what variables are observed and what variables are hidden.

Definition 2.17 (Causal sufficiency) *A set of observable variables* $\mathbf{V}$ *is said to be* causally sufficient *if and only if any common cause of two or more variables in* $\mathbf{V}$ *is also in* $\mathbf{V}$.

Causal sufficiency, already mentioned in the introduction, basically asserts that there are no hidden confounders for a subset of variables in $\mathbf{V}$; i.e., that all variables that may bias the causal analysis

are observed. In practice, this is untestable, but discussed a lot, as the outcome of most algorithms depends on it. For instance, had the GENDER variable not been measured in our example of Subsection 2.1.1, then the set {DRUG, RECOVERY} would not have been causally sufficient: a common cause for both remaining variables would have been left out of the analysis, and we could not have examined the separate results in Table 2.2, which show the true causal effect of the drug. On the contrary, the example where GENDER is replaced by BLOOD PRESSURE is also causally sufficient without BLOOD PRESSURE, as it is not a common cause for the two other variables. Not including it in the analysis does not bias the causal conclusions, as in this case the true causal effect can directly be read from Table 2.1.

Lack of causal sufficiency is an important issue: the potential presence of hidden confounders changes the way we discover direct causation, needs new notation, concepts, and algorithms, and weakens the causal conclusions that we can draw from observational data. We deal with these issues more extensively throughout Chapter 4; until then, we assume a causally sufficient context.

2.3.2 Conditional Independence and Causation with Causal Sufficiency

In this subsection, we examine more formally how conditional independence and direct causation are related in causally sufficient cases (later, in Subsection 4.1.3 we revisit the properties described here in causally insufficient cases).

Assume a DAG-isomorphic probability distribution over a causally sufficient $\mathbf{V}$. Direct causation $X \dashrightarrow Y$ (as per Definition 2.6) implies that X and Y are dependent *given any conditioning set* (Pearl & Verma, 1991, see definitions of potential and genuine causes):

$$X \dashrightarrow Y \implies \forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}). \tag{2.15}$$

The contraposition of this implication asserts that as soon as we find a set $\mathbf{S}$ that makes X and Y independent, we can exclude direct causation, which we note $X \not\dashrightarrow Y$:

$$\exists \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \perp\!\!\!\perp Y \mid \mathbf{S}) \implies X \not\dashrightarrow Y. \tag{2.16}$$

The exact converse of (2.15) does not hold. Note that the right-hand side of the implication is perfectly symmetrical with respect to X and Y, whereas the premise is the asymmetric causation from X to Y. But, relying on causal sufficiency, we can write:

$$\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}) \implies X \dashrightarrow Y \text{ or } Y \dashrightarrow X. \tag{2.17}$$

Using (2.17), we can theoretically determine all adjacencies of a perfect map with a criterion of conditional independence, but we cannot orient the edges. This seems to indicate that indeed, we can detect only "causal adjacencies," but not the direction of causation. Enter the V-structure: we recall from Definition 2.14 that conditioning on a collider (a common child of two nonadjacent variables) makes two disconnected parents dependent. In causal terms: conditioning on a common effect makes two independent causes dependent. Formally, we have:

$$\begin{aligned} &X \dashrightarrow Z \dashleftarrow Y \text{ and } X \not\dashrightarrow Y \text{ and } Y \not\dashrightarrow X \\ &\qquad \implies \Big(\exists \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y, Z\} : (X \perp\!\!\!\perp Y \mid \mathbf{S}) \text{ and } (X \not\perp\!\!\!\perp Y \mid \mathbf{S} \cup \{Z\})\Big). \end{aligned} \tag{2.18}$$

The exact converse does not hold either. But (still with causal sufficiency) we can find an equivalence relation defining a V-structure: first, we certify the existence of a link between X and Z and between Y and Z with (2.17). Then, we identify Z as an unshielded collider if conditioning on it creates a dependency between X and Y:

$$\begin{aligned} X \dashrightarrow Z \dashleftarrow Y \iff \Big(& \big(\exists \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y, Z\} : (X \perp\!\!\!\perp Y \mid \mathbf{S}) \text{ and } (X \not\perp\!\!\!\perp Y \mid \mathbf{S} \cup \{Z\})\big) \\ & \text{and } \big(\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Z\} : (X \not\perp\!\!\!\perp Z \mid \mathbf{S})\big) \\ & \text{and } \big(\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{Y, Z\} : (Y \not\perp\!\!\!\perp Z \mid \mathbf{S})\big)\Big). \end{aligned} \tag{2.19}$$

Actually, typical structure-learning algorithms first establish the existence of a link between two variables by seeking a certificate equivalent to, or implicating the premise of, (2.17), and insert a temporary undirected link. They then look for V-structures, where they can insert the causal direction. Note again that there is no guarantee that all links can be oriented into causal arcs, and that in general we therefore cannot recover the full causal structure with conditional independence, because of the issue of independence equivalence. In causality, this is known as *causal underdetermination* (Spirtes et al., 2001, p. 62): given a joint probability distribution $\mathrm{P}(\mathbf{V})$, the correct causal graph can only be specified by its adjacencies and V-structures. Later, we show extended graph notations to take this fact into account and to easily represent equivalence classes.

Finally, if we combine (2.9), (2.17) and (2.19), we find, for a perfect causal map $\mathcal{G}$ (using the symbol $\dashrightarrow$ to denote direct causation and $\rightarrow$ to denote an arc in the graph):

$$\begin{aligned} X \text{ and } Y \text{ adjacent in } \mathcal{G} &\iff X \dashrightarrow Y \text{ or } Y \dashrightarrow X \\ X \rightarrow Z \leftarrow Y &\iff X \dashrightarrow Z \dashleftarrow Y. \end{aligned} \tag{2.20}$$

It is sometimes possible to orient further arcs in a graph by looking at already-oriented arcs and propagating constraints, preventing acyclicity and the creation of additional V-structures other than those already detected. This is discussed in Subsections 3.1.3 and 4.2.3.

2.3.3 Prediction for Interventions: the *do*-Calculus

In any given analysis, obtaining and agreeing on a causal network is only the first step. Further analysis, such as what-if scenarios, policy evaluation, intervention assessment, etc., can be performed using the causal graph. Most tools reasoning on a causal graph use the *do*-calculus, a set of rules that allow the evaluation of the causal effect of interventions on the data-generating system.

We need new notation to represent the effect of an intervention on a probability distribution, as opposed to the effect of an observation. To this end, we use *do*-expressions. A general *do*-expression is of the type

$$\mathrm{P}(\mathbf{Y} = \mathbf{y} \mid do(\mathbf{X} = \mathbf{x}), \mathbf{W} = \mathbf{w}), \tag{2.21}$$

and should be read as "the probability of $\mathbf{Y} = \mathbf{y}$ given that $\mathbf{W}$ is observed to have value $\mathbf{w}$ and $\mathbf{X}$ is intervened on so that it has value $\mathbf{x}$." Like Pearl (2000), we also use the following equivalent shorthand notation, denoting the intervention values with a hat: $\mathrm{P}(\mathbf{y} \mid \hat{\mathbf{x}}, \mathbf{w})$. How to evaluate such an expression requires some knowledge of the causal graph. The variables affected by an intervention on $\mathbf{X}$ are the effects of $\mathbf{X}$ (which we know only through the causal graph), whereas

the variables whose distribution is affected by the $\mathbf{W} = \mathbf{w}$ observation are all those that are not independent of $\mathbf{W}$ (which we can know from the joint distribution alone).

Graphically, we can represent an intervention by the modification of the graphical structure. Let X be a tunable variable and Y a response variable possibly causally influenced by X. We can distinguish several types of interventions, in increasing levels of complexity, along with their notation:

1. Unconditionally setting X to some value x. We then observe Y: $\mathrm{P}(Y \mid do(X = x))$;
2. Setting X to some value, depending on some control variable Z: $\mathrm{P}(Y \mid do(X = x \mid Z = z))$, or more generally, $\mathrm{P}\big(Y \mid do(X = g(z))\big)$;
3. Replacing the original conditional distribution of X by a new distribution, potentially with a new set of parents: $\mathrm{P}\big(Y \mid do(\mathrm{P}'(X \mid \mathbf{Pa}(X)'))\big)$.

Intervention 1 is a special case of Intervention 2, which is itself a special case of Intervention 3. Interventions 2 and 3 are subject to the acyclicity constraint: $Z \notin \mathbf{De}(X)$ and $\mathbf{Pa}(X)' \cap \mathbf{De}(X) = \emptyset$, respectively. Realizing that they create causal dependencies between Z and X, and $\mathbf{Pa}(X)'$ and X, respectively, we must see to it that we do not create causal feedback loops, forbidden by our model based on DAGs.

Consider the classical example in Figure 2.5, which shows a causal network for a simple system. The top-level variable describes whether the sky is cloudy (C). Depending on this, it may rain (R), or we may decide to turn on the sprinkler (S). Both R and S cause the grass in the garden to be wet (W).

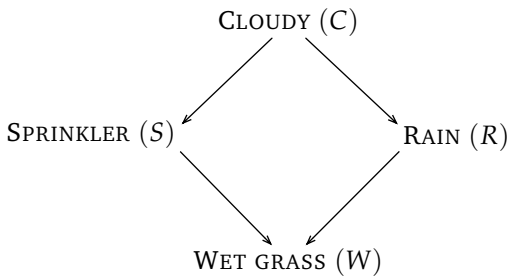


Figure 2.5: The **SPRINKLER** example.

Now suppose that we perform the following interventions on the system, corresponding to the three types of interventions we have identified:

(a) We decide to turn on the sprinkler independently of all other variables;
(b) We turn on the sprinkler if it is not raining;
(c) Depending on RAIN and on a new variable TEMPERATURE (T)—which we suppose independent from CLOUD and RAIN—we turn it off or on.

Figure 2.6 shows what are called *manipulated graphs* for these three scenarios. The variable we intervene on is SPRINKLER (S).

Now, we can examine the effect of an atomic intervention of the type of (a), $\mathrm{P}(W \mid do(S = on))$: the incoming arcs of S are suppressed; i.e., setting S affects only its effect and not its cause:

$$\mathrm{P}(W) \neq \mathrm{P}(W \mid do(S = on)) \tag{2.22}$$
$$\mathrm{P}(C) = \mathrm{P}(C \mid do(S = on)) \tag{2.23}$$
$$\mathrm{P}(R) = \mathrm{P}(R \mid do(S = on)). \tag{2.24}$$

On the contrary, observing S, as in $\mathrm{P}(W \mid S = on)$, also affects the probability distribution of its cause, and thus of the other effect of its cause:

$$\mathrm{P}(W) \neq \mathrm{P}(W \mid S = on) \tag{2.25}$$
$$\mathrm{P}(C) \neq \mathrm{P}(C \mid S = on) \tag{2.26}$$
$$\mathrm{P}(R) \neq \mathrm{P}(R \mid S = on). \tag{2.27}$$

The new joint probability distribution resulting from an intervention is called *postintervention* distribution (as opposed to the original, *preintervention* distribution). The following definition shows how to compute them in case (a), which can be regarded as the most important "primitive" of causal-effect assessment, also used in causal-reasoning techniques like those presented later in this thesis. For details on more complicated interventions, refer to Hagmayer et al. (2005), Woodward (2003) or to Chapter 4 of Pearl (2000).

Definition 2.18 (Postintervention distribution) *Given a causal Bayesian network $\mathcal{B} = \langle \mathcal{G}, \mathbf{P} \rangle$ over variables $\mathbf{V}$, the* postintervention distribution $\mathrm{P}(\mathbf{V} \mid do(X = x))$*, also denoted $\mathrm{P}(\mathbf{V} \mid \hat{x})$, after an intervention $do(X = x)$, is:*

$$\mathrm{P}(\mathbf{V} \mid \hat{x}) = \begin{cases} \prod_{Y \in \mathbf{V}, Y \neq X} \mathrm{P}(Y \mid \mathbf{Pa}_{\mathcal{G}}(Y)) & \text{if } X = x, \\ 0 & \text{if } X \neq x. \end{cases} \tag{2.28}$$

The *truncated factorization* (2.28) states that the postintervention probability distribution is computed as if the manipulated variable X had no incoming causal influence (i.e., no direct causes), and as if $\mathrm{P}(X = x)$ had probability one. This makes sense, as forcing X to have a certain value effectively ignores its direct causes and "natural" distribution. With this formula, we have an expression for the whole joint probability distribution.

Most of the time, however, computing the whole distribution $\mathrm{P}(\mathbf{V} \mid \hat{x})$ is prohibitively long. Can we somehow reuse the simpler, less complex conditional distributions of the type $\mathrm{P}(Y \mid \mathbf{Pa}_{\mathcal{G}}(Y))$ that we have directly from the Bayesian network $\mathcal{B}$? This is what the *do*-calculus is about.

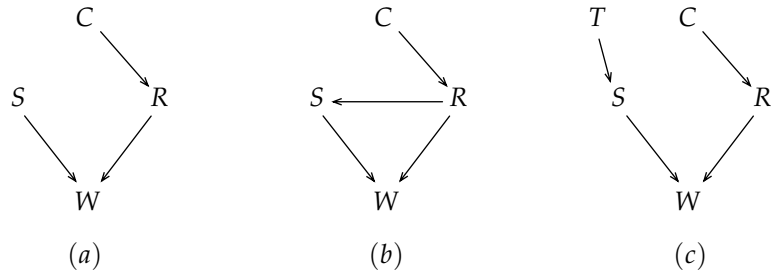


Figure 2.6: The manipulated graphs.

Manipulated graphs are used in the rules of the *do*-calculus, originally described in Pearl (1995). With $X, Y \in \mathbf{V}$, the following manipulated graphs are defined:

- $\mathcal{G}_{\overline{\mathbf{X}}}$ is the graph where all arcs going into nodes in $\mathbf{X}$ have been deleted (we suppress the influence of the causes for $\mathbf{X}$);
- $\mathcal{G}_{\underline{\mathbf{X}}}$ is the graph where all arcs going out of nodes in $\mathbf{X}$ have been deleted (we suppress the influence of $\mathbf{X}$ on its effects).

Definition 2.19 (Rules of *do*-calculus) *For all disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{W} \subset \mathbf{V}$ in a causal graph $\mathcal{G}$, we have the following rules.*

- **Rule 1: Insertion and deletion of observations**

$$(\mathbf{Y} \perp\!\!\!\perp \mathbf{Z} \mid \mathbf{X} \cup \mathbf{W})_{\mathcal{G}_{\overline{\mathbf{X}}}} \implies \mathrm{P}(\mathbf{Y} \mid do(\mathbf{X}), \mathbf{Z}, \mathbf{W}) = \mathrm{P}(\mathbf{Y} \mid do(\mathbf{X}), \mathbf{W}) \tag{2.29}$$

- **Rule 2: Action/observation exchange**

$$(\mathbf{Y} \perp\!\!\!\perp \mathbf{Z} \mid \mathbf{X} \cup \mathbf{W})_{\mathcal{G}_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}} \implies \mathrm{P}(\mathbf{Y} \mid do(\mathbf{X}), do(\mathbf{Z}), \mathbf{W}) = \mathrm{P}(\mathbf{Y} \mid do(\mathbf{X}), \mathbf{Z}, \mathbf{W}) \tag{2.30}$$

- **Rule 3: Insertion and deletion of actions**

$$(\mathbf{Y} \perp\!\!\!\perp \mathbf{Z} \mid \mathbf{X} \cup \mathbf{W})_{\mathcal{G}_{\overline{\mathbf{X}},\overline{\mathbf{Z \setminus An(W)}}}} \implies \mathrm{P}(\mathbf{Y} \mid do(\mathbf{X}), do(\mathbf{Z}), \mathbf{W}) = \mathrm{P}(\mathbf{Y} \mid do(\mathbf{X}), \mathbf{W}) \tag{2.31}$$

Rule 1 establishes an equality between two probabilities where the only difference lies in an additional observation $\mathbf{Z}$, and gives a familiar *d*-separation condition, but in the manipulated graph. It reaffirms the graph manipulation consisting of removing the incoming arcs for $\mathbf{X}$ as the correct representation of a single intervention on $\mathbf{X}$ and maintains *d*-separation as a valid criterion for reading conditional independence off the graph.

Rule 2 gives a condition for the equivalence of observing $\mathbf{Z}$ and setting $\mathbf{Z}$ with respect to the conditional distribution of $\mathbf{Y}$: $\mathbf{Z}$ and $\mathbf{Y}$ must be *d*-separated in the graph where all outgoing arcs from $\mathbf{Z}$ have been deleted. This is related to what is known as the *back-door criterion* and has allowed us to solve our problem resulting from Simpson's paradox. To exchange observation for intervention, we require all influences from $\mathbf{Z}$ on $\mathbf{Y}$ to go through the direct effects of $\mathbf{Z}$ (and hence, $\mathbf{Z}$ and $\mathbf{Y}$ to be *d*-separated once we remove the effect of $\mathbf{Z}$ on its children). Otherwise, we are in a situation similar to the one in Figure 2.1 (a) and we must adjust for the confounding variables by conditioning on them (or by setting them to certain values). Thus they appear in the conditioning set of the *d*-separation condition of Rule 2, *d*-separate $\mathbf{Z}$ from $\mathbf{Y}$, and in this context allow the equivalence of observation and intervention.

Finally, Rule 3 tells when it is that adding an intervention on $\mathbf{Z}$ has no effect on $\mathbf{Y}$: namely, when $\mathbf{Z}$ and $\mathbf{Y}$ are *d*-separated in the manipulated graph, just like S and C, respectively, in Figure 2.6 (a). If $\mathbf{X}$ is additionally being set, then we must manipulate the graph accordingly. (The technical reason for limiting the arc deletion to nonancestors of $\mathbf{W}$ can be found in Pearl, 1995.)

These rules, although not straightforward, can be understood intuitively if we visualize the graphs and can be applied to solve a *do*-expression (see Pearl, 2000, pp. 86–87, for examples), and this transformation can be efficiently done algorithmically (Galles & Pearl, 1995). In Chapter 5, we make extensive use of the *do*-calculus to introduce causal-explanation trees, a method for finding how to account for observations in a causal context.

2.3.4 Causal Models

The rules of the *do*-calculus rely on a causal graph and on conditional probability distributions—in other words, on a *causal Bayesian network*—to solve a *do*-expression. The role of a *do*-expression is to evaluate the effect of an intervention in a given context, and can be seen as the primitive of causal reasoning: virtually all causal questions can be cast in terms of effects of interventions. Here are a few examples:

- The *causal effect* of X on Y can be evaluated by comparing the "natural" distribution of Y with the distribution of Y after an intervention on X. A change in $\mathrm{P}(Y)$ reveals a (direct or indirect) causal link from X to Y. Similarly, a direct causal effect from X to Y, as defined on page 13, can be specified by a change in $\mathrm{P}(Y)$ after an intervention on X when all other variables have also been manipulated.
- A *confounder* for X and Y can be seen as a variable which influences both X and Y when intervened on, and which must be held constant to avoid biased estimates of the causal effect between X and Y, if any.
- *Spurious correlation* between X and Y is correlation that is due to a common cause, such that neither an intervention on X nor an intervention on Y would change the distribution of the other variable.
- An *explanatory variable* for some observation $E = e$ can be defined as a variable that increases the probability of e when intervened on.[6]

The causal graph together with the conditional distributions for each variable are enough to evaluate a *do*-expression: they thus constitute a *causal model*, according to the following definition.

Definition 2.20 (Causal model) *A* causal model $\mathcal{M}_{\mathbf{V}}$ *for a set of variables* $\mathbf{V}$ *is an algebraic tool that can be used to evaluate the effect of interventions on the system as formalized by a do-expression of the type* $\mathrm{P}(\mathbf{y} \mid \hat{\mathbf{x}}, \mathbf{w})$. *If the model can evaluate any such legal expression, it is known as* fully specified.

Causal models are thus, first and foremost, tools for evaluating interventions, thereby learning about the manipulation-invariant properties of the data-generating system, as explained in the introduction in Figure 1.3. This is what distinguishes them from traditional predictive models, which often make predictions based on the assumption that the joint probability $\mathrm{P}(\mathbf{V})$ does not change. Causal models are able to determine how the system dynamics change in response to an intervention on a set of variables $\mathbf{X}$, and can thus adapt their internal representation of the postintervention distribution $\mathrm{P}'(\mathbf{V})$.

Throughout this chapter, we have looked at a specific instance of causal model for a causally sufficient context: a causal Bayesian network $\mathcal{B} = \langle \mathcal{G}, \mathbf{P} \rangle$, where $\mathcal{G}$ is a fully oriented DAG that provides the set of all direct cause–effect relationships in $\mathbf{V}$, and $\mathbf{P}$ is a set of conditional probability distributions describing and quantifying each such relationship. Causal Bayesian networks, however, are by no means the only possible form of causal model. An important alternative, also mentioned in the introduction but not used in this thesis, is to use structural equations of the form

$$X := f_X(\mathbf{Pa}(X), U_X) \tag{2.32}$$

[6] We discuss the interpretation and definition of explanatory variables in more detail in Chapter 5.

for each node X in the causal graph $\mathcal{G}$ (U_X being a stochastic noise factor), instead of a conditional probability distribution as in Bayesian networks. In a functional setting, an intervention $do(X = x)$ is represented by a postintervention model where the original function (2.32) determining X is replaced by

$$X := x, \tag{2.33}$$

all other equations remaining untouched (Pearl, 2000).

The rest of the theoretical part of this thesis is concerned with two main points: how to obtain the causal graph $\mathcal{G}$ (Chapters 3 and 4) and how to reason on a causal model with the concept of causal explanations (Chapter 5). In Chapters 3 and 4, we are thus primarily concerned with the qualitative part of the causal model and do not specify whether the quantitative model is a causal Bayesian network or a structural-equation model; in Chapter 5, we use the notation of *do*-expression used in Definition 2.20, leaving it up to analysts to choose the concrete quantitative model that is best suited to their needs.

Chapter Summary

In this chapter, we have described the need for causal networks by showing how causal reasoning helps to solve Simpson's paradox. We have shown the relationships that exist between conditional independence, causation, and graphs, and have learned about the causal underdetermination issue. We have become acquainted with Bayesian networks and causal networks, and discovered the basic tool used to reason on causal networks, the do-calculus. Finally, we have defined what a causal model is: a tool for evaluating the effect of interventions.

CHAPTER

3

Structure Learning with Causal Sufficiency

This chapter explains how it is possible to learn the structure of a causal graph from observational data with a causally sufficient set of variables; i.e., where we can assume that there are no unknown hidden causes. We begin by introducing the needed notation, then show the standard algorithms to perform this task. Then, we propose a novel approach to structure learning based on the local neighborhood of each variable called the Markov blanket. We show how this opens the door to a family of fast and flexible algorithms. We end this chapter with a series of benchmarks and comparisons.

3.1 Structure Learning: the Basics

We first start with the basics of structure learning: how to deal with causal underdetermination, how to represent it, and how to systematically identify unambiguous structures.

3.1.1 Automatic Learning and Underdetermination

In the previous chapter, we saw how we could relate causation to conditional independence. In this chapter, we examine how to automatically build a causal graph form observational data with the help of conditional independence (Pearl & Verma, 1991). The goal of structure-learning algorithms is to obtain a structure that is a perfect map of the input data, in the sense of Definition 2.12: every *d*-separation and *d*-connection relation on the graph should map, respectively, to a conditional-independence and conditional-dependence relation in the data. If the full joint distribution of the data were known, it would be possible to check this property, but it is not using only the data. What is done instead to check a conditional-independence statement of the form $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ is a *statistical test of conditional independence*. This means that the Boolean outcome of the test is subject to the Type I and Type II statistical test error rates (see, e.g., Rice, 1995).

While designing algorithms, it is useful (if only to ensure correctness) to suppose that we have a consistent conditional-independence oracle that never fails nor returns contradictory results. In

practice, this can be implemented if the causal structure is known and if the problem is DAG-isomorphic, as in this case we can build a perfect map of the problem, and directly use the *d*-separation criterion in lieu of a conditional-independence test when checking for $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$. This is also a reason why such algorithms are tested on artificial data generated by a known structure: because we can use the never-failing oracle to validate the algorithm.

But even with a conditional-independence oracle, we know that we have the fundamental problem of causal underdetermination: several different graphs can imply the same set of conditional-independence relations, as long as they have the same undirected structure and the same V-structures (Property 2.16). What structure-learning algorithms do to account for this ambiguity is return a special kind of graph called *partially directed acyclic graph*, or PDAG (pronounced "peedag", Spirtes et al., 2001).

Definition 3.1 (Partially directed acyclic graph) *A* partially directed acyclic graph *or* PDAG *is a graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}_{\mathcal{E}} \rangle$ according to Definition 2.1 on page 12, where $\mathcal{E} = \{\rightarrow, -\}$; i.e., there are two possible edge types, $\rightarrow$ and $-$. $X - Y$ is considered equivalent to $Y - X$.*

In PDAGs, some of the edges can be directed and some others may remain undirected. Actually, following Property 2.16, if we use PDAG to represent the equivalence class, then this PDAG must also have the same undirected structure and the same V-structure.

This property actually implies further orientation possibilities: for instance, any new orientation of one of the edges that are not in a V-structure is not allowed to create a V-structure. In the same spirit, any new orientation is not allowed to make the graph cyclic. Actually, there is a PDAG in which all oriented edges are exactly those that have the same orientation in all members of the equivalence class. This PDAG is known as the *completed PDAG* (CPDAG—"seepeedag"), *maximally oriented PDAG*, or *essential graph*, depending on the author (Andersson et al., 1997). Later, we describe a set of rules that lead to the CPDAG given a PDAG where only V-structures are oriented.

Then, CPDAGs, not general PDAGs, are used to represent the independence equivalent class of causal graphs (Pearl, 2000, p. 19). When checking algorithms based on conditional independence for correctness, it is customary to compare their output (in the form of a PDAG) with the CPDAG of the equivalence class of the generating graph.

Consider the graphical example in Figure 3.1, and suppose the true causal structure of some process is (a). The set of conditional-independence relations drawn from it does not allow us to distinguish between (a), (b) and (c); they actually all belong to the equivalence class shown by the CPDAG (d).

One could wonder how the arc $Y \rightarrow U$ was directed in the CPDAG (d): there is no justifying V-structure for it. Actually, it is the *absence* of a V-structure involving it that unambiguously dictates the orientation in this case: as previously described, any further orientation may not create a new V-structure.

Figure 3.2 shows graphs which are not in this equivalence class. (a) has a reversed arc $U \rightarrow Y$ and thus two extra V-structures $Z \rightarrow Y \leftarrow U$ and $W \rightarrow Y \leftarrow U$. (b) has no reversed arc, but has an extra V-structure $Z \rightarrow X \leftarrow W$. (c) has the missing V-structure $Z \rightarrow Y \leftarrow W$ and is not even a DAG, as it contains a cycle.

3.1.2 IC: an Algorithm Specification

We now know that structure-learning algorithms, in general, can only learn the structure up to the independence-equivalence class, specified by the CPDAG of the original data-generating structure. In this subsection, we describe the Inductive Causation (IC) algorithm, which is the outline of a systematic way to build the structure of the causal graph.

Like many other structure-learning algorithms, IC works in different phases. It starts with an empty graph with d nodes, representing the $d = |\mathbf{V}|$ variables in our analysis. Its main steps are:

1. Find the undirected structure: add an undirected edge between X and Y if no set $\mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\}$ can be found such that $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$.
2. Determine the V-structures. For each connected triple $X - Z - Y$ where X and Y are nonadjacent, direct the edges and add a V-structure $X \rightarrow Z \leftarrow Y$ if $Z \notin \mathbf{S}_{XY}$, i.e., if and only if X and Y are *dependent* given Z.
3. Direct the remaining edges. Follow different rules/constraints to direct the remaining edges whenever possible, avoiding the creation of new V-structures and of cycles.

The justification for this algorithm can almost be directly read from the relations between causation and conditional independence in (2.17) and (2.19). Undirected links are inserted whenever no d-separating set can be found for a pair of variables X, Y (which, according to (2.17), means either $X \dashrightarrow Y$ or $Y \dashrightarrow X$, and thus $X \rightarrow Y$ or $Y \rightarrow X$ in the perfect map that we are trying to build). Then, V-structures are oriented if an intermediary variable Z makes them dependent.

Although the set $\mathbf{S}_{XY}$ such that $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$ is not unique, we can prove with d-separation that if $X - Z - Y$ is not a V-structure, then any such set must include Z: If $X - Z - Y$ is not a V-structure, then it is one of $X \rightarrow Z \rightarrow X$, $X \leftarrow Z \leftarrow X$, or $X \leftarrow Z \rightarrow X$. Definition 2.11 tells us that in order for X and Y to be d-separated, these three paths must be blocked exactly the same way: by including Z in the d-separating set; therefore, if $Z \notin \mathbf{S}_{XY}$, then we have a V-structure $X \rightarrow Z \leftarrow Y$.

The last step ensures that the returned PDAG is maximally oriented, respecting the rules that no new V-structure may be created and that the graph must remain acyclic. (The exact rules are detailed in the next subsection, in Algorithm 3.2.)

How can this algorithm ensure that the returned PDAG is the CPDAG of the equivalence class, i.e., the specification of a perfect map? After all, we are only checking local conditions for pairs

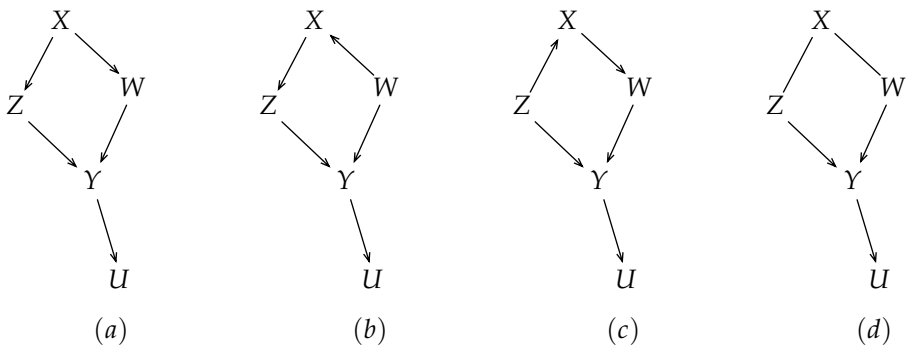


Figure 3.1: DAGs (a), (b), (c), belonging to the equivalence class of the CPDAG (d).

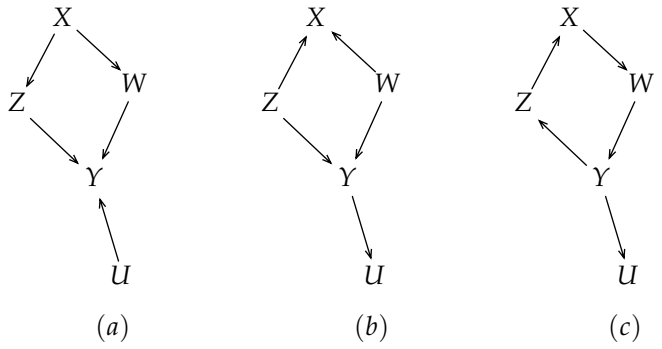


Figure 3.2: Graphs not in the equivalence class shown in Figure 3.1 (d).

or triplets of variables, and Definition 2.12 says that the d-separation/conditional-independence equivalence must hold for all possible subsets of $\mathbf{V}$. We have to explicitly mention all the assumptions that this algorithm relies on.

First, *consistency of the conditional-independence tests* used to implement the $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ criterion. This allows us to prove that the returned graph converges to the correct CPDAG in the large-sample limit. Then, we assume *causal sufficiency*, which we have already discussed, and which asserts that all common causes of two or more variables in $\mathbf{V}$ are also in $\mathbf{V}$. Finally, there are two conditions that we need to examine more carefully, the *causal Markov condition* and the *faithfulness condition* (Spirtes et al., 2001).

Definition 3.2 (Causal Markov condition) *A probability distribution* $\mathrm{P}(\mathbf{V})$ *is said to fulfill the* causal Markov condition *for a causal graph* $\mathcal{G}$ *if and only if:*

$$(X \not\perp\!\!\!\perp Y \mid \mathbf{Pa}_{\mathcal{G}}(X)) \implies X \dashrightarrow Y, \tag{3.1}$$

where $\dashrightarrow$ *denotes direct or indirect causation: either* $X \dashrightarrow Y$*, or there exists a causal chain between* X *and* Y *of the form* $X \dashrightarrow Z_1 \dashrightarrow Z_2 \dashrightarrow \cdots \dashrightarrow Y$.

The causal Markov condition establishes a link between association and causation, in the sense that every variable Y that is dependent on X given all causes of X, $\mathbf{Pa}_{\mathcal{G}}(X)$, is a direct or indirect effect of X. In a way, this is stronger than causal sufficiency, as it implies, roughly spoken, that every association can be explained with some form of causal mechanism within the observed variables $\mathbf{V}$. Hence, $\mathbf{V}$ must contain all causes explaining the associations between the variables in $\mathbf{V}$; moreover, there may not be variables whose association is not due to a causal mechanism (for instance, the example of supposed correlation between water levels in Venice and gain prices in England, cited by Sober, 1987). Hausman & Woodward (1999) challenge and discuss the general validity of the causal Markov condition: in an actual causal analysis, determining whether all associations can be explained causally boils down to determining whether we can reasonably expect the variables $\mathbf{V}$ to be interpreted as causing each other or not—the spurious effects of lack of causal sufficiency notwithstanding. For instance, the causal Markov condition probably does not hold for variables representing the pixels of handwritten characters, whereas it can be expected to hold in an acyclic causal system describing risk factors for a certain disease or natural disaster (Scheines, 2002).[7]

[7] For realistic examples, see the causal networks used for benchmarking the algorithms, described in Section 3.4.

Concretely, the causal Markov condition ensures that there is a causal graph over $\mathbf{V}$ where all causal dependencies in the data can be represented.

The *faithfulness condition* requires the probability distribution $\mathrm{P}(\mathbf{V})$ to be DAG-isomorphic, and thus ensures, together with the causal Markov condition, that there is a graph representing all causal dependencies which additionally is a perfect map. We showed with the XOR example that not all distributions are DAG-isomorphic; determining whether such violations occur often is practice with real datasets is still an open question (Steel, 2005).

Together, the causal Markov and the faithfulness conditions thus ensure the existence of a graph $\mathcal{G}^\star$ which is a perfect map (and thus can exactly represent all dependence and independence relations in the data), and where the arcs denote direct causation (in the sense of Definition 2.6). Under these assumptions, Pearl & Verma (1991) proved that IC returned the correct CPDAG representing the equivalence class containing $\mathcal{G}^\star$.

3.1.3 The PC Algorithm

What makes the whole structure-learning problem interesting is how to efficiently implement Step 1 of the IC algorithm: it is indeed impossible to perform $d(d-1)$ full subset searches on $\mathbf{V} \setminus \{X, Y\}$. In addition to the exponential time complexity that this search would represent, performing t conditional-independence tests with a Type I error rate of α makes the probability to have no Type I error at all $(1-\alpha)^t$, which rapidly goes down as t grows. The Type II error rate itself grows as the number of variables $|\mathbf{S}_{XY}|$ in the conditioning set grows. Practical algorithms aim to keep the number of tests low and the conditioning set small.

An example of an improvement over a naïve implementation of IC is the PC algorithm (named after the first names of its authors, Peter Spirtes and Clark Glymour), which proposes an implementation of Step 1 of the IC algorithm. Pseudocode for the PC algorithm is listed in Algorithm 3.1.

PC first starts with a full graph $\mathcal{G}$, where all variables are connected with undirected edges. Then (lines 4–15), unlike IC, PC will not take an (X, Y) pair of variables and then look for a d-separating set until all possible sets have been examined, but instead has an iterative approach. In a first pass, it will check whether all variable pairs are independent given the empty set. This hopefully allows the removal of many links between all variables that are unconditionally independent in quadratic complexity. Then, it goes through all (X, Y) pairs of variables again, trying to d-separate them by conditioning on one single variable Z. As a bonus, it reuses the intermediate result from the first pass: now it does not need to search the whole set $\mathbf{V} \setminus \{X, Y\}$ for a fitting Z: searching $\mathbf{Bd}_{\mathcal{G}}(X)$ or $\mathbf{Bd}_{\mathcal{G}}(Y)$ (i.e., the variables that are still connected to X and Y in the graph being built $\mathcal{G}$) is enough, thanks to the faithfulness assumption (Spirtes et al., 2001). For the next pass, PC examines all subsets of size 2 in its search for d-separating sets, etc., until either (i) the size of the conditioning set reaches $|\mathbf{V}| - 2 = |\mathbf{V} \setminus \{X, Y\}|$ (i.e., we condition on all other variables), or (ii) in the current graph, there is no variable that still has as many edges as the size of the conditioning set for the current pass. In case (i), we have performed all possible conditional-independence tests; in case (ii), we know (provided the outcome of all previous tests was correct) that no further test would result in a new conditional-independence relation.

We can pass to PC the *fan-in parameter* f, indicating the maximum size of the conditioning sets that PC should build. It defaults to $f = |\mathbf{V}| - 2$. For a given f, this step of PC has a complexity of

$\mathcal{O}(d^{f+2})$ calls to the conditional-independence oracle, where $d = |\mathbf{V}|$ is the number of variables. In the worst case of a fully connected graph with default fan-in, the complexity is exponential.

The second step of PC (lines 16–18) is just like the second step of IC; and so is the last step. We now detail this last step, which orients the graph maximally (listed in Algorithm 3.2, separately because we will reuse it later in other algorithms).

The three rules are shown graphically in Figure 3.3. Rule (a) prevents the creation of a new V-structure. If a V-structure existed for $X \to Y \leftarrow Z$, then it would have been detected in the second step of the algorithm. Now, if we already know $X \to Y$, the only consistent orientation possibility for $Y - Z$ is $Y \to Z$. Rule (b) simply preserves acyclicity: any orientation that introduce a cycle makes the graph fall outside of what reasoning tools can work with today. Finally, in Rule (c), $X - Y$ is oriented as $X \to Y$ as this is again the only consistent orientation possibility: orienting $Y \to X$ would cause a problem with subsequent orientations of $X - Z$ and $X - W$. Either this would lead to the creation of a new, forbidden V-structure $Z \to X \leftarrow W$, or we would introduce a cycle trying to avoid the V-structure.

These three rules have been proven complete by Meek (1995): their repeated application indeed leads to the maximal possible orientation of the passed PDAG given the V-structure.

The main problem with IC and PC is that they still perform a lot of conditional-independence tests, searching many subsets, and are thus slow to build large graphs. While the general structure-learning problem is NP-hard (Chickering et al., 1994; Hulten et al., 2003) and no algorithm is known to make fewer than $\mathcal{O}(2^d)$ calls to a conditional-independence oracle (Chickering & Meek, 2006), we can still greatly reduce the constant factor and be more efficient than PC, especially in cases where we can assume a constant graphical connectivity degree.

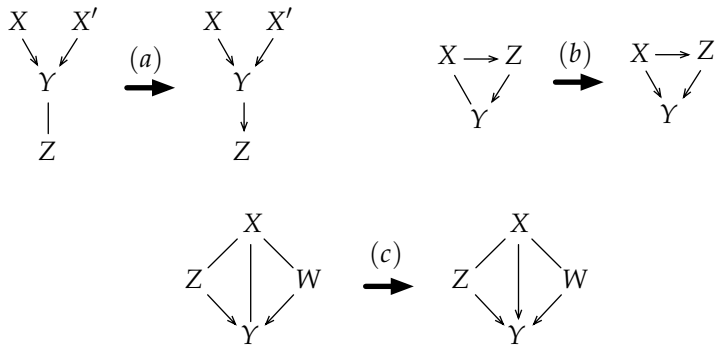


Figure 3.3: Rules to maximally orient a PDAG.

3.2 STRUCTURE LEARNING BASED ON MARKOV BLANKETS

In the remainder of this chapter, we present our contribution to structure learning in a causally sufficient context. Our main idea is to reduce the number of conditional-independence tests the algorithms have to make, and to try to keep the size of the conditioning set low: a small number of tests makes the algorithm faster; keeping a small conditioning set increases the power of the tests. To do this, we show how we can benefit from first determining a local neighborhood around each

Algorithm 3.1 The PC algorithm

procedure $\mathcal{G} = \text{PC}(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot)\ ,\ \mathbf{V},\ f\ )$
Input: $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$: a conditional-independence oracle
$\mathbf{V}$: set of variables in the analysis
f : maximum size of conditioning sets; default $= |\mathbf{V}| - 2$
Output: $\mathcal{G}$: maximally oriented partially directed acyclic graph

// Initialization
$\mathcal{G} \leftarrow$ fully connected unoriented graph over $\mathbf{V}$
$i \leftarrow 0$

// Remove extra adjacencies
while $\exists (X - Y)$ such that $|\mathbf{Bd}_{\mathcal{G}}(X)| > i$ **and** $i \leq f$ **do**
  for each $X - Y$ such that $|\mathbf{Bd}_{\mathcal{G}}(X)| > i$ **do**
    for each $\mathbf{S} \subseteq \mathbf{Bd}_{\mathcal{G}}(X) \setminus \{Y\}$ such that $|\mathbf{S}| = i$ **do**
      if $(X \perp\!\!\!\perp Y \mid \mathbf{S})$ **then**
        remove link $X - Y$ from $\mathcal{G}$
        $\mathbf{S}_{XY}, \mathbf{S}_{YX} \leftarrow \mathbf{S}$
        break
      end if
    end for
  end for
  $i \leftarrow i + 1$
end while

// V-structure detection
for each $X - Z - Y$ such that X, Y are nonadjacent **and** $Z \notin \mathbf{S}_{XY}$ **do**
  create V-structure $X \rightarrow Z \leftarrow Y$
end for

// Constraint propagation
return MAXIMALLYORIENTPDAG($\mathcal{G}$)
end procedure

Algorithm 3.2 Maximal orientation of a PDAG

procedure $\mathcal{G} =$ MAXIMALLYORIENTPDAG($\mathcal{G}$)
Input: $\mathcal{G}$: partially directed acyclic graph
Output: $\mathcal{G}$: maximally oriented partially directed acyclic graph

while $\mathcal{G}$ is changed by some rule **do**
  for each X, Y, Z such that $X \rightarrow Y - Z$ **do**
    orient as $X \rightarrow Y \rightarrow Z$ *// Rule* (a)
  end for
  for each X, Y such that $X - Y$ **and** $\exists$ directed path from X to Y **do**
    orient as $X \rightarrow Y$ *// Rule* (b)
  end for
  for each X, Y s.t. $X - Y$ **and** $\exists$ nonadjacent Z, W s.t. $X - Z \rightarrow Y$ **and** $X - W \rightarrow Y$ **do**
    orient as $X \rightarrow Y$ *// Rule* (c)
  end for
end while
return $\mathcal{G}$
end procedure

variable, and then perform the adjustments needed to transform the partial results into a correct PDAG.

Learning a local neighborhood has the important advantage that it can be cast as a *feature-selection* problem. Coming from the machine-learning community, feature selection (John et al., 1994; Guyon & Elisseeff, 2003) is a common technique that aims to reduce the number of variables used for building more efficient or more robust regression or classification models by eliminating those that are irrelevant to the target. Techniques have evolved to be able to handle nonlinear relationships between variables, redundant variables, in both discrete and continuous domains. In certain circumstances, the output of a feature-selection algorithm coincides with local neighborhoods of variables in a causal graph; this neighborhood is known as the *Markov blanket*.

The Markov blanket of a variable X is the smallest set containing all variables carrying information about X that cannot be obtained from any other variable.[8]

Definition 3.3 (Markov blanket) *The* Markov blanket *of a node X is the smallest set of variables $\mathbf{Mb}(X)$ such that $\forall Y \in \mathbf{V} \setminus \mathbf{Mb}(X) \setminus \{X\} : (X \perp\!\!\!\perp Y \mid \mathbf{Mb}(X))$.*

In this section, we show how the feature-selection task and the causal-graph-construction task can both be stated to some extent as identification of the Markov blanket.

3.2.1 Background on Feature Selection

We are given a dataset of n samples $D = \{(\mathbf{x}_i, y_i), 1 \leq i \leq n\}$. Each data point $(\mathbf{x}_i, y_i)$ has $d-1$ inputs, modeled as a vector $\mathbf{x}_i \in \mathbb{R}^{d-1}$, and an output, or *target*, $y_i \in \mathbb{R}$ (we use $d-1$ and not d for the size of $\mathbf{x}_i$ for consistency with our previous notation). The data points are assumed to be drawn i.i.d. from a joint probability distribution over the random variables $\mathbf{V} = \mathbf{X} \cup \{Y\}$. The result of the feature-selection task is a set of retained variables $\mathbf{F} \subseteq \mathbf{X}$. How many variables to retain, and which ones, depends on the particular algorithm; this selection usually maximizes some trade-off between efficiency and classification/regression error of a given learning task.

John et al. (1994) propose a classification of the input variables $\mathbf{X}$ with respect to their relevance to the target Y in terms of conditional independence. In the following definitions, we write X_i to note the ith input variable, and $\mathbf{X}_{\setminus i}$ to denote all input variables but the ith one.

Definition 3.4 (Strong relevance) *A variable X_i is said to be* strongly relevant *to the target Y if and only if*

$$\mathrm{P}(Y \mid \mathbf{X}_{\setminus i}) \neq \mathrm{P}(Y \mid \mathbf{X}_{\setminus i}, X_i).$$

A variable is strongly relevant to the target when it carries information about Y that no other variable carries. This is expressed in the definition by a change in the probability distribution of the target between conditioning on all other variables, $\mathbf{X}_{\setminus i}$, and also including X_i in the conditioning set. If X_i carries no exclusive information about Y, the two distributions will be identical.

[8] Some authors write "Markov blanket" without the notion of smallest set, and use "Markov boundary" to note the smallest Markov blanket $\mathbf{Mb}(X)$.

Definition 3.5 (Weak relevance) *A variable X_i is said to be* weakly relevant *to the target Y if and only if it is not strongly relevant and*

$$\exists \mathbf{S} \subseteq \mathbf{X}_{\setminus i} : \mathrm{P}(Y \mid \mathbf{S}) \neq \mathrm{P}(Y \mid \mathbf{S}, X_i).$$

We speak of weak relevance of a variable X_i when there exists a certain context $\mathbf{S}$ in which it carries information about the target. However, this is not necessarily exclusive information, as it may be available from other variables.

Corollary 3.6 (Irrelevance) *A variable X_i is said to be* irrelevant *to the target Y when it is neither strongly nor weakly relevant, i.e., if and only if*

$$\forall \mathbf{S} \subseteq \mathbf{X}_{\setminus i} : \mathrm{P}(Y \mid \mathbf{S}) = \mathrm{P}(Y \mid \mathbf{S}, X_i).$$

A variable is irrelevant if it carries no information about the target at all, no matter the context.

In general, the set of strongly relevant features can be empty, and weakly relevant features may still exist (Tsamardinos & Aliferis, 2003). In the next subsection, we explain how the faithfulness condition allows us to exclude this particular case. For our purposes, we assume that we use feature-selection algorithms that return the set of all strongly relevant variables, and only those. We will discuss with experiments in Section 3.4 whether this is a reasonable assumption.

In order to easily relate this discussion to structure learning, we rewrite the result $\mathbf{F}_Y$ of the feature-selection task with target Y in terms of conditional independence:

$$\mathbf{F}_Y = \big\{ X \;\big|\; (X \not\perp\!\!\!\perp Y \mid \mathbf{V} \setminus \{X, Y\}) \big\}, \tag{3.2}$$

with $\mathbf{V} = \mathbf{X} \cup \{Y\}$. This is the set of the variables that are dependent on the target Y, conditioned on all others. Note that if we repeat the feature-selection task using as target another variable $X \in \mathbf{V}$ yielding a result $\mathbf{F}_X$, we have:

$$X \in \mathbf{F}_Y \iff Y \in \mathbf{F}_X. \tag{3.3}$$

This follows as a direct consequence of (3.2) due to the symmetry of the conditional-independence relation $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ with respect to X and Y.

3.2.2 Markov Blankets in Feature Selection and Causal Graphs

We now show how feature-selection techniques can be used for causal-structure learning through properties of the Markov blankets.

Let us first note that relating feature selection and causal-structure learning is not new. Several algorithms identifying the Markov blanket of a single variable with techniques inspired from causal-structure learning have been proposed as the optimal solution to the feature-selection problem in the case of a faithful distribution (Aliferis et al., 2010*a*,*b*). Tsamardinos & Aliferis (2003) show that for faithful distributions, the Markov blanket of a variable Y is exactly the set of strongly relevant features, and prove its uniqueness. They propose the Incremental Association Markov Blanket (IAMB) algorithm to determine the Markov blanket. With the same faithfulness assumption, the Min-Max Markov Blanket algorithm (MMMB, Tsamardinos et al., 2003) identifies the Markov blanket of a variable Y by calling a subroutine Min-Max Parents and Children (MMPC). This subroutine

finds the direct parents and children of Y with association measures and conditional-independence tests. MMPC is again called on each of these nodes to find potential spouses of Y. False positives are then discarded with conditional-independence tests. MMMB was further discussed by Peña et al. (2005), who propose AlgorithmMB, a similar approach based on scores and conditional-independence tests to retrieve $\mathbf{Mb}(Y)$. The HITON_MB algorithm (Aliferis et al., 2003) is similar in its main steps, and selects an optimal subset of the Markov blanket of a target variable given the faithfulness assumption. Nilsson et al. (2007) also propose a theoretical algorithm for consistent identification of strongly relevant features in polynomial time for the class of strictly positive distributions. They also argue that some common backward feature-elimination algorithms like Recursive Feature Elimination (Guyon et al., 2002) are actually consistent, in the sense that they return the set of strongly relevant features in the large-sample limit.[9]

These are examples of using causal-structure learning or similar constraint-based techniques to help feature selection (see Guyon et al., 2007, for a review of those techniques). We wish to propose a framework to do the converse so as to benefit from feature-selection ideas for causal-structure learning. To this end, we first examine properties of the Markov blanket of a causal graph over $\mathbf{V}$. For the rest of the discussion, we assume that the joint probability distribution over all variables $\mathbf{V}$ is faithful.

If interpreted in the context of a causal graph, the Markov blanket of X coincides with certain nodes.

Property 3.7 *In a DAG $\mathcal{G}$ that has the perfect-map property, $\mathbf{Mb}(X)$ (the Markov blanket of X) is the set of parents, children, and children's parents (spouses) of X.*

> **Proof** We use the perfect-map property of $\mathcal{G}$ to use the *d*-separation and the conditional-independence criteria interchangeably.
>
> We first show the implication "Y is a parent, child, or spouse of $X \implies Y \in \mathbf{Mb}(X)$." Parents and children of X are directly linked to X. As no set can *d*-separate X from Y owing to the direct link, we conclude $Y \in \mathbf{Mb}(X)$. If Y is a spouse of X, then there is a subgraph $X \to Z \leftarrow Y$. First, $Z \in \mathbf{Mb}(X)$ because of its direct link, as shown above. Then, we have $(X \rightleftharpoons Y \mid Z)$ because conditioning on the collider Z, by definition, opens a path between X and Y. So we have to conclude $Y \in \mathbf{Mb}(X)$. This proves the first implication.
>
> We now show the implication "$Y \in \mathbf{Mb}(X) \implies Y$ is a parent, child or spouse of X." Take some $Y \in \mathbf{Mb}(X)$ and assume it is not parent, child, or spouse of X: then there is at least one path from X to Y of length 2 where the middle node is not a collider, or at least one path of length greater than 2, which, structurally, must contain at least one noncollider. In either case, the first noncollider node on this path can only be a parent, child, or spouse of X, so it is in $\mathbf{Mb}(X)$ as shown above. Moreover, conditioning on it blocks the path by definition because the node is a noncollider. This means that there is a set $\mathbf{Z}$ such that $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ and hence that Y cannot be in $\mathbf{Mb}(X)$ because the definition of $\mathbf{Mb}(X)$ requires minimality. This leads to a contradiction and proves the second implication. Together, the two implications mean that $\mathbf{Mb}(X)$ and the set of parents, children, and spouses of X are identical. □

As a direct consequence of this, note that we have:

$$X \in \mathbf{Mb}(Y) \iff Y \in \mathbf{Mb}(X).$$

[9] Actually, their definition of consistency has to do with returning the set of features relevant to the Bayes classifier, which is slightly stronger than strong relevance as used here.

The following statement is a key property of Markov blankets.

Property 3.8 (Total conditioning) *In the context of a perfect map $\mathcal{G}$, we have:*

$$\forall X, Y \in \mathbf{V}: \; X \in \mathbf{Mb}(Y) \iff (X \not\perp\!\!\!\perp Y \mid \mathbf{V} \setminus \{X, Y\}).$$

This property says that the Markov blanket of each node is the set of all variables that are dependent on it, conditioned on all other variables. In order to prove this important property, we use the following two lemmas and a corollary.

Lemma 3.9 *In a DAG $\mathcal{G}$, any (undirected) path π of length $\ell > 2$ can be blocked by conditioning on any two consecutive nodes in π.*

> **Proof** It follows from Definition 2.11 that a path π is blocked when either at least one collider (or one of its descendants) is not in the conditioning set $\mathbf{S}$, or when at least one noncollider is in $\mathbf{S}$. It therefore suffices to show that conditioning on two consecutive nodes always includes a noncollider. This is the case because two consecutive colliders would require bidirected arrows, which is a structural impossibility with simple DAGs. □

Lemma 3.10 *In a DAG $\mathcal{G}$, two nodes X, Y are d-connected given all other nodes $\mathbf{S} = \mathbf{V} \setminus \{X, Y\}$ if and only if any of the following conditions holds:*

(i) There is an arc from X to Y or from Y to X (i.e., $X \to Y$ or $X \leftarrow Y$);
(ii) X and Y have a common child Z (i.e., $X \to Z \leftarrow Y$).

> **Proof** We prove this by first proving an implication and then its converse.
>
> ($\Longleftarrow$) If (i) holds, then X and Y cannot be d-separated by any set. If (ii) holds, then Z is included in the conditioning set and d-connects X and Y by Definition 2.11.
>
> ($\Longrightarrow$) X and Y are d-connected given a certain conditioning set when at least one path remains open. Using the conditioning set $\mathbf{S}$, paths of length > 2 are blocked by Lemma 3.9 since $\mathbf{S}$ contains all nodes on those paths. Paths of length 2 contain a mediating variable Z between X and Y; by Definition 2.11, $\mathbf{S}$ blocks them unless Z is a common child of X and Y. Paths of length 1 cannot be blocked by any conditioning set. So the two possible cases where X and Y will be d-connected are (i) or (ii). □

Corollary 3.11 *Two variables X, Y are dependent given all other variables $\mathbf{S} = \mathbf{V} \setminus \{X, Y\}$ if and only if any of the following conditions holds:*

(i) $X \multimap Y$ or $Y \multimap X$ (X causes Y or Y causes X);
(ii) $X \multimap Z \multimap Y$ (X and Y have a common effect Z).

> **Proof** This follows directly from Lemma 3.10 owing to the perfect-map structure, which ensures that there exists a DAG where conditional independence and d-separation map one-to-one. Lemma 3.10 can then be reread in terms of conditional independence and causation instead of d-separation and arcs. □

We can now prove the total-conditioning property.

> **Proof of Property 3.8** This is a direct consequence of Corollary 3.11, where points (i) and (ii) lead to the definition of the Markov blanket of Y (following Property 3.7) as (i) all its causes and effects, and (ii) the other direct causes of its effects. This is equivalent to $\mathbf{Mb}(Y)$ in $\mathcal{G}$. □

In other words, in a causal graph, the parents, children, and spouses of Y store information about Y that cannot be obtained from any other variable. Note that $\mathbf{Mb}(Y)$ then has exactly the property of the output of feature selection, $\mathbf{F}_Y$, as characterized in (3.2). This links feature selection and causal-structure learning in the sense that

$$\mathbf{F}_Y = \mathbf{Mb}(Y).$$

Note that if we work with a DAG-isomorphic distribution, then we know that a perfect map exists and that the set of parents, children, and spouses of Y is uniquely defined: this means that $\mathbf{Mb}(Y)$ is unique, too.

Markov blankets alone, however, do not fully specify a causal graph. Thus, feature selection, even if guaranteed to find only strongly relevant features, cannot be directly used to construct the graph as we want it to be. The problem is that spouses of Y, even if not directly linked in the original graph, would be linked in $\mathbf{F}_Y$ and $\mathbf{Mb}(Y)$. An additional step is needed to transform the Markov blankets into parents, children, and spouses.

3.2.3 Recovering the Local Structure

The result of feature selection can be graphically shown by an undirected graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}_\mathcal{E} \rangle$ with $\mathcal{E} = \{-\}$, where $X - Y \iff X \in \mathbf{F}_Y$. This graph is close to the original causal graph in that it contains all arcs as undirected links, and additionally links spouses together, and is called the *moral graph* of the original directed graph (Lauritzen & Spiegelhalter, 1988, p. 166).

Definition 3.12 (Moral graph) *The* moral graph $\mathcal{G}^m$ *of a DAG* $\mathcal{G}$ *is the graph obtained by making a copy of* $\mathcal{G}$*, creating new edges between all parents of the nodes with more than one parent, and making all edges undirected.*

The extra step needed to transform this graph into a causal PDAG is the deletion of the spouse links and the orientation of the arcs, a task which we call "resolving the Markov blankets."

An existing algorithm can resolve the Markov blankets, i.e., use Markov-blanket information to infer the local structure around a node: the Grow-Shrink (GS) algorithm, proposed by Margaritis & Thrun (1999). The full algorithm first finds the Markov blanket for each variable, and performs further conditional-independence tests around each variable to infer the structure locally. It then uses a heuristic to remove cycles possibly introduced by previous steps. We list in Algorithm 3.3 (using our notation) the steps of the algorithm responsible for building the local structure using the Markov-blanket information, as this is exactly the task we address. In the code, $\mathbf{Bd}(X)$ stands for the *boundary* of X; i.e., the set of its direct neighbors in the graph $\mathcal{G}$. It is different from $\mathbf{Mb}(X)$ in that whereas $\mathbf{Mb}(X)$ is passed as input to the algorithm and is fixed, $\mathbf{Bd}(X)$ depends on the graph $\mathcal{G}$, which is modified throughout the algorithm. As before, we note a conditional-independence test with $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$. More will be said about the actual implementation of such tests in Section 3.3.

The GS algorithm makes two passes over all variables and the members of their Markov blankets (or direct neighbors in the second pass). It first removes the possible spouse links between linked variables X and Y by looking for a d-separating set around X and Y. In a second pass, it orients the arcs whenever it finds that conditioning on a middle node creates a dependency. While searching

Algorithm 3.3 Resolve the Markov blankets with the Grow-Shrink algorithm

```
1: procedure 𝒢 = ResolveMarkovBlankets_GrowShrink( (· ⫫ · | ·), Mb(·) )
   Input:  (· ⫫ · | ·) : a conditional-independence oracle
           Mb(·) : the Markov-blanket information for each node X ∈ V
   Output: 𝒢 : partially oriented DAG

   // Compute graph structure
2:  𝒢 ← moral graph according to Mb(·)
3:  for each X ∈ V and Y ∈ Mb(X) do
4:      B ← smallest set of {Bd(X) \ {Y}, Bd(Y) \ {X}}
5:      for each S ⊆ B do
6:          if (X ⫫ Y | S) then
7:              remove link X − Y from 𝒢
8:              break
9:          end if
10:     end for
11: end for

    // Orient edges
12: for each X ∈ V and Y ∈ Bd(X) do
13:     for each Z ∈ Bd(X) \ Bd(Y) \ {Y} do
14:         orient Y → X    // to be corrected if a test yields independence
15:         B ← smallest set of {Mb(Y) \ {Z}, Mb(Z) \ {Y}}
16:         for each S ⊆ B do
17:             if (Y ⫫ Z | S ∪ {X}) then
18:                 remove orientation Y → X
19:                 break
20:             end if
21:         end for
22:         if Y → X then
23:             break
24:         end if
25:     end for
26: end for

27: return 𝒢
28: end procedure
```

for the appropriate conditioning set, GS selects the smallest base search set (set **B** in Algorithm 3.3) for each phase. This has two very desirable effects. First, it reduces the number of tests, which is useful because each phase contains a subset search, exponential in time complexity with respect to the searched set. Second, it reduces the average size of the conditioning set, which increases the power of the statistical tests, thus helping to reduce the number of Type II errors.

While the GS approach considerably reduces the number of tests to be performed with respect to a large subset search, it is possible to perform fewer tests while still correctly identifying the structure and orienting the arcs, and decreasing the average conditioning-set size. A helpful observation is that orientation and removal of the spouse links can be done together in a single pass. We know, as discussed in the previous section, that only arcs in V-structures can be oriented: fortunately, V-structures are identified exactly when we identify a spouse link to be removed. Two spouses X and Y that are not directly linked in the original causal graph can be d-separated by some set of nodes. Thus, if we can find a set $\mathbf{S}_{XY}$ that makes X and Y conditionally indepen-

dent, we know that the link between them is a spouse link to be removed. Moreover, we know that any node Z which is part of the intersection of their Markov blankets and not included in $\mathbf{S}_{XY}$ is a collider and thus a common child, and that the triplet (X, Z, Y) is actually a V-structure $X \to Z \leftarrow Y$ in the original graph. This follows from the definition of *d*-separation. If we can find an efficient search algorithm to find such *d*-separating sets, then we can propose a fast novel structure-learning algorithms based on these principles.

An approach based on this observation has two main benefits. First, it only searches the triangles, i.e., the cliques of three nodes, in the moral graph. Assuming that the information about the Markov blanket is correct, only triangles can hide spouse links and V-structures. Second, for each connected pair $X - Y$ in a triangle, decisions about possible spouse links and arc orientation are taken together and thus faster.

Pseudocode for the proposed search algorithm is listed in Algorithms 3.4 and 3.5, where the notation $\mathcal{G}^{\backslash XY}$ denotes the moral graph $\mathcal{G}$ where all direct links involving X or Y have been removed. The algorithm uses the following concept.

Definition 3.13 (Collider sets) *In an undirected graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, let* $\mathbf{Tri}(X - Y)$ *(with $X, Y \in \mathbf{V}$ and $(X, Y) \in \mathbf{E}$) be the set of vertices forming a triangle with X and Y:*

$$\mathbf{Tri}(X - Y) = \{Z \in \mathbf{V} \mid (X, Z) \in \mathbf{E}, (Y, Z) \in \mathbf{E}\}.$$

Suppose that $\mathcal{G}$ is the moral graph of the DAG representing the causal structure of a DAG-isomorphic dataset. A set of vertices $\mathbf{Z} \subseteq \mathbf{Tri}(X - Y)$ then has the collider-set property *for the pair (X, Y) if it is the largest set that fulfills*

$$\exists \mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\} \setminus \mathbf{Z} : (X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY}) \tag{3.4}$$

$$\text{and} \quad \forall Z_i \in \mathbf{Z} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}_{XY} \cup \{Z_i\}). \tag{3.5}$$

The set $\mathbf{S}_{XY}$ is then a d-separating set for X, Y.

Collider sets are useful because they identify the common children of two nodes.

Property 3.14 *When it exists, the subset $\mathbf{Z}$ that has the collider-set property for the pair (X, Y) is the set of all direct common effects of X and Y.*

> **Proof** We prove this using $\mathbf{Z}$ and a corresponding $\mathbf{S}_{XY}$ that fulfills (3.4).
>
> ($\Longrightarrow$) ($Z_i \in \mathbf{Z} \implies X \multimap Z_i \leftarrow Y$.) By (3.4) and (3.5), we know that each Z_i opens a dependence path between X and Y (which are independent given $\mathbf{S}_{XY}$) by conditioning on $\mathbf{S}_{XY} \cup \{Z_i\}$. By Definition 2.11, conditioning on Z_i opens a path if Z_i is either a colliding node or one of its descendants. As, by definition, $\mathbf{Z} \subseteq \mathbf{Tri}(X - Y)$ holds, we are in the first case. We conclude that Z_i is a direct effect of both X and Y.
>
> ($\Longleftarrow$) ($X \multimap Z_i \leftarrow Y \implies Z_i \in \mathbf{Z}$.) Note that (3.4) and (3.5) together are implied in presence of a V-structure $X \multimap Z_i \leftarrow Y$. Thus, a direct effect is compatible with the conditions. The fact that $\mathbf{Z}$ captures all direct effects follows from the maximization of its cardinality. □

The algorithm uses the presence of a collider set for X and Y to determine that $X - Y$ is a spouse link. We need to prove that, indeed, X and Y are nonadjacent if a collider set is found. Moreover, we show unicity of the collider set.

Lemma 3.15 *In the context of a causal graph that has the perfect-map property, the set* $\mathbf{Z}$ *that has the collider-set property for a given pair* (X, Y) *is not defined if and only if* X *is a direct cause or a direct effect of* Y*. This set* $\mathbf{Z}$ *is unique for nonadjacent* X *and* Y*.*

Proof If X is a direct cause or a direct effect of Y, X and Y are adjacent in the causal graph. Thus, there can be no d-separating set $\mathbf{S}_{XY}$ as required by (3.4), and thus the collider set $\mathbf{Z}$ is not defined. If X and Y are not related by direct causation, they are not adjacent and thus a d-separating set $\mathbf{S}_{XY}$ can be found. Then, a collider set is always defined, even when there is no other node Z_i that could fulfill (3.5): in this case, $\mathbf{Z} = \emptyset$ is a valid collider set.

We now show unicity, using interchangeably the criteria of d-separation and conditional independence, as allowed by the faithfulness assumption. Suppose that, for a pair (X, Y), two sets $\mathbf{Z}$, $\mathbf{W}$ have been found that fulfill the collider-set property, with the corresponding d-separating sets $\mathbf{S}^{\mathbf{Z}}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\} \setminus \mathbf{Z}$ and $\mathbf{S}^{\mathbf{W}}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\} \setminus \mathbf{W}$ fulfilling (3.4). Let $\mathbf{Z}^\star = \mathbf{Z} \setminus \mathbf{W}$. Owing to symmetry, proving that $\mathbf{Z}^\star$ is empty also proves $\mathbf{Z} = \mathbf{W}$.

Suppose that $\mathbf{Z}^\star \neq \emptyset$; i.e., that we can find at least one Z in $\mathbf{Z}^\star$. Then, by definition, we have that $\left(X \perp\!\!\!\perp Y \mid \mathbf{S}^{\mathbf{Z}}_{XY}\right)$ and $\left(X \not\perp\!\!\!\perp Y \mid \mathbf{S}^{\mathbf{Z}}_{XY} \cup \{Z\}\right)$. We now have two cases: either (i) $Z \notin \mathbf{S}^{\mathbf{W}}_{XY}$, or (ii) $Z \in \mathbf{S}^{\mathbf{W}}_{XY}$. In the former case (i), consider the set $\mathbf{W}' = \mathbf{W} \cup \{Z\}$. Then $\mathbf{W}'$ also fulfills the collider-set property with the same d-separating set $\mathbf{S}^{\mathbf{W}}_{XY}$: the only additional condition is $\left(X \not\perp\!\!\!\perp Y \mid \mathbf{S}^{\mathbf{W}}_{XY} \cup \{Z\}\right)$. This holds because, as shown by Property 3.14, Z is a direct child of X and Y, and conditioning on it opens a path, no matter what the conditioning set is. But all this is in contradiction with the definition stating that any set fulfilling this property must be the largest set to do so, because the cardinality of $\mathbf{W}'$ is greater than that of $\mathbf{W}$.

In the latter case (ii), the d-separating set $\mathbf{S}^{\mathbf{W}}_{XY}$ contains Z. But this is impossible owing to the same reason that Z is a direct child of both X and Y and that thus any set containing Z cannot d-separate X and Y. We therefore conclude $\mathbf{Z}^\star = \emptyset$ and $\mathbf{Z} = \mathbf{W}$, which leads to the uniqueness of the set fulfilling the collider-set property. □

The purpose of Algorithms 3.4 and 3.5 is thus to find these collider sets (Algorithm 3.5 is listed separately because it is reused in the next chapter). The outer loop goes over all triangle links and performs a collider-set search for each of them. Let $X - Y$ be one of these links: if it is not a spouse link, the search procedure will return **null** as d-separating set. Otherwise, it will return a (possibly empty) set for X and Y.[10] The collider set can be inferred by removing the d-separating set from the triangle nodes $\mathbf{Tri}(X - Y)$: as $\mathbf{Tri}(X - Y)$ contains nodes on a path of length 2 between X and Y, finding a d-separating set that does not contain some of these nodes proves that they can only be colliders according to the definition of d-separation.[11] For instance, if the procedure produces an empty set for a given linked pair $X - Y$, then X and Y are unconditionally independent, and therefore all nodes in $\mathbf{Tri}(X - Y)$ are colliders.

Two caveats must be observed during this search, however. First, there might be other active, d-connecting paths between X and Y that are not going through any node of $\mathbf{Tri}(X - Y)$. Those nodes must be blocked by appropriate conditioning on the boundary of X or Y as determined by the base conditioning set at line 2 of Algorithm 3.5. Second, this base conditioning set must be checked not to include any descendant of possible colliders. If it did, it would open a d-connecting path as per Definition 2.11. This check is performed at lines 8–15. At line 8, we build a set $\mathbf{D}$ that includes all possible descendants of currently conjectured colliders that intersect the set $\mathbf{B}$. The subsequent loop makes sure none of these is opening a path between X and Y.

[10] Note that returning an empty d-separating set in $\mathbf{S}_{XY}$ is different from returning **null**, signaling the absence of any such set when there is direct causation between X and Y.

[11] The next paragraphs describe patterns where this is not exactly true and show how the algorithm still deals with them correctly.

Algorithm 3.4 Resolve the Markov blankets with collider sets

procedure $\mathcal{G} = \text{RESOLVEMARKOVBLANKETS\_COLLIDERSETS}(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot)\,,\ \mathbf{Mb}(\cdot)\ )$
Input: $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$: a conditional-independence oracle
$\mathbf{Mb}(\cdot)$: the Markov-blanket information for each node $X \in \mathbf{V}$
Output: $\mathcal{G}$: partially oriented DAG

$\mathcal{G} \leftarrow$ moral graph according to $\mathbf{Mb}(\cdot)$
$\mathbf{C} \leftarrow \{\}$, an empty list of orientation directives

for each edge $X - Y$ part of a fully connected triangle **do**
  $\mathbf{S}_{XY} \leftarrow \text{FINDDSEPARATINGSET}(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot)\,,\ \mathcal{G},\ X - Y\ )$
  if $\mathbf{S}_{XY} \neq$ **null then** *// save orientation directives*
    remove link $X - Y$ from $\mathcal{G}$
    $\mathbf{Z} \leftarrow \mathbf{Tri}(X - Y) \setminus \mathbf{S}_{XY}$ *// this is the collider set*
    for each $Z \in \mathbf{Z}$ **do**
      $\mathbf{C} \leftarrow \mathbf{C} \cup \{(X \rightarrow Z \leftarrow Y)\}$
    end for
  end if
end for

for each orientation directive $(X \rightarrow Z \leftarrow Y) \in \mathbf{C}$ **do** *// orient edges*
  if edges $X - Z$ and $Y - Z$ still exist in $\mathcal{G}$ **then**
    create V-structure $X \rightarrow Z \leftarrow Y$
  end if
end for

return $\mathcal{G}$
end procedure

Algorithm 3.5 Find a *d*-separating set for a triangle link in a moral graph

procedure $\mathbf{S}_{XY} = \text{FINDDSEPARATINGSET}(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot)\,,\ \mathcal{G},\ X - Y\ )$
Input: $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$: a conditional-independence oracle
$\mathcal{G}$: a moral graph (undirected)
$X - Y$: an edge in $\mathcal{G}$ that can be a spouse link
Output: $\mathbf{S}_{XY}$: if non-**null**, a set such that $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$

$\mathbf{B} \leftarrow$ smallest set of $\{\mathbf{Bd}(X) \setminus \mathbf{Tri}(X - Y) \setminus \{Y\},\ \mathbf{Bd}(Y) \setminus \mathbf{Tri}(X - Y) \setminus \{X\}\}$
for each $\mathbf{S} \subsetneq \mathbf{Tri}(X - Y)$ **do** *// subset search*
  $\mathbf{Z} \leftarrow \mathbf{B} \cup \mathbf{S}$
  if $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ **then**
    return $\mathbf{Z}$
  end if
  $\mathbf{D} \leftarrow \mathbf{B} \cap \{\text{nodes reachable by } W \text{ in } \mathcal{G}^{\setminus XY} \mid W \in (\mathbf{Tri}(X - Y) \setminus \mathbf{S})\}$
  $\mathbf{B}' \leftarrow \mathbf{B} \setminus \mathbf{D}$
  for each $\mathbf{S}' \subsetneq \mathbf{D}$ **do** *// descendant of collider may be opening a path*
    $\mathbf{Z} \leftarrow \mathbf{B}' \cup \mathbf{S}' \cup \mathbf{S}$
    if $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ **then**
      return $\mathbf{Z}$
    end if
  end for
end for

return null
end procedure

Theorem 3.16 *In the large-sample limit, for DAG-isomorphic, causally sufficient datasets, the procedure* ResolveMarkovBlankets_ColliderSets() *correctly identifies all V-structures and all spouse links, assuming consistent statistical tests.*

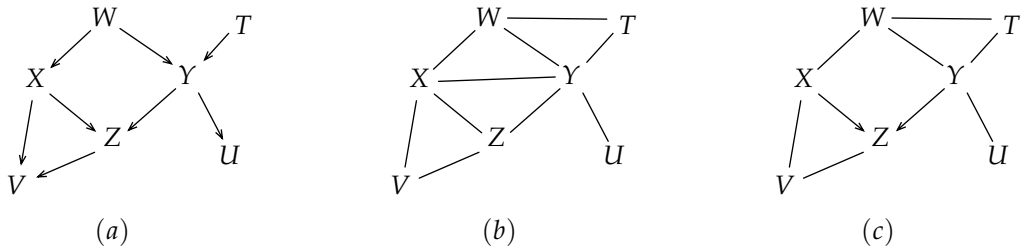


Figure 3.4: Sample local causal structure (a) and corresponding moral graph (b). On (c), the spouse link and orientation information that the collider-set search for the linked pair $X - Y$ gives.

Before proving this theorem, we begin with a graphical example. Consider the sample local structure in Figure 3.4, imagine it is part of a larger network, and suppose we are performing the search done by Algorithm 3.5. We are looking for a d-separating set for X and Y. Looking at the original graph, we see that $\{W\}$ is the smallest such set; let us see how the algorithm finds it. We have: $\mathbf{Tri}(X - Y) = \{W, Z\}$, $\mathbf{Bd}(X) = \{W, Y, Z, V\}$ and $\mathbf{Bd}(Y) = \{W, X, Z, U, T\}$. The base conditioning set $\mathbf{B}$ will thus be the smallest of $\big\{\{V\}, \{U, T\}\big\}$, thus $\mathbf{B} = \{V\}$. At this stage, conditioning on V is justifiable: one cannot exclude situations where X and Y are d-connected given the empty set through T and V, for instance if T and V both had a common cause farther away in the network. But actually in this example, all tests containing V in the conditioning set yield dependence, because V is a descendant of the collider Z and thus opens a path by the definition of d-separation. Eventually, in the iteration where $\mathbf{S} = \{W\}$, we will find conditional independence in the nested loop at lines 10–15. As $\mathbf{Tri}(X - Y) \setminus \mathbf{S} = \{Z\}$, $\mathbf{D}$ will be assigned the value $\{V\}$ and $\mathbf{B}'$ will be empty, so that we will perform exactly one extra test at line 12 with the conditioning set $\mathbf{S}_{XY} = \{W\}$, which yields independence. This in turn allows us, back in the main procedure in Algorithm 3.4, to identify the link $X - Y$ as a spouse link and determine (line 9) that the set $\mathbf{Tri}(X - Y) \setminus \mathbf{S}_{XY} = \{Z\}$ is the set of all direct effects of X and Y; i.e., fulfills the collider-set property.

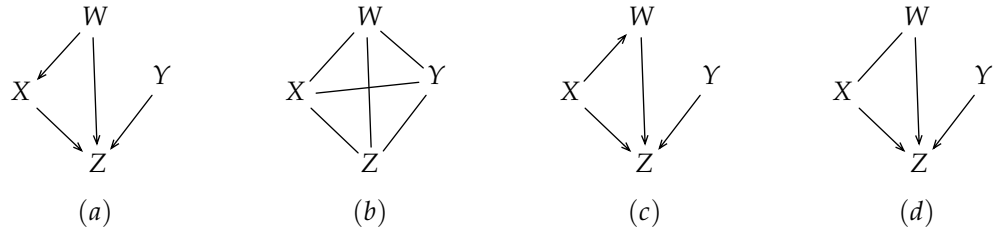


Figure 3.5: Another sample local causal structure (a) and corresponding moral graph (b). On (c), a wrong result if orientation is done immediately at line 10 of Algorithm 3.4. On (d), the correct (non-)orientation if the condition at line 15 is added.

For some structures, the order in which arcs are removed and oriented must happen such that all spouse links are removed before proceeding to orientation. Consider another example, shown in Figure 3.5, and suppose again that we are looking for a d-separating set for the pair (X, Y). As X and Y are unconditionally independent, $\mathbf{S}_{XY} = \emptyset$ is a valid d-separating set. We may thus remove

the link $X - Y$, and, considering that $\mathbf{Tri}(X - Y) = \{W, Z\}$, we could want to orient $X \to Z \leftarrow Y$ and $X \to W \leftarrow Y$ (leaving the spouse link $W - Y$ to be removed later). This would be wrong, precisely because $W - Y$ is a spouse link, and thus the orientation $X \to W \leftarrow Y$ is not allowed if one of the links to be oriented does not actually exist in the original graph, by (2.19). This is the reason why all orientation directives are saved in a list **C** at line 10 of Algorithm 3.4. After all spouse links have been removed, the orientations are done at line 16 only when both links to be oriented still exist, thus ensuring the existence of the V-structure $X \to Z \leftarrow Y$.

We now prove more formally that this algorithm is correct.

> **Proof of Theorem 3.16** First, we note that in a moral graph, a node X is connected to its parents, children, and spouses. Thus, all spouse links to be removed are in the moral graph, and, by the definition of spouse, each spouse link between X and Y corresponds to at least one unshielded collider for the pair (X, Y). Additionally, by the definition of unshielded collider, X and Y are nonadjacent, so that for each spouse link $X - Y$ there is a set $\mathbf{S}_{XY}$ such that $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$ by the contraposition of (2.17). So, when such a set $\mathbf{S}_{XY}$ is found, the link $X - Y$ is removed, and for each Z such that $X - Z - Y$ and $Z \notin \mathbf{S}_{XY}$, we orient the triplet as $X \to Z \leftarrow Y$ for the exact same reason that allows IC (or PC) to do the same in Step 2 of the algorithm (Pearl, 2000): the V-structure-identification property (2.19). The proof boils down to proving that the proposed search procedure always identifies a d-separating set $\mathbf{S}_{XY}$ when there is one.
>
> If some S_{XY} exists, then the link between X and Y is a spouse link by the definition of a moral graph, which implies that X and Y have a nonempty set of common effects **Z**. Each $Z \in \mathbf{Z}$ is linked to both X in Y and is thus in $\mathbf{Tri}(X - Y)$ by definition. Let us assume we can d-separate X and Y by some set: then, by the definition of d-separation, only conditioning on a common effect or a descendant of a common effect can create a dependency. In Algorithm 3.5, all possible colliders (line 3) and descendants of currently conjectured colliders (line 8) undergo a subset search, such that there will always be one iteration where all colliders and their descendants are left out of the conditioning set. It is then enough to show that all d-connecting paths between X and Y that are not due to conditioning on a collider or collider's descendant go through the base conditioning set as determined at line 2.
>
> To prove this, we note that the subset search at line 3 will always go through an iteration where it blocks all such d-connecting paths of length 2, i.e., patterns of the type $X \to W \to Y$ and $X \leftarrow W \to Y$. As a direct consequence of our working on the moral graph, all longer dependency paths go both through a node W in the set of immediate neighbors $\mathbf{Bd}(X)$ of X, and through a node in $\mathbf{Bd}(Y)$. Let us look at $\mathbf{Bd}(X)$. We have two cases: either (i) $W \in \mathbf{Tri}(X - Y)$ and will eventually be blocked by the subset search at line 3, or (ii) $W \in \mathbf{Bd}(X) \setminus \mathbf{Tri}(X - Y)$ (and thus $W \in \mathbf{Bd}(X) \setminus \mathbf{Tri}(X - Y) \setminus \{Y\}$ because $W \neq Y$). This set is exactly the set selected as base conditioning set at line 2, blocking all such paths, up to some symmetry with Y. Our being able to choose the smaller of the two possible base conditioning sets is due to reasons of symmetry. □

The complexity of the whole algorithm iterating over all triangle links, in terms of number of conditional-independence tests, is $\mathcal{O}(d^2 2^\alpha)$, where d is the number of variables in the analysis and $\alpha = \max_{X \in \mathbf{V}} |\mathbf{Mb}(X)| - 1$. In the worst case of a fully connected graph, where $\mathbf{Mb}(X) = \mathbf{V} \setminus \{Y\}$, the complexity is exponential in the number of variables owing to the subset search. But in practice, the original graphs are often sparse enough so that the actual run time is not exponential. Many algorithms (e.g., MMMB, HITON_MB, AlgorithmMB, GS) perform subset searches in the (possibly augmented) Markov blanket and thus rely on graph sparseness to be efficient. Although the complexity of the procedure RESOLVEMARKOVBLANKETS_COLLIDERSETS() is the same as that of RESOLVEMARKOVBLANKETS_GROWSHRINK(), we show in the experimental results in Section 3.4 that in practice, the former performs fewer tests with a smaller average conditioning-set size, while still providing comparable accuracy in structure learning.

We did not exclusively design the algorithm so as to always use the smallest possible conditioning set for the tests: there is a trade-off between obtaining the minimal possible conditioning set and keeping the total number of tests low in the average case. Later, in the empirical evaluation of this algorithm, we examine three behavioral criteria: the total number of tests, the average size of the conditioning set, and the maximum size of the conditioning set.

3.2.4 A Generic Algorithm Based on Feature Selection

The subroutine explained in the previous section lets us take a list of Markov blankets for each variable represented as the moral graph, and efficiently turn it into a causal PDAG. This enables us to use the output of feature-selection methods for causal-structure learning in a generic approach. Algorithm 3.6 lists pseudocode for the three main steps of this approach:

1. Find the conjectured Markov blanket of each variable with feature selection and build the moral graph;
2. Remove spouse links and orient V-structures using collider sets;
3. Propagate orientation constraints.

The constraint-propagation rules of Step 3 are already known to us: they are those used by the PC algorithm, and were listed in Algorithm 3.2.

The challenge with this approach is twofold. One issue is efficiency: consistent but slow feature-selection algorithms will not beat existing causal learning algorithms, as they have to be run as many times as the number of variables d. The second, and larger, issue is that consistent feature-selection algorithms are needed in order to prove the correctness of this generic algorithm, in the sense that the result of feature selection should be equal to the set of strongly relevant features. This requirement is not always fulfilled. Hardin et al. (2004) study an Support Vector Machine (SVM) classifier (Bruges, 1998) and discuss feature selection based on the $\mathbf{w}$ weights: although irrelevant variables are not selected in the large-sample limit, they show that the weights of the weakly relevant variables can be arbitrarily close to those of the strongly relevant variables owing to the large-margin behavior of SVMs. Forward feature selection has been shown to miss strongly relevant variables (Guyon & Elisseeff, 2003). Nilsson et al. (2007) also describe forward selection as inconsistent, but claim that backward feature elimination is actually consistent in the large-sample limit.[12] For finite datasets, Statnikov et al. (2006) further show (among others) that even the weights of the irrelevant variables can become greater than those of relevant variables, and that weakly relevant variables can be selected more often than strongly relevant variables in some cases.

These considerations are taken into account in our approach. In the next section, we describe an instantiation of the generic algorithm with an existing backward feature-elimination algorithm. Expecting the feature selection to be too inclusive, i.e., to include features that are not strongly relevant, we add the filtering condition at line 5 of the generic outline in Algorithm 3.6: in order to link X and Y in the moral graph, we require the feature selection performed for X to have selected variable Y, and conversely, we require X to have been selected by the feature selection performed for Y. This does not theoretically guarantee the absence of "false positives," however. Further in the section, we replace the feature-selection step with a provably consistent algorithm in the multivariate Gaussian case, and analyze its complexity and behavior.

[12] This is subject to the assumption that the underlying classifier must itself be consistent, in the sense that it must return the Bayes classifier in the large-sample limit.

3.3 ALGORITHMS FOR CAUSAL FEATURE SELECTION

In this section, we show two algorithms (and a variant) as instantiations of the generic approach previously described. First, we explain an algorithm based on the Recursive Feature Elimination (RFE) algorithm (Guyon et al., 2002) as a direct application of existing methods. We then describe Total Conditioning (TC), a fast algorithm that can be proved correct under the multivariate Gaussian assumption. We also show a variant, TC_{bw}, that improves accuracy with low sample sizes by using an explicit backward feature-selection heuristic. In Section 3.4, we report on experiments including these algorithms.

3.3.1 An RFE-Based Approach

To empirically test the soundness of the approach, we propose to use RFE over a Support Vector Regression (SVR) learner (Smola & Schölkopf, 1998) with a linear kernel, assuming for this example that we will deal with multivariate Gaussian data. RFE is an instance of a backward feature-elimination algorithm. Given some learner (in this case, SVR), it iteratively trains it, ranks the features according to some criterion, and removes the feature (or the p features) with the smallest ranking criterion. This criterion can be the weights $\mathbf{w}$ attributed to the features by the learner, or some sensitivity measure of the features (Guyon et al., 2002). In our case, we have used the $\mathbf{w}$ weights of SVR.

Using RFE, the Markov-blanket identification is done in two steps:

1. Use RFE to rank the features according to their weights in the trained model and to provide what can be seen as a relevance ordering of the features;
2. Determine the size of the Markov blanket and thus the number of variables to select from the list returned by RFE.

We do not have a theoretical guarantee that RFE/SVR will return the Markov-blanket variables. Although Nilsson et al. (2007) shows that RFE/SVM as described in Guyon et al. (2002) is consistent (i.e., returns strongly relevant variables in the large-sample limit), the limitations of ranking variables on the $\mathbf{w}$ weights of an SVM with finite datasets have also been highlighted (Hardin et al., 2004; Statnikov et al., 2006). For now, we thus use this feature-selection step as a heuristic.

In order to determine the number of variables to select from the ranked list returned by RFE, we use the following criterion: starting with the first variable from the list, accept a new variable in the Markov blanket if the cross-validated training error of the SVR decreases with the new variable, and stop and return the current list if adding the next variable increases the error.

The symmetry condition (3.3), $X \in \mathbf{F}_Y \iff Y \in \mathbf{F}_X$, might not be satisfied: we rely on the check at line 5 of the generic approach of Algorithm 3.6 to make sure that we do not select spurious features in the Markov blanket. This conservative approach implies that we expect RFE to select at least all strongly relevant variables, plus possibly some others that we hope to identify with this simple condition.

As a conditional-independence test at lines 5 and 12 of the collider-set search in Algorithm 3.5, we can use the distribution-free Recursive Median (RM) algorithm proposed by Margaritis (2005) to detect the V-structure and remove the spouse links, or a z-test as used in Scheines et al. (1995) in the case of Gaussian data.

Algorithm 3.6 Causal-structure learning with feature selection

procedure $\mathcal{G} =$ GENERICSTRUCTURELEARNING(D)
  Input: D : $n \times d$ dataset with n d-dimensional data points
  Output: $\mathcal{G}$: maximally oriented partially directed acyclic graph

  // Step 1: Markov-blanket construction
  for each variable $X \in \mathbf{V}$ **do**
    $\mathbf{F}_X \leftarrow$ FEATURESELECTIONALGORITHM(X, D)
  end for
  for each pair (X, Y) such that $Y \in \mathbf{F}_X$ and $X \in \mathbf{F}_Y$ **do** *// symmetry check*
    add X to $\mathbf{Mb}(Y)$ and Y to $\mathbf{Mb}(X)$
  end for

  // Step 2: Spurious arc removal & V-structure detection
  $\mathcal{G} \leftarrow$ RESOLVEMARKOVBLANKETS($\mathbf{Mb}(\cdot)$)

  // Step 3: Constraint propagation
  $\mathcal{G} \leftarrow$ MAXIMALLYORIENTPDAG($\mathcal{G}$)

  return $\mathcal{G}$
end procedure

Algorithm 3.7 An RFE-based feature-selection step

procedure RFEFEATURESELECTION()
  Input: X : the target variable to perform feature selection for
  D : $n \times d$ dataset with n d-dimensional data points
  Output: $\mathbf{S}$: the set of selected variables

  $\mathbf{w} \leftarrow$ weights of $\mathbf{V} \setminus X$ according to RFE(SVR)
  $\mathbf{P} \leftarrow$ predictor variables sorted according to $\mathbf{w}$
  $\mathbf{S} \leftarrow \emptyset$
  $error_{opt} \leftarrow \mathrm{var}[X]$ *// MSE of constant function*
  $error \leftarrow$ TRAIN(cross-validated SVR with predictor $(\mathbf{P})_1$))

  while $error < error_{opt}$ **do**
    $error_{opt} \leftarrow error$
    $\mathbf{S} \leftarrow \mathbf{S} \cup \{ (\mathbf{P})_1 \}$ *// add beneficial predictor*
    $\mathbf{P} \leftarrow \mathbf{P} \setminus \{ (\mathbf{P})_1 \}$
    $error \leftarrow$ TRAIN(cross-validated SVR with predictors $\mathbf{S} \cup \{ (\mathbf{P})_1 \}$)
  end while

  return S
end procedure

Although we expect the resulting graph to be accurate in the large-sample limit (see Section 3.4), we also expect the run time of such an approach to be much higher compared to existing algorithms. Training the SVR has a cubic complexity in terms of the number of samples, $\mathcal{O}(n^3)$. To get an accurate ranking, RFE runs the training $d-1$ times. Then, a new SVR learner is trained and cross-validated several times (we used a 5-fold cross-validation) to get the validation error, which is repeated for each variable in the actual Markov blanket. The complexity for the whole feature-selection step is then $\mathcal{O}(d^2n^3)$, with a large constant factor. We thus emphasize that this RFE-based feature selection is not meant as a valid practical instantiation of the generic algorithm, but rather as a proof of concept to validate the approach. In order to be practical, the feature-selection step has to be redesigned so that it is done efficiently when run for all variables. This is what the next algorithm is meant to address in the specific case of multivariate Gaussian variables.

3.3.2 The TC Algorithm

We now propose in the procedure TCFEATURESELECTION() (Algorithm 3.8) another instantiation of the feature-selection call at line 3 of the generic approach of Algorithm 3.6. (We write "TC" to refer to the whole algorithm and not only to the feature-selection procedure, referred to as TCFEATURESELECTION().)

For a given target variable X, TC estimates the coefficients of a multiple regression problem, considering all other variables $\mathbf{V} \setminus X$ as predictors. It then returns the significant predictors, according to a t-test on the coefficient of each variable. Its short listing is in Algorithm 3.8.

Algorithm 3.8 The Total Conditioning feature-selection step

1: **procedure** TCFEATURESELECTION()
   Input: X : the target variable to perform feature selection for
   D : $n \times d$ dataset with n d-dimensional data points
   Output: $\mathbf{S}$: the set of selected variables
2: $\mathbf{b} \leftarrow$ weights of $\mathbf{V} \setminus X$ in the problem of regressing X on $\mathbf{V} \setminus X$
3: $\mathbf{S} \leftarrow \{$predictors whose b weight is significant according to t-test$\}$
4: **return S**
5: **end procedure**

The conditional-independence tests to be performed at lines 5 and 12 of the collider-set search in Algorithm 3.5 are done using partial correlation.

Definition 3.17 (Partial correlation) *In a variable set* $\mathbf{V}$*, the* partial correlation *between two random variables* $X, Y \in \mathbf{V}$ *given* $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$*, noted* $\rho_{XY \cdot \mathbf{Z}}$*, is the correlation of the residuals* R_X *and* R_Y *resulting from the least-squares linear regression of* X *on* $\mathbf{Z}$ *and of* Y *on* $\mathbf{Z}$*, respectively.*

We now prove that TC is correct in the large-sample limit (subject to the consistency of the statistical tests) under the faithfulness and causal-sufficiency assumptions.

Theorem 3.18 *If the variables are jointly distributed according to a multivariate Gaussian, TC returns the maximally oriented PDAG of the Markov equivalence class of the DAG representing the causal structure of the data-generating process in the large-sample limit.*

Proof An edge is added between X and Y in the feature selection if we find that $\rho_{XY\cdot\mathbf{V}\setminus\{X,Y\}} \neq 0$. We conclude $(X \not\perp\!\!\!\perp Y \mid \mathbf{V}\setminus\{X,Y\})$ owing to the multivariate Gaussian distribution. Corollary 3.11 says that this implies that X causes Y or Y causes X, or that they share a common child. Therefore, each V-structure is turned into a triangle by the end of the feature-selection step. The collider-set search then examines each link $X - Y$ that is part of a triangle, and by Property 3.14, we know that if the search for a set $\mathbf{Z}$ that has the collider-set property succeeds, there must be no link between X and Y. We know by the same lemma that this set includes all colliders for the pair (X,Y), so that all V-structures are correctly identified. Step 3 is the same as in the IC or PC algorithms; see Pearl & Verma (1991); Spirtes et al. (2001). □

The main points leading to the correctness of TC are thus the equivalence of a zero regression weight for some predictor Y while regressing X on all variables $\mathbf{V}\setminus X$ and a zero partial correlation $\rho_{XY\cdot\mathbf{V}\setminus\{X,Y\}}$, and the fact that this is zero if and only if $(X \perp\!\!\!\perp Y \mid \mathbf{V}\setminus\{X,Y\})$ holds in a Gaussian context (Baba et al., 2004). Then, our feature-selection step (Algorithm 3.8) gives the Markov blanket for each node, and the collider-set search (Algorithm 3.4) then takes care of identifying the V-structures and removing the spouse links.

Efficiency of TC

The other advantage of using linear regression and partial correlation is that it yields a fast algorithm. Actually, *all* regression weights and parameters needed for the feature-selection step of TC can be efficiently computed by inverting the sample correlation matrix $\mathbf{R} \in [-1,1]^{d\times d}$. Building graphs by inverting the correlation matrix is what is typically done with Gaussian Markov random fields, a special case of undirected graphical models (see for instance Talih, 2003).

The weight computation and the statistical-significance tests are performed as follows. Let $\hat{b}_{ij}$ be the maximum-likelihood estimator of the true regression weight b_{ij} of predictor X_j when X_i is the dependent variable, such that it solves the multiple-regression equation for target X_i in the sense that it minimizes the sum of the squared residuals

$$SS_R = \sum_{k=1}^{n}\left(x_{ik} - \sum_{j=1, j\neq i}^{d} \hat{b}_{ij}x_{jk}\right)^2 \tag{3.6}$$

where x_{ik} is the value of X_i for the kth sample. If we have the inverse correlation matrix $\mathbf{R}^{-1} = (r^{ij})$, the vector $\mathbf{b}$ at line 2 of Algorithm 3.9 can be found in linear time: $\hat{b}_{ij} = -r^{ij}/r^{ii}$ (Raveh, 1985). For instance, the list of weights to predict variable X_1 with all others is

$$\mathbf{b}_1 = (\hat{b}_{12}, \hat{b}_{13}, \cdots, \hat{b}_{1d}) = -(r^{12}, r^{13}, \cdots, r^{1d})/r^{11}. \tag{3.7}$$

The distribution of these weights is known (Judge et al., 1988):

$$\frac{\hat{b}_{ij} - b_{ij}}{\hat{\sigma}_{ij}} \sim t_{(n-(d-1))}, \tag{3.8}$$

where $\hat{\sigma}_{ij}$ is the standard error of the jth predictor for variable X_i; i.e., that it follows a t distribution with a number of degrees of freedom df = number of samples − number of predictors $= n - (d - 1)$. For our null hypothesis $H_0 : b_{ij} = 0$, we need $\hat{\sigma}_{ij}$ in addition to $\hat{b}_{ij}$ to compute the t-statistics $\hat{b}_{ij}/\hat{\sigma}_{ij}$. The estimate of the coefficient error $\hat{\sigma}_{ij}$ can be expressed as

$$\hat{\sigma}_{ij} = \hat{\sigma}_i\sqrt{\omega^{jj}/n}, \tag{3.9}$$

where $\hat{\sigma}_i$ is an estimator of the standard error of the regression for target X_i, and ω^{jj} is the jth diagonal element of the inverse correlation matrix of the predictors (Judge et al., 1988, p. 243). (How to obtain the inverse correlation matrix of the predictors from the $\mathbf{R}^{-1}$ matrix in quadratic time is discussed in the next subsection.) The standard error $\hat{\sigma}_i$ can also be obtained in linear time from $\mathbf{R}^{-1}$ as follows.

Without loss of generality, we assume a zero mean and a unit standard deviation for all variables. Then $\sigma_i^2 = 1 - R_i^2$, where R_i^2 is the coefficient of determination of the regression for target X_i. This coefficient can be expressed as the scalar product of the $\mathbf{b}_i$ vector with the vector $\mathbf{r}_i$ of the pairwise correlation coefficients of the predictors with the target X_i (Raveh, 1985), which we read directly from the correlation matrix $\mathbf{R}$:

$$R_i^2 = \mathbf{b}_i^T \mathbf{r}_i. \tag{3.10}$$

An unbiased estimator $\hat{\sigma}_i$ for σ_i is then

$$\hat{\sigma}_i = \sqrt{\frac{n(1 - \mathbf{b}_i^T \mathbf{r}_i)}{n - d}}. \tag{3.11}$$

To sum up, we have a complexity of $\mathcal{O}(nd^3)$ to build and invert the correlation matrix, and $\mathcal{O}(d^3)$ to check for significance. This comes from having to obtain d times the inverse correlation matrix of $d-1$ predictors in $\mathcal{O}(d^2)$, and then checking their significance in linear time. The overall complexity of TC, including the collider identification and the constraint-propagation steps, is then $\mathcal{O}(nd^3 + d^2 2^\alpha)$.

The weaknesses of this approach are its infeasibility when the correlation matrix $\mathbf{R}$ does not have full rank (including the case $n < d$; i.e., when there are fewer samples than variables), the low power of the statistical tests with small datasets, and multicollinearity in the predictors. The symptoms of the last two points are that the t-tests do not refute the null hypothesis of zero weight because (i) there is not enough data to support it, or (ii) multicollinearity makes the weights lower than they should be, such that it becomes harder to interpret them as depicting the independent contribution of each predictor. We address this problem in the next subsection.

Significance Tests

Independently of low sample sizes or multicollinearity, the statistical tests on the weights of the linear-regression equations are a delicate point in TC. The choice of the Type I error rate α needs investigating as it significantly influences the result of the algorithm.

In a network of d nodes, the feature-selection step performs $d(d-1)/2$ tests to determine the undirected skeleton. We will falsely reject the null hypothesis $b_{ij} = 0$ about $m \cdot \alpha$ times on average, where $m < d(d-1)/2$ is the difference in the number of edges between the original DAG $\mathcal{G}_0$ and the complete graph. We will thus add on average $m \cdot \alpha$ wrong edges. We can set the significance level for the individual tests to be inversely proportional to $d(d-1)/2$ to avoid this problem (assuming a large m and thus rather sparse graphs), and check that it does not affect the Type II error rate too much, which we do now.

According to (3.8), the expression $(\hat{b}_{ij} - b_{ij})/\hat{\sigma}_{ij}$ follows a t distribution with $n - (d-1)$ degrees of freedom. If we call $\Psi(\cdot)$ the cumulative distribution function of a t distribution with $n - (d-1)$

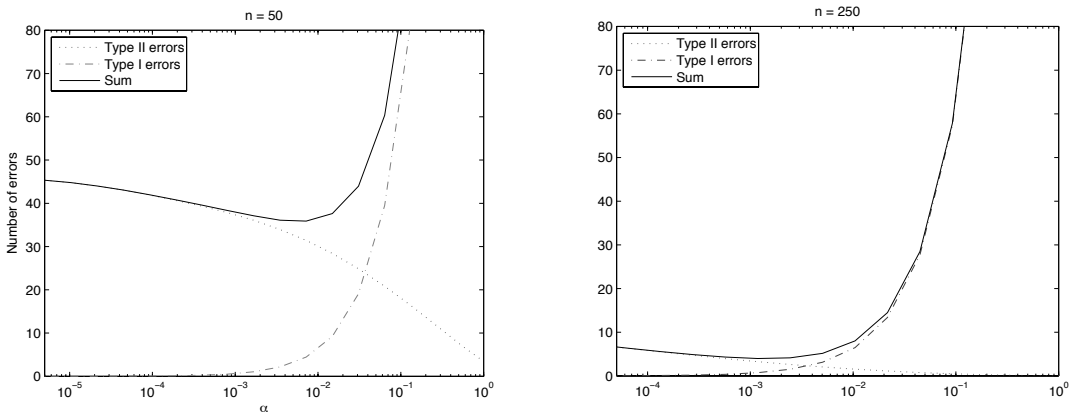


Figure 3.6: Expected Type I and II errors as a function of α with two different sample sizes: $n = 50$ and $n = 250$.

degrees of freedom, we can write the Type II error rate β for each regression weight:

$$\beta_{ij} = \Psi(\Psi^{-1}(1-\alpha/2) - |b_{ij}|/\hat{\sigma}_{ij}). \tag{3.12}$$

The values for $\hat{\sigma}_{ij}$ can be computed from the inverse correlation matrix $\mathbf{R}^{-1}$ and thus depend on the particular dataset being analyzed, but the true b_{ij} are unknown. What we could do in theory to optimize α is to minimize the average number of extraneous (T_e) and missing (T_m) links:

$$T = T_e + T_m = m \cdot \alpha + \sum_{(i,j)\in\mathbf{E}} \beta_{ij}, \tag{3.13}$$

where m is the number of edges missing in the original DAG compared to a full graph, and $\mathbf{E}$ is the set of arcs in the original DAG, so that $m + |\mathbf{E}| = d(d-1)/2$. As m, $\mathbf{E}$ and b_{ij} are unknown, we can only find an upper bound for the number of missed links T_m, provided (i) we can estimate the graph sparseness to approximate m; (ii) we assume $|b_{ij}| \geq \delta$; and (iii) we choose $\mathbf{E}^\star$ such that it maximizes the sum in (3.14), with $|\mathbf{E}^\star| = d(d-1)/2 - m$. Then we have:

$$T_m \leq \sum_{(i,j)\in\mathbf{E}^\star} \Psi(\Psi^{-1}(1-\alpha/2) - \delta/\hat{\sigma}_{ij}). \tag{3.14}$$

Actually, we found this bound to be too loose for practical use, as it significantly overestimates T_m. However, we can model the Type I and Type II error rate as a function of α for artificial problems whose sparseness and regression weights are known. This is shown in Figure 3.6 for a specific instance of an **ALARM** dataset (see Section 3.4 for details on this network) with two different sample sizes, $n = 50$ (left) and $n = 250$ (right). We did not use this information to tune α in the experiments, as it cannot be obtained without prior knowledge, but the curves showed that an α inversely proportional to $d(d-1)/2$ has the same order of magnitude as the optimal α on the datasets we analyzed.

What we also see is that the Type I error curve rapidly goes up, whereas the Type II error curve is upper-bounded by the total number of links in the original graph. In terms of pure number of errors, setting a low α will thus be more beneficial than setting a higher α to get a low β. It is worth discussing, however, depending on the particular problem to solve, which is more desirable: missing causal links or getting extra causal links. In terms of Bayesian networks, getting too few links prevents the model from being able to reconstruct the full joint probability distribution,

because we lose the independence-map property; whereas getting too many links implies having to estimate more parameters from the same data and thus makes a subsequent parameter-learning task more complex.

3.3.3 The TC_{bw} Algorithm

Despite correctness of TC, with a low number of samples n it fails to have enough evidence for rejecting the null hypothesis of zero regression weight, and thus misses links (see detailed results in Section 3.4), even for a high α. We now try to address this particular issue by successively eliminating the most insignificant predictors and reevaluating the remaining ones. This is actually a backward stepwise-regression method. Pseudocode for this heuristic is listed in Algorithm 3.9.

Intuitively, the problem to solve is that the regression weights cannot be high enough for significance with small sample sizes. By removing the most insignificant predictors and thus the most likely to actually be zero, we scale down the regression problem and increase the power of the tests. How many insignificant predictors to remove can be discussed; in our implementation, we compared $p = 1$ to $p = (\text{number of predictors})/2$ and found that the latter yielded results that were just as good, with an important speed gain.

This stepwise regression raises some issues; notably, Tibshirani (1994) argues that the repeated tests on non-changing data are biased and that the remaining **b** coefficients are too large. We thus expect TC_{bw} to be biased and to include more false positives than TC. Ideally, one would need a criterion to predict when the additional false positives would outweigh the benefits of reducing the false negatives. Whether such a criterion, which would allow us to know a priori whether TC or TC_{bw} should be used, can be found, is an open question.

Solving a standard multiple-regression problem with d predictors traditionally has complexity $\mathcal{O}(nd^3)$. Naïvely solving d regression problems $d-1$ times (in the case $p = 1$) or $\log d$ times (in the case $p = (\text{number of predictors})/2$) would have a complexity of $\mathcal{O}(nd^5)$ or $\mathcal{O}(nd^4 \log d)$, respectively. But we can avoid re-inverting matrices in the inner loop of the stepwise regression thanks to the following result.

Let $\mathbf{\Sigma} = \mathbf{X}^T\mathbf{X}$ be n times the correlation matrix $\mathbf{R}$, where $\mathbf{X}$ is the $n \times d$ matrix representing a dataset where all variables have zero mean and unit standard deviation. Then we can use $\mathbf{\Sigma}^{-1}$ to

Algorithm 3.9 The Total Conditioning/backward feature-selection step

1: **procedure** TCBWFEATURESELECTION()
   Input: X : the target variable to perform feature selection for
   D : $n \times d$ dataset with n d-dimensional data points
   Output: **S** : the set of selected variables
2: **P** $\leftarrow$ **V** $\setminus X$ *// all predictors*
3: **S** $\leftarrow \emptyset$ *// significant predictors*
4: **while** **P** $\neq \emptyset$ **and** **P** $\neq$ **S** **do**
5: **b** $\leftarrow$ weights of **P** in the problem of regressing X on **P**
6: **S** $\leftarrow$ **S** $\cup$ {predictors whose b weight is significant according to t-test}
7: **P** $\leftarrow$ **P** $\setminus$ {the p less significant predictors}
8: **end while**
9: **return S**
10: **end procedure**

linearly find the weights of the regression problems and their standard error, which are needed for the t-tests. Suppose we find that variable X_1 is the weakest predictor, and want to reevaluate the weights of the other predictors at line 5 of TC_{bw}. Let $\mathbf{X}_{\setminus i}$ be the dataset where variable X_i has been removed. Then we need the matrix $\mathbf{\Omega}^{-1}$ to solve the new problem, where $\mathbf{\Omega} = \mathbf{X}_{\setminus 1}^T \mathbf{X}_{\setminus 1}$. As a special case of Strassen's blockwise matrix inversion formula, we have:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_{11} & \mathbf{c}^T \\ \mathbf{c} & \mathbf{\Omega} \end{bmatrix}$$

$$\implies \mathbf{\Sigma}^{-1} = \begin{bmatrix} \frac{1}{\sigma_{11} - \mathbf{c}^T\mathbf{\Omega}^{-1}\mathbf{c}} & -\frac{\mathbf{c}^T\mathbf{\Omega}^{-1}}{\sigma_{11} - \mathbf{c}^T\mathbf{\Omega}^{-1}\mathbf{c}} \\ -\frac{\mathbf{\Omega}^{-1}\mathbf{c}}{\sigma_{11} - \mathbf{c}^T\mathbf{\Omega}^{-1}\mathbf{c}} & \mathbf{\Omega}^{-1} + \frac{\mathbf{\Omega}^{-1}\mathbf{c}\mathbf{c}^T\mathbf{\Omega}^{-1}}{\sigma_{11} - \mathbf{c}^T\mathbf{\Omega}^{-1}\mathbf{c}} \end{bmatrix}. \tag{3.15}$$

Let $\sigma^{ij} = (\mathbf{\Sigma}^{-1})_{ij}$ and $\mathbf{b} = \mathbf{\Omega}^{-1}\mathbf{c}$. Then $\mathbf{b}$ are the weights of the regression of X_1 on $X_2, \cdots, X_d$ and can be computed without knowing $\mathbf{\Omega}^{-1}$ (Raveh, 1985), see (3.7). We have:

$$\sigma^{11} = 1/(\sigma_{11} - \mathbf{c}^T\mathbf{b}) \tag{3.16}$$

and, $(\mathbf{\Sigma}^{-1})_{\setminus 1}$ being the matrix $\mathbf{\Sigma}^{-1}$ where the first row and column have been removed,

$$(\mathbf{\Sigma}^{-1})_{\setminus 1} = \mathbf{\Omega}^{-1} + \mathbf{b}\mathbf{b}^T/(\sigma_{11} - \mathbf{c}^T\mathbf{b}). \tag{3.17}$$

We can thus compute $\mathbf{\Omega}^{-1}$ given $\mathbf{\Sigma}^{-1}$ with complexity $\mathcal{O}(d^2)$ as follows:

$$\mathbf{\Omega}^{-1} = (\mathbf{\Sigma}^{-1})_{\setminus 1} - \sigma^{11}\mathbf{b}\mathbf{b}^T. \tag{3.18}$$

This trick is also used in TC to find the inverse correlation matrix of the predictors from the inverse correlation matrix of the whole variable set.

In TC_{bw}, we thus use (3.18) so that we never need to invert another matrix again once $\mathbf{\Sigma}^{-1}$ has been obtained. This leads to a complexity of $\mathcal{O}(d^2)$ for stepwise elimination of a predictor. In the most computationally expensive case $p = 1$, this elimination of row and column of the inverse matrix is repeated at most $d - 2$ for each variable, yielding a complexity of $\mathcal{O}(nd^4)$ for the whole feature-selection step for all variables. The overall complexity of TC_{bw} is then $\mathcal{O}(nd^4 + d^2 2^\alpha)$. We are only adding one degree of complexity in d with respect to TC with the additional stepwise regression.

3.4 Benchmarks and Experimental Results

In this subsection, we report on experiments and results on two points separately. First, we test our procedure described in Algorithm 3.4 to recover the local structure with the collider-set search given all Markov blankets, and compare it to the relevant steps of the GS algorithm, which are listed in Algorithm 3.3, with 5 different network topologies. For the sake of comparison, we also run the reference PC algorithm (Spirtes et al., 2001), initialized with the moral graph instead of the fully connected graph. Second, we conduct experiments to investigate how the full structure-learning algorithms behave. We first use the RFE-based approach. We then systematically compare TC, TC_{bw} and several reference algorithms, varying the dataset size and the network size.

3.4.1 Experimental Setup

In order to test the accuracy of the various algorithms, we chose to sample data from the following known networks, from the Bayes net repository (Elidan, 2001):

- **ALARM** (Beinlich et al., 1989). This network has become a de-facto standard benchmark for structure-learning algorithms: it contains 37 nodes, 46 arcs, 4 of which are undirected in the PDAG of the equivalence class. It was originally designed to help interpret monitoring data to alert anesthesiologists to various situations in the operating room. It is depicted in Figure 3.7.
- **INSURANCE** (Binder et al., 1997): 27 nodes, 52 arcs, 18 being undirected in its PDAG. It was designed to evaluate car-insurance risks. This network has fewer nodes than **ALARM** but is denser, see Figure 3.8.
- **HAILFINDER** (Abramson et al., 1996): 56 nodes, 66 arcs, 17 of which are undirected in its PDAG. It is a normative system that forecasts severe summer hail in northeastern Colorado. See Figure 3.9.
- **CARPO** (created by Alex Dagum with contributions from Mark Peot): 61 nodes, 74 arcs, 24 being undirected in its PDAG. It is meant to help diagnose the carpal-tunnel syndrome. The version we used has three disconnected subgraphs, one of which is a single variable, and a relatively flat causal structure, as can be seen in Figure 3.10.
- A subset of **DIABETES** (Andreassen et al., 1991) with 104 nodes, 149 arcs, 8 being undirected in its PDAG, which was designed as a preliminary model for insulin-dose adjustment. This subset is made of 6 repeating patterns (there are 24 in the original network) of 17 nodes, plus 2 external nodes linked to every pattern. The first two of these patterns are shown in Figure 3.11.

We performed three series of experiments.

1. We compared our algorithm resolving the Markov blanket to the relevant steps of the Grow-Shrink algorithm, as described in Subsection 3.2.3, and to PC;
2. We tested the RFE-based approach and compared it to PC;
3. Finally, we compared TC and TC_{bw} to three reference algorithms and examined their accuracy, run time, and number of tests while varying the network structure, the network size, and the sample size.

The chosen reference algorithms are:

1. The PC algorithm. PC is, like TC and TC_{bw}, exponential in the worst case, when graphs are not sparse enough: we discuss which structural elements make PC or TC exhibit the exponential behavior;
2. The full Grow-Shrink algorithm, as described in Margaritis & Thrun (1999);
3. A state-of-the-art Bayesian structure-learning algorithm that works with continuous datasets, the Bach-Jordan scoring algorithm (Bach & Jordan, 2003), coupled with a greedy search in the space of DAGs. Note that Bayesian structure-learning algorithms are often score-based and return fully oriented DAGs. Maximizing the chosen score function might not minimize the number of structural errors, as we report on these results.

For all simulation experiments, we generated the datasets by using the 5 graphs as a structure for a linear structural-equations model: the parentless variables were sampled as Gaussians with

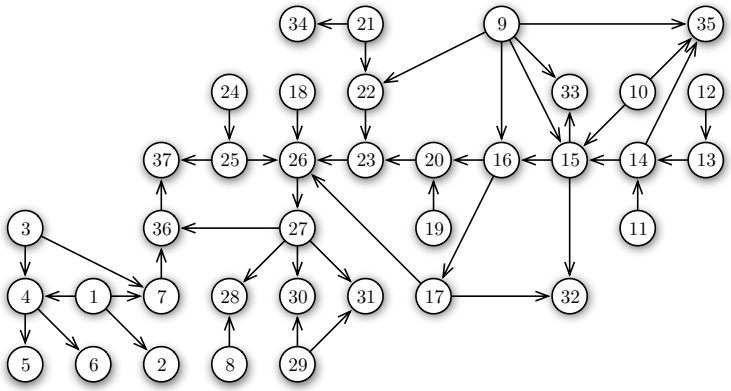


Figure 3.7: The ALARM network.

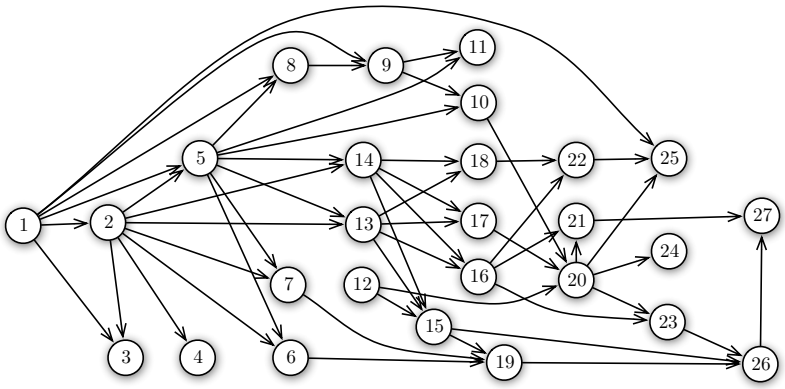


Figure 3.8: The INSURANCE network.

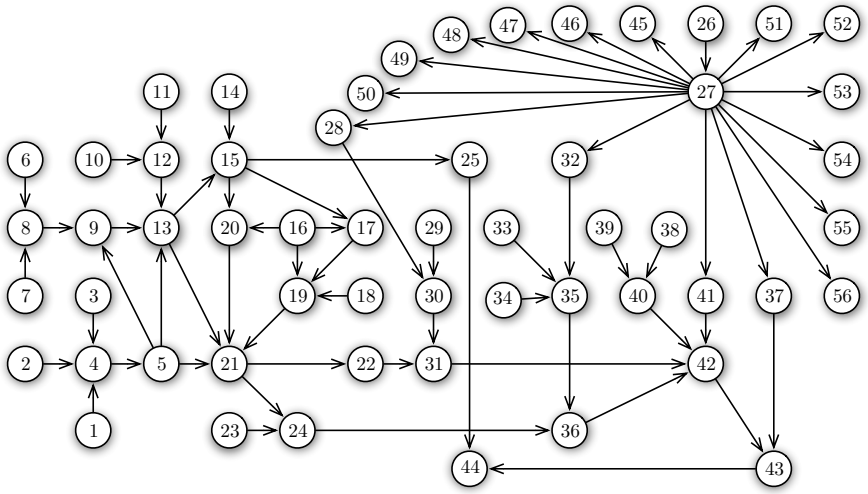


Figure 3.9: The HAILFINDER network.

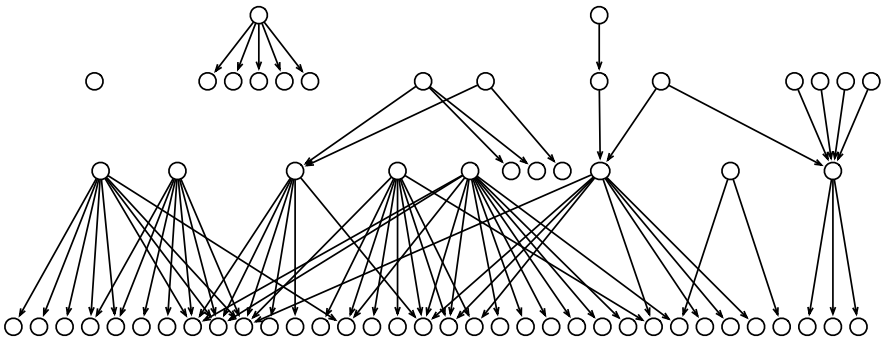


Figure 3.10: The **Carpo** network.

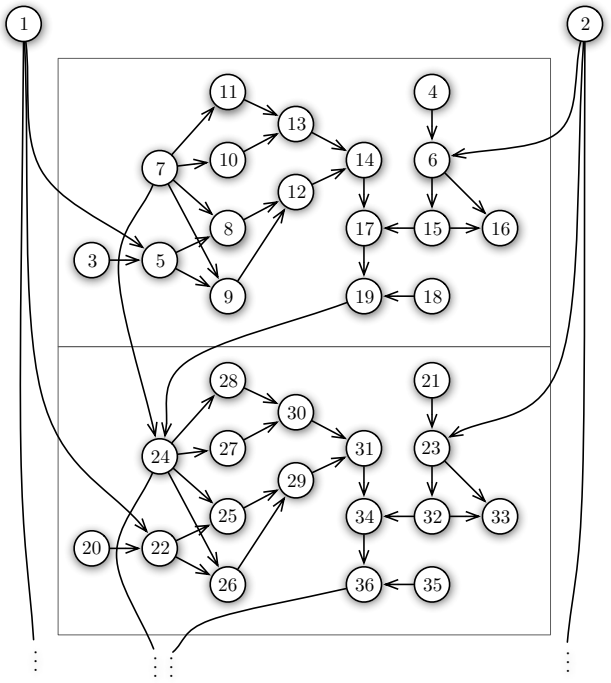


Figure 3.11: Two of the six patterns of the **Diabetes** network.

zero mean and unit standard deviation; the other variables were defined as a linear combination of their parents with coefficients randomly distributed uniformly between 0.2 and 1, similarly to what was done in Scheines et al. (1995) for the evaluation of PC. The disturbance terms were also normally distributed. We compared the number of tests, the size of the conditioning sets, and the structural errors in case of runs with artificial data. A structural error is an arc addition, deletion, or reversal with respect to the original graph.

We used the implementation of PC proposed by Leray & François (2004) in the BNT Structure Learning Matlab package. The implementation of TC and TC_{bw} was also done in Matlab. The statistical tests were done using Fisher's z-transform of the partial correlation, unless otherwise stated. For PC and GS, we chose the default value of $\alpha = 0.05$; we note though that the optimal value of α is problem dependent and that especially with low sample sizes, hand tuning α can return better results than those listed here. For both TC and TC_{bw}, we set $\alpha = 2/(d(d-1))$, according to the discussion at the end of Subsection 3.3.2.

3.4.2 Results and Discussion

We split the results and the discussion in three separate subsections: first, Markov-blanket resolution, then RFE approach, and finally the full TC and TC_{bw} algorithms.

Local-Structure Recovery with Markov-Blanket Information

In this series of experiments, we compare RESOLVEMARKOVBLANKETS_COLLIDERSETS() (denoted by CS) to RESOLVEMARKOVBLANKETS_GROWSHRINK() (denoted by GS) and to a modified version of PC, where the graph being built is initialized with the moral graph (instead of the full graph in the original version of PC). This represents exactly the Markov-blanket information available to the two other algorithms and allows a direct comparison. Note that we observe the PDAG that PC obtains *before* the constraint-propagation step building the maximally oriented PDAG, such that, in all three tested algorithms, we only expect the V-structures to be oriented.

We tested the three algorithms on each network using two methods to check for conditional independence: first, using a conditional-independence oracle with the original graph (which is equivalent to a perfect test); and second, using Fisher's z-transform of the sample partial-correlation coefficient as computed on artificial data, with significance $\alpha = 0.05$. Using the oracle always yields correct graphs.

Table 3.1 shows the results of these experiments. We first list the results obtained when using the conditional-independence oracle to decide upon conditional independence. For GS, we ran two versions of Algorithm 3.3: one, which we name GS(1), where the subset searches at lines 5 and 16 proceed with decreasing sizes of the chosen subset $\mathbf{S}$, and another, GS(2), with increasing subset sizes. GS(1) usually leads to fewer tests, but with larger conditioning sets. The order of the subset searches for our method (lines 3 and 10 in Algorithm 3.5) was fixed to decreasing subset sizes, as this always led to fewer tests *and* smaller conditioning sets.

The results for the modified PC algorithm are only shown for the sake of comparison: PC is a general-purpose algorithm which is not specialized in such local-structure recognition given the Markov blankets. What the comparison shows, however, is that, whenever this Markov-blanket information is available or cheap to obtain, there are much more efficient approaches.

Algorithm	ALARM	INSURANCE	HAILFINDER	CARPO	DIABETES
modified PC					
# tests	11331	773572	19543985⋆	2025250⋆	93134⋆
avg. $\lvert\mathbf{Z}\rvert$	4.36	7.65	5.75⋆	5.47⋆	4.64⋆
max $\lvert\mathbf{Z}\rvert$	10	16	6⋆	6⋆	6⋆
GS(1)					
# tests	1485	6435	2809	209342	5414
avg. $\lvert\mathbf{Z}\rvert$	2.62	3.63	2.66	7.46	2.73
max $\lvert\mathbf{Z}\rvert$	8	11	7	15	10
GS(2)					
# tests	1472	7180	2979	200621	6197
avg. $\lvert\mathbf{Z}\rvert$	2.20	3.05	2.31	7.39	2.39
max $\lvert\mathbf{Z}\rvert$	7	8	7	15	8
CS					
# tests	214	1288	593	294	943
avg. $\lvert\mathbf{Z}\rvert$	1.80	2.69	2.30	1.79	2.13
max $\lvert\mathbf{Z}\rvert$	5	6	6	8	7

Table 3.1: Number of tests and size of the conditioning sets (noted $|\mathbf{Z}|$) as performed by various algorithms to recover the local network structure of the networks given perfect Markov-blanket information. The star (⋆) notes PC results where the maximum size of the conditioning set has been set to 6 to avoid prohibitive run times.

GS(1) and GS(2) are close to one another in all scores, and outperform PC by several orders of magnitude in the number of tests and significantly in average and maximum size of the conditioning sets (except, artificially, for the results marked with a star), because it uses the Markov-blanket information better. Our approach, however, is one order of magnitude better than GS(1) and GS(2) in terms of number of tests, while still using smaller average and maximum conditioning-set sizes in all tested networks. Especially striking are the results on the **CARPO** network: this is an example where CS saves a lot of time by ignoring the numerous links that are not part of triangles, whereas GS(1) and GS(2) also checks those, with the often large Markov blankets (Figure 3.10).

We then performed the same experiments, but using the statistical tests on data sampled from the networks as described in the previous sections. We used a fixed sample size $n = 500$ and averaged over 9 different samplings for each network. We only compared PC, GS(1) and CS on this series of experiments, preferring GS(1) to GS(2) because of the lower number of tests it usually performs. The exhaustive results are listed in Table 3.2 for the sake of completeness, and the sum of the structural errors is also shown in Figure 3.12 for easier visualization.

First, we see that we get similar results as in Table 3.1 as far as the number of tests and size of the conditioning sets are concerned: CS is faster and consistently performs fewer tests with smaller conditioning sets, which leads to an increased power of the tests. However, that is sometimes balanced out by the reliance of CS on a single series of tests both to remove spouse links and to orient (possibly multiple) V-structures at the same time, thus leading to a greater penalty if the outcome of a test is wrong with respect to the initial graph.

We see that GS(1) and PC can beat CS on certain arc scores; PC, in particular, is good at avoiding arc-orientation mistakes in these experiments. GS(1), which checks not only triangle links but all links to try to orient them, makes more orientation mistakes, especially on the **CARPO** network. PC tends to miss a few more arcs than CS, which in turn misses a few more than GS(1). But taken as a whole, CS beats GS(1) significantly on **INSURANCE**, **HAILFINDER**, and **CARPO**, while performing

Algorithm	ALARM	INSURANCE	HAILFINDER	CARPO	DIABETES
mod. PC					
# tests	2850 ± 285	13461 ± 3247	$9681105^{\dagger}$	412791 ± 104080	57153 ± 9910
avg. $\lvert\mathbf{Z}\rvert$	2.97 ± 0.17	3.50 ± 0.33	$5.54^{\dagger}$	5.17 ± 0.15	4.37 ± 0.14
max $\lvert\mathbf{Z}\rvert$	6	6	$6^{\dagger}$	6	6
arcs:					
missing	5.44 ± 0.53	9.56 ± 1.01	$6^{\dagger}$	14.22 ± 1.64	9.56 ± 1.88
extra	0.33 ± 0.5	0.11 ± 0.33	$\mathbf{0}^{\dagger}$	$\mathbf{0.22} \pm 0.44$	$\mathbf{1.11} \pm 0.60$
reversed	**0**	$\mathbf{0.22} \pm 0.67$	$1^{\dagger}$	0.11 ± 0.22	$\mathbf{2.00} \pm 1.39$
TOTAL	5.78 ± 0.72	9.89 ± 1.47	$7^{\dagger}$	14.56 ± 1.86	12.67 ± 2.69
GS(1)					
# tests	1304 ± 60	4544 ± 195	2415 ± 63	129265 ± 17033	5239 ± 46
avg. $\lvert\mathbf{Z}\rvert$	2.66 ± 0.10	3.66 ± 0.04	2.62 ± 0.02	7.49 ± 0.08	2.76 ± 0.01
max $\lvert\mathbf{Z}\rvert$	8	11	7.89 ± 0.33	15	10
arcs:					
missing	$\mathbf{1.56} \pm 0.53$	$\mathbf{5.44} \pm 0.53$	$\mathbf{3.11} \pm 0.33$	**0**	6.11 ± 0.78
extra	0.56 ± 0.73	0.33 ± 0.71	1 ± 0.71	$\mathbf{0.22} \pm 0.44$	2.78 ± 1.64
reversed	1.11 ± 1.05	3.67 ± 2.12	8 ± 2.29	16.78 ± 2.49	2.67 ± 2.00
TOTAL	3.22 ± 1.81	9.44 ± 2.39	12.11 ± 2.74	17 ± 2.62	$\mathbf{11.55} \pm 3.03$
CS					
# tests	173 ± 3	782 ± 19	507 ± 18	308 ± 14.39	907 ± 4
avg. $\lvert\mathbf{Z}\rvert$	1.55 ± 0.03	2.36 ± 0.02	2.08 ± 0.03	1.90 ± 0.12	2.17 ± 0.01
max $\lvert\mathbf{Z}\rvert$	5	6	5	8	7
arcs:					
missing	$\mathbf{1.56} \pm 0.73$	6.33 ± 0.5	3.44 ± 0.52	**0**	$\mathbf{5.11} \pm 1.17$
extra	$\mathbf{0.44} \pm 0.53$	$\mathbf{0.22} \pm 0.44$	0.67 ± 0.70	0.33 ± 0.50	1.44 ± 1.51
reversed	0.11 ± 0.33	0.33 ± 0.5	$\mathbf{0.11} \pm 0.33$	**0**	7.11 ± 1.05
TOTAL	$\mathbf{2.11} \pm 0.96$	$\mathbf{6.89} \pm 1.03$	$\mathbf{4.22} \pm 1.27$	$\mathbf{0.33} \pm 0.50$	13.66 ± 2.20

Table 3.2: Number of tests, size of the conditioning sets (noted $\lvert\mathbf{Z}\rvert$), and structural errors as returned by GS(1) and CS to recover the local network structure of the networks given perfect Markov-blanket information. Results are given in the form "mean $\pm$ standard deviation over the 9 datasets." The best performer for each type of structural error has been highlighted in bold. All runs of PC were done with a forced maximum size of the conditioning set of 6. The dagger (†) notes PC results from a single dataset instead of 9 because of the long completion times. Represented graphically in Figure 3.12.

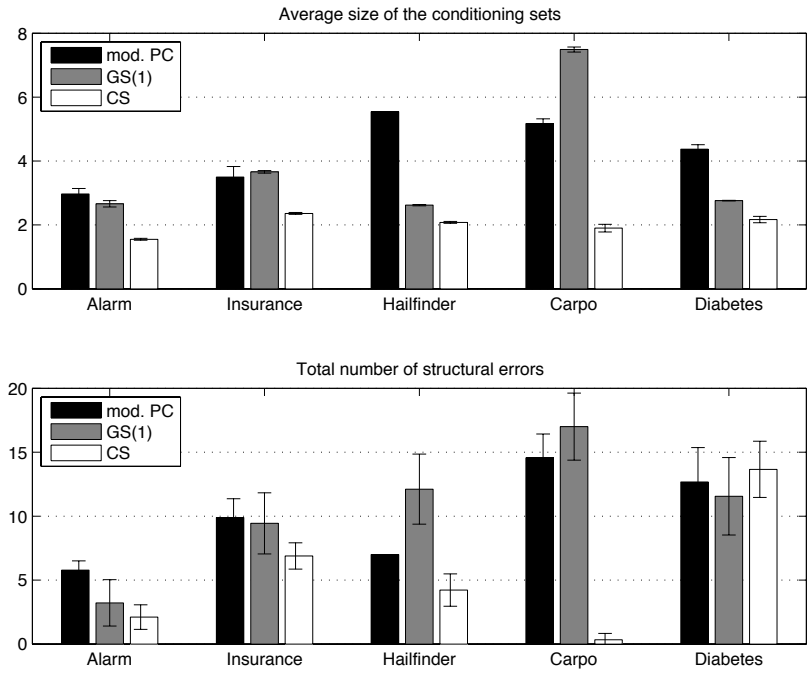


Figure 3.12: Average size of the conditioning sets and total number of errors for the three local-structure-discovery algorithms on various networks. Graphical representation corresponding to the results in Table 3.2.

slightly better on **ALARM** and being slightly outperformed on **DIABETES**. Based on these results, we will now use our collider-set search as the method of choice to break up the Markov blankets for the next series of experiments.

RFE-Based Approach

In this series of experiments, we tested our RFE-based approach on the **ALARM** network with sample sizes $n = 100$, 200, 300, 400 and 500. Table 3.3 lists the results and shows the number of errors as measured at different stages of the algorithm:

1. Right after the Markov-blanket identification, without adjustment. This compares the true Markov blanket of each variable with the identified Markov blanket as returned by Algorithm 3.7;
2. After building the moral graph. This notably excludes variables from Markov blankets if they do not satisfy the symmetry condition (3.3) in the symmetry check performed at line 5 in the generic approach described in Algorithm 3.6;

3a. After removal of the spouse links using the Recursive Median (RM) test (Margaritis, 2005) to check for conditional independence in the continuous domain;

3b. Alternatively, after removal of the spouse links using a test on Fisher's z-transform of partial correlation;

4a. After removal of the spouse links using RM *and* after maximal orientation. This is actually the result that can be compared to other full structure-learning algorithms;

4b. After removal of the spouse links using partial-correlation tests *and* after maximal orientation;

5. Finally, we show how PC performs on the same instance for comparison.

Note that the RM test is a Bayesian distribution-free conditional-independence test. In this case, where we use multivariate Gaussian distributed data, we do not expect it to perform better than the specialized z-test. We nevertheless include it in this series of experiments for two reasons. First, it allows the collider-set search to be also distribution-free, in the sense that if "distribution-free feature selection" can be performed efficiently and consistently in the first phase, applying a subsequent collider-set search does not make more assumptions on the distribution. Second, it allows us to evaluate the cost of using a distribution-free algorithm on Gaussian data.

Detailed results are in Table 3.3 and the total number of structural errors is shown graphically in Figure 3.13. What we can read from the results is that, generally, the selected Markov blankets contain all variables from the true Markov blanket plus one or two additional variables. Starting at $n = 300$, on average, fewer than two variables were missed. Many spurious variables were selected, however, even for the larger datasets. This confirms our expectation that the RFE approach will also select weakly relevant features: on average, the Markov blankets in the **ALARM** network have a size of 3.5, and on average 5.5 variables are selected per variable.

The symmetry check requiring Y to be part of $\mathbf{Mb}(X)$ and X to be part of $\mathbf{Mb}(Y)$ in order to add a link between X and Y fulfills its purpose, as even in the case $n = 200$, where on average about 73 variables enter wrong Markov blankets, only 4 extra links are added in the moral graph. Incidentally, we thus argue that a global analysis can be beneficial to achieve better results on local tasks: we see here that determining via RFE the Markov blanket of a single variable is too inclusive, but that validating the selected variables globally, for instance with our Markov-blanket symmetry check, allows us to significantly reduce the number of false positives.

After the collider-set search, the number of missing and extra arcs can both either increase or decrease. If the number of missing links increases, it is because the collider-set search found d-separation too often while variables were actually dependent. If it decreases, it means that the missing arcs in the moral graph were spouse links, as their absence is not penalized in the PDAG any more. If the number of extra arcs increases, then the collider-set search failed to identify spouse links; if it decreases, then the collider-set search could also remove links that were not spouse links (which in turn possibly led to wrong orientations). Also, determining which part of the algorithm is responsible for a missed, extra, or reversed edge in a PDAG or CPDAG is not evident. As the feature-selection step is not alone responsible for the extra or missing links, the collider-set search is not responsible for all orientation mistakes. In the collider-set search, if a

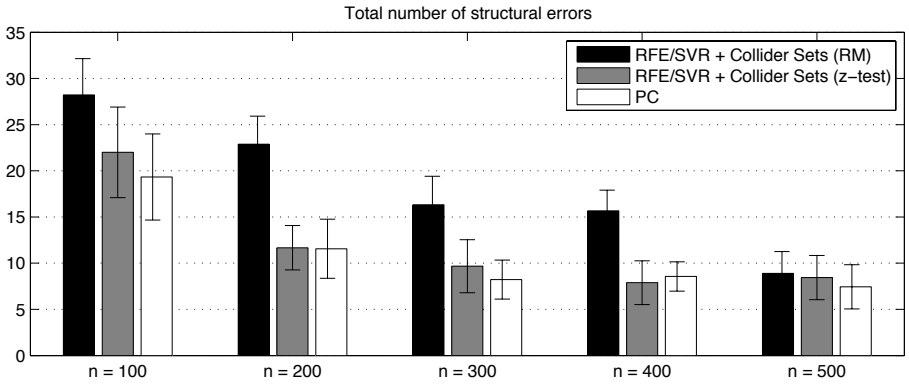


Figure 3.13: Total number of errors for the CPDAGs returned by the three global structure-discovery algorithms on the **ALARM** network with various sample sizes. Graphical representation corresponding to the results in Table 3.3.

Stage	$n = 100$	$n = 200$	$n = 300$	$n = 400$	$n = 500$
1. Mb$(\cdot)$ ident.					
missing variables	17.33 ± 3.33	3.33 ± 1.48	0.78 ± 1.11	0.78 ± 1.48	0.33 ± 1.48
extra variables	80.66 ± 15.17	72.89 ± 9.25	74.78 ± 13.69	72.56 ± 9.99	73.33 ± 9.99
2. Moral graph					
missing arcs	23.44 ± 3.54	7.89 ± 3.48	4.33 ± 3.91	4.56 ± 3.47	4.33 ± 3.87
extra arcs	4.67 ± 2.24	4.11 ± 1.69	3.78 ± 1.20	3.11 ± 1.62	3.89 ± 2.20
TOTAL	28.11 ± 3.82	12.00 ± 2.55	8.11 ± 4.01	7.67 ± 4.06	8.22 ± 4.38
3a. PDAG/RM					
missing arcs	17.44 ± 2.35	10.00 ± 1.80	6.89 ± 2.67	6.33 ± 2.60	4.56 ± 1.51
extra arcs	4.78 ± 2.17	4.22 ± 1.56	3.33 ± 1.12	3.22 ± 1.48	3.44 ± 2.01
reversed arcs	1.22 ± 1.20	2.11 ± 1.27	3.11 ± 0.78	1.78 ± 0.97	0.67 ± 0.60
TOTAL	23.44 ± 1.67	16.33 ± 3.12	13.33 ± 2.65	11.33 ± 2.78	8.67 ± 2.37
3b. PDAG/z-t.					
missing arcs	11.89 ± 2.32	3.33 ± 1.32	2.67 ± 1.94	2.11 ± 1.17	2.56 ± 1.51
extra arcs	5.22 ± 2.22	4.00 ± 1.58	3.22 ± 1.20	2.78 ± 1.30	3.44 ± 2.01
reversed arcs	0.33 ± 0.50	0.56 ± 0.73	1.22 ± 0.67	0.78 ± 1.09	0.44 ± 1.13
TOTAL	17.44 ± 3.32	7.89 ± 2.15	7.11 ± 2.98	5.67 ± 2.12	6.44 ± 2.40
4a. CPDAG/RM					
missing arcs	17.44 ± 2.35	10.00 ± 1.80	6.89 ± 2.67	6.33 ± 2.60	4.56 ± 1.51
extra arcs	4.78 ± 2.17	4.22 ± 1.56	3.33 ± 1.12	3.22 ± 1.48	3.44 ± 2.01
reversed arcs	6.00 ± 3.87	8.67 ± 2.12	6.11 ± 2.32	6.11 ± 3.59	0.89 ± 0.60
TOTAL	28.22 ± 3.93	22.89 ± 3.02	16.33 ± 3.08	15.67 ± 2.24	8.89 ± 2.37
4b. CPDAG/z-t.					
missing arcs	11.89 ± 2.32	3.33 ± 1.32	2.67 ± 1.94	2.11 ± 1.17	2.56 ± 1.51
extra arcs	5.22 ± 2.22	4.00 ± 1.58	3.22 ± 1.20	2.78 ± 1.30	3.44 ± 2.01
reversed arcs	4.89 ± 3.33	4.33 ± 1.73	3.78 ± 1.09	3.00 ± 1.80	2.44 ± 1.13
TOTAL	22.00 ± 4.90	$\mathbf{11.67} \pm 2.40$	9.67 ± 2.87	$\mathbf{7.89} \pm 2.37$	8.44 ± 2.40
5. CPDAG/PC					
missing arcs	12.11 ± 2.52	7.44 ± 1.42	4.22 ± 0.97	5.67 ± 1.12	4.78 ± 0.83
extra arcs	4.56 ± 2.19	2.67 ± 1.87	2.78 ± 1.48	2.11 ± 0.93	2.00 ± 1.66
reversed arcs	2.67 ± 1.80	1.44 ± 1.51	1.22 ± 1.09	0.78 ± 1.09	0.67 ± 0.87
TOTAL	$\mathbf{19.33} \pm 4.66$	$\mathbf{11.56} \pm 3.2$	$\mathbf{8.22} \pm 2.11$	8.56 ± 1.59	$\mathbf{7.44} \pm 2.40$

Table 3.3: Structural errors at various stages of the RFE-based approach, showing the missing, extra and reversed arcs with respect to the original graph. For Step 1, identification of the Markov blanket, the figures are averages over the 37 variables; i.e., the count of the extra or missing variables per Markov blanket, and thus not directly comparable to the other steps. The sums of the errors for the CPDAGs are represented in Figure 3.13.

wrong spouse link is removed, it is because a wrong V-structure has been identified, so that the absence of an arc will be linked to the wrong orientation of the falsely recognized V-structure. It is also possible to construct cases where missing a variable in the feature-selection step will lead not only to a missing arc, but also to the detection of a spurious V-structure, even if all subsequent tests are perfect.

For the PDAGs obtained using z-tests, the number of missing arcs always decreases with respect to the moral graph, and so does the number of extra links for $n \geq 200$. We find that the more general RM test seems to return independence too often, as for $n \geq 200$ more links are missing in the PDAG than in the moral graph. (On highly nonlinear data, we would, however, expect RM to perform better than a z-test, which assumes Gaussianity.)

The CPDAGs do not have a number of adjacency errors different from their PDAGs; this step can only add directionality errors. We have nevertheless copied the results in order to improve the readability and to make the comparison with PC easier. Although the RFE approach can outperform PC in adjacency errors, PC still consistently makes fewer directionality errors. We remark, however, that the overall performance of RFE/SVR with z-tests is quite comparable to that of PC, as also shown in Figure 3.13, which empirically justifies the intuition behind this approach.

TC and TC_{bw} vs. Competitors

For this series of experiments, we performed more systematic testing of TC, TC_{bw}, PC, the full GS, and the Bach-Jordan method on datasets sampled from **ALARM**, **INSURANCE**, **HAILFINDER**, **CARPO**, and **DIABETES**, varying the sample size. The Bach-Jordan method consists of a scoring function based on Mercer kernels coupled with a greedy search in the space of DAGs and was designed to learn Bayesian networks. It does not guarantee that the formal semantics of a causal graph are respected in the large-sample limit, but has been included in the experiments for the sake of comparison. Other possible competitors such as SCA (Friedman et al., 2000) or AlgorithmMB (Peña et al., 2005) were inapplicable because generalizing them to handle continuous variables requires techniques that are too computationally expensive, notably because of score-based subroutines that are hard to generalize.

The structural errors, as before, are missing, extra, and reversed arcs in the returned CPDAG with respect to the generating graph. For the Bach-Jordan method, similarly to what was done in Fu (2005), we converted the returned DAG to its essential graph first before checking for structural errors to avoid penalizing statistically equivalent structures. For all experiments, we also compare the run times and the number of tests performed by TC, TC_{bw}, GS, and PC.

Specific to the Bach-Jordan method is the issue of choosing the appropriate kernel parameters; i.e., in our case, the σ width in the Gaussian kernel. Bach & Jordan (2003) claim that the algorithm is in general robust to the choice of σ. We have found, however, that for varying sample sizes, the number of structural errors is quite sensitive to σ. As the authors do not propose a heuristic to set it, we systematically tested the algorithm with $\sigma = 2$, 1, 0.5, and 0.3 for each run, and chose the outcome with the smallest sum of structural errors. We chose these four possible values so as to cover a range of meaningful width for the Gaussian kernel, given our datasets. In general, smaller datasets preferred $\sigma = 2$, while the larger ones preferred a smaller σ. The change of σ is not directly visible in the following plots of the errors, but it often leads to "zigzags" in the

Bach-Jordan curves. This is due to our choice to only test a fixed number of values for σ and not to perform a full optimization of this hyperparameter for each run. The results shown are thus not the best results obtainable with this method, but we cannot optimize fully on σ either without the risk of overfitting.

ALARM The figures on page 68 show the structural errors, run times and number of statistical tests against the number of samples for **ALARM**. For each sample size, 5 datasets were drawn from the model; the error bars indicate the standard deviation over these 5 runs.

The numbers of extra and missing links seem to clearly decrease on average for all algorithms with an increasing number of samples, except for Bach-Jordan, which sometimes has a tendency to add more links when more data points are available. Note that Bach-Jordan's σ changes between the last two runs, explaining the abrupt change in the extra arcs. The number of reversed arcs is less satisfactory, in particular for TC. The explanation is that TC misses many arcs with low sample sizes, and thus does not actually get the opportunity to make many directionality errors for these cases. TC_{bw} exhibits a related behavior, although much less strong. We also see that Bach-Jordan makes the most directionality errors (this is actually valid for all networks). GS reaches repeatedly a zero extra-arc score for $n > 1000$, although it misses some more often than the others.

Starting at about 200 samples, TC equals or outperforms PC, GS and Bach-Jordan. TC_{bw} beats both TC and PC, and the converging curves of TC and TC_{bw} show that the stepwise regression becomes unnecessary with about 400 samples. On average, TC was about 20 times faster than the implementation of PC we used, although the factor tended to decrease with larger sample sizes. TC_{bw} was naturally slower than TC, although only marginally so compared to the speed difference with PC.

Overall, the constraint-based methods seem to perform about equally well for $n > 400$, and TC_{bw} and GS perform slightly better than PC for low sample sizes. Note that for low sample sizes, TC is always outperformed by TC_{bw}, PC, and GS, but is often the one to perform best when the sample size becomes larger. The graphs in Figure 3.15 show that TC and TC_{bw} are fastest, although GS performed fewer tests that TC_{bw}.

INSURANCE For this network, we find similar behaviors to **ALARM**, shown in the graphs on page 69. The most striking difference is the clear tendency of Bach-Jordan to add more arcs when more data is available for this more densely connected network. Between the 5th and 6th sample sizes, there is again a change of σ. Comparing the curves of the missing and extra arcs, we see that this changes the trade-off between false negatives and false positives.

In this case, too, TC_{bw} outperforms TC with low sample sizes (because it misses fewer arcs) but is outperformed with bigger datasets (because it adds too many). Both PC and GS, while being slightly better than TC_{bw} for $n < 100$, are outperformed starting at about $n = 500$. Note the overall good performance of GS in terms of arc-orientation errors. The corresponding curve also decreases more smoothly with larger datasets. The Bach-Jordan method is unexpectedly fast on this dataset, although poor in accuracy. The pattern of the number of statistical tests is very similar to that of **ALARM**.

HAILFINDER This network poses a problem to PC: we divided its run time and the number of tests by 10 in the graphs of Figure 3.19 on page 70. Because of its long run times, PC was run only

once for each point in the plots, so that the error bars are missing. PC runs into trouble because of the node cluster around variable 27 in the network (see Figure 3.9): it tries to separate it from the other nodes by doing subset searches on its large number of neighbors. In order to speed it up, we set the maximum node fan-in parameter to 6, so that PC would not attempt to conduct conditional-independence tests with conditioning sets larger than 6 (we see in Figure 3.19 how this imposes an upper bound on the run times of PC). TC and TC_{bw} do not run into this problem, because this cluster is correctly left alone after the feature-selection step, done in $\mathcal{O}(d^3)$ operations. Note that TC, TC_{bw}, and GS *would* also spend a long time on this cluster if all neighbors of variable 27 were its parents, because they would contain a lot of extra spouse links to be checked with an exponential number of combinations. But this example shows that a local lack of sparseness is fatal to the efficiency of PC, whereas other algorithms can still deal with it if the density of the connections is caused by children rather than parents.

This network shows more clearly the missing arc problem that TC has with low sample size, and the benefits of using TC_{bw} rather than TC here, at least for $n < 2000$. On this network, GS performs overall rather well. It is beaten by TC only for $n > 2000$, but performs better than all others for $n < 200$. Bach-Jordan still exhibits the same tendency to add more arcs when more data is available. For this dataset, σ changes twice, between the 4th and 5th, and between the 5th and 6th dataset sizes. The 5th sample size seems to have generated an unfavorable dataset for PC, as the number of extra arcs is particularly high.

Examining the run times shows TC to be the fastest. This is important especially with larger sample sizes, as TC is often both the fastest and most accurate algorithm.

CARPO The results for this network are shown on page 71. The structural particularity of this network is the multiple occurrences of a single variable having many children. Overall, PC performs badly on this network. For $n < 200$, GS is the clear winner: all other algorithms make more errors. The plain TC, especially, misses many arcs. For $n > 500$, however, both TC and TC_{bw} slightly but consistently outperform GS. At $n = 800$, TC beats TC_{bw}. Bach-Jordan, although fast on this instance, adds again too many extra links, and makes numerous directionality errors.

DIABETES This is our largest and final test network. The error patterns are most similar to those of the **INSURANCE** network, with the exception of Bach-Jordan, which performs more poorly here. We can detect two changes of σ: between the 3rd and 4th, and between the 5th and 6th sample sizes, which explains the "zigzags" in Figure 3.22 (a) on page 72.

Starting at $n > 1000$, all constraint-based methods seem to yield similar overall accuracy. GS is better in terms of directionality errors; TC and TC_{bw} are better in terms of missed links. For $n > 4000$, TC and TC_{bw} have the same accuracy and slightly beat GS and PC, while they are beaten significantly for $n < 800$. We note that the extra links added by GS allow it to obtain a better directionality accuracy than in our first series of experiments, where it was given the exact moral graph as input.

Discussion

Both TC and TC_{bw} slightly but consistently beat the other competitors when the sample size exceeds one or two thousand, depending on the network. They are usually weaker with low sam-

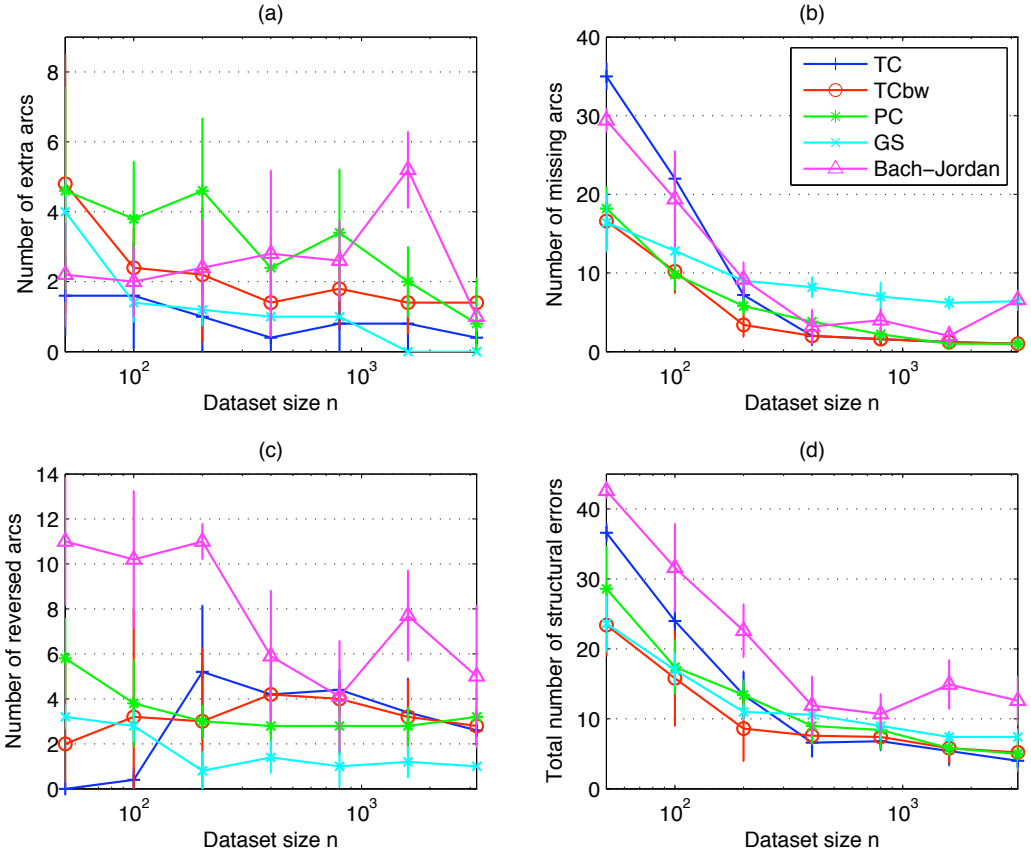


Figure 3.14: Differentiated errors on **ALARM** as a function of the sample size n: (a) extra arcs; (b) missing arcs; (c) reversed arcs; (d) total sum.

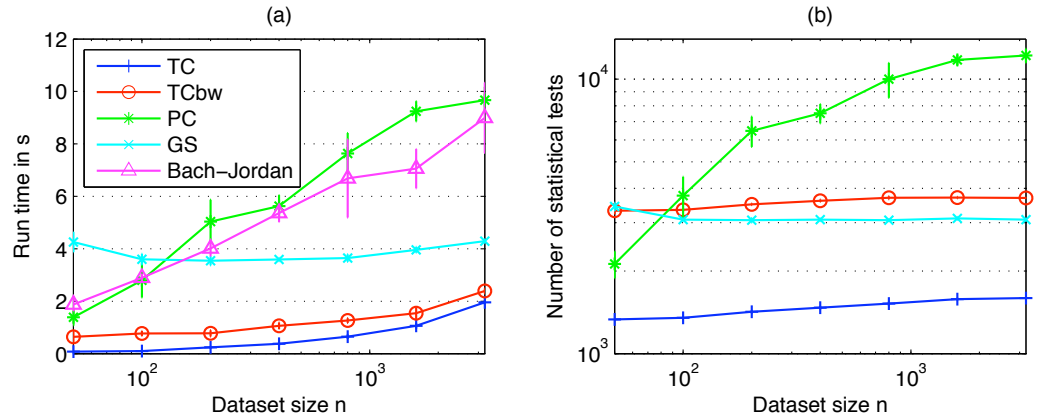


Figure 3.15: ALARM: (a) run times and (b) number of statistical tests as a function of the sample size n.

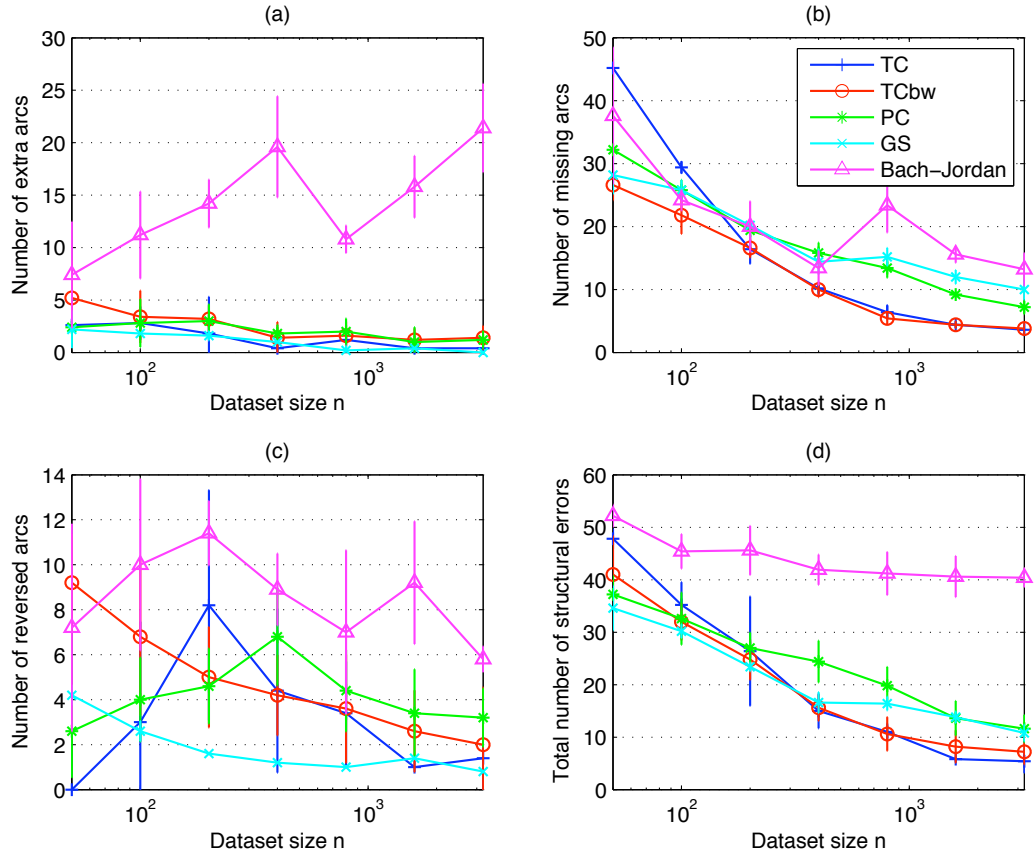


Figure 3.16: Differentiated errors on INSURANCE as a function of the sample size n: (a) extra arcs; (b) missing arcs; (c) reversed arcs; (d) total sum.

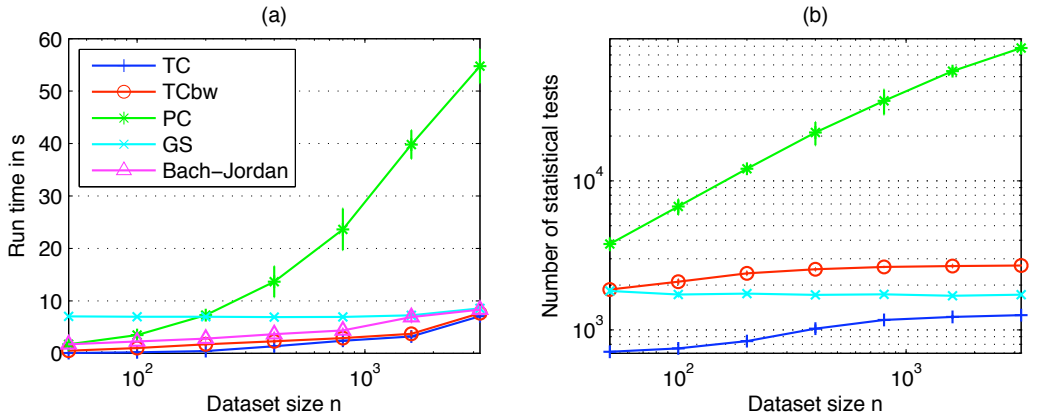


Figure 3.17: INSURANCE: (a) run times and (b) number of statistical tests as a function of the sample size n.

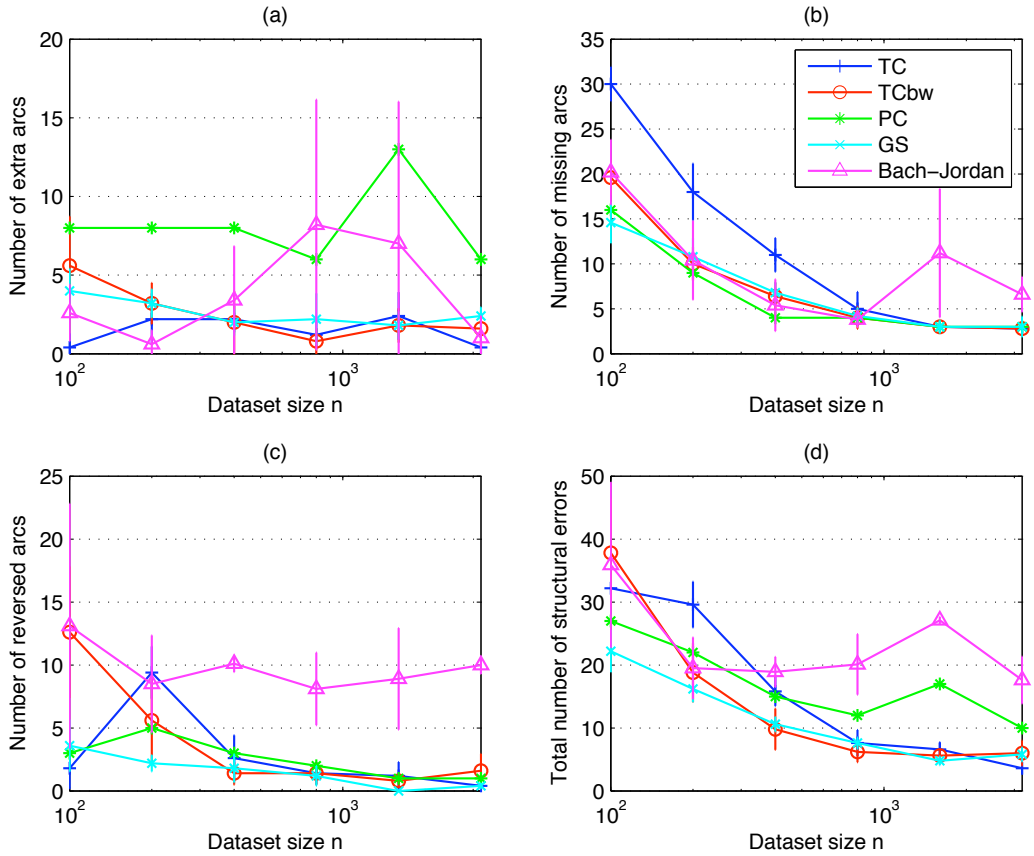


Figure 3.18: Differentiated errors on **HAILFINDER** as a function of the sample size n: (a) extra arcs; (b) missing arcs; (c) reversed arcs; (d) total sum.

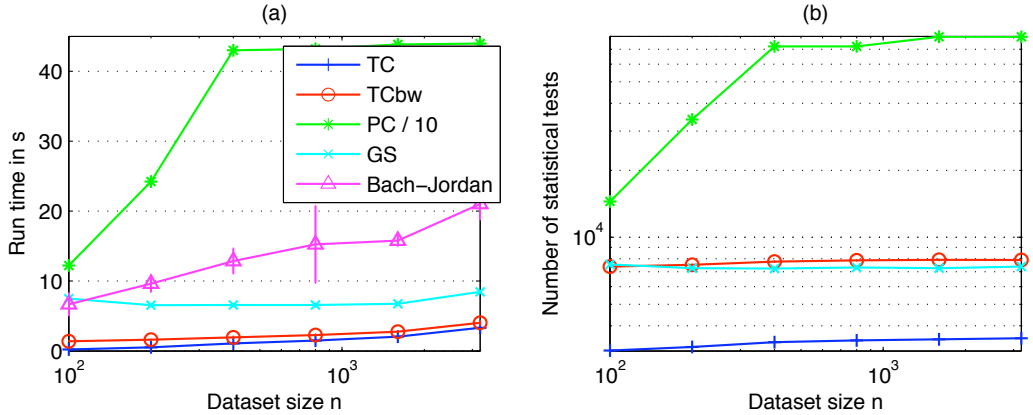


Figure 3.19: **HAILFINDER**: (a) run times and (b) number of statistical tests as a function of the sample size n.

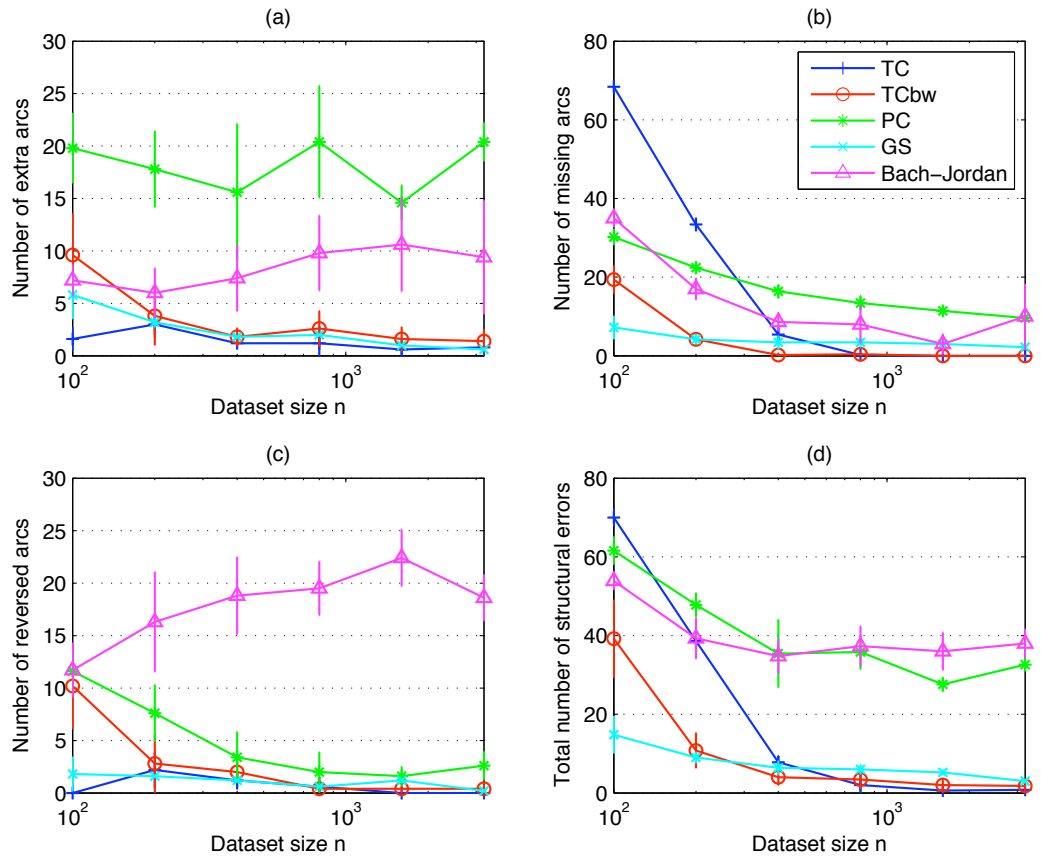


Figure 3.20: Differentiated errors on **CARPO** as a function of the sample size n: (a) extra arcs; (b) missing arcs; (c) reversed arcs; (d) total sum.

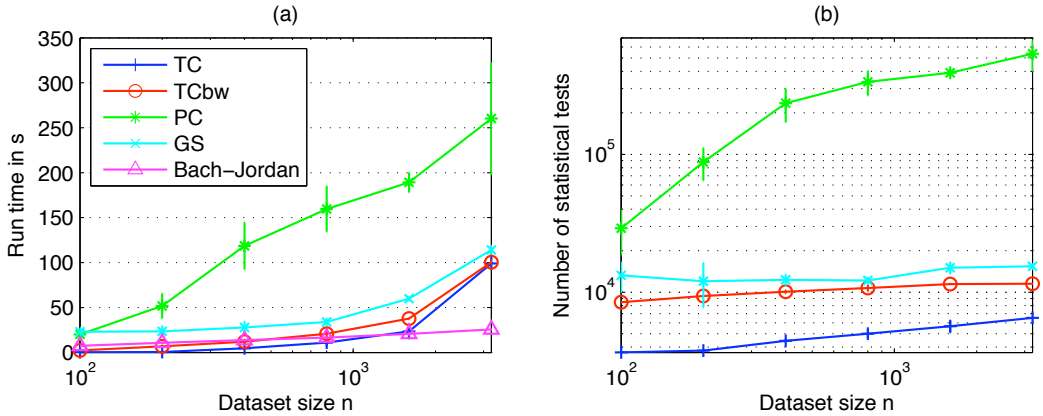


Figure 3.21: **CARPO**: (a) run times and (b) number of statistical tests as a function of the sample size n.

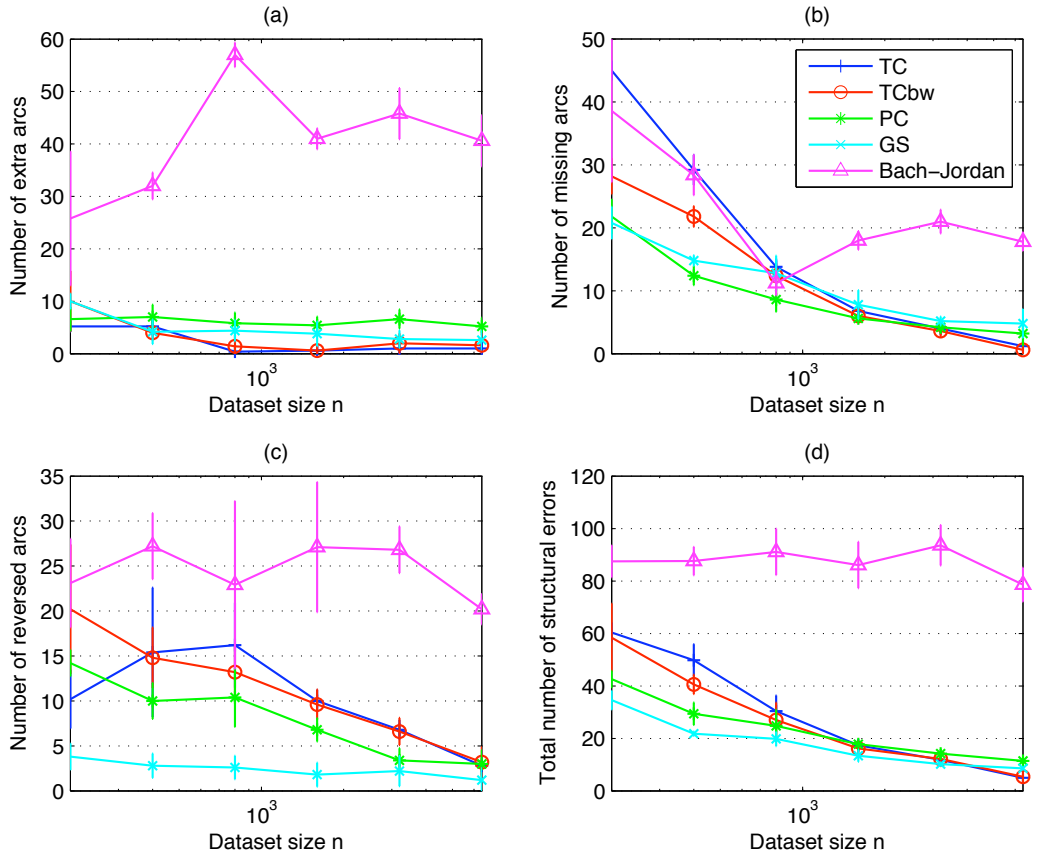


Figure 3.22: Differentiated errors on **DIABETES** as a function of the sample size n: (a) extra arcs; (b) missing arcs; (c) reversed arcs; (d) total sum.

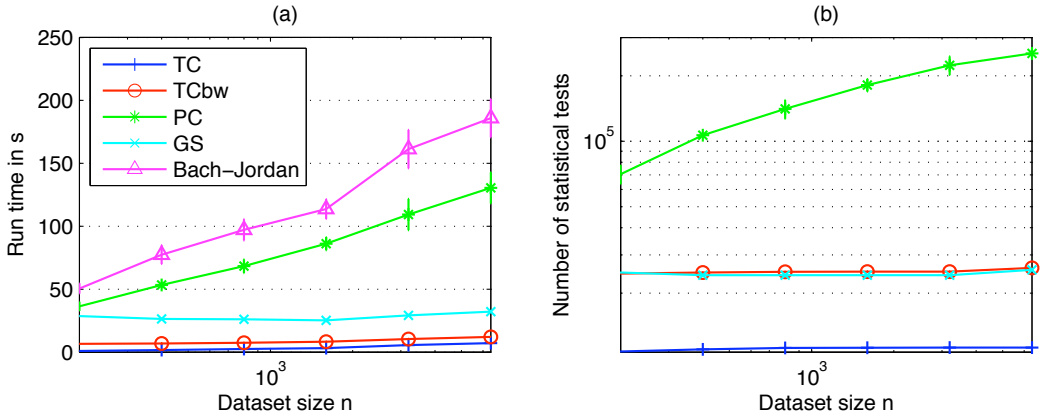


Figure 3.23: **DIABETES**: (a) run times and (b) number of statistical tests as a function of the sample size n.

ple sizes because of missed arcs, but still comparable to the state of the art. GS beats PC with small datasets, because of the way that PC goes through conditioning sets for the statistical tests (Tsamardinos et al., 2006, discuss in detail this particular issue in the case of tests with discrete variables). The score-based Bach-Jordan method was found difficult to tune with the parameter σ. For the multivariate Gaussian case, its performance is usually worse than the other tested algorithms. This also reflects the fact that PC, GS, TC and TC_{bw} with z-tests are all "tuned" for multivariate Gaussian data. The additional errors made, and additional time spent, by Bach-Jordan reflect the cost of being more generic.

In terms of run time, PC is slowed down by nodes with a high graphical degree, whereas TC or GS handle them without the exponential time-complexity growth if they are not part of triangles, as in **HAILFINDER**. In general, TC and TC_{bw} resolve all conditional-independence relations (up to married parents) in the feature-selection step in $\mathcal{O}(d^3)$ and $\mathcal{O}(d^4)$, respectively, whereas all PC can do in $\mathcal{O}(d^{2+\alpha})$ is resolve conditional-independence relations with conditioning sets of cardinality α. It is then reasonable to expect algorithms like GS, TC and TC_{bw} to scale better than PC on sparse networks where nodes have a small number of parents. The exponential growth in PC can be seen in cases where the nodes have a high degree, be they parents or children; in TC and GS, it is due to large fully-connected triangle structures and to spouse links coming from the Markov blanket-construction step. And whereas these large structures imply a high degree, the converse is not true (for instance in the **HAILFINDER** network). So, PC will exhibit an exponential behavior on all problem instances where TC and GS also exhibit this behavior, but the converse is not true.

It is interesting to mention what kind of high-degree structure is more likely to appear. If it is a node with many children (as node 27 in **HAILFINDER**), which we call an *explosion pattern*, TC can handle it efficiently. If it is a node with many parents, an *implosion pattern*, then none of these algorithms can recognize it in polynomial time. Explosion patterns correspond to a single cause that has many effects; implosion patterns correspond to many causes leading to the same effect. It remains open to discussion to know which one is more likely to occur with real-life datasets.

GS could not be beaten on small sample sizes. It is yet an unsolved challenge for TC and TC_{bw} to handle problems where the number of variables exceeds the number of samples, as in gene-expression networks, thus leading to an attempt at inverting a matrix that does not have full rank. Regularizing the covariance matrix might help make TC_{bw} more robust in this case. Computationally, TC_{bw} does incur a small run-time penalty with respect to TC, but the number of tests that TC_{bw} performs is usually comparable to GS.

TC_{bw} helps solve problems with TC and small datasets, but still cannot operate below the $n = d$ threshold. The exact sample size where TC_{bw} stops performing better than TC does not appear to be a simple function of n or d but depends on the structure of the network. It would be useful to investigate when the feature-selection addition of TC_{bw} becomes irrelevant. And as GS is more accurate with small sample sizes, finding a similarly testable condition predicting the threshold where TC is more accurate than GS would allow us to merge the approaches into a single algorithm that can determine which Markov-blanket approach to use in order to achieve better results.

Across all networks, Bach-Jordan makes the most orientation errors. Recall that Bach-Jordan returns a fully oriented DAG and not a PDAG: in the best case, it thus returns a member of the equivalence class of the generating graph, and we know that two members of this equivalence class only differ in arc orientations. Owing to their score-based approach, Bach-Jordan uses a criterion that ignores the independence-equivalence problem and maximize a goodness-of-fit measure of

the returned DAG. Whereas this is reasonable in a predictive setting, it is problematic in a causal context: the reversal of one arc changes the causal interpretation completely.

The structural performance of Bach-Jordan justifies again the need for algorithms that explicitly address the causal-underdetermination problem and return equivalence classes as PDAGs instead of DAGs. Although PDAGs have the disadvantage that they may contain undirected edges, which must be manually oriented before proceeding with the causal analysis, they clearly represent several statistically indistinguishable models rather than making a causally unjustified choice. This leaves it up to the analysts to decide on how to fully orient the graph, ensuring that any orientation of the undirected edges that does not create new V-structures is still statistically compatible with the training data.

3.4.3 Conclusion

Causal discovery and feature selection are closely related: optimal feature selection discovers Markov blankets as sets of strongly relevant features, and causal discovery reveals Markov blankets as direct causes, direct effects, and common causes of direct effects.

This has enabled us to propose a generic causal-structure-learning algorithm based on Markov blankets: by performing feature selection for each variable, we can construct the undirected moral graph as an approximation of the causal graph; a second step, the collider-set identification, is needed in order to transform the Markov blankets into parents, children, and spouses.

In the worst case, separating the Markov blankets into causes, effects, and other causes of common effects has exponential complexity with respect to the number of variables, but can actually be solved efficiently provided the graph is sparse enough—a common assumption of many algorithms. We proposed a procedure to transform the moral graph into a causal graph in linear time, assuming a constant number of edges per node. We experimentally compared it favorably to the similar steps of the Grow-Shrink algorithm.

Determining the Markov blanket with existing backward feature elimination like RFE eliminates the irrelevant variables in the large-sample limit, but remains too inclusive. Global corrections have to be made, such as for instance insuring that a variable in the selected Markov blanket of a target also includes this target in its own selected Markov blanket. We have conducted experiments that have confirmed that this adjustment discards most false positives, and thus provided a hint that the approach is consistent in the large-sample limit. The main challenge is to perform feature selection for all variables in an efficient way.

This task is tractable with the multivariate Gaussian assumption. We have presented the TC and the TC_{bw} algorithms, which fit into the described framework, and have compared them to PC, GS, and a Bayesian structure-learning method. For small sample sizes, GS usually makes fewer structural errors, and TC/TC_{bw} are more accurate and more efficient for larger sample sizes.

Evidence shows the superiority of the Markov-blanket approaches as described in this chapter (even if not all Markov-blanket algorithms consistently beat PC on all networks). As support for this claim, we invoke the high run times of PC, and the good low and high sample-size accuracy of GS and TC/TC_{bw}, respectively. Not only are Markov-blanket techniques much more scalable, they can be more accurate; they are also more easily modifiable to construct only parts of the network that are deemed relevant by some criterion.

The biggest challenges in causal-structure learning include robust and consistent distribution-free structure learning with continuous and potentially highly nonlinear data. We hope that a framework such as the one we have presented, based on feature selection, can provide a good basis for further research in this domain, allowing future algorithms to be scalable enough to tackle large problems thanks to the principle of learning local neighborhoods.

Chapter Summary

In causally sufficient cases, automatic causal-structure learning can be done with algorithms. Often, conditional-independence tests are made to determine compatible structures. However, owing to causal underdetermination, algorithms return partially directed graphs. Structure learning in general is NP hard, but techniques identifying a local neighborhood for each node, for instance the Markov blanket, can be several orders of magnitude more efficient than standard approaches like PC. We have shown how the task can be split into Markov-blanket identification and local adjustments, and have proposed efficient algorithms for both.

CHAPTER

4

Structure Learning without Causal Sufficiency

Up to now, we have seen causal structure-discovery techniques that assume that all causes of more than one variable are observed—the so-called causal-sufficiency assumption. In practice, however, it is untestable, and often violated. In this chapter, we examine the case where we cannot assume causal sufficiency. First, we revisit our understanding of causation with possible hidden variables in mind, and show how structure-learning algorithms deal with this problem. We then propose a novel algorithm, MBCS*, which uses ideas from the previous chapter, first learning local neighborhoods as Markov blankets. We show with experiments that MBCS* is comparable to the state of the art in accuracy, while being several orders of magnitude faster on large problems.

4.1 HIDDEN VARIABLES AND CONFOUNDERS

The problem of hidden variables is central to causal-structure learning. Indeed, certain types of hidden variables can completely change the outcome of the normal algorithms which assume causal sufficiency. In this section, we will try to understand how hidden variables are problematic, what assuming causal sufficiency actually has allowed the previously described algorithms to do, and what it means, algorithmically, to drop causal sufficiency.

4.1.1 Effect of Hidden Variables

Even in theory, causal sufficiency cannot be tested (although special cases may invalidate it, as we see below). On the contrary, the faithfulness assumption, for instance, could be tested by enumerating all possible disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subsetneq \mathbf{V}$ ($\mathbf{X}, \mathbf{Y} \neq \emptyset$), and checking that the model and the data agree on conditional independence of $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{Z}$. This is not done in practice because of the exponential time complexity of this check. Causal sufficiency is more fundamental. It is often regarded as the most controversial assumption as it is generally impossible to ensure that all possible *common causes* are measured.

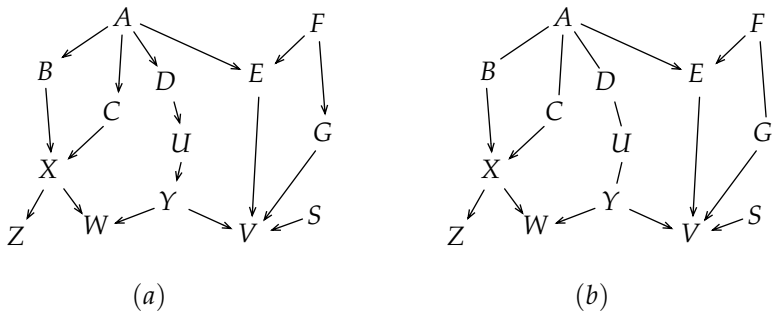


Figure 4.1: (a) A sample fully specified causal structure to study the effect of hidden variables; (b) its corresponding maximally oriented PDAG.

But why does causal sufficiency focus on hidden causes specifically, and does not address hidden effects, for instance? In short, the reason is that unobserved common causes bias the result of the causal analysis and can lead to the spurious detection of cause–effect relationships, whereas unobserved effects that are themselves not causes of two or more other variables cannot.

Let us consider an example to illustrate this: Assume we are in a context where the data-generating structure is described by the causal graph in Figure 4.1 (a). Suppose that one of the variables in this graph is unobserved but that we are not aware of this and use an algorithm that assumes causal sufficiency to learn the structure. What is the penalty that we get for ignoring the possibly spurious effects of this hidden variable? We examine a few examples, depending on the kind of variable that is hidden, and compare the structure of the graph we obtain when hiding a variable with the reference structure shown in Figure 4.1 (b), which is the maximally oriented PDAG representing the equivalence class of the causal graph, and thus the best a general structure-learning algorithm could return, as explained in the previous chapter.

Suppose that the hidden variable has no effect: take Z, for instance, which is the effect of a single cause X. Hiding Z has no negative impact on the result of the algorithm in the rest of the network, as shown in Figure 4.2 (a): all links but the one from X to Z are identical to the reference PDAG.

If we hide a variable like S, which has no parents and which has only one effect, the adjacencies of the rest of the graph will also be unaffected. The orientation possibilities, however, can be reduced, as V-structures may disappear in the operation. In this case (Figure 4.2 (b)), the orientations $Y \to V$, $E \to V$, and $G \to V$, which are all part of a V-structure with the $S \to V$ link, all stay, as they themselves form three other detectable V-structures: $Y \to V \leftarrow E$, $E \to V \leftarrow G$, and $Y \to V \leftarrow G$.

Causally, it is reassuring to know that the disappearance of an effect like Z has no impact on the analysis. The disappearance of a cause like S, though, even if has a single effect, can lead to a loss of precision in the end graph.

If we hide a more central variable, like U or G, which have exactly one cause and one effect, then it simply disappears from the graph, and the link from itself and its effect becomes a link from its cause to its effect, as we can see in Figure 4.2 (c). For U, we did not know which of D or Y was the cause or the effect anyway, so the new link between D and Y remains undirected. The disappearance of G, which was in the V-structure $G \to V \leftarrow E$, creates the new V-structure $F \to V \leftarrow E$. The rest of the network, in both cases, is unaffected; no new spurious links are created.

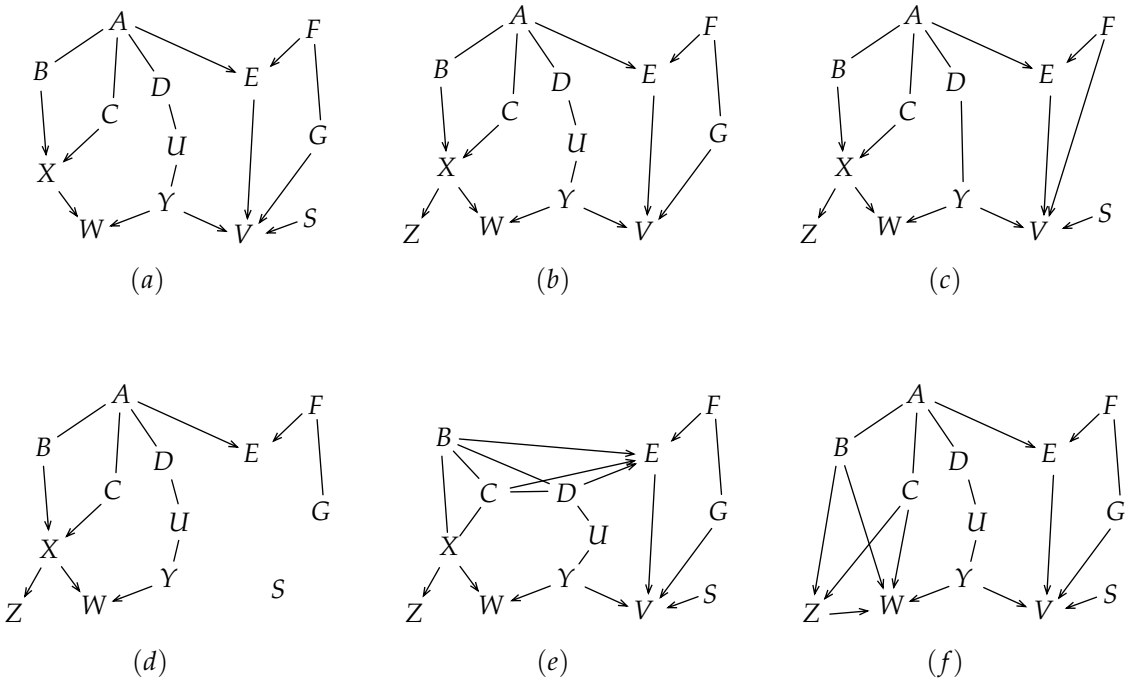


Figure 4.2: Various PDAGs showing the effect of learning the DAG of Figure 4.1 (a) with various variables hidden: (a) Z; (b) S; (c) G and U; (d) V; (e) A; (f) X.

From the causal point of view, this follows intuition: removing a variable in a chain like $F \to G \to V$ should indeed link F to V. There *is* a causal action from F to V: if we additionally examine more precisely how this mechanism works by observing a mediating variable G, so much the better; if we cannot observe it, then the rougher mechanism $F \to V$ is still detected, without affecting the rest of the network; it is a matter of how fine-grained the analysis is.

Should we remove a leaf variable like V, which has no effects, we would again only lose very local information about the causes of V, without affecting the rest of the network (Figure 4.2 (d). But what happens if a root cause for several variables, like A, goes unobserved? This is a more problematic case. Indeed, all its direct effects, B, C, D, and E, are dependent on one other, but independent given A. If A cannot be conditioned on, all effects become connected, as shown in Figure 4.2 (e). In turn, this has consequences on orientation in the rest of the network: the V-structure $B \to X \leftarrow C$ cannot be identified any more, as B and C are now adjacent (recall from Definition 2.14 that the two causes in a V-structure must be nonadjacent), which again prevents $X - Z$ from being oriented.

Let us sum up: in these few examples, forgetting about effects does not cause problems in that the rest of the network structure is preserved; forgetting about causes of multiple effects, however, falsely changes the network structure, inserting wrong links and even wrong orientation. We can characterize precisely the incurred structural penalty of hiding a given variable, if learned with an algorithm that assumes causal sufficiency.

Property 4.1 *Suppose data is generated according to its perfect causal map, DAG* $\mathcal{G}_0 = \langle \mathbf{V}, \mathbf{E} \rangle$*, with some variable* $H \in \mathbf{V}$ *hidden. Let* $\mathbf{P} = \mathbf{Pa}_{\mathcal{G}_0}(H)$ *and* $\mathbf{C} = \mathbf{Ch}_{\mathcal{G}_0}(H)$*. Define the undirected graph*

$$\mathcal{G}_u = \langle \mathbf{V} \setminus \{H\}, \mathbf{E}' \rangle, \text{ where } X - Y \iff \forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y, H\} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}) .$$

Then the following is true in G_u:

(*i*) *For all $P \in \mathbf{P}$ and $C \in \mathbf{C}$: $P - C$;*

(*ii*) *The subgraph $\mathcal{G}_{\mathbf{C}} = \langle \mathbf{C}, \mathbf{E}_{\mathbf{C}} \rangle$ where $\mathbf{E}_{\mathbf{C}} = \mathbf{E}' \cap (\mathbf{C} \times \mathbf{C})$, is a clique; i.e., it is fully connected.*

G_u, the undirected version of the causal graph, can be obtained Step 1 of the IC algorithm, for instance (see Subsection 3.1.2 on page 31). This property describes the structural penalty of ignoring that a variable is hidden and running an algorithm that assumes causal sufficiency: hiding a variable H links all parents to all children, and additionally spuriously links all children together.

Proof (*i*) For all pairs (P, C) where $P \in \mathbf{P}$ and $C \in \mathbf{C}$, there is by construction an open dependency path $P \to H \to C$. According to d-separation (Definition 2.11), it can only be blocked by conditioning on H. Thus, because $\mathcal{G}_0$ is a perfect map of our problem, every set $\mathbf{S}_{PC} : (P \perp\!\!\!\perp C \mid \mathbf{S}_{PC})$ must include H, which is impossible since H is hidden. We conclude that no such $\mathbf{S}_{PC}$ exists, which by construction of $\mathcal{G}_u$ means that P and C are adjacent.

(*ii*) A similar argument can be applied for the children, as between any two children C_1 and C_2 there is a dependency path $C_1 \leftarrow H \to C_2$ that cannot be blocked. □

In general, deleting a node H can introduce up to $n_p(n_c - 1) + n_c(n_c - 1)/2$ spurious links, with $n_p = |\mathbf{Pa}_{\mathcal{G}_0}(H)|$ and $n_c = |\mathbf{Ch}_{\mathcal{G}_0}(H)|$. As special cases of this property, we see that, for instance, ignoring nodes without effects does not introduce spurious links, and deleting a node that has exactly one effect (like U) also does not cause a structural mismatch. The true, structure-changing problems arise when nodes that have more than a single effect are hidden: this is why the definition of causal sufficiency, as stated on page 21, is concerned with the observation of all *common causes of two or more variables*, and ignores other hidden variables.

We can use Property 4.1 to extract graph-transformation rules that describe this structural penalty. Let $\mathcal{G}_0$ be the original data-generating graph (for instance, the one in Figure 4.1 (*a*)), and $\mathcal{G}_{cs}$ the PDAG learned with an algorithm that assumes causal sufficiency, run with no hidden variables (for instance, the one in Figure 4.1 (*b*)). Then, the graph $\mathcal{G}_{\mathbf{H}}$ learned with the same algorithm, but from which we hide the set of variables $\mathbf{H}$, can be obtained as follows: Start from $\mathcal{G}_{cs}$, and then iteratively for each $H \in \mathbf{H}$:

1. Remove H as well as its incoming and outgoing edges;
2. For each parent $P \in \mathbf{Pa}_{\mathcal{G}_0}(H)$, connect P to every child $C \in \mathbf{Ch}_{\mathcal{G}_0}(H)$. Direct it as $P \to C$ when P and H used to be part of a V-structure; i.e., when $\exists P' \neq P$ such that $P \to H \leftarrow P'$;
3. For each child $C_1 \in \mathbf{Ch}_{\mathcal{G}_0}(H)$, connect C_1 to every other child $C_2 \in \mathbf{Ch}_{\mathcal{G}_0}(H) \setminus \{C_1\}$. Direct it as $C_1 \to C_2$ when H and C_2 used to be part of a V-structure; i.e., when $\exists X \neq H$ such that $H \to C_2 \leftarrow X$.

(Note that Rule 3 can potentially lead to conflicting orientations for a single link. Although this is a problem for DAGs and PDAGs, contradicting orientations are very useful for algorithms that do not assume causal sufficiency, precisely because they allow identification of hidden variables. We describe how in the next subsection.)

We now apply these rules to the last example: Suppose we hide variable X from Figure 4.1 (*a*). We first copy $\mathcal{G}_{cs}$ (which in our case is the graph in Figure 4.1 (*b*)) and remove X and all its edges. Then, according to Rule 2, we draw links from each parent $\{B, C\}$ to each child $\{Z, W\}$, orienting them from parent to children because of the V-structure $B \to X \leftarrow C$. Then, following Rule 3, we add a link between Z and W, directing it as $Z \to W$ because of the V-structure $X \to W \leftarrow Y$. The

graph we obtain, indeed, is the one learned by a traditional algorithm, shown in Figure 4.2 (f). While the links from B and C make sense as B and C are ancestors of Z and W in $\mathcal{G}_{cs}$, the link $Z \to W$ is a spurious cause–effect relationship that shows up because X was not observed.

Now that we know the effect of hiding variables, we have to determine what we can learn algorithmically about hidden variables, if anything. Not observing a variable means information loss, and in general we cannot recover this information. What we can learn about, however, is the presence or absence of hidden variables between *pairs of observed variables*, again with conditional-independence criteria. We explain how in the next subsections.

4.1.2 Latent-Structure Projections

Let us now suppose we are learning from data whose (unknown) actual causal DAG is:

$$X \to Y \leftarrow H_1 \to H_2 \to Z \leftarrow W. \tag{4.1}$$

Further assume that H_1 and H_2 are hidden. Structure-learning algorithms such as those presented in the previous chapter would find these adjacencies:

$$X - Y - Z - W.$$

This would be the graph resulting from Step 1 of IC, for instance. Now, if we proceed and orient the V-structures (IC Step 2), conditional-independence tests will reveal that $(X \perp\!\!\!\perp Z)$ and $(X \not\perp\!\!\!\perp Z \mid Y)$ hold, which is a sufficient condition for the V-structure $X \to Y \leftarrow Z$. Similarly, $(Y \perp\!\!\!\perp W)$ and $(Y \not\perp\!\!\!\perp W \mid Z)$ is a sufficient condition for the V-structure $Y \to Z \leftarrow W$. However, in a DAG—like in a PDAG—these two overlapping V-structures are incompatible. This is a concrete example where Rule 3 of the transformation rules described in the previous subsection leads to two contradicting orientation directives for the edge $Y - Z$.

This contradiction is a hint that there are hidden variables. The simplest DAG compatible with these findings needs the addition of an extra variable H:

$$X \to Y \leftarrow H \to Z \leftarrow W. \tag{4.2}$$

Actually, (4.2) is called the *projection* of the hidden (or *latent*) structure in (4.1) (Pearl, 2000).

Definition 4.2 (Projection of latent structure) *Given a perfect causal DAG $\mathcal{G}_0 = \langle \mathbf{V}, \mathbf{E} \rangle$ with hidden variables $\mathbf{L}$ and observed variables $\mathbf{O}$ such that $\mathbf{V} = \mathbf{L} \cup \mathbf{O}$ and $\mathbf{L} \cap \mathbf{O} = \emptyset$, another DAG $G_p = \langle \mathbf{O} \cup \mathbf{H}, \mathbf{E}' \rangle$ is called the* projection *of $\mathcal{G}_0$ if:*

- *Every introduced variable $H \in \mathbf{H}$ has no parent and exactly two children $C_1, C_2 \in \mathbf{O}$;*
- *For all $X, Y \in \mathbf{O}$ and $\mathbf{Z} \subseteq \mathbf{O} \setminus \{X, Y\}$: $(X \not\perp\!\!\!\perp Y \mid \mathbf{Z})_{\mathcal{G}_0} \iff (X \not\perp\!\!\!\perp Y \mid \mathbf{Z})_{\mathcal{G}_p}$.*

In a projection of a latent structure, we describe a less complex version of the graph we want to learn and of its hidden structures, such that every independence relation is preserved. Interestingly, the latent structure is not influenced by unobserved effects, as long as they are not themselves the cause of two or more other variables. This follows from what we observed in Subsection 4.1.1: hiding variables that do not have more than one effect does not introduce spurious links. This means that only hidden causes are relevant and worth studying, because they have this effect of changing the network.

In the projection of a latent structure as defined by Pearl (2000), all introduced hidden variables are parentless and have only two direct effects. Verma (1993) proved that any hidden structure has at least one projection. Notice that we cannot recover the information about the two separate hidden variables H_1 and H_2 in this example. In the projection, information can thus be lost with respect to the true latent structure.

Projections of latent structures are important because they describe a new learning target. Finding a way to identify the complete hidden structure is impossible: the number of possible hidden structures is infinite. With the causal-sufficiency assumption, we considered that the ground truth was representable as a DAG. Without the causal-sufficiency assumption, we use the projection of the latent structure as ground truth. We could represent it with a DAG that includes extra variables to show where the hidden structures are, but it is customary to use another notation, called a *mixed ancestral graph* (MAG—Spirtes et al., 1996, 2001).

Definition 4.3 (Mixed ancestral graph) *A* mixed ancestral graph *or* MAG *is a graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}_{\mathcal{E}} \rangle$ according to Definition 2.1 on page 12, where $\mathcal{E} = \{\rightarrow, \leftrightarrow\}$; i.e., there are two possible edge types, $\rightarrow$ and $\leftrightarrow$. $X \leftrightarrow Y$ is considered equivalent to $Y \leftrightarrow X$.*

MAGs are essentially DAGs that allow for a new kind of edge: the bidirected arrow, which represents a hidden cause for a pair of variables. For example, the MAG representing (4.1) is:

$$X \rightarrow Y \leftrightarrow Z \leftarrow W.$$

Notice that the bidirected arrow, if it indicates a hidden cause in the projection, does not preclude a direct link between Y and Z, so we cannot exclude direct causation $Y \rightarrowtriangle Z$ or $Z \leftarrowtriangle Y$ in addition to a hidden cause if we detect the two conflicting V-structures as shown here.

As *d*-separation is defined on DAGs (Definition 2.11), we can define a generalization of *d*-separation for MAGs called *m-separation* (Drton & Richardson, 2004). To facilitate notation, we use the following "wildcard" arrow patterns: $\overset{*}{\rightarrow}$, $\overset{*}{\leftarrow}$, and $\overset{*}{-\!\!*}$, where the $*$ endpoint denotes an unspecified arrow end. For instance, the pattern $X \overset{*}{\rightarrow} Z \overset{*}{-\!\!*} Y$ matches the following six MAG substructures:

$$\begin{array}{ll} X \rightarrow Z \leftarrow Y & X \leftrightarrow Z \leftarrow Y \\ X \rightarrow Z \leftrightarrow Y & X \leftrightarrow Z \leftrightarrow Y \\ X \rightarrow Z \rightarrow Y & X \leftrightarrow Z \rightarrow Y. \end{array}$$

Definition 4.4 (*m*-Separation) *In a MAG $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, two nodes X and Y are* m-separated *given a set of nodes $\mathbf{Z}$ ($X, Y \notin \mathbf{Z}$), noted (like d-separation) $(X \not\rightleftharpoons Y \mid \mathbf{Z})_{\mathcal{G}}$, if and only if every undirected path from X to Y is blocked by $\mathbf{Z}$. A path of length $\ell = 1$ is never blocked; a path $\boldsymbol{\pi} = (V_0, V_1, V_2, \cdots, V_{\ell-1}, V_\ell)$ (with $V_0 = X$, $V_\ell = Y$) of length ℓ is* blocked *by $\mathbf{Z}$ whenever at least one node V_i, $1 \leq i < \ell$, on $\boldsymbol{\pi}$ is blocked by $\mathbf{Z}$. V_i is blocked by $\mathbf{Z}$ if and only if these two conditions hold:*

- *V_i is serially connected ($\overset{*}{\rightarrow} V_i \rightarrow$, $\leftarrow V_i \overset{*}{\leftarrow}$) or is a diverging node ($\leftarrow V_i \rightarrow$) on $\boldsymbol{\pi}$ and $V_i \in \mathbf{Z}$;*
- *V_i is a converging node ($\overset{*}{\rightarrow} V_i \overset{*}{\leftarrow}$) on $\boldsymbol{\pi}$ and neither V_i nor any $D \in \mathbf{De}(V_i)$ is in $\mathbf{Z}$.*

X and Y are m-connected *given $\mathbf{Z}$, noted (like d-connection) $(X \rightleftharpoons Y \mid \mathbf{Z})_{\mathcal{G}}$ if and only if they are not m-separated given $\mathbf{Z}$. When $\mathbf{Z}$ only contains one variable, $\mathbf{Z} = \{Z\}$, we write $(X \not\rightleftharpoons Y \mid Z)_{\mathcal{G}}$ and $(X \rightleftharpoons Y \mid Z)_{\mathcal{G}}$ instead of, respectively, $(X \not\rightleftharpoons Y \mid \{Z\})_{\mathcal{G}}$ and $(X \rightleftharpoons Y \mid \{Z\})_{\mathcal{G}}$. Similarly, two sets of random variables $\mathbf{X}$ and $\mathbf{Y}$ are m-separated or m-connected given $\mathbf{Z}$ if and only if for each $X \in \mathbf{X}$ and*

each $Y \in \mathbf{Y}$ pairwise m-separation or m-connection holds, respectively. When the context is clear, we drop the subscript "$_{\mathcal{G}}$."

The *m*-separation criterion can be used to characterize a MAG as a perfect map for a hidden structure, just like *d*-separation is used to characterize a DAG as a perfect map when there are no hidden variables. As we use the same notation for *d*- and *m*-separation, we can simply reuse the perfect-map equivalence (2.9), which asserts that *m*-separation and conditional independence coincide one-to-one (Drton & Richardson, 2004):

$$\forall \mathbf{X}, \mathbf{Y}, \mathbf{Z} \text{ disjoint subsets of } \mathbf{V} : \big((\mathbf{X} \not\perp\!\!\!\perp_{\mathcal{G}} \mathbf{Y} \mid \mathbf{Z}) \iff (\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}) \big). \tag{4.3}$$

4.1.3 Conditional Independence and Causation without Causal Sufficiency

In Subsection 2.3.2, we examined the relations between conditional independence and causation in general, and deduced what it meant for causal graphs in (2.20) on page 23. We need to revisit this result, now taking into account the possibility of hidden variables.

Recall (2.17), which, for a causally sufficient $\mathbf{V}$, reads:

$$\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}) \implies X \dashrightarrow Y \text{ or } Y \dashrightarrow X.$$

This is not true any more if we do not assume that we observe all causes. Indeed, we could be detecting a spurious form of adjacency as described in Property 4.1 instead of true causation between X and Y. What holds instead is:

$$\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}) \implies \text{one of } \begin{cases} X \dashrightarrow Y \\ Y \dashrightarrow X \\ X \dashleftarrow H \dashrightarrow Y \\ X \dashleftarrow H \dashrightarrow Y \text{ and } X \dashrightarrow Y \\ X \dashleftarrow H \dashrightarrow Y \text{ and } Y \dashrightarrow S. \end{cases} \tag{4.4}$$

We allow the introduction of an hypothetical additional variable H, which is a new cause for both X and Y. In the MAG which represents the projection, this is all summed up by the presence of some link between X and Y: $X \mathrel{*\!\!-\!\!*} Y$.

The conditions for a V-structure are also changed. Recalling them from (2.19):

$$\begin{aligned} X \dashrightarrow Z \dashleftarrow Y \iff \Big(& \big(\exists \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y, Z\} : (X \perp\!\!\!\perp Y \mid \mathbf{S}) \text{ and } (X \not\perp\!\!\!\perp Y \mid \mathbf{S} \cup \{Z\}) \big) \\ & \text{and } \big(\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{X, Z\} : (X \not\perp\!\!\!\perp Z \mid \mathbf{S}) \big) \\ & \text{and } \big(\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{Y, Z\} : (Y \not\perp\!\!\!\perp Z \mid \mathbf{S}) \big) \Big). \end{aligned}$$

Now, without causal sufficiency, any of the following possibilities can match the set of conditional-independence relations defined by (2.19), in addition to $X \dashrightarrow Z \dashleftarrow Y$:

$$\begin{array}{ll} X \dashleftarrow H_1 \dashrightarrow Z \dashleftarrow Y & X \dashleftarrow H_1 \dashrightarrow Z \dashleftarrow Y \text{ and } X \dashrightarrow Z \\ X \dashrightarrow Z \dashleftarrow H_2 \dashrightarrow Y & X \dashrightarrow Z \dashleftarrow H_2 \dashrightarrow Y \text{ and } Z \dashleftarrow Y \\ X \dashleftarrow H_1 \dashrightarrow Z \dashleftarrow H_2 \dashrightarrow Y & X \dashleftarrow H_1 \dashrightarrow Z \dashleftarrow H_2 \dashrightarrow Y \text{ and } X \dashrightarrow Z \\ X \dashleftarrow H_1 \dashrightarrow Z \dashleftarrow H_2 \dashrightarrow Y \text{ and } Z \dashleftarrow Y & X \dashleftarrow H_1 \dashrightarrow Z \dashleftarrow H_2 \dashrightarrow Y \text{ and } X \dashrightarrow Z \text{ and } Z \dashleftarrow Y \end{array} \tag{4.5}$$

In the MAG, all these possibilities are represented by a structure matching the pattern $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$. The detection of several of these patterns can lead to full orientation of arrows: for instance, $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ and $Z \mathrel{*\!\!\rightarrow} Y \mathrel{\leftarrow\!\!*} W$ are combined into $X \mathrel{*\!\!\rightarrow} Z \leftrightarrow Y \mathrel{\leftarrow\!\!*} W$.

As for the causally sufficient case, there are rules for further orienting a graph. These are detailed in the next section, where we describe how (4.4) and (4.5) are used by structure-learning algorithms that do not assume causal sufficiency.

4.2 ALGORITHMS WITHOUT CAUSAL SUFFICIENCY

This section examines structure-learning algorithms that do not assume causal sufficiency. There are various kinds of such algorithms. For instance, assuming continuous variables with linear causal influences, Silva et al. (2006) propose an algorithm to recover hidden variables that are the cause of more than two observed variables and to infer the relationships between the hidden variables themselves. The authors check additional constraints on the covariance matrix, known as *tetrad constraints* (Scheines et al., 1995), entailed by special kinds of hidden structures.

There are also more specialized techniques to deal with hidden variables. Elidan et al. (2001) look for structural signatures of hidden variables in a learned DAG model. Boyen et al. (1999) describe a technique that looks for violation of the causal Markov condition to infer the presence of latent variables in Bayesian networks. Once a hidden variable is identified, Elidan & Friedman (2001) discuss how to treat as a discrete variable and how to choose the best-fitting number of states for it. There are also many techniques for learning Bayesian networks from *partially observed* variables; for instance, Friedman's (1998) structural EM algorithm.

In the rest of this section, we examine algorithms that learn the projection of the hidden causal structure as defined in Subsection 4.1.2. Some of the few algorithms that solve this task are IC* by Pearl & Verma (1991); Pearl (2000), and Fast Causal Inference (FCI) by Spirtes et al. (1995, 2001). We first describe IC* and FCI, and where their efficiency bottleneck is. Next, we describe how we can generalize the approach developed in the previous chapter to the causally insufficient case. We present the Markov Blanket/Collider Set (MBCS*) algorithm and compare it experimentally against the FCI reference.

4.2.1 MAGs, PAGs and Equivalence Classes

Like their causally sufficient counterpart, structure-learning algorithms that do not assume causal sufficiency can only obtain an equivalence class of the generating structure. In order to represent those graphs, we introduce yet another kind of graph, called *partial ancestral graph* (PAG—Spirtes et al., 1996).

Definition 4.5 (Partial ancestral graph) *A* partial ancestral graph *or* PAG *is a graph $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}_\mathcal{E} \rangle$ according to Definition 2.1 on page 12, where $\mathcal{E} = \{\rightarrow, \bullet\!\!\rightarrow, \bullet\!\!-\!\!\bullet, \leftrightarrow\}$. $X \leftrightarrow Y$ and $X \bullet\!\!-\!\!\bullet Y$ are considered equivalent to, respectively, $Y \leftrightarrow X$ and $Y \bullet\!\!-\!\!\bullet X$.*

The role of a PAG is to represent a set of independence-equivalent MAGs. PAGs are thus to MAGs what PDAGs are to DAGs. This is summarized in Table 4.1. By "ground truth," we mean the target

	Causally sufficient case		**Causally insufficient case**	
	Ground-truth representation	Equivalence-class representation	Ground-truth representation	Equivalence-class representation
Graph type	DAG (directed acyclic graph)	PDAG (partially directed acyclic graph)	MAG (mixed ancestral graph)	PAG (partial ancestral graph)
Edge types	$\rightarrow$	$\rightarrow$, $-$	$\rightarrow$, $\leftrightarrow$	$\rightarrow$, $\circ\!\!\rightarrow$, $\circ\!\!-\!\!\circ$, $\leftrightarrow$
			Edge patterns: $*\!\!-\!\!*$, $*\!\!\rightarrow$; (PAG) $*\!\!-\!\!\circ$	

Table 4.1: Relationships between DAGs, PDAGs, MAGs, and PAGs.

graph that we learn—a DAG in the causally sufficient case, and the MAG representation of the projection of the hidden structures in the causally insufficient case.

We should explicate the relations between the graph types: the representation of the equivalence class is a pattern that matches one or more DAGs (in causally sufficient cases) or MAGs (in causally insufficient cases). In a PDAG, the $-$ arrow is a pattern (for either $\rightarrow$ or $\leftarrow$ in a DAG), and a PDAG $\rightarrow$ directly matches a DAG $\rightarrow$.

Similarly, by PAG, we represent a set of MAGs. PAGs allow four kinds of arrows: $\rightarrow$, $\circ\!\!\rightarrow$, $\circ\!\!-\!\!\circ$, and $\leftrightarrow$. $X \rightarrow Y$ in the PAG denotes direct causation $X \rightarrow Y$ in the projection; $X \leftrightarrow Y$ indicates the presence of a latent cause $X \leftarrow H \rightarrow Y$ (without excluding direct causation), and translates to the same bidirected link in the MAG; $X \circ\!\!\rightarrow Y$ denotes either true causation $X \rightarrow Y$ or a latent cause $X \leftarrow H \rightarrow Y$ (or a combination of both); finally, $X \circ\!\!-\!\!\circ Y$ denotes potential causation from $X \rightarrow Y$ or $Y \rightarrow X$ and/or a latent common cause $X \leftarrow H \rightarrow Y$ in the projection, and is thus the most "agnostic" link.

Additionally, for MAGs and PAGs, an asterisk as an arrowhead is a wildcard for any possible endpoints of a link, such that $X *\!\!\rightarrow Y$, for instance, means in a PAG any of $X \rightarrow Y$, $X \circ\!\!\rightarrow Y$, and $X \leftrightarrow Y$. Additionally, we also use the notation $X *\!\!-\!\!* \underline{Z} *\!\!-\!\!* Y$ to indicate that Z is a *definite noncollider* for X and Y, such that any of $X *\!\!\rightarrow Z \rightarrow Y$, $X \leftarrow Z \leftrightarrow Y$, or $X \leftarrow Z \rightarrow Y$ can occur, but not $X *\!\!\rightarrow Z \leftarrow\!\!* Y$. Why we need an explicit notation for noncolliders is made clear later.

To illustrate how MAGs and PAGs are related to a latent structure, consider the causal graph shown in Figure 4.3 (a). There, the hidden variable H is a cause of 3 observed variables, and L is a hidden variable in the causal chain from Z to W. All other variables are observed. In (b), we show the projection of (a): note that we lose information about L and about the similarity of H_1, H_2, and H_3, which are actually the same variable for the reasons examined in Subsection 4.1.1. The corresponding MAG is shown in (c), and in (d) the PAG that represents the class of independence-equivalent MAGs of which (c) is a member. Note how the causal-underdetermination problem influences PAG learning: for instance, the model shown in (e), if learned as a PAG, will also be represented as in (d).

4.2.2 IC*: Adaptation of IC

Pearl (2000) showed how IC can very easily be extended so that it builds a PAG to represent an equivalence class of latent structure. The modified algorithm is called IC* and is outlined here.[13]

[13] Originally, Pearl & Verma (1991) had used "IC" to refer to the general algorithm that does not assume causal sufficiency.

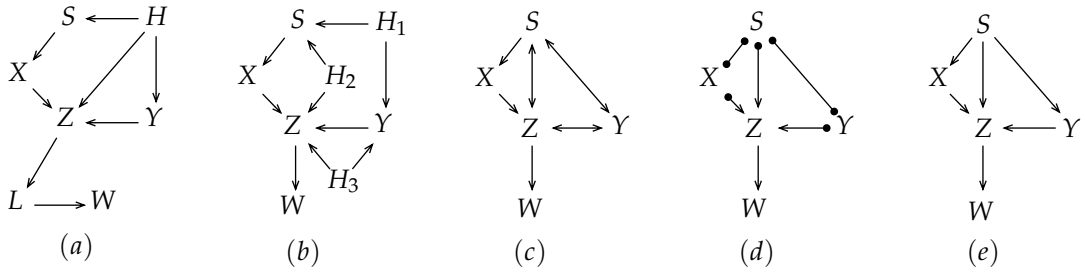


Figure 4.3: (a) Example of a causal structure with the hidden variables H and L. (b) Projection of (a). (c) MAG representing the projection (b). (d) PAG representing the class of projections that are independence-equivalent to (c). (e) Another structure with no hidden variable whose learned PAG is (d).

The main difference with IC is that instead of creating V-structures like in a PDAG, we now just add *arrow heads* into the identified colliders, independently of what the other arrow endpoints are:

1. Adjacencies: insert the "agnostic link" $X \bullet\!\!-\!\!\bullet Y$ if $\nexists \mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$;
2. V-structures: for all substructures $X \ast\!\!-\!\!\ast Z \ast\!\!-\!\!\ast Y$ where X and Y are nonadjacent, add arrowheads into Z as $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y$ if $Z \notin \mathbf{S}_{XY}$;
3. Orientations: use rules to further orient "agnostic" endpoints wherever possible.

The second difference with respect to PDAG learning is in the set of rules applied in Step 3 to further orient the graph. Those rules are detailed in the next subsection together with the FCI algorithm.

Again, like IC, IC* is more an algorithm specification than a real algorithm: in particular, Step 1 is problematic, because it requires a subset search, which has an exponential time complexity. FCI is to IC* what PC is to IC: it details how to perform this search more efficiently.

4.2.3 The FCI Algorithm

To the best of our knowledge, the Fast Causal Inference (FCI) algorithm is regarded as the state-of-the-art implementation of a PAG-learning algorithm. We list its pseudocode in Algorithm 4.1. The notation $\mathbf{Bd}_{\mathcal{G}}(X)$ stands for the set of direct neighbors of X in the graph being constructed $\mathcal{G}$ (which potentially changes at each iteration). The set $\mathbf{ExtDSep}(X, Y)$ is defined below.

If we compare FCI with PC (Algorithm 3.1 on page 35), we see that lines 2–18 are almost identical: FCI looks for d-separating sets of increasing size in the direct neighborhood of each pair of connected variables $X - Y$, and then creates V-structures accordingly. The V-structures, however, are only used temporarily: FCI goes through another round of searches at lines 19–27, further trying to d-separate linked pairs. This is so because in a MAG, examining the local neighborhood for each pair (X, Y) as defined by $\mathbf{Bd}(X) \cup \mathbf{Bd}(Y)$ is not enough to guarantee that we will always find a d-separating set there if one exists.

We show this with an example, taken from Spirtes et al. (2001). Suppose we have the hidden structure shown in Figure 4.4 (a), and its corresponding projection as MAG in (b), and we are looking for a set $\mathbf{S}_{XY}$ such that $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$. On (a), it is easy to see that $\{H_1, H_2\}$ is such a set. H_1 and H_2, however, are hidden variables, and are not available to FCI. In (c), we show what

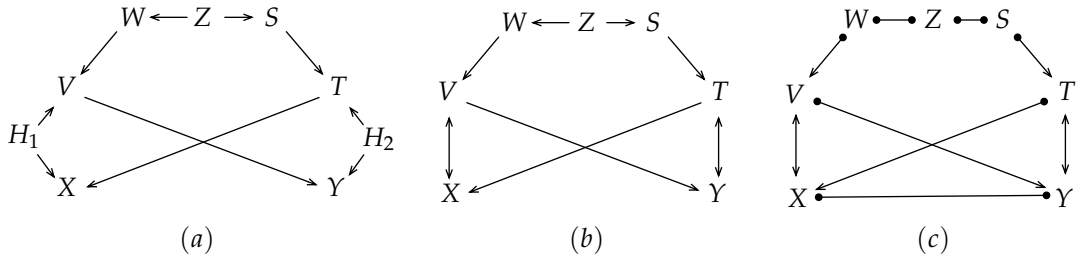


Figure 4.4: (a) Example of a hidden structure with H_1 and H_2 unobserved. (b) The MAG representing the projection of (a). (c) The PAG learned by FCI if interrupted right after line 18.

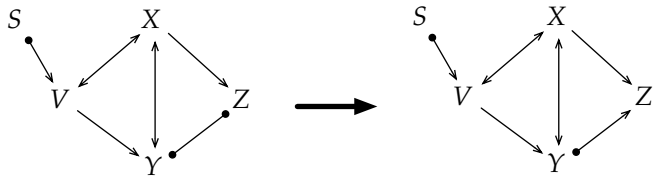


Figure 4.5: Path $\pi = \langle S, V, X, Z, Y \rangle$ is a DDP for Z. Rule 4 adds an arrowhead into Z if $Z \notin \mathbf{S}_{SY}$.

adjacencies FCI detects if it limits its search for a d-separating set to the same nodes as those that PC considers: we see that link $X \bullet\!\!-\!\!\bullet Y$ is in (c) but not in (b). Actually, the joint neighborhoods of X and Y are $\{V, T\}$, and no subset of this can make X and Y independent. Not conditioning on V opens the path $X \leftrightarrow V \rightarrow Y$. So V must be conditioned on, and T as well, for reasons of symmetry. But then, conditioning on both opens the path $X \leftrightarrow V \leftarrow W \leftarrow Z \rightarrow S \rightarrow T \leftrightarrow Y$. The only way to find the needed set is to include one of $\{W, Z, S\}$ in it, none of which is in $\mathbf{Bd}(X)$ or $\mathbf{Bd}(Y)$.

Enter then the second search for a d-separating set at lines 19–27. The searched nodes are now referred to using the $\mathbf{ExtDSep}(X, Y)$ notation. $\mathbf{ExtDSep}(X, Y)$ is the union of $\mathbf{PossibleDSep}(X, Y)$ and $\mathbf{PossibleDSep}(Y, X)$ (Spirtes et al., 1995). $\mathbf{PossibleDSep}(X, Y)$ is the set of nodes Z where there is an undirected path π between X and Z such that for each subpath $S \ast\!\!-\!\!\ast W \ast\!\!-\!\!\ast T$ of π, either (a) W is a collider, or (b) W is not marked as a noncollider and S, W, T are a triangle. (A triangle is a set of three nodes all adjacent to one another.) This definition is such that $\mathbf{ExtDSep}(X, Y)$ reaches out for precisely the nodes like W and S in the previous example, which we know can block additional open paths.

After this extended search, FCI reorients everything as $\bullet\!\!-\!\!\bullet$, the agnostic link, and orients V-structure as specified by Step 2 of IC*. The last step is maximally orienting the PAG (Step 3 of IC*). We list the orientation rules as a separate procedure in Algorithm 4.2, as we reuse them in our algorithm.[14]

Rule 1 preserves acyclicity. Rule 2 honors the noncollider constraint when one of the two endpoints is an arrowhead. Rule 3 orients double-triangle structures; for instance, it orients $S \bullet\!\!\rightarrow Z$ in Figure 4.3 (c). Rule 4 needs the following definition (Spirtes et al., 1995).

Definition 4.6 (Definite discriminating path) *In a PAG $\mathcal{G}$, π is a* definite discriminating path *or* DDP *between S and Y (S, Y nonadjacent) for Z ($Z \neq S, Y$) if and only if π is an undirected path*

[14] Although Pearl (2000) lists only two rules for the IC* algorithm, FCI proposes more rules, leading to more orientation possibilities. We describe only the more complete rules as stated in Spirtes et al. (1995, 2001).

Algorithm 4.1 The FCI algorithm

procedure $\mathcal{G} = \text{FCI}(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot), \mathbf{V}, f\ )$
Input: $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$: a conditional-independence oracle
$\mathbf{V}$: set of variables in the analysis
f : maximum size of conditioning sets; default $= |\mathbf{V}| - 2$
Output: $\mathcal{G}$: maximally oriented partial ancestral graph

// Initialization
$\mathcal{G} \leftarrow$ fully connected graph over $\mathbf{V}$ with $\bullet\!\!-\!\!\bullet$ links
$i \leftarrow 0$

// Remove extra adjacencies
while $\exists (X \bullet\!\!-\!\!\bullet Y)$ such that $|\mathbf{Bd}_{\mathcal{G}}(X)| > i$ **and** $i \leq f$ **do**
  for each $X \bullet\!\!-\!\!\bullet Y$ such that $|\mathbf{Bd}_{\mathcal{G}}(X)| > i$ **do**
    for each $\mathbf{S} \subseteq \mathbf{Bd}_{\mathcal{G}}(X) \setminus \{Y\}$ such that $|\mathbf{S}| = i$ **do**
      if $(X \perp\!\!\!\perp Y \mid \mathbf{S})$ **then**
        remove link $X \bullet\!\!-\!\!\bullet Y$ from $\mathcal{G}$
        $\mathbf{S}_{XY}, \mathbf{S}_{YX} \leftarrow \mathbf{S}$
        break
      end if
    end for
  end for
  $i \leftarrow i + 1$
end while
for each $X \ast\!\!-\!\!\ast Z \ast\!\!-\!\!\ast Y$ such that X, Y are nonadjacent **and** $Z \notin \mathbf{S}_{XY}$ **do**
  create (temporary) V-structure $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y$
end for

// Remove additional extra adjacencies
for each pair of adjacent variables X, Y **do**
  for each $\mathbf{S} \subseteq \mathbf{ExtDSep}(X,Y) \setminus \{X,Y\}$ **do**
    if $(X \perp\!\!\!\perp Y \mid \mathbf{S})$ **then**
      remove link $X \bullet\!\!-\!\!\bullet Y$ from $\mathcal{G}$
      $\mathbf{S}_{XY}, \mathbf{S}_{YX} \leftarrow \mathbf{S}$
      break
    end if
  end for
end for

reorient every link as $\bullet\!\!-\!\!\bullet$

// Orient V-structures
for each $X \ast\!\!-\!\!\ast Z \ast\!\!-\!\!\ast Y$ such that X, Y nonadjacent **do**
  if $Z \notin \mathbf{S}_{XY}$ **then**
    create V-structure $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y$
  else
    mark Z as noncollider: $X \ast\!\!-\!\!\ast \underline{Z} \ast\!\!-\!\!\ast Y$
  end if
end for

return MAXIMALLYORIENTPAG$(\ \mathcal{G}, \forall (X,Y) : \mathbf{S}_{XY}\ )$
end procedure

Algorithm 4.2 Maximal orientation of a PAG

```
1: procedure G = MaximallyOrientPAG( G, a list of sets S_XY )
   Input:   G :  partial ancestral graph
            S_XY : for (some) nonadjacent pairs (X, Y): a d-separating set of variables
   Output:  G :  maximally oriented partial ancestral graph
2:   while G is changed by some rule do
3:     for each X *-• Y such that there is a directed path from X to Y do        // Rule 1
4:       orient as X *-> Y
5:     end for
6:     for each X *-> Z •-* Y do                                                 // Rule 2
7:       orient as X *-> Z -> Y
8:     end for
9:     for each X *-> Z <-* Y with S *-• Z and S ∈ S_XY do                       // Rule 3
10:      orient as S *-> Z
11:    end for
12:    for each definite discriminating path π between S and Y for Z do          // Rule 4
13:      if X *-* Y where X is adjacent to Z on π and X, Z, Y are a triangle then
14:        if S_SY exists and Z ∉ S_SY then
15:          orient as X *-> Z <-* Y
16:        else
17:          mark Z as a noncollider X *-* Z *-* Y
18:        end if
19:      end if
20:    end for
21:  end while
22:  return G
23: end procedure
```

between S and Y containing Z, Z precedes Y on π, every vertex V between S and Z on π is a collider or a definite noncollider on π, and:

1. *If V and V′ are adjacent on π and V′ is between V and Z, then $V \ast\!\!\rightarrow V'$ on π;*
2. *If V is between S and Z on π and V is a collider on π, then $V \rightarrow Y$ in $\mathcal{G}$, otherwise $Y \ast\!\!\rightarrow V$ in $\mathcal{G}$.*

Figure 4.5 shows an example for a DDP and for Rule 4, which produces the orientation $Y \bullet\!\!\rightarrow Z$. For a more extensive justification and a proof of these rules, see Spirtes et al. (1995, 2001).

The time complexity of FCI makes it non-scalable for larger networks. In particular, the two subset searches at lines 6 and 20 of Algorithm 4.1 are computationally costly in dense networks. In the next section, we present an algorithm that takes another approach at PAG learning to tackle problems larger than those that FCI can handle.

4.3 Efficient PAG Learning: the MBCS* Algorithm

In this section, we propose a PAG-learning algorithm, MBCS*, which is more efficient than FCI in the sense that it performs much fewer conditional-independence tests, whose average conditioning-set size is smaller. We show in Section 4.4 that MBCS* compares very favorably to FCI on test networks in terms of computational tractability, while achieving similar accuracy.

Our basic intention for MBCS*, like the novel algorithms introduced in the previous chapter, is to first learn a local neighborhood for each variable, and then make local adjustments so that the resulting graph is indeed a causal PAG.

4.3.1 Markov Blankets without Causal Sufficiency

As for our generic algorithm for the causally sufficient case, presented in Subsection 3.2.4, the local neighborhood we will be learning for each node is the Markov blanket of each variable.Recalling the definition of Markov blanket from Definition 3.3: $\mathbf{Mb}(X)$ is defined as the smallest set of variables $\mathbf{Mb}(X)$ such that $\forall Y \in \mathbf{V} \setminus \mathbf{Mb}(X) \setminus \{X\} : (X \perp\!\!\!\perp Y \mid \mathbf{Mb}(X))$.

In a DAG that has the perfect-map property, we saw in Property 3.7 on page 38 that the Markov blanket is the set of parents, children, and children's parents of a given node X, and is unique. How does this translate to the causally insufficient case for MAGs? To answer this, we need a new graphical definition.

Definition 4.7 (District) *In a MAG $\mathcal{G}$, a node D is part of the* district *of a node X, noted $D \in \mathbf{Dis}_{\mathcal{G}}(X)$, if and only if there is a path between X and D where all edges are bidirected.*

If the MAG is a DAG (i.e., has no bidirected links) then the district of any node X is empty. In a way, the district of X can be seen as the nodes connected to X by a series of hidden common causes.

Property 4.8 *In a MAG that has the perfect-map property, the Markov blanket $\mathbf{Mb}(X)$ of a node X is the set of parents, children, children's parents (spouses) of X, as well as the district of X and of the children of X, and the parents of each node of these districts.*

> **Proof** First, in the case where the MAG is a DAG (i.e., where all districts are empty), Property 4.8 reduces to the familiar property that in a DAG that has the perfect-map property, $\mathbf{Mb}(X)$ is the set of parents, children, and spouses of X (Property 3.7). We now prove its generalization, using the perfect-map property of $\mathcal{G}$ to use the d-separation and the conditional-independence criteria interchangeably.
>
> We first show the implication "Y is a parent of X, child of X, spouse of X, member of the district of X, member of the district of a child of X, or parent of a node in these districts $\implies$ $Y \in \mathbf{Mb}(X)$." If Y is a parent of X, a child of X, or is directly linked to X with a bidirected link, then no set can m-separate X from Y owing to the direct link and we conclude $Y \in \mathbf{Mb}(X)$. If Y is a spouse of X, a member of the district of X, a member of the district of a child of X, or a parent of a node in these districts of X, then there is a path from X to Y where every node is a collider: $X \mathrel{*\!\!\rightarrow} Z_1 \leftrightarrow Z_2 \leftrightarrow \cdots \leftrightarrow Z_m \mathrel{\leftarrow\!\!*} Y$. First, $Z_1 \in \mathbf{Mb}(X)$ because of its direct link, as shown above. Then, we have $(X \rightleftharpoons Z_2 \mid Z_1)$ because conditioning on the collider Z_1, by definition, opens a path between X and Z_2. So we have to conclude $Z_2 \in \mathbf{Mb}(X)$. Similarly, we find $(X \rightleftharpoons Z_i \mid Z_1, Z_2, \cdots, Z_{i-1})$, and finally, $(X \rightleftharpoons Y \mid Z_1, Z_2, \cdots, Z_m)$, for the same reason; so $Y \in \mathbf{Mb}(X)$. This proves the first implication.
>
> We now show the implication "$Y \in \mathbf{Mb}(X) \implies$ Y is a parent of X, child of X, spouse of X, member of the district of X, member of the district of a child of X, or parent of a node in these districts." Take some $Y \in \mathbf{Mb}(X)$ and assume it is not a parent of X, a child of X, a spouse of X, a member of the district of X, a member of the district of a child of X, or a parent of a node in these districts: then there is at least one path from X to Y of length $\ell \geq 2$ that contains at least one noncollider. In either case, the first noncollider nodes on this path can only be a parent of X, a child of X, a spouse of X, a member of the district of

X, a member of the district of a child of X, or a parent of a node in these districts, so it is in $\mathbf{Mb}(X)$ as shown above. Moreover, conditioning on it blocks the path by definition because the node is a noncollider. This means that there is a set $\mathbf{Z}$ such that $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$ and hence that Y cannot be in $\mathbf{Mb}(X)$ because the definition of $\mathbf{Mb}(X)$ requires minimality. This leads to a contradiction and proves the second implication, which together with the first implication proves the equivalence. □

4.3.2 MCBS⋆

MBCS⋆ reuses algorithmic ideas from the previous chapter: first, it detects the Markov blankets for each variable; second, it examines the triangle structures to identify colliders and noncolliders; finally, it uses the same orientation rules as FCI to obtain the maximally oriented PAG. We detail the first two steps below. MBCS⋆ is a variant of the generic algorithm based on Markov blankets that we presented in the previous chapter, with the modifications required for handling the lack of causal sufficiency correctly.

In Chapter 3, we presented portions of the algorithms separately (Markov-blanket construction, collider-set search, resolving the Markov blanket). Here, for simplicity, we present MCBS⋆ all together in Algorithm 4.3. The only call it makes is the collider-set search, for which we reuse the previously defined Algorithm 3.5 on page 44.

Step 1: Learning the Markov Blanket

The first phase of MBCS⋆ builds an undirected graph where each variable is connected to all members of its Markov blanket.

We use algorithmic ideas from Margaritis & Thrun (1999) to learn the Markov blanket of a node with a linear number of conditional-independence tests (proof in the supplemental material of Margaritis & Thrun, 1999).[15] This technique is used in lines 4–14 of Algorithm 4.3. The resulting graph is an undirected graph very similar to the moral graph defined in the context of DAGs (Definition 3.12). We overload the same term in the context of MAGs to denote the undirected graph where each node is connected to its parents, children, spouses, district, children's district, district's parents, and children's district's parents. With respect to the target MAG, it therefore contains spurious links to members of the Markov blanket that are not parents and children. We call them *SD links* (for Spouse/District). Removal of these links is done in the second step of MBCS⋆, which follows.

Step 2: Removing the SD Links

In the second step of MBCS⋆, each undirected edge must be identified as either an SD link to be removed, or a true link of the original MAG to be kept. Direct parents and children are dependent given any conditioning set, while spouses and district members (and their parents) can be made independent. For each link $X - Y$, a search is thus performed to try to m-separate the two connected nodes. This search can be limited to the smallest of the Markov blankets of X and

[15] As in Chapter 3, we could use various feature-selection techniques to learn the Markov blanket. This is a generalization of this approach, which will be the topic of future research.

Algorithm 4.3 The MBCS* algorithm

procedure $\mathcal{G} = \text{MBCS}^\star(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot), \mathbf{V}\ )$
Input: $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$: a conditional-independence oracle
$\mathbf{V}$: set of variables in the analysis
Output: $\mathcal{G}$: maximally oriented partial ancestral graph

// Initialization
$\mathcal{G} \leftarrow$ empty graph over $\mathbf{V}$

// Find Markov blankets
for each $X \in \mathbf{V}$ **do**
  $\mathbf{S} \leftarrow$ empty set of Markov-blanket variables
  while $\exists Y \in \mathbf{V} \setminus \{X\}$ such that $(X \not\perp\!\!\!\perp Y \mid \mathbf{S})$ **do**
    add Y to $\mathbf{S}$
  end while
  while $\exists Y \in \mathbf{S}$ such that $(X \perp\!\!\!\perp Y \mid \mathbf{S} \setminus \{Y\})$ **do**
    remove Y from $\mathbf{S}$
  end while
  for each $Y \in \mathbf{S}$ **do**
    add link $X \bullet\!\!-\!\!\bullet Y$
  end for
end for

// Add noncollider constraints
for each $X \bullet\!\!-\!\!\bullet Z \bullet\!\!-\!\!\bullet Y$ such that X, Y nonadjacent **do**
  mark as noncollider $X \bullet\!\!-\!\!\bullet \underline{Z} \bullet\!\!-\!\!\bullet Y$
end for

// Adjust local structures (collider-set search)
$\mathbf{C} \leftarrow$ empty list of collider-orientation directives
for each $X \ast\!\!-\!\!\ast Y$ in a fully connected triangle **do**
  $\mathbf{S}_{XY} \leftarrow \text{FINDDSEPARATINGSET}(\ (\cdot \perp\!\!\!\perp \cdot \mid \cdot),\ \mathcal{G},\ X - Y\ )$
  if $\mathbf{S}_{XY} \neq$ **null then** *// save orientation directives*
    remove link $X \ast\!\!-\!\!\ast Y$ from $\mathcal{G}$
    $\mathbf{Z} \leftarrow \mathbf{Tri}(X - Y) \setminus \mathbf{S}_{XY}$ *// this is the collider set*
    for each $Z \in \mathbf{Z}$ **do**
      $\mathbf{C} \leftarrow \mathbf{C} \cup \{(X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y)\}$
    end for
    for each $Z \in \mathbf{S}_{XY}$ **do**
      mark as noncollider $X \ast\!\!-\!\!\ast \underline{Z} \ast\!\!-\!\!\ast Y$
    end for
  end if
end for

for each orientation directive $(X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y) \in \mathbf{C}$ **do** *// orient edges*
  if edges $X \ast\!\!-\!\!\ast Z$ and $Y \ast\!\!-\!\!\ast Z$ still exist in $\mathcal{G}$ **then**
    create V-structure $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y$
  end if
end for

return $\text{MAXIMALLYORIENTPAG}(\ \mathcal{G},\ \forall(X,Y)\colon \mathbf{S}_{XY}\ )$
end procedure

Y, as by definition they contain all nodes that can minimally make them independent from each other, provided they are linked by an SD link. If such a m-separating set $\mathbf{S}_{XY}$ is found, the link is removed. Interestingly, in the causally insufficient case as well, identifying a m-separating set $\mathbf{S}_{XY}$ also identifies the collider set for X and Y.

Collider sets are useful because each node in them satisfies the property of a collider (2.14) and reveals a V-structure, whereas a V-structure in a causally insufficient context should be understood as a structure such as $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ (rather than the traditional $X \rightarrow Z \leftarrow Y$). Suppose $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$: then each node Z (connected by a non-SD link to both X and Y) not in $\mathbf{S}_{XY}$ is a collider. This follows from the fact that for each path $X \mathrel{*\!\!-\!\!*} Z \mathrel{*\!\!-\!\!*} Y$ where $Z \notin \mathbf{S}_{XY}$, the only structural possibility is to have arrowhead pointing into Z according to the definition of m-separation. Similarly, if $Z \in \mathbf{S}_{XY}$, then any orientation is possible, save for a collider. These two types of constraints appear respectively in lines 25 and 28 of Algorithm 4.3. Note that more noncollider constraints are added in line 16: in the case $X \bullet\!\!-\!\!\bullet Z \bullet\!\!-\!\!\bullet Y$ with X, Y nonadjacent, we know that Z cannot be a V-structure owing to the following lemma.

Lemma 4.9 *In the moral graph $\mathcal{G}_m$ of a DAG or a MAG $\mathcal{G}$, whenever the pattern $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ occurs in $\mathcal{G}$, then X and Y are linked in $\mathcal{G}_m$.*

> **Proof** The pattern $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ means any of $X \rightarrow Z \leftarrow Y$, $X \leftrightarrow Z \leftarrow Y$, $X \rightarrow Z \leftrightarrow Y$, or $X \leftrightarrow Z \leftrightarrow Y$. In any of these four cases, we have $Y \in \mathbf{Mb}(X)$ as per Property 4.8 as Y is, respectively, a spouse of X, a parent of a member of the district of X, a member of the district of a child of X, or a member of the district of X. As per Definition 3.12, Y is thus linked to X in $\mathcal{G}_m$. □

The search for collider sets and simultaneously for m-separating sets at line 20 is performed following the implementation proposed in the previous chapter; see also the discussion on page 45 about why V-structure orientations must be delayed to line 34 instead of being made immediately at line 25.

Correctness

It appears that, whereas FCI is a good deal more complicated than PC, the Markov-blanket ideas from Section 3.2 generalize quite well as MBCS*. How can we guarantee that indeed, we do consider all potential m-separating sets and do not take incorrect shortcuts? After all, if we apply a straightforward generalization of the PC approach to solve causally insufficient problems, we can easily show that it leads to problematic situations and to spurious arcs (see example around Figure 4.4).

To offer this guarantee, we now prove that MBCS* correctly identifies all adjacencies and V-structures. The final orientation step (Algorithm 4.2) requires conditional-independence information for Rules 3 and 4: we also prove that MBCS* provides all necessary information.

Theorem 4.10 *After line 14, MBCS* has correctly identified all Markov blankets for each variable.*

> **Proof** There are several ways to show this. An easy way is to observe that (i) line 6 will in general add at least all variables Y of the Markov blanket of X plus some more; and (ii) line 9 will always remove from $\mathbf{S}$ all variables that are not in the Markov blanket.
>
> (i) At least all Markov-blanket variables are selected, because they are by definition characterized by the fact that no matter $\mathbf{S}$, $(X \perp\!\!\!\perp Y \mid \mathbf{S})$ never holds. Certain other variables can be included in $\mathbf{S}$,

however: all weakly relevant variables may be selected. Let us recall that a weakly relevant variable Y for X is such that $\exists \mathbf{Z} : (X \not\perp\!\!\!\perp Y \mid \mathbf{Z})$ (which follows from Definition 3.5 on page 37). In our case, it can happen that $\mathbf{S} = \mathbf{Z}$ when we test $(X \perp\!\!\!\perp Y \mid \mathbf{S})$, so that Y is added to $\mathbf{S}$ although it is not in $\mathbf{Mb}(X)$.

(ii) When we reach line 9, we know that $\mathbf{Mb}(X) \subseteq \mathbf{S}$, and we have to show that each weakly relevant feature $W \in \mathbf{S} \setminus \mathbf{Mb}(X)$ will be removed. To prove this, we need to show that $(X \perp\!\!\!\perp W \mid \mathbf{S} \setminus \{W\})$ holds. We distinguish two cases: either (ii^a) $\mathbf{Mb}(X) = \mathbf{S} \setminus \{W\}$, or (ii^b) $\mathbf{Mb}(X) \subsetneq \mathbf{S} \setminus \{W\}$. Case (ii^a) is trivial, because by the definition of a Markov blanket, $(X \perp\!\!\!\perp W \mid \mathbf{Mb}(X))$ always holds if $W \in \mathbf{V} \setminus \mathbf{Mb}(X) \setminus \{X\}$. For case (ii^b), suppose that $(X \not\perp\!\!\!\perp W \mid \mathbf{S} \setminus \{W\})$ holds. This means that there exists at least one variable $W' \in \mathbf{S} \setminus \mathbf{Mb}(X)$, $W' \neq W$, which opens a dependency path between X and W. The only possibility for this, as dictated my m-separation in Definition 4.4, is for W' to be a collider on a path from X to W: $X \cdots Z \mathrel{*\!\!\rightarrow} W' \mathrel{\leftarrow\!\!*} W$, where Z is the member of the Markov-blanket of X closest to W'. Now, the link $Z \mathrel{*\!\!\rightarrow} W'$ must actually be of the form $Z \leftrightarrow W'$ for this path to be opened by the definition of m-separation, but then we would have to conclude that $W' \in \mathbf{Mb}(X)$ by the properties of $\mathbf{Mb}(X)$ in MAGs (Property 4.8). This leads to a contradiction, so in case (ii^b) we also conclude $(X \perp\!\!\!\perp W \mid \mathbf{S} \setminus \{W\})$, and can thus deduce $(X \perp\!\!\!\perp W \mid \mathbf{S} \setminus \{W\})$ for case (ii) in general. Hence, all weakly relevant variables W are removed from $\mathbf{S}$ at line 9.

As Step (ii) always follows Step (i), after Step (ii) we have correctly identified all Markov blankets. □

Theorem 4.11 *The collider-set search correctly identifies all colliders and removes all links $X \bullet\!\!-\!\!\bullet Y$ introduced by the construction of the moral graph where $\exists \mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \perp\!\!\!\perp Y \mid \mathbf{S})$.*

Proof Lemma 3.15 on page 42 says the collider set for a pair (X, Y) exists and is unique whenever $\exists \mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\} : (X \perp\!\!\!\perp Y \mid \mathbf{S})$. As MBCS* removes a link as soon as a collider set is identified, it will remove such links. □

MBCS* calls the MAXIMALLYORIENTPAG() procedure, which needs d-separating-set information and noncollider constraints to apply the orientation rules. If we wish to prove that MBCS* maximally orients the returned graph, as FCI does, we need to prove that either MBCS* passes the information required to apply all orientation rules to MAXIMALLYORIENTPAG(), or, when MBCS* fails to pass this information, that it has already oriented the concerned arcs.

Theorem 4.12 *The MBCS* algorithm passes to the* MAXIMALLYORIENTPAG() *procedure both the required $\mathbf{S}_{XY}$ sets and the noncollider constraints of the type $X \mathrel{*\!\!\rightarrow} \underline{Z} \mathrel{\leftarrow\!\!*} Y$, where X and Y are nonadjacent.*

Proof We first show that the sets $\mathbf{S}_{XY}$ are correctly passed by MBCS* for the two rules that need them, Rules 3 and 4. Rule 3 of the procedure MAXIMALLYORIENTPAG() orients patterns $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ with $S \bullet\!\!-\!\!* Z$ into $S \mathrel{*\!\!\rightarrow} Z$ if $S \in \mathbf{S}_{XY}$—see the example in Figure 4.6 (a). So, whenever the MAG shows this pattern, $\mathbf{S}_{XY}$ must exist, and S must be in $\mathbf{S}_{XY}$. We distinguish two cases: either $S \in \mathbf{Tri}(X - Y)$ or $S \notin \mathbf{Tri}(X - Y)$. In the former case, $\mathbf{S}_{XY}$ has been assigned a value because, owing to $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$, the spouse link $X - Y$ has been removed. In the latter case, where either or both of the pairs (X, S) or (S, Y) are not adjacent, as in Figure 4.6 (b), Rule 3 is not needed to orient $S \mathrel{*\!\!\rightarrow} Z$ as $S \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ is a V-structure recognized by MBCS* before calling MAXIMALLYORIENTPAG().

We must now show that we have $S \in \mathbf{Mb}(Y)$ whenever the conditions for Rule 4 are fulfilled and we additionally have $X \mathrel{*\!\!\rightarrow} Z \mathrel{\leftarrow\!\!*} Y$ in the original MAG. This guarantees that $\mathbf{S}_{SY}$ exists and is assigned a value by MBCS*, thus leading to correct orientation. Let V be the second node on the DDP π between S and Y for Z. In any case, $V \neq Z$; otherwise, we cannot apply Rule 4 anyway, as this rule requires a triangle (X, Z, Y) and would then require S and Y to be adjacent: this would contradict Definition 4.6. So V is between S and Z on π. Now, there are two possibilities. If V is a definite noncollider for its neighbors on

π, then by Definition 4.6 $Y \circ\!\!\rightarrow V$. It follows that S is either a parent of, or a member of the district of, V; and V is either a child of, or a member of the district of, Y; and thus by Property 4.8, $S \in \mathbf{Mb}(Y)$. If V is a collider, then we consider its neighbor V', where $V \leftrightarrow V'$, by Definition 4.6 (i) and because V is a collider. If V' is a definite noncollider, then, similarly $Y \circ\!\!\rightarrow V'$, so that V is either a parent of, or a member of the district of, V', and S, being a parent of V, is again in $\mathbf{Mb}(Y)$ by Property 4.8. If V' is again a collider, we check the same conditions with its neighbor V'', and so on. Eventually, either we find a noncollider and we are in the same situation as just described, or we reach Z with a path of colliders only, of the type $S \circ\!\!\rightarrow V_1 \leftrightarrow V_2 \leftrightarrow \cdots \leftrightarrow V_m \leftrightarrow Z \leftarrow\!\!\circ Y$. S is thus either a parent of, or a member of the district of, Z; and Z is either a child of, or a member of the district of, Y, so that, here too, $S \in \mathbf{Mb}(Y)$.

We now show that MBCS* identifies all noncollider constraints of the type $X \circ\!\!-\!\!\circ \underline{Z} \circ\!\!-\!\!\circ Y$, where X and Y are nonadjacent. We distinguish two cases: either $X \notin \mathbf{Mb}(Y)$ or $X \in \mathbf{Mb}(Y)$. In the former case, X and Y are not directly linked in the moral graph, and thus, by Lemma 4.9, Z cannot be a collider. In the latter case, where $X \in \mathbf{Mb}(Y)$, there is a link in the moral graph between X and Y. If this is an SD link, it will be removed by the collider-set search, and will be included in $\mathbf{S}_{XY}$ whenever it is needed to make X and Y independent. This case can only occur if Z is a noncollider as per Definition 4.4. □

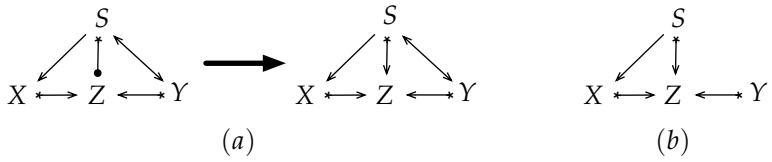


Figure 4.6: (a) Example for Rule 3 when $S \in \mathbf{S}_{XY}$ and $S \in \mathbf{Tri}(X - Y)$. (b) Example where $S \notin \mathbf{Tri}(X - Y)$, but where Rule 3 is not needed for the same orientation owing to Z's being an unshielded collider for (S, Y).

Theorem 4.13 *Assuming faithfulness, the MBCS* algorithm returns the maximally oriented PAG representing the equivalence class of MAGs to which the latent-structure projection belongs.*

Proof Theorem 4.10 and Theorem 4.11 show that all adjacencies are detected correctly, and Theorem 4.12 shows that all collider and noncollider constraints are also detected correctly. The theorem thus holds, provided MAXIMALLYORIENTPAG() (Spirtes et al., 2001) returns the maximally oriented PAG given a PAG where all unshielded colliders are oriented and all noncollider constraints of the type $X \circ\!\!-\!\!\circ \underline{Z} \circ\!\!-\!\!\circ Y$, where X and Y are nonadjacent, are identified. □

4.4 Benchmarks and Experimental Evaluation

In this section, we report on tests we conducted to compare MBCS* to FCI, which is, to the best of our knowledge, the only other published implementation of a PAG-learning algorithm. As in the previous chapter, we have two separate benchmarks, one that compares the number of conditional-independence tests made by the algorithms, and one that compares the structural accuracy of the PAG they return with respect to the reference MAG.

4.4.1 Experimental Setup

We compare FCI and MBCS* with a series of experiments. We took two standard benchmark networks, **Alarm** and **Hailfinder**, and for each of them, chose to hide 0, 1, 2, and 3 variables,

creating in total 8 learning problems. According to the variable numbering in Figures 3.7 and 3.9 on page 57, the sets of hidden variables were, for **ALARM**, $\{27\}$, $\{27,9\}$, and $\{27,9,25\}$; for **HAILFINDER**, we chose the sets $\{21\}$, $\{21,35\}$, and $\{21,35,13\}$.

Hiding at most 3 variables may seem like making it too easy for the algorithms, but the variables we chose were rather densely connected nodes, with multiple parents and/or children, such that the relative graph density increased quite significantly. We should keep in mind that learning the structure without assuming causal sufficiency, even with no hidden variable, is harder than with causal sufficiency, because more possible scenarios must be investigated to explain each association between two variables. The complexity of the graph construction increases also significantly with the graph density; we know that deleting central nodes (such as the ones we chose) adds links from all their causes to their effects, and additionally connects all effects together, as noticed in Property 4.1.

In a first series of experiments, the algorithms were run with a conditional-independence oracle, which is equivalent to perfect conditional-independence tests. Conditioning on hidden variables was prohibited. These experiments are meant to compare the behavior of the algorithms by simulating the best case where the tests never fail.

In a second series of tests, multivariate Gaussian datasets (with 500 data points) were sampled from the networks, and data corresponding to the hidden variables was removed. The algorithms were run with Fisher's z-test on partial correlation as conditional-independence test. This was repeated 5 times for each learning problem.[16] For FCI, we used the authors' implementation in TETRAD (Scheines et al., 1995). MBCS* was implemented in Matlab.

4.4.2 Results and Discussion

Table 4.2 shows the results of the first series of experiments with the conditional-independence oracle. We report on the number of conditional-independence tests in the t column and the weighted number of tests in the wt column, as the number of tests alone does not reflect the quality of the algorithm. If τ_i is the size of the conditioning set of the ith test, then $wt = \sum_{i=1}^{t} \tau_i$. The third column r is the approximate ratio between the number of tests that MBCS* makes and the number made by FCI.

In the t column, we see that MBCS* makes up to 3 orders of magnitude fewer conditional-independence tests than FCI on the tested networks. As the **ALARM** network becomes denser with the hiding of certain variables, the difference between FCI and MBCS* becomes even more apparent both in the number of tests and in the weighted number of tests. The inverse phenomenon is observed for **HAILFINDER**, where the difference between FCI and MBCS* gets smaller: this is because this network is more densely connected, and both algorithms exhibit a behavior gradually evolving towards the worst case of the fully connected graph. FCI slowly "catches up" with the advantage that MBCS* has in these circumstances.

Figure 4.7 compares FCI and MBCS* with randomly generated datasets. The results essentially show that the difference of accuracy between FCI and MBCS* is not significant in either way. On each learning problem, the returned PAGs have been checked for correctness with respect to the

[16] We would have liked to both vary the number of samples for each dataset and include more test networks, but the run times of FCI on the larger instances, even when run with an upper limit on the maximum size of conditioning sets, were prohibitive, ranging up to a week for dense networks on a 2-GHz dualcore machine.

Alg.	ALARM				HAILFINDER			
	#hid. v.	t	wt	r	#hid. v.	t	wt	r
MBCS*	0	2,237	12,123	$\frac{1}{\mathbf{4}}$	0	5,333	35,841	$\frac{1}{\mathbf{423}}$
FCI		9,340	27,666			2,254,774	20,153,894	
MBCS*	1	3,397	18,113	$\frac{1}{\mathbf{6}}$	1	6,516	42,379	$\frac{1}{\mathbf{353}}$
FCI		21,497	95,497			2,302,707	20,448,775	
MBCS*	2	5,208	27,576	$\frac{1}{\mathbf{6}}$	2	7,205	46,291	$\frac{1}{\mathbf{322}}$
FCI		31,018	145,322			2,324,503	20,608,841	
MBCS*	3	7,527	42,133	$\frac{1}{\mathbf{30}}$	3	18,244	117,209	$\frac{1}{\mathbf{143}}$
FCI		231,096	1,612,106			2,622,312	22,888,622	

Table 4.2: Comparison of MBCS* and FCI where conditional-independence tests are done using a conditional-independence oracle. We report the number of tests t; the weighted number of tests wt, where each test contributes to wt a summand equal to the size of its conditioning set; and the ratio of t for FCI over the t for MBCS*: $r \approx t$ (MBCS*) $/t$ (FCI).

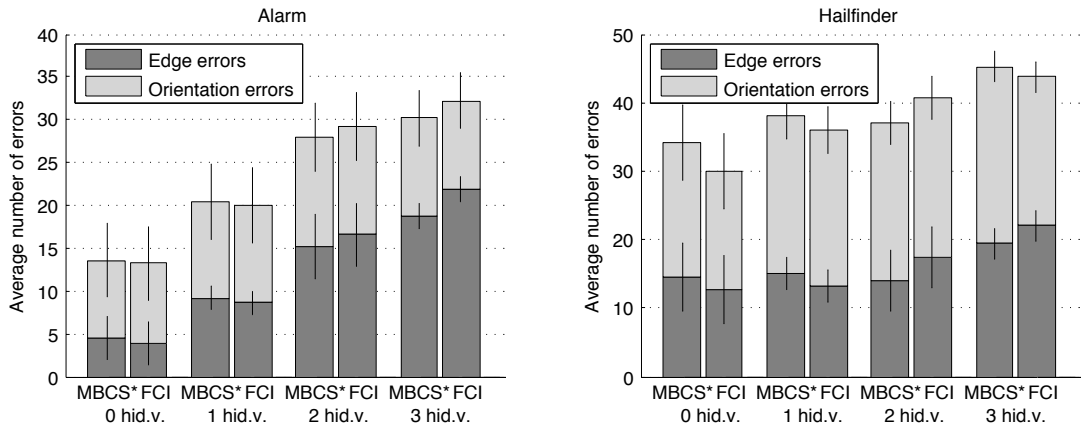


Figure 4.7: Comparison of MBCS* and FCI where conditional-independence tests are done using Fisher's z-test. We compare the number of edge errors (missing/extraneous) and orientation errors (including missing/extraneous hidden variables). Error bars show the standard deviation over the 5 runs.

maximally oriented PAG $\mathcal{G}_0$ theoretically obtainable (as returned by the first series of experiments). The discrepancies were classified either as edge errors (when an arc was missing or extraneous in the returned PAG with respect to $\mathcal{G}_0$), or orientation errors (when a predicted arc in the returned PAG was indeed present in $\mathcal{G}_0$, but had a reversed direction or different end points).

On all 8 learning problems, both edge and orientation errors of MBCS* and FCI are similar within the margin indicated by the standard-deviation error bars, except when 3 variables were hidden, where MBCS* seems to be slightly better than FCI in terms of the number of edge errors. As the rules applied to maximally orient the constructed PAG are the same for both algorithms, we deduce that the Markov-blanket-construction and V-structure-detection steps in MBCS* pass to the MAXIMALLYORIENTPAG() procedure a structure very similar to the one constructed by FCI.

On **ALARM**, we see that the orientation errors do not increase much as more variables are hidden; on **HAILFINDER**, they account for more than half of all structural errors, for both algorithms. We note that the overall relatively high number of errors comes from the failure of statistical tests with limited sample size. This indicates that structure learning in causally insufficient cases is a hard problem and that low-sample-size situations where tests typically fail must be investigated further.

4.4.3 Conclusion

In this chapter, we have been concerned with structure learning in causally insufficient cases. We have been able to generalize the generic approach presented in Chapter 3 so that it also works in these cases: interestingly, we have found that, here too, a Markov-blanket-based approach greatly simplifies the structure-learning task.

This has allowed us to propose a PAG-learning algorithm, MBCS*, which is much more efficient than the reference FCI algorithms on networks that are sufficiently sparse, making up to three orders of magnitude fewer conditional-independence tests to retrieve the same structure. MBCS* is based on a first phase that identifies the Markov blanket of the underlying MAG, and then makes local adjustments to remove the spurious links and identify all colliders; the last step involving orientation rules is the same as for FCI.

We have experimental evidence that the structural accuracy of MBCS* is as good as that of FCI. The reduced practical complexity makes MBCS* solve in minutes problems that FCI would need several days to solve. In that sense, MBCS* makes a whole new range of problems computationally tractable thanks to its local-neighborhood approach.

PAG-learning algorithms may be criticized for their relatively high structural-error rate, as shown in our experimental results. In particular, orientation errors are frequent. These errors include cases where missing variables fail to be detected (e.g., in the returned graph, we read $\rightarrow$ instead of $\bullet\!\!\rightarrow$, or $\bullet\!\!-\!\!\bullet$ instead of $\leftrightarrow$), or are wrongly detected (e.g., $\leftrightarrow$ instead of $\bullet\!\!\rightarrow$). These errors can be problematic, because the main advantage of structure-learning algorithms that do not assume causal sufficiency is precisely the presence of the edge types $\leftrightarrow$ and $\rightarrow$ in the returned PAG: they signal, respectively, the definitive presence or absence of a hidden cause behind two variables. The other edge types, $\bullet\!\!\rightarrow$ and $\bullet\!\!-\!\!\bullet$, do not assert the presence or absence of hidden causes.

An interesting topic for future research could then be to focus on the cases where direct causation can be established even without causal sufficiency, and where a hidden variable is known to exist.

In these two cases, obtaining more robust results than what a general algorithm like FCI or MBCS* returns would contribute to the global causal analysis by providing statistical facts that could, for instance, justify the use of the faster algorithms that do assume causal sufficiency, or explicitly assert that hidden confounders exist.

Another interesting research direction in causal analysis without causal sufficiency is to find criteria and algorithms that, starting from a PAG, help analysts to further specify the causal graph. In particular, the two edge types $\bullet\!\!\rightarrow$ and $\bullet\!\!-\!\!\bullet$ can be oriented more specifically. An algorithm could inspect a given PAG and suggest a manipulation whose outcome would allow to orient as many of these partially unoriented edges as possible, in an effort towards obtaining a fully specified causal model.

Chapter Summary

When we drop causal sufficiency, we face new underdetermination problems. With the formalism of MAGs and PAGs, it is possible to learn an independence-equivalence class of projections of the hidden structures. Causal inference rules change and must be adapted to include the possibility of hidden causes, whereas hidden effects that have no further causes can safely be ignored. Automatic structure learning can still be done with algorithms like FCI, a generalization of PC; local-structure learning coupled with local adjustments here again mean that it is possible to greatly increase algorithmic efficiency without any loss of accuracy.

CHAPTER

5

Causal Reasoning with Explanations

Constructing a causal network is only part of a more comprehensive causal analysis. Causal networks are good at extracting explanations about the observed state of a subset of variables. In this chapter, we explain the desiderata of an explanation and confront them with the concept of explanation proposed by existing methods. We also address the necessity of taking into account causal approaches, and we discuss several more traditional methods of obtaining explanations. We then introduce causal-explanation trees, based on the construction of explanation trees using the measure of causal-information flow. The causal-information flow itself makes use of the do-calculus and thus requires causal information. Finally, we compare our approach to several other methods on known networks.

5.1 Extracting Model Information with Explanations

In data analysis, *evidence explanation* refers to the task of obtaining more information about the state of certain observed variables in the form of an *explanation*, i.e., a set of variables and values that helps the analyst understand the observed state (Lacave & Dìez, 2002). The explanation for some observation should answer the question "why?" or "how come?" For example, if we see that the grass is wet, a relevant explanation could be that it has been raining; or if the price of a product happens to fall very low, that the demand for it was minimal.

In this section, we discuss evidence explanation and characterize more formally what makes an explanation relevant, in the sense of intuitively understandable by the analyst. Then, in Section 5.2, we list the standard approaches to explanation as well as some recent methods for making them more concise, and explain some of their drawbacks. We elaborate on what can be done to improve their relevance and how to deal with the problem of multiple explanations. We then introduce a causal criterion for selecting the explanatory variables and present causal-explanation trees in Section 5.3. Finally, we detail experiments and comparisons in Section 5.4.

5.1.1 A Relevant Explanation

To explain *why* a given system is observed in a given state, we must intuitively convey some information about the causal mechanisms that lead to the observation made. If we observe that it is raining and some explanation tells us that "it is raining because the grass is wet," we do not find this a relevant explanation as it contradicts our understanding of how the system works; that the grass being wet cannot make it rain.

We might regard the wet grass as a relevant explanation of why we *believe* that it has rained, however, in the sense that seeing that the grass is wet makes us give more credibility to the possibility of rain—something must have made the grass wet, after all. This kind of explanation is referred to by Lacave & Dìez (2002) not as evidence explanation but as explanation of the *reasoning process*: When we have observed some evidence, what is the process that makes us update our beliefs with respect to the other variables in the network? But determining that the wet grass "explains" our belief that it has rained is only possible because we identify the rain as a relevant explanation for the wet grass in the first place. In this sense, evidence explanation can be seen as more fundamental that explanation of the reasoning process.

Suppose we obtain an explanation, which we denote with $\mathbf{H} = \mathbf{h}$, for the evidence denoted by $\mathbf{E} = \mathbf{e}$: an intuitive interpretation of this result is that manually setting $\mathbf{H} = \mathbf{h}$ will be a favorable configuration to observe $\mathbf{E} = \mathbf{e}$ (or, at least, that observing the system in a configuration compatible with $\mathbf{H} = \mathbf{h}$ is more favorable to observe $\mathbf{E} = \mathbf{e}$ than doing nothing and just checking if $\mathbf{E} = \mathbf{e}$).

As Halpern & Pearl (2005) discuss, explanations need to be causal to be consistent with users' knowledge of the mechanisms of the system. It is therefore important that the explanations be given in a data-generating direction, such that users can infer interventional rules from the given explanations (for instance, "if we can make it rain somehow, then we know that the grass will be wet," as opposed to an impossible "let us make the grass wet so that it rains").

If explanations should be given in a causal way, they subject to the availability of causal information. In order to perform such analyses, we assume that we have causal model that can predict the effect of interventions on certain variables, as discussed in Subsection 2.3.4. Later, we use the formalism of causal Bayesian networks, but in general, such approaches are adaptable to any fully specified causal model.

As we describe later, explanation algorithms that were proposed in the past do not make use of causal information, but work with the joint probability distribution of the variables of interest. We show with examples that failing to regard causal relationships between the variables can easily lead to counterintuitive results that are easily misinterpreted.

5.1.2 Explanandum and Observation

We now propose a formalization of the context of evidence explanation.

Assuming a system with variables $\mathbf{V}$, we may observe the values of some variables: $\mathbf{O} = \mathbf{o}$. Even though several variables $\mathbf{O}$ may be observed, or in an extreme case every variable is observed, the explanation can be focused only on a specific subset $\mathbf{E} \subseteq \mathbf{O}$. The state $\mathbf{E} = \mathbf{e}$ is then called *explanandum*. The set of explanatory variables $\mathbf{H} \subsetneq \mathbf{V}$ can include both observed and unobserved

variables, and an explanation is then defined as an assignment $\mathbf{H} = \mathbf{h}$ (compatible with $\mathbf{O} = \mathbf{o}$ for variables both in $\mathbf{H}$ and in $\mathbf{O}$). The purpose of explanation algorithms is to find one or more explanations of the type $\mathbf{H} = \mathbf{h}$, given an explanandum $\mathbf{E} = \mathbf{e}$ and observed variables $\mathbf{O} = \mathbf{o}$.

These relationships are summarized in Figure 5.1 and by the following relations:

$$\mathbf{E} \subseteq \mathbf{O} \subseteq \mathbf{V} \tag{5.1}$$

$$\mathbf{E} \neq \mathbf{V} \tag{5.2}$$

$$\mathbf{H} \subseteq \mathbf{V} \setminus \mathbf{E} \tag{5.3}$$

$$\mathbf{H} \neq \emptyset. \tag{5.4}$$

What (5.1) and (5.2) say is that the explanandum variables must be observed, and can be the whole set of observed variables; and while all variables can be observed, the explanandum cannot be all variables, since there would be nothing to explain it with. Indeed, (5.3) and (5.4) say that the explanatory variables are a nonempty subset of the non-explanandum variables.

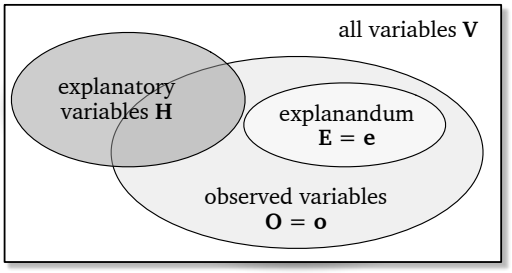


Figure 5.1: The sets of variables in an explanation context.

We insist on the distinction between the explanandum $\mathbf{e}$ and the observations $\mathbf{o}$ (Chajewska & Halpern, 1997). Observations constitute all our knowledge about the current state of a system, and this knowledge might not exactly coincide with what we want explained. Consider for example the case where we wish to know why the grass is wet while we know it has been raining. We do not seek an explanation for why it has rained, only for why the grass is wet. A perfectly valid explanation is that the grass is wet because it is raining if no other factors can sufficiently explain the facts.

An algorithm respecting this should then determine, for each variable in $\mathbf{O}$, whether its observed state is relevant to explain $\mathbf{e}$, and for each unobserved variable in $\mathbf{V} \setminus \mathbf{O}$, whether knowing its state adds "explanatory power" to the proposed explanation. If it does not do so and treats observations and explanandum equally, we have the following alternative: either (i) give it all our observations as input: then we do not lose information about the state of the network, but we are unable to specify exactly what should be explained; or (ii) discard the observations that we do not want to explain: then we do lose this additional knowledge that we have about the state of the network. Either choice is not satisfactory.

To summarize the desiderata exposed in this section, we wish our explanations to give us causal information by detailing the mechanisms that lead to the explanandum, using all the available information we have about the state of the network. Let us note that often, in more complex networks, there is not one definitive relevant explanations, but there could be several ones, equally justified: algorithms should hence not focus on returning a single explanation, but rather a set of relevant explanations.

5.2 TRADITIONAL APPROACHES

This section reviews some of the major techniques for finding explanations and and discusses how they live up to our desiderata about explanations.

5.2.1 Most Probable Explanation and Variants

A common non-causal measure of explanatory power is the conditional probability of the explanatory variables $\mathbf{H}$ given the explanandum $\mathbf{e}$. The *most probable explanation* (MPE) approach (Pearl, 1988) then considers $\mathbf{h}^\star = \arg\max_{\mathbf{h}} \mathrm{P}(\mathbf{h} \mid \mathbf{e})$ as the best explanation (or, alternatively, looks for the k best explanations by maximizing this probability). The explanandum $\mathbf{e}$ is in the case of MPE equal to the full set of observations $\mathbf{O} = \mathbf{o}$, and the set $\mathbf{H}$ is $\mathbf{V} \setminus \mathbf{E}$. This list can be long and uninformative because of lack of concision; moreover, it is hard to distinguish between long explanations, whose respective probabilities are low and close to one another anyway.

In the *partial abduction* approach (Shimony, 1991), the set of explanatory variables is a strict subset $\mathbf{H} \subsetneq \mathbf{V} \setminus \mathbf{E}$. The set of variables $\mathbf{X} = \mathbf{V} \setminus \mathbf{H} \setminus \mathbf{E}$ excluded from the explanation is then marginalized out before the maximum is computed: we look for $\arg\max_{\mathbf{h}} \sum_{\mathbf{x}} \mathrm{P}(\mathbf{h}, \mathbf{x} \mid \mathbf{e})$. This is the *maximum a posteriori* (MAP) model approach. The excluded variables $\mathbf{X}$ are selected either by a user, or via automated analysis of the network. Automatically selecting the relevant explanatory variable is a nontrivial issue (Shimony, 1991).

Partial abduction is computationally more expensive than standard MPE, because it cannot be readily solved by message-passing algorithms, but approximations exist (e.g., Park, 2002). On the other hand, it leads to more concise explanations than MPE.

Further efforts to make explanations more concise include de Campos et al. (2001), where the k most probable explanations are found and then simplified based on relevance and probabilistic criteria; and Henrion & Druzdzel (1990), where partial assignments are also allowed but only within a set of variable assignments. This is known as *scenario-based explanation*. The best explanation is the one with the highest posterior probability.

There are several concerns with these approaches—MPE/MAP or scenario-based—maximizing some conditional probability of the explanatory variables (Chajewska & Halpern, 1997). First, they do not distinguish the explanandum and the observations, such that the additional state information that is not meant to be explained is excluded from a possible explanation. Furthermore, there is no distinction between observing an explanatory variable X in a certain state x, and forcing it to have the value x. Thus, depending on the choice of the explanatory variables, the intuitive interpretation (as described in the previous section) stating that setting $\mathbf{H} = \mathbf{h}$ will be a favorable configuration for observing that $\mathbf{E} = \mathbf{e}$ does not hold.

MPE and, to a lesser extent, MAP model explanations, are not robust: little changes in the network will often change the result of the analysis, even though the changes occur in parts of the network largely independent of the explanandum (Chan & Darwiche, 2006).

Common to the methods in this subsection is that they order explanations by $\mathrm{P}(\mathbf{h}, \mathbf{e})$ (this is equivalent to $\mathrm{P}(\mathbf{h} \mid \mathbf{e})$ as $\mathrm{P}(\mathbf{e})$ is constant for a given $\mathbf{e}$): this joint probability alone cannot be considered to determine the explanatory power of $\mathbf{h}$ on $\mathbf{e}$. Some of these problems are illustrated by the experiments in Section 5.4.

5.2.2 SE Analysis

In *SE analysis*, Jensen (2001) additionally considers the sensitivity of an explanation $\mathbf{h}$ with respect to the explanandum. Less sensitive explanations ensure that small changes in the network's parameters will not lead to severely different explanations, so that the explanation is stable with respect to the specification of the network.

SE analysis also works by comparing two explanations $\mathbf{h}_i$ and $\mathbf{h}_j$, usually with *Bayes' factor* or the *likelihood ratio* (Jeffreys, 1961):

$$\text{Bayes' factor} = \frac{\text{posterior ratio}}{\text{prior ratio}} = \frac{\mathrm{P}(\mathbf{h}_i \mid \mathbf{e}) \,/\, \mathrm{P}(\mathbf{h}_j \mid \mathbf{e})}{\mathrm{P}(\mathbf{h}_i) \,/\, \mathrm{P}(\mathbf{h}_j)} = \frac{\mathrm{P}(\mathbf{e} \mid \mathbf{h}_i)}{\mathrm{P}(\mathbf{e} \mid \mathbf{h}_j)}.$$

An empirical interpretation of Bayes' factor given by Jeffreys (1961) is that if it is less than 1 it is in favor of $\mathbf{h}_j$, if less than 3 it is a slight support for $\mathbf{h}_i$; if it is between 3 and 12, it is a positive support; and higher than 12, it is a strong support for $\mathbf{h}_i$. In order to find the best explanations, Bayes' factors are computed for all legal assignments of the variables $\mathbf{H}$, and these are ranked accordingly.

In Yuan & Lu (2007), Bayes' factors are used to search for explanations consisting of only a few variables by ranking them by their Bayes' factor computed as the ratio between the probability of the explanation given the explanandum and its opposite:

$$\text{Bayes' factor} = \frac{\mathrm{P}(\mathbf{h} \mid \mathbf{e})}{1 - \mathrm{P}(\mathbf{h} \mid \mathbf{e})}.$$

An exhaustive search is performed over all subsets of the explanatory variables, and the best-ranking explanations are shown to be more concise in a sample network than MPE, Shimony's (1991) MAP and the simplifications described by de Campos et al. (2001).

In addition to the computationally expensive enumeration of all legal assignments, these methods also suffer from the same shortcomings and MPE and variants: additional observations other than the explanandum are discarded, and the causal directionality is ignored in the selection of the relevant explanatory variables.

5.2.3 Explanation Trees

The method proposed by Flores (2005) constructs a set of best explanations while at the same time giving a preference for concise explanations, summarizing the results of the analysis in an *explanation tree*. We describe this method in more detail, as the causal-explanation-tree method, described the next section, is based on a similar representation.

Definition 5.1 (Explanation tree) *An* explanation tree *for an explanandum* $\mathbf{E} = \mathbf{e}$ *is a tree in which every node* X *is an explanatory variable (with* $X \in \mathbf{V} \setminus \mathbf{E}$*), and each branch out of* X *is a specific instantiation* $x \in \mathcal{X}$ *of* X*. A path from the root to a leaf is then a series of assignments* $X = x$*,* $Y = y, \cdots, Z = z$*, summarized as* $\mathbf{P} = \mathbf{p}$*, which constitutes a full explanation.*

Figure 5.2 shows an example of an explanation tree that represents the following explanations: (i) $X = x$ and $Y = y$; (ii) $X = x$' and $Z = z$; and (iii) $X = x$' and $Z = z$'. The leaves usually carry labels reflecting a relevance score for the explanation. Depending on the semantics of the tree as

Algorithm 5.1 Flores's (2005) Explanation-tree algorithm

```
1: procedure T = ExplanationTree( H, e, p; α, β )
   Input:   H :  set of explanatory variables
          E = e :  explanandum
          P = p :  path of variable assignments
            α, β :  stopping criteria
   Output:  T :  an explanation tree
2:    X* ← arg max_{X∈H} Σ_{Y∈H} Inf( X; Y | e, p )
3:    if max_{Y∈H\X*} Inf( X; Y | e, p ) < α or P(p | e) < β then
4:       return ∅
5:    end if
6:    T ← new tree with root X*
7:    for each x ∈ X* do
8:       T' ← ExplanationTree( H \ X*, e, p ∪ {x} )
9:       add a branch x to T with subtree T' and
10:      assign it the label P(p, x | e)
11:   end for
12:   return T
13: end procedure
```

determined by the constructing algorithm, one may also read the partial paths $X = x$ and $X = x$' as two additional explanations; in that case, these subpaths may also carry labels.

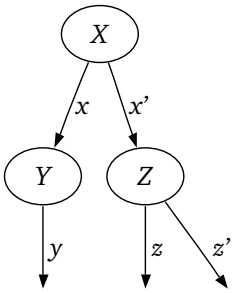


Figure 5.2: A sample explanation tree.

Flores's (2005) algorithm, summarized in Algorithm 5.1, builds such an explanation tree. Starting with an empty tree, the variable to use as the next node is selected to be the one that, given the explanandum, reduces the uncertainty the most in the rest of the explanatory variables according to some measure. The nodes that are on the path being grown are added to a conditioning set, so that the part of the explanatory space they already cover is taken into account. Two stopping criteria are used to determine when to stop growing the tree: the minimum posterior probability β of the current branch, and the minimum amount of uncertainty reduction α that must be achieved by adding a new variable. Among all explanations represented by the final tree, the best one is the one with the largest posterior probability $\mathrm{P}(\mathbf{p} \mid \mathbf{e})$.

The algorithm is parametrized with the measure of uncertainty reduction $\textsc{Inf}(\, X; Y \mid \mathbf{e}, \mathbf{p}\,)$.[17] For our implementation, we used the conditional mutual information, a generalization of the traditional information-theoretic concept of mutual information.

[17] See Flores (2005) for additional cases where max at line 3 is replaced by min or avg, and INF is the Gini index.

Definition 5.2 (Mutual information) *The* mutual information *between two random variables X and Y, denoted by $I(X;Y)$, is defined as:*

$$I(X;Y) = \sum_{x \in \mathcal{X}} \mathrm{P}(x) \sum_{y \in \mathcal{Y}} \mathrm{P}(y \mid x) \log \frac{\mathrm{P}(y \mid x)}{\mathrm{P}(y)}. \tag{5.5}$$

Mutual information is a symmetrical measure of how much one variable tells about another one. It has the property that $I(X;X) = 0 \iff (X \perp\!\!\!\perp Y)$. What we need here is mutual information given some observations represented by $\mathbf{e}$ and $\mathbf{p}$ in Algorithm 5.1; for that purpose, we use the conditional mutual information.

Definition 5.3 (Conditional mutual information) *The* conditional mutual information *between two random variables X and Y given a set of variables $\mathbf{Z}$ ($X, Y \notin \mathbf{Z}$), denoted by $I(X;Y \mid \mathbf{Z})$, is the expected value of the mutual information given values of $\mathbf{Z}$:*

$$\begin{aligned} I(X;Y \mid \mathbf{Z}) &= \mathbb{E}_{\mathbf{Z}}[I(X;Y) \mid \mathbf{Z}] \\ &= \sum_{\mathbf{z} \in \mathcal{Z}} \mathrm{P}(\mathbf{z}) \sum_{x \in \mathcal{X}} \mathrm{P}(x \mid \mathbf{z}) \sum_{y \in \mathcal{Y}} \mathrm{P}(y \mid x, \mathbf{z}) \log \frac{\mathrm{P}(y \mid x, \mathbf{z})}{\mathrm{P}(y \mid \mathbf{z})}. \end{aligned}$$

If the context $\mathbf{Z} = \mathbf{z}$ is given, we simply do not take the expected value over $\mathbf{Z}$, and what we get is

$$I(X;Y \mid \mathbf{z}) = \sum_{x \in \mathcal{X}} \mathrm{P}(x \mid \mathbf{z}) \sum_{y \in \mathcal{Y}} \mathrm{P}(y \mid x, \mathbf{z}) \log \frac{\mathrm{P}(y \mid x, \mathbf{z})}{\mathrm{P}(y \mid \mathbf{z})}. \tag{5.6}$$

Again, it is easy to show that there is equivalence between zero conditional mutual information and conditional independence: $I(X;Y \mid \mathbf{Z}) = 0 \iff (X \perp\!\!\!\perp Y \mid \mathbf{Z})$. If X fully determines Y, then knowing one is enough to know the other with certainty, and full information is shared.

Back to explanation trees. These trees are interesting in that they can present many mutually exclusive explanations in a compact form. Flores (2005) also argues that explanations as constructed by Algorithm 5.1 are reasonable and more sensible than (k-)MPE in the sense that on simple networks, the returned explanations are those that we expect. Four elements, however, are subject to discussion in this approach.

First, on line 2 of Algorithm 5.1, variables are added to the tree by order of how much information they provide about the remaining variables in the set of explanatory variables. But this does not measure the information of the added variables shared with the explanandum. Moreover, the explanandum actually grows as the tree is constructed, since there is no difference between the constructed path $\mathbf{p}$ and $\mathbf{e}$ at line 2: selecting an explanatory variable is like explaining this additional variable as well. Thus, this maximization cannot be interpreted as selecting variables that reduce the uncertainty in the explanandum.

Second, the algorithm makes no distinction between explanandum and observations. To try to fix this, we could either additionally condition on observations $\mathbf{O} = \mathbf{o}$, or marginalize out $\mathbf{O}$ altogether. The former case is no different from adding $\mathbf{o}$ to the explanandum $\mathbf{e}$, and the latter case excludes all $X \in \mathbf{O}$ from explanations, so both cases are unsatisfactory.

Third, the criterion for choosing the best explanation is the probability of the explanation path given the evidence $\mathrm{P}(\mathbf{p} \mid \mathbf{e})$, and not how likely the system is to have produced the evidence we are trying to explain with a configuration $\mathbf{p}$, $\mathrm{P}(\mathbf{e} \mid \mathbf{p})$. The two measures are linked, but since

several explanations can cover almost an equal share of the explanation space and often only one will be included in the explanation tree, the criterion $\mathrm{P}(\mathbf{p} \mid \mathbf{e})$ will miss explanations which could have explained the evidence well but do not cover as large a fraction of the explanation space.

Fourth, causal considerations are ignored: there is no distinction between ancestors and descendants of a variable, only allowing explanations of the type "it rains because the grass is wet."

From an end-user perspective though, trees are a good solution for representing several competing explanations compactly and in a readable way. We introduce in the next section a modified approach, which may more aptly address the issues discussed here.

5.3 CAUSAL-EXPLANATION TREES

To construct meaningful explanations, we wish to exploit the causal information that we have a causal model. To ensure that such a model can be built, we assume that the corresponding joint probability distribution fulfills the faithfulness condition and that the set of variables is causally sufficient (Pearl, 2000; Spirtes et al., 2001). As a reminder, faithfulness of the distribution ensures that there is a unique graph whose arcs depict all (conditional) dependencies of the distribution, and only those; causal sufficiency forbids hidden common causes for variables in $\mathbf{V}$, such that we can build a DAG whose arcs represent direct causation.

We will also represent our explanations with the help of trees, as we can compactly represent several alternatives. In order to construct our tree causally, we must forget about the symmetrical mutual-information criterion and look for a causal criterion instead.

5.3.1 Causal-Information Flow

Causal models allow us to evaluate the effect of interventions (Subsection 2.3.4): a general rule gives the joint distribution of variables after an intervention, known as postintervention distribution (Definition 2.18), and the *do*-calculus gives rules that can be used to evaluate conditional postintervention distributions more efficiently. If we wish to include causality as the central concept in an explanation, we need to have a measure of strength of causation between two variables. One natural way to do this is to see what kind of postintervention distribution a variable of interest Y has after manipulating another variable X, and in particular to quantify the change in the distribution of Y. This is what the *causal-information flow* does.

We first state the definition of unconditional causal-information flow.

Definition 5.4 (Unconditional causal-information flow) *The (unconditional)* causal-information flow *from X to Y, written $I(X \to Y)$, is:*

$$I(X \to Y) = \sum_{x \in \mathcal{X}} \mathrm{P}(x) \sum_{y \in \mathcal{Y}} \mathrm{P}(y \mid \hat{x}) \log \frac{\mathrm{P}(y \mid \hat{x})}{\mathrm{P}^{\star}(y)}, \tag{5.7}$$

where $\mathrm{P}^{\star}(y) = \sum_{x' \in \mathcal{X}} \mathrm{P}(x')\mathrm{P}(y \mid \hat{x}')$ *(Ay & Polani, 2006).*

We notice how (5.7) is similar to mutual information (5.5). The log term measures the divergence between the probability of $Y = y$ (noted by the special $\mathrm{P}^{\star}$ expression to avoid cases where it

can become zero) and its probability when X is intervened on so as to have value x (recall that $\mathrm{P}(y \mid \hat{x})$ is a shorthand for $\mathrm{P}(Y = y \mid do(X = x))$). The two sums take the expected value of that divergence. What we obtain is a measure of how setting X, on average, affects Y.

We now make this definition more general by providing a conditional version, as we did for conditional mutual information, taking into account that now, we can condition in two different ways: normal conditioning, meaning observation, and *do*-conditioning, meaning intervention. In both cases, we take the expected value over variables we condition on.

Definition 5.5 (Causal-information flow) *The (conditional)* causal-information flow *from X to Y when intervening on $\mathbf{Z}$ and observing $\mathbf{W}$, written $I(X \to Y \mid \hat{\mathbf{Z}}, \mathbf{W})$, is:*

$$I(X \to Y \mid \mathbf{W}, \hat{\mathbf{Z}}) = \sum_{\mathbf{z} \in \mathcal{Z}} \mathrm{P}(\mathbf{z}) \sum_{\mathbf{w} \in \mathcal{W}} \mathrm{P}(\mathbf{w} \mid \hat{\mathbf{z}}) \sum_{x \in \mathcal{X}} \mathrm{P}(x \mid \mathbf{w}, \hat{\mathbf{z}}) \sum_{y \in \mathcal{Y}} \mathrm{P}(y \mid \mathbf{w}, \hat{x}, \hat{\mathbf{z}}) \log \frac{\mathrm{P}(y \mid \mathbf{w}, \hat{x}, \hat{\mathbf{z}})}{\mathrm{P}^{\star}(y \mid \mathbf{w}, \hat{\mathbf{z}})}, \tag{5.8}$$

where $\mathrm{P}^{\star}(y \mid \mathbf{w}, \hat{\mathbf{z}}) = \sum_{x' \in \mathcal{X}} \mathrm{P}(x' \mid \mathbf{w}, \hat{\mathbf{z}}) \mathrm{P}(y \mid \mathbf{w}, \hat{x}', \hat{\mathbf{z}})$ *(Ay & Polani, 2006).*

If the contexts $\hat{\mathbf{z}}$ and $\mathbf{w}$ are known, then, just as for conditional mutual information, we do not need to take the expected value, and we can write:

$$I(X \to Y \mid \mathbf{w}, \hat{\mathbf{z}}) = \sum_{x \in \mathcal{X}} \mathrm{P}(x \mid \mathbf{w}, \hat{\mathbf{z}}) \sum_{y \in \mathcal{Y}} \mathrm{P}(y \mid \mathbf{w}, \hat{x}, \hat{\mathbf{z}}) \log \frac{\mathrm{P}(y \mid \mathbf{w}, \hat{x}, \hat{\mathbf{z}})}{\mathrm{P}^{\star}(y \mid \mathbf{w}, \hat{\mathbf{z}})}.$$

The expression $I(X \to Y \mid \mathbf{w}, \hat{\mathbf{z}})$ measures the amount of information flowing from X to Y if we intervene on $\mathbf{Z}$, setting it to $\mathbf{z}$ (i.e., if we block the causal flow on all paths going through $\mathbf{Z}$) while observing that $\mathbf{W} = \mathbf{w}$. This is the form of causal-information flow that we will be interested in.

For DAG-isomorphic probability distributions, $I(X \to Y \mid \mathbf{w}, \hat{\mathbf{z}}) = 0$ if and only if all directed paths (if any) from X to Y go through $\mathbf{Z}$ or $\mathbf{W}$ in the corresponding causal graph. For binary variables, if Y is a deterministic function of X regardless of $\mathbf{Z}$ and $\mathbf{W}$, then $I(X \to Y \mid \mathbf{w}, \hat{\mathbf{z}}) = 1$. In causal Bayesian networks, we have no causal feedback loops, so if $I(X \to Y) > 0$, then $I(Y \to X) = 0$. In essence, this is another way to assert that the corresponding graph has no cycles.

In the next subsection, we use causal-information flow to build what we call a causal-explanation tree.

5.3.2 Tree Construction

Like the non-causal-explanation trees, the causal-explanation-tree method takes advantage of a tree representation. The tree is grown so as to ensure that explanations in any path are causal: variables can be selected as explanatory only if they causally influence the explanandum, as measured by the causal-information flow from explanatory variables to the explanandum.

Our method is shown in Algorithm 5.2. At line 2, we use the *do*-conditioning on the already-build path $\mathbf{p}$, and we allow inputting additional observed variables $\mathbf{O}$ in an additional conditioning set. We replace Y from (5.8) with the state of the explanandum e. This means that we are not computing an average measure of how much an explanatory variable X causally contributes to

E, but that we are looking at the specific case $E = e$ instead. We suppress the corresponding summation and divide by the prior probability $\mathrm{P}(e \mid \mathbf{o}, \hat{\mathbf{p}})$, so that we end up computing:

$$I(X \to e \mid \mathbf{o}, \hat{\mathbf{p}}) = \sum_{x \in \mathcal{X}} \frac{\mathrm{P}(x \mid \mathbf{o}, \hat{\mathbf{p}})\mathrm{P}(e \mid \mathbf{o}, \hat{x}, \hat{\mathbf{p}})}{\mathrm{P}(e \mid \mathbf{o}, \hat{\mathbf{p}})} \log \frac{\mathrm{P}(e \mid \mathbf{o}, \hat{x}, \hat{\mathbf{p}})}{\sum_{x' \in \mathcal{X}} \mathrm{P}(x' \mid \mathbf{o}, \hat{\mathbf{p}})\mathrm{P}(e \mid \mathbf{o}, \hat{x}', \hat{\mathbf{p}})},$$

ensuring that the expected value $\mathbb{E}_E\left[I(X \to e \mid \mathbf{o}, \hat{\mathbf{p}})\right] = \sum_{e \in \mathcal{E}} \mathrm{P}(e \mid \mathbf{o}, \hat{\mathbf{p}}) I(X \to e \mid \mathbf{o}, \hat{\mathbf{p}})$ equals $I(X \to E \mid \mathbf{o}, \hat{\mathbf{p}})$.

Using this criterion, the explanation tree is then built as follows: the root node is selected as being $\arg\max_X I(X \to e \mid \mathbf{o})$; i.e., the node which has the maximum information flow to the state of the explanandum. The important thing is that we may condition on $\mathbf{o}$ without confusing observation and explanandum. Furthermore, we also allow selection of an observed variable $X \in \mathbf{O}$ in addition to unobserved variables, consistently with our desiderata. When X is observed ($X \in \mathbf{O}$), we know its value x. We must then compute the pointwise causal-information flow from x to e to determine the causal contribution of X in this circumstance. This is, with $\mathbf{o}'$ being $\mathbf{o}$ without the observation $X = x$:

$$I(x \to e \mid \mathbf{o}', \hat{\mathbf{p}}) = \log \frac{\mathrm{P}(e \mid \mathbf{o}', \hat{\mathbf{p}}, \hat{x})}{\sum_{x' \in \mathcal{X}} \mathrm{P}(x' \mid \mathbf{o}', \hat{\mathbf{p}})\mathrm{P}(e \mid \mathbf{o}', \hat{x}', \hat{\mathbf{p}})}.$$

Once a variable $X^\star$ has been selected, outgoing branches are added, one for each possible value x of $X^\star$ (or only one if $X \in \mathbf{O}$ is observed, as we then know which value X has). The tree is then grown recursively: for each new leaf, the next explanatory variable is selected as being $\arg\max_Y I(Y \to e \mid \mathbf{o}, \hat{x})$, and so on, where the *do*-conditioning set always reflects the values of the variables selected from the root to the current leaf.

We use only one stopping criterion, the minimum causal-information flow α we accept as a causal information contribution. The algorithm furthermore explicitly allows us to restrict the search set for explanatory variables $\mathbf{H}$ (defaulting to $\mathbf{V} \setminus \{E\}$). Finally, each leaf is labeled with a score indicating how relevant the current explanation is. As for the score, we use the causal-information flow again:

$$I(\mathbf{p} \to e \mid \mathbf{o}). \tag{5.9}$$

This measures, given the observational context $\mathbf{o}$, the pointwise information flow from the whole path to the explanandum e. In (5.9), the whole path is denoted by $\mathbf{p}$, but at line 8 of Algorithm 5.2, it is denoted by $\mathbf{p}, x$ to take into account the last branch being built. We have to make sure that variables selected in $\hat{\mathbf{p}}$ are removed from $\mathbf{o}$ if needed: this happens if explanatory variables are selected among the observed variables.

This score thus measures how much performing the interventions represented by the proposed explanation changes the probability of the explanandum (given the observations) with respect to the prior probability of the explanandum. Higher values indicate better explanations; negative values indicate that the probability of the explanandum actually decreases with the proposed explanation.

In order to improve the explanation tree, we can get rid of the subtrees that only contribute bad explanations (i.e., with a score lower than zero). This is done at line 10. We cannot exclude these branches straight off because we may select at line 2 a variable X that has a very unfavorable

Algorithm 5.2 Causal-explanation-tree algorithm

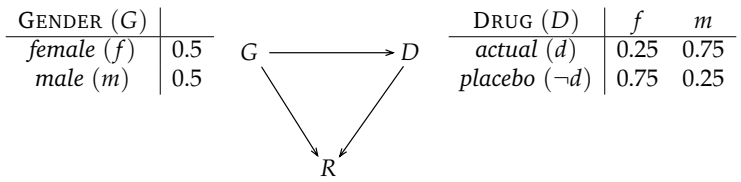
```
1: procedure T = CausalExplanationTree( H, o, e, p̂; α )
   Input:   H :  set of explanatory variables
          O = o :  observation set
          E = e :  explanandum
              p̂ :  path of interventions
              α :  stopping criterion
   Output:  T :  a causal-explanation tree
2:    X* ← arg max_{X∈H} I(X → e | o, p̂)
3:    if I(X* → e | o, p̂) < α then
4:       return ∅
5:    end if
6:    T ← new tree with root X*
7:    for each x ∈ X* do
8:       c ← I(p, x → e | o)
9:       T' ← CausalExplanationTree( H \ {X*}, o, e, p̂ ∪ {x̂} )
10:      if c > 0 or some leaf of T' has a contribution c' > 0 then
11:         add a branch x to T with subtree T' and
12:         assign it the contribution c
13:      end if
14:   end for
15:   return T
16: end procedure
```

state $\tilde{x}$ although X is a good variable on average. Moreover, even if $X = \tilde{x}$ contributes badly to the explanation, it may be possible further down the tree to find a variable that makes the whole explanation score positive again. We show such examples in the next section.

Using the causal-information-flow criterion brings us two advantages over standard (conditional) mutual information: first, we automatically only consider variables that can causally influence the explanandum. Second, when selecting the ith variable on a tree branch, we take into account the previously selected variables 1 through $i-1$ causally, as they enter the conditioning set of variables that have been intervened on. They are thus mixed up neither with additional observations $\mathbf{o}$, nor with the explanandum e.

In practice, computing a causal-information flow of the type $I(X \to e \,|\, \mathbf{o}, \hat{\mathbf{p}})$ at line 2 of Algorithm 5.2 requires the distributions $\mathrm{P}(X \,|\, \mathbf{o}, \hat{\mathbf{p}})$, $\mathrm{P}(E \,|\, \mathbf{o}, \hat{\mathbf{p}})$, and $\mathrm{P}(E \,|\, \mathbf{o}, \hat{X}, \hat{\mathbf{p}})$ (i.e., $\mathrm{P}(E \,|\, \mathbf{o}, \hat{x}, \hat{\mathbf{p}})$ for all $x \in \mathcal{X}$). Additionally, $\mathrm{P}(e \,|\, \mathbf{o})$ is needed to label the leaves, but as it is only dependent on e and $\mathbf{o}$, we compute it only once. We can further avoid unnecessary computations by using the graphical-reachability criterion from a candidate node X to E, blocking paths going through $\mathbf{O} \cup \mathbf{P}$. The inference steps were implemented using the factor graph message-passing algorithm (Frey et al., 2001).

The complexity of this algorithm, in terms of number of calls to an inference engine per node in the constructed tree, is $\mathcal{O}(dc)$, where d is the number of explanatory variables $|\mathbf{H}|$ and c is the average domain size of the variables, e.g., 2 for binary variables. For comparison, Flores's (2005) approach is $\mathcal{O}(d^2c^2)$.



GENDER (G)	
female (f)	0.5
male (m)	0.5

DRUG (D)	f	m
actual (d)	0.25	0.75
placebo ($\neg d$)	0.75	0.25

RECOVERY (R)	f,d	$f,\neg d$	m,d	$m,\neg d$
recovery (r)	0.2	0.3	0.6	0.7
death ($\neg r$)	0.8	0.7	0.4	0.3

Figure 5.3: The **DRUG** network.

5.4 BENCHMARKS AND EXPERIMENTAL EVALUATION

We cannot formally prove that our method provides "better explanations." We saw that the definition of explanation was very much linked to an explanation-generating algorithm, and thus, algorithms can be arbitrarily "good" at generating their own flavor of explanations. What we try to do in this section, however, if to pose one query pertaining to some network and see what explanations various algorithms return. In this sense, we argue that causal-explanation trees provide more intuitive and more interpretable explanations.

5.4.1 Experimental Setup

For a series of networks, we asked selected algorithms to output an explanation or a series of explanations for a given explanandum. We compared causal-explanation trees (CET) with parameter $\alpha = 0$ to Most Probable Explanation (MPE), Bayes' factor (BF) following Yuan & Lu (2007), and standard (non-causal) explanation trees (ET) with parameters $\alpha = 0.02$ and $\beta = 0$, which returned trees with a reasonable depth.

The three networks we used for our comparative evaluation are small enough that we can still relatively easily interpret them "manually" and judge the quality of the explanations empirically, while being interesting enough to be easily misinterpreted.

The **DRUG** network, shown in Figure 5.3, comes from Chapter 6 of Pearl (2000) and was already discussed as motivating example in Chapter 2. It represents the outcome of an experiment designed to check the efficiency of a new drug on male and female patients. The males have a natural recovery rate of 70%; taking the drug decreases it to 60%. Similarly, 30% of females recover naturally, but only 20% when given the drug. Thus, both the absence of the drug and being a male can explain a good recovery rate.

The **ACADEME** network, shown in Figure 5.4, depicts the relationships among various marks given to students following a course. The FINAL MARK is determined by some OTHER outside factors and an intermediate mark (TP MARK), which is in turn determined by the student's abilities in THEORY and PRACTICE as well as EXTRA (extra-curricular activities) in this tested subject.

Finally, the **ASIA** network (Lauritzen & Spiegelhalter, 1988), shown in Figure 5.5, models the relationships between two indicators, X-RAY results and DYSPNEA, of severe diseases for a person. TUBERCULOSIS (more likely if a VISIT TO ASIA has occurred) and LUNG CANCER (more likely when the person is a SMOKER) both increase abnormal X-ray results and dyspnea; BRONCHITIS also

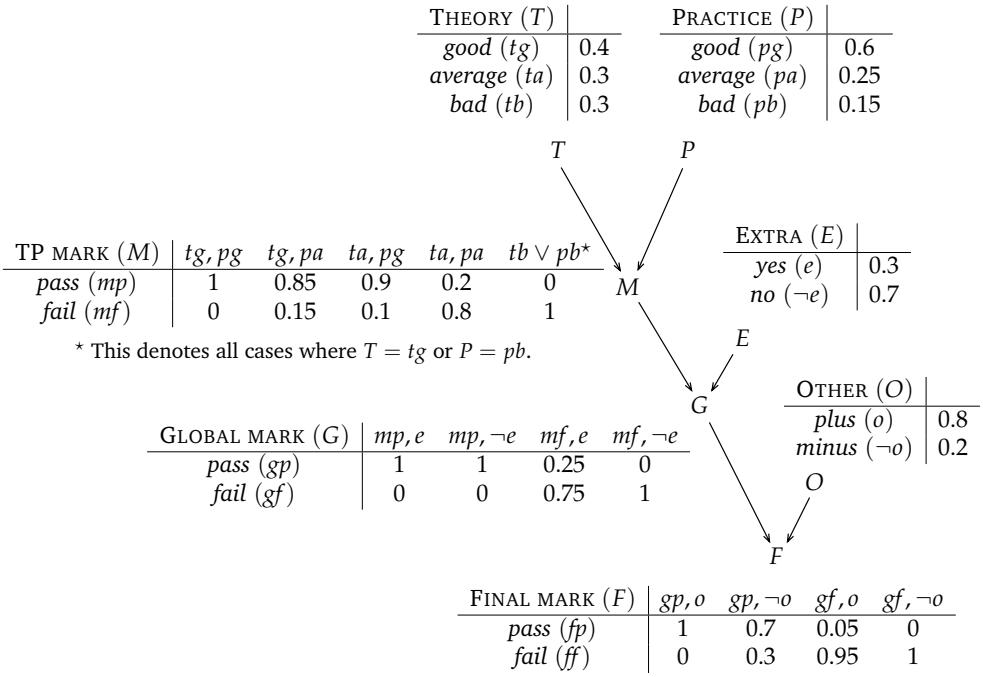


Figure 5.4: The **ACADEME** network.

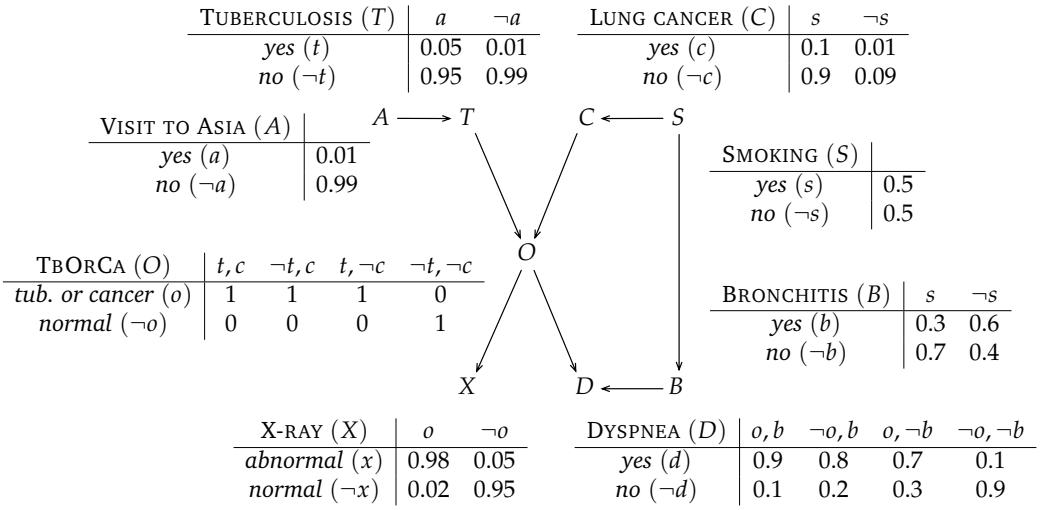
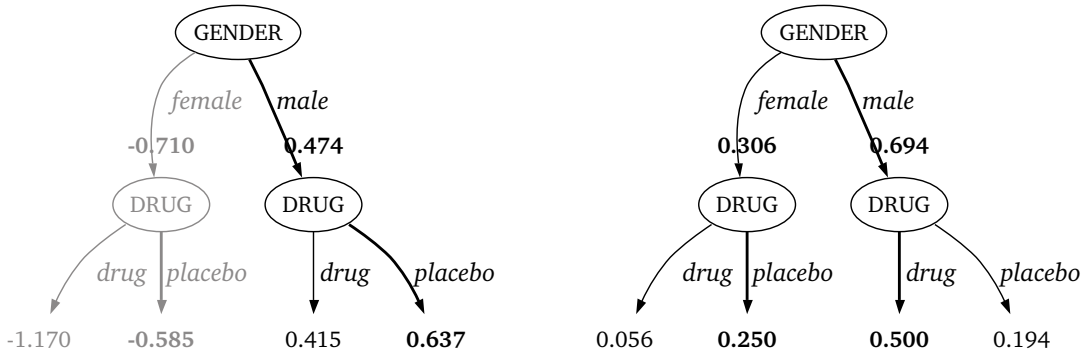


Figure 5.5: The **ASIA** network.



(*a*) Causal-explanation tree (*b*) Explanation tree

(*c*) MPE: $\mathrm{P}(\textsc{Gender} = \textit{male}, \textsc{Drug} = \textit{drug} \mid \textsc{Recovery} = \textit{recovery}) = 0.5$

(*d*) BF: $\mathrm{BF}(\textsc{Gender} = \textit{male}) = 2.27$
$\mathrm{BF}(\textsc{Gender} = \textit{male}, \textsc{Drug} = \textit{placebo}) = 1.68$
$\mathrm{BF}(\textsc{Drug} = \textit{drug}) = 1.25$

Figure 5.6: **DRUG**: explain RECOVERY = *recovery*.

causes increased dyspnea. TBORCA is a modeling artifact, excluded from the analysis in the two tree algorithms: it is defined as the disjunction of TUBERCULOSIS and LUNG CANCER, indicating they both have an identical and indistinguishable effect on X-RAY and DYSPNEA.

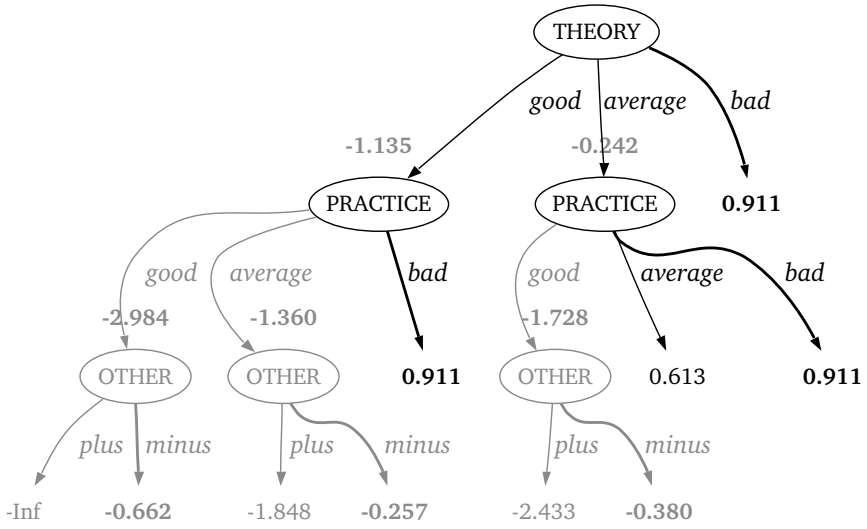
5.4.2 Results and Discussion

We now describe the explanation queries we handed out to the various algorithms and comment on the results they returned.

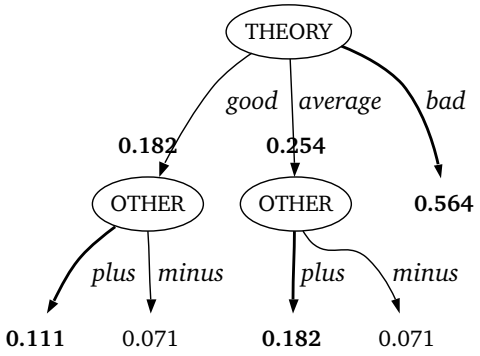
DRUG In Figure 5.6, we try to explain a recovery. This query is especially interesting because we remember from Section 2.1 that it was problematic to determine whether the drug has a beneficial effect or not. The following discussion had led us to think that although the drug seemed to have a good effect on the combined male and female tables (Table 2.1), it actually made both males and females recover less often (Table 2.2). We thus expect an explanation where $\textsc{Drug} = \textit{drug}$ is identified as detrimental and $\textsc{Gender} = \textit{male}$ as beneficial to the recovery.

All approaches correctly indicate that $\textsc{Gender} = \textit{male}$ largely accounts for the recovery. However, ET selects $\textsc{Gender} = \textit{male} \wedge \textsc{Drug} = \textit{drug}$ as the best explanation according to the leaves' labels—just like MPE. (We recall that the label assigned to a path $\mathbf{p}$ by ET is $\mathrm{P}(\mathbf{p} \mid e)$, where e is the explanandum; it can be expected that the best explanation according to ET will be the same as the one returned by MPE, provided all variables are eventually selected by ET.) But this contradicts the natural idea of explanation, since the drug has a negative impact on the recovery! The most favorable situation for recovery, $\textsc{Gender} = \textit{male} \wedge \textsc{Drug} = \textit{placebo}$, gets an even lower score than another unfavorable situation, $\textsc{Gender} = \textit{female} \wedge \textsc{Drug} = \textit{placebo}$.

CET labels the leaves more sensibly: branches where the drug was not given have a higher rank. Moreover, the branches where $\textsc{Gender} = \textit{female}$ have a negative label, indicating that they actually *decrease* the probability of recovery. Notice that in our rendition of the tree, we grayed out the



(*a*) Causal-explanation tree



(*b*) Explanation tree

(*c*) MPE: $\mathrm{P}(\textsc{Theory} = \textit{bad}, \textsc{TP mark} = \textit{fail}, \textsc{Global mark} = \textit{fail}, \textsc{Extra} = \textit{no}, \textsc{Other} = \textit{positive}, \textsc{Practice} = \textit{good} \mid \textsc{Final mark} = \textit{fail}) = 0.208$

(*d*) BF: $\mathrm{BF}(\textsc{Theory} = \textit{bad}) = 3.02$
$\mathrm{BF}(\textsc{Theory} = \textit{bad}, \textsc{Extra} = \textit{no}) = 2.78$
$\mathrm{BF}(\textsc{Theory} = \textit{bad}, \textsc{Other} = \textit{negative}) = 2.53$

Figure 5.7: **Academe**: explain Final mark = *fail*.

branches and subtrees where explanations had a negative label, thus ignoring the condition at line 10 of Algorithm 5.2 when growing the tree. Branches with negative labels, indeed, are not completely useless. Here, for instance, the whole *female* subtree has negative labels; however, we can still learn by looking at this subtree that a female patient who was given the drug is less likely to recover than one who was given the placebo. The CET algorithm continues to grow the subtree on the *female* branch because it determines that, as for males, knowing the state of DRUG can improve the explanation at this stage: even if GENDER $=$ *female* $\wedge$ DRUG $=$ *placebo* is a bad explanation (label -0.585), it is still better than merely GENDER $=$ *female* (label -0.71). In principle, it could be that further down the tree, the explanation becomes good again.

As for BF: although the first two returned explanations are sensible, the third one mistakenly selects DRUG $=$ *drug* as a good explanation.

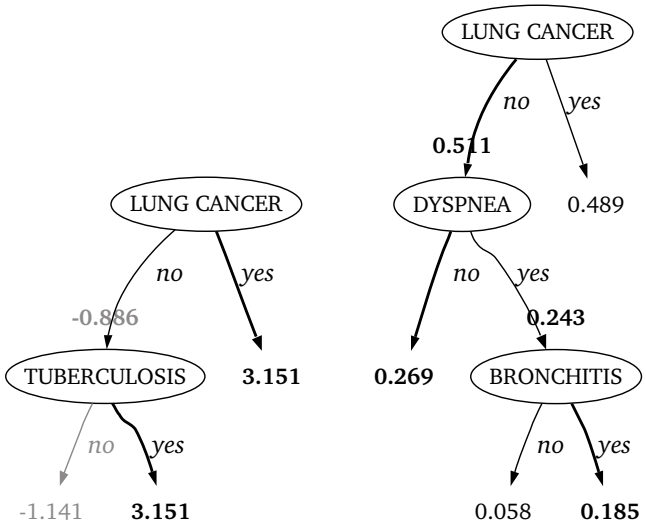
ACADEME In Figure 5.7, the explanandum was set to FINAL MARK $=$ *fail*; i.e., we wish to explain why a student failed the course. TP MARK and GLOBAL MARK have been excluded from the possible explanatory variables in the two tree algorithms as they are modeling artifacts. On this example, we ran CET with $\alpha = 0.1$ in order to keep the tree from getting too deep.

ET gives the highest score to the explanation THEORY $=$ *bad* (0.564). We could have expected PRACTICE to also be part of alternative explanations, as it influences the final mark very similarly to THEORY. This is what CET does, including PRACTICE to explain the final failure when THEORY is average or good. When THEORY is bad already, then this is a good enough explanation for overall failure; in other cases, then bad PRACTICE is an equally good explanation (all with label 0.911). This is enough to have TP MARK be *fail* deterministically. EXTRA and OTHER would be selected here too if we had run the algorithm with $\alpha = 0$, as even with a failed TP MARK there is a slight chance of not failing altogether, but here the contribution was considered too small to be worth noticing—it would have made the graph too large to display conveniently. The gray subtrees containing OTHER represent bad explanations, and thus have negative labels; they were grown by CET for the same reason that made it grow the *female* branch in Figure 5.6 (a): although the end explanation is bad, it is still a bit better than stopping before reaching it. For instance, OTHER $=$ *minus* improves both partial explanations THEORY $=$ *good* $\wedge$ PRACTICE $=$ *good* and THEORY $=$ *good* $\wedge$ PRACTICE $=$ *average*.

MPE includes PRACTICE $=$ *good* in its long list of states, which, intuitively, does not seem likely as a good explanation for overall failure. The only reason why this state is chosen is that it has a higher prior probability than any other, which conditioning on FINAL MARK $=$ *fail* is not enough to make vanish. BF provides more concise explanations, but, like ET, ignores PRACTICE altogether, although a bad practice can account for failure similarly to a bad theory.

In this network, the causal structure is rather obvious—the parametrization of the causal Bayesian network is what makes it interesting. Unfortunately, this is also what makes it hard to read. It is hard to know right away the contributions of the various variables on the final mark without carefully looking at the conditional probability distributions. In that sense, a good explanation can be seen as the graphical representation of a series of smartly chosen probability inference and marginalization steps that make the whole network more easily interpretable.

ASIA In Figure 5.8, we wish to explain abnormal X-ray results. Whereas both tree algorithms select LUNG CANCER as the best explanation, they differ on how to explain when it is absent:



(a) Causal-explanation tree (b) Explanation tree

(c) MPE: $\mathrm{P}(\textsc{Bronchitis} = \textit{yes}, \textsc{Dyspnea} = \textit{yes}, \textsc{Lung cancer} = \textit{yes}, \textsc{TbOrCa} = \textit{yes}, \textsc{Smoker} = \textit{yes}, \textsc{Tuberculosis} = \textit{yes}, \textsc{Visit to Asia} = \textit{no} \,|\, \textsc{X-ray} = \textit{abnormal}) = 0.24$

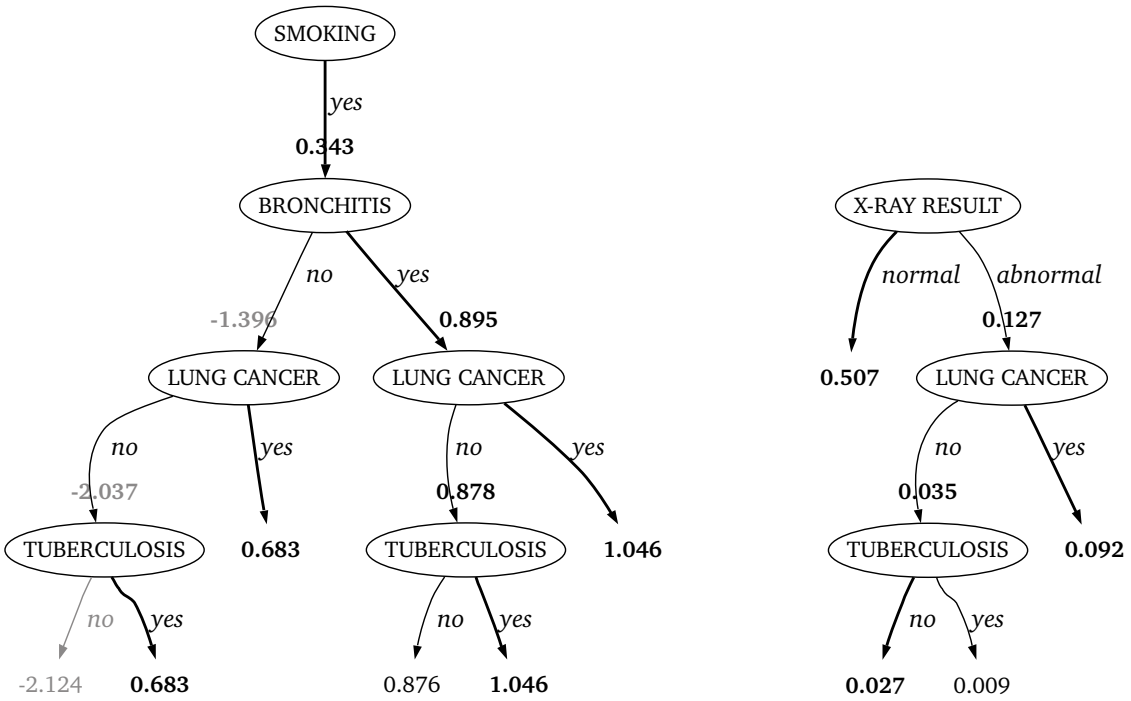
(d) BF: $\mathrm{BF}(\textsc{TbOrCa} = \textit{yes}) = 19.60$
$\mathrm{BF}(\textsc{TbOrCa} = \textit{yes}, \textsc{Visit to Asia} = \textit{no}) = 19.21$
$\mathrm{BF}(\textsc{TbOrCa} = \textit{yes}, \textsc{Lung cancer} = \textit{yes}) = 16.42$

Figure 5.8: **Asia**: explain X-ray $=$ *abnormal*.

CET selects, justifiably, Tuberculosis, but ET uses Dyspnea and then Bronchitis, which are not causes of X-ray and cannot explain it, especially not when we know that no lung cancer is present. We can see here that the CET explanation Lung cancer $=$ *no* $\wedge$ Tuberculosis $=$ *yes* is good, although the first branch Lung cancer $=$ *no* had a negative label. We can also see that the way CET reports that Lung cancer $=$ *yes* and Tuberculosis $=$ *yes* have exactly the same effect on X-ray (because of the node TbOrCa) is not evident and is hidden in that several labels get the exact same value (here, 3.151). In these cases, which variable is selected first depends on the order in which they are searched at line 2 of Algorithm 5.2. Inverting them in the explanation tree would have been perfectly valid as well, since they both have the same causal-information flow to the explanandum.

We also notice that ET gives a higher label to Lung cancer $=$ *no* than to Lung cancer $=$ *yes*. In the context of the explanandum X-ray $=$ *abnormal*, this is certainly counterintuitive, but is due to the fact that not having lung cancer has a much higher prior probability than having lung cancer. ET cannot ignore this, whereas CET, using *do*-conditioning on the proposed explanation, effectively ignores these prior probabilities. This is the right thing to do: we should not care how improbable it would be to observe the system in a state compatible with a given explanation, as long as the proposed explanation makes the explanandum much more likely to happen.

MPE surprisingly excludes a visit to Asia, when this is expected to make abnormal X-rays more likely through Tuberculosis. BF, while still providing very concise explanations, is stuck on the middle node TbOrCa, which we cannot exclude from the considered variables right away, and, like ET, ignore the important Tuberculosis.



(*a*) Causal-explanation tree (*b*) Explanation tree

(*c*) MPE: $\mathrm{P}(\textsc{Bronchitis} = \mathit{yes}, \textsc{X-ray} = \mathit{normal}, \textsc{Lung cancer} = \mathit{no}, \textsc{TbOrCa} = \mathit{no}, \textsc{Smoker} = \mathit{yes}, \textsc{Tuberculosis} = \mathit{no}, \textsc{Visit to Asia} = \mathit{no} \mid \textsc{Dyspnea} = \mathit{yes}) = 0.46$

(*d*) BF: $\mathrm{BF}(\textsc{Bronchitis} = \mathit{yes}) = 6.14$
$\mathrm{BF}(\textsc{Bronchitis} = \mathit{yes}, \textsc{Visit to Asia} = \mathit{no}) = 5.89$
$\mathrm{BF}(\textsc{Bronchitis} = \mathit{yes}, \textsc{Tuberculosis} = \mathit{no}) = 5.84$

Figure 5.9: Asia: explain $\textsc{Dyspnea} = \mathit{yes} \mid \textsc{Smoking} = \mathit{yes}$.

In Figure 5.9, we try a slightly more complex query. We wish to explain the presence of dyspnea for a smoker. In practice, this means that we have two observed variables in our system: SMOKING $=$ *yes* and DYSPNEA $=$ *yes*, but we only wish to explain the latter (explaining SMOKING would be irrelevant anyway, since there are no causes for it in our network). This is an important practical example, because knowing that the patient smokes is important information that we should not discard, but which should still be available as part of the explanation, if deemed potentially relevant.

It turns out that only CET is still able to select SMOKING as explanatory variable: ET can only add this observation to the explanandum and thus cannot select it. Instead, the best explanation according to ET is normal X-rays, which does not seem very likely. Although ET does select the important LUNG CANCER and TUBERCULOSIS, it ignores the largest factor according to BF and CET, namely BRONCHITIS. After the branch SMOKING $=$ *yes*, CET exhausts all relevant causes for DYSPNEA; here too, we can see that explanations where LUNG CANCER $=$ *yes* and TUBERCULOSIS $=$ *yes* get the same label in common subtrees, indirectly signaling their indistinguishable effect.

5.4.3 Conclusion

It seems difficult to discuss explanations without discussing causality. When we ask for an explanation, we expect to learn about the causes of a given event, not about what may be correlated with this event. We think that explaining a given state with variables that can causally influence it is what makes the explanations returned by our causal-explanation-tree method more intuitive, interpretable, and useful than others.

The selection of causal variables cannot simply be solved by pre-selecting the set of variables that a non-causal algorithm like MPE, BF or ET considers. The selection criterion itself has to be causal as well as immune to problems like the misinterpretation of the effect of the drug in our first example. Postintervention distributions and the *do*-calculus are ideal tools for getting this causal criterion right, as this allows us to make a difference between observations and interventions on the system, treat observations differently from explanandum, and use *do*-conditioning for the proposed explanations to quantify their causal effects and ignore their prior probability distributions. In turn, this allows a precise labeling of explanations according to how much more probable the explanandum becomes given an explanation, which we think is superior to the traditional measure involving the conditional probability of the explanation given the explanandum used in MPE, BF, and ET.

Inevitably, techniques like causal-explanation trees cannot extract more information from the causal Bayesian network than what is already in the Bayesian network. In particular, CET makes no further assumptions, and treats the network as the summary of all assumptions that have been made about the system to analyze. In particular, it is up to the network to faithfully represent the underlying joint probability distribution and exhaustive set of causal relationships among the variables under study. The only practical assumption made by our algorithm is that all variables are discrete, which helps in calculating the causal-information flow; we have not ventured into trying to compute conditional probability distributions of the type $\mathrm{P}(Y \mid do(\mathbf{X}), \mathbf{Z})$ of arbitrary shapes.

In the future, it would be worth to investigate the extent to which partially specified input networks can still be reliably used as input for causal-explanation inference systems like CET. Indeed, the evaluation of a postintervention probability $\mathrm{P}(y \mid \hat{x})$ rarely requires knowledge of the full causal

structure. Although the general formula (2.28) for the full postintervention distribution requires knowing the graphical parents for each variable in $\mathbf{V} \setminus \{X\}$, the rules of the *do*-calculus operate given *d*-separation in manipulated graphs (see Definition 2.19 on page 26) and may not require complete knowledge of the graph.

Automatic causal structure-learning algorithms usually return partially directed structure, as we know from Chapters 3 and 4: it would be interesting to know in which circumstances causal explanations can be produced automatically given a dataset instead of a full causal model, and try to build the necessary parts of the causal model on the fly as information is requested by the explanation algorithm.

Chapter Summary

In this chapter, we have moved to the next step after causal network construction and specification, and examined explanations as an application of reasoning and inference that can be done given a causal model. In particular, we have presented causal-explanation trees, where explanations are presented as a tree, compactly representing several explanations graphically. Unlike other techniques, we do not condition on the explanandum to maximize the probability of the explanatory variables $\mathrm{P}(\mathbf{h} \mid \mathbf{e})$*, but focus on* $\mathrm{P}(\mathbf{e} \mid do(\mathbf{H} = \mathbf{h}))$ *instead, by means of the causal-information flow. We thus only select causes of the explanandum as explanations. The algorithm makes an explicit distinction between observation and explanandum, and labels the leaves so as to reflect how a proposed explanation changes the probability of the explanandum, making the tree intuitive and easy to interpret.*

CHAPTER

6

Applications

This chapter is devoted to two real-world applications of causal analysis involving techniques or concepts developed in the previous chapter. Real causal analysis, still in its infancy, has many shortcomings owing to stringent assumptions like causal sufficiency or the problem of observational equivalence. In these two applications, we describe how we built the causal models, the challenges we faced in the particular circumstances, and the kind of causal reasoning we did. Section 6.1 examines the assessment of quality risk in pharmaceutical manufacturing, and Section 6.2 explores how to find a causal-relevance criterion for admitting international students into a master's program.

6.1 Risk Assessment in Pharmaceutical Manufacturing

In this section, we present a statistical approach based on causal Bayesian network construction to quantitatively measure a company's current exposure to failures and defects in product quality or compliance to government regulations. Specifically, we use the causal model in two ways. First, as an *explanatory approach*, we wish to causally explain the values of variables with causal-explanation trees. Then, as an *exploratory approach*, we aim to evaluate "what-if" scenarios (for instance, with alternative methods or policies) by using the *do*-calculus to assess the effect of interventions on the structure of the model.

Building the causal structure in the context of pharmaceutical manufacturing raises some challenges: in particular, there is no automated way to collect the needed data. This application thus does not make use of the automatic structure-learning algorithms developed in Chapters 3 and 4. Instead, we design and apply a methodology for model selection and probability elicitation based on expert knowledge. The causal model allows analysts to more rigorously compare practices across different manufacturing sites, creates the opportunity for risk remediation, and lets us evaluate alternative methods and approaches.

6.1.1 Introduction and Context

In the context of this application, we are concerned with the development of a risk-assessment model for quantifying the risk hazard in a setting where risk involves various challenges related to government compliance and product quality and availability. One of the initial motivations for developing such models was the U.S. Food and Drug Administration's (FDA) new initiative, Pharmaceutical Current Good Manufacturing Practices for the 21st century (U.S. Food and Drug Administration, 2002), which launched a series of projects to provide "the most effective public health protection." One key aspect in this initiative is the use of quality and risk-management techniques in order to better identify the highest risk elements in a drug-manufacturing process (U.S. Food and Drug Administration, 2004). This calls for the development of risk-management tools to manage manufacturing process risks not only for the FDA but also for the pharmaceutical companies which have to make sure that their plants do not have a high risk of producing defective drugs or a high risk of not being compliant to the FDA. As we were unable to use the traditional structure-learning algorithms, we have devised a general methodology to build risk-assessment models based on causal networks. Our approach can also be employed in other contexts characterized by scarce or unreliable data, considerable qualitative information, and expert knowledge.

Risk quantification is a key function in many industries and has been performed using a plethora of different tools ranging from statistical methods that offer no insight or justification of their output, to management techniques and expert brainstorming. For this application, we adopt an approach whose central concept is a causal model, a causal Bayesian network modeling the different sources of risk and their inter-relationships. We build the structure of the causal graph using principles defined in the mid 1980s (Pearl, 1986) using expert-driven modeling. Bayesian networks have been used in different areas where overall risk assessment is as important as finding the most probable or the most risky variable, but causal-reasoning tools have not explicitly been used on them. As we know from Chapters 2 and 5, using the *do*-calculus on a causal network allows us to assess the effect of interventions on variables in an unbiased way (Pearl, 2000), and makes it possible to extract intuitive causal explanations about a given state.

Applications using data to build the structure of the graph include hard-drive-failure diagnosis tools (Fine & Ziv, 2003) and other computer-diagnostic tools (Brodie et al., 2002), medical decision-support systems (Heckerman et al., 1992; Heckerman & Nathwani, 1992), production-line analysis (Wolbrecht et al., 2000), and others (Heckerman et al., 1995; Neil et al., 2008). The construction of the network in studies of process performance raises some challenges that are not easily overcome, foremost being the lack of quantitative variables. Many processes involve variables that are difficult to quantify and analyze rigorously, such as employee training, organizational clarity, etc., so that defining the level of granularity, the structure of the network, and its parameters involve additional challenges. Such applications include reliability analysis (Neil et al., 2001), defect prediction (Fenton et al., 2007*b*, 2008), software cost estimation (Stamelosa et al., 2005), or accident causation (Marsh & Bearfield, 2004), to name a few.

When asked about the causes which might induce a decrease in process performance, two experts might contradict each other not only about the causes of risk but also about the strength of the cause–effect relationship. To extract quantitative risk measures from a pool of experts, methods can be applied such as the Delphi approach (Turoff & Hiltz, 1996) or Six Sigma (Barney & McCarty, 2002). However, those methods cannot manage data and expert knowledge at the same level. The fishbone/Ishikawa cause–effect diagrams (Ishikawa, 1990) are for instance recommended by six sigma experts to perform root-cause analysis. These are graphs of potential causes for defects

that are entirely built from expert knowledge, but significantly differ from the causal graphs defined through Chapter 2. In particular, whereas data cannot be used to refine a purely qualitative Ishikawa diagram, causal Bayesian networks can be updated with data whenever it becomes reliably available, and can be used for various causal-reasoning tasks, for instance, explanations (Chapter 5). In this context, they can be seen as a quantitative, more formal complement to an Ishikawa diagram.

In the next subsections, we show how to build a causal network from expert knowledge to assess the risk of quality failure in a process. First, we describe our methodology to build a fully specified causal model from an Ishikawa diagram and a team of experts. Then, we describe the model we obtained following this methodology in the context of pharmaceutical manufacturing processes. Finally, we discuss several examples of explanatory and exploratory analysis that we did with the resulting model.

6.1.2 Methodology for Creating a Quality-Risk Causal Model

This section describes the risk-assessment problem we address and describes a methodology to construct the structure and determine the parameters of the causal model.

Quality Risk

Assuming a set of severity levels $\{s_1, \cdots, s_n\}$, quality risk measures the frequency of the following failures in the different severity levels: On the one hand, *fitness-for-use* failures occur when the quality, efficacy or safety of the end product/service is not ensured. Quality relates to a product conforming to its specification. Efficacy ensures that a product has a positive effect (e.g., a drug treats a patient or a service brings value to the customer). Safety assumes that the end product is not harmful (e.g., no patient suffers from fatal side effects or no service is damaging to other business units). In the case of pharmaceutical manufacturing processes, a problem with the raw material, blending or product compounds is typically related to fitness-for-use risk. It has a direct impact on the physical product or how it is processed. On the other hand, *compliance* failures occur when quality, efficacy or safety in the process fails to be controlled.

These two components of quality risk are orthogonal and do not imply each other. A manufacturing plant for instance can produce high-quality products but might not report any information about the manufacturing process. Its fitness-for-use risk is low but its compliance risk is high. Conversely, another site might have many controls in place to ensure that all good manufacturing practice (GMP) systems are implemented. This does not mean that their yield is high since many batches might have to be re-processed or thrown away because of low quality.

Assuming that the process is stationary, a frequency measure is enough to account for failures on different time scales: the number of failures over t time periods is simply the product of the unit frequency times t. Quality risk can therefore also be interpreted as a measure of the required time for one failure to occur. High-risk processes would for instance generate a failure every hour, while low-risk processes might induce only a failure every year.

A seemingly straightforward approach to compute quality risk would be to collect all reports on defective end products and noncompliance events, to count the number of times such events occurred and to return the corresponding frequencies as defined in the quality risk. Unfortunately,

this approach does not work if data collection is not under control. Lack of failure reports does not mean no events—it only means no reporting of events, which may be due to political or cultural factors. One way to circumvent such issues is to extract knowledge from non-statistical sources, such as experts. This does not preclude any use of data but adds assumptions about how statistics should be interpreted. The remainder of this section describes a method based on causal networks that allows for such expert knowledge to be used in conjunction with statistics.

Determining the Structure

There are a few methodologies and guidelines to build the structure from expert data. For instance, Neil et al. (2000) discuss a general approach for building large-scale Bayesian networks as well as how to choose the suitable level of granularity. Marsh & Bearfield (2008) describe how Bayesian networks can be built from event trees/fault trees in the context of safety assessment; Marquez et al. (2008) develop specialized inference algorithms to reason over these Bayesian network built from fault trees. In this section, we propose to build the structure starting from a fishbone diagram. The fishbone (or Ishikawa) diagram is a cause–effect graph built by a panel of experts to identify potential causes for a problem. It is the result of a technique designed by Ishikawa (1990) in the sixties to track the root causes for out-of-specification measurements. Figure 6.1 (a) shows such a diagram designed by industry experts to analyze poor performance of pharmaceutical manufacturing processes. It is composed of a main horizontal line representing the first layer of causes (horizontal arrow pointing to the poor-process-performance box). Other arrows correspond to categories of causes. The leaves of this tree-structured diagram are the potential causes. The latter can then be ranked for further analysis and are meant to be investigated individually.

The causal network adds further structure to the approach by providing a way to investigate links between the causes. These links should be direct cause–effect relationships and can appear between causes that are not in the same branch of the Ishikawa diagram. The art of building the causal network resides in how accurately we can represent the causal influences between the causes listed by the Ishikawa diagram. Whereas the Ishikawa diagram can be constructed by tracking back the causes of a failure by repeatedly asking "why?" until a root cause is identified (Ishikawa, 1990), the causal network requires that, for each potential cause, one should determine the direct effects. Once these are established, the structure of the causal network is almost defined. We list a simple procedure for building the structure of a causal graph given an Ishikawa diagram. We assume, for each step, that a moderator asks the experts questions, and that the experts meet to reach a consensus.

Build the Structure of a Causal Network from Expert Knowledge

1. Get an Ishikawa diagram and define an initial set of variables/nodes for the causal network by flattening the hierarchy appearing in the diagram.
2. Start from the initial causes by asking, "What other variable can be directly influenced by this one?" and connect them to it by a directed link. Repeat with all variables until no new link is added.
3. Draw the full resulting graph and submit it to the team of experts for review, asking again for each link if this represents a direct cause–effect relationship. Remove the links if the perceived causal effect can be accounted for by another causal path in the graph.
4. For each variable, determine its levels/states.

Estimating the Parameters

Once the structure of the causal network is defined, the parameters determining the conditional probability tables must be estimated. Without data, there is only a limited set of information sources that can be used. The first source of information is the literature, such as survey reports (Tran et al., 2005). The second and main source of information is expert knowledge. The structure of the causal network can be used to develop a questionnaire: the graph provides the variable to measure and the values to consider. However, setting all the entries of a probability table is usually cumbersome and not easy to grasp for someone not trained in statistics. Furthermore, complex interactions between several causes are usually difficult to assess directly. Parametrized probabilistic models such as noisy-OR or noisy-MAX can be used to make conditional-probability elicitation at each node more intuitive.

The noisy-OR model (Pearl, 1986) can be applied with binary variables and assumes independence between all the causes of a variable. In a medical setting for instance, an effect variable Y (e.g., a disease) might be caused by any of the anomalies $X_1, \cdots, X_d$ independently. Each of these n causes is assumed to have an independent probability p_i of causing Y. The full conditional probability table for Y can then be computed as

$$\mathrm{P}(Y \mid X_1, X_2, \cdots, X_d) = \prod_{i=1}^{d} (1 - p_i)^{x_i}, \tag{6.1}$$

where x_i can only be 1 or 0. This model only requires us to specify d parameters instead of $2^d - 1$ for the full probability table. The noisy-MAX model (Dìez, 1993) is the equivalent of the noisy-OR model for graded variables. Instead of specifying one probability p_i for each cause X_i, it requires the definition of $p_{ij} = \mathrm{P}(Y \mid X_i = x_{ij})$, where x_{ij} is the jth possible value for variable X_i; e.g., the intensity of the anomaly. The use of such models has been already advised in the design of diagnostic tools from expert knowledge using causal networks (Kraaijeveld & Druzdzel, 2005). Other techniques used for probability elicitation include for instance Fenton et al. (2007*a*), which describes the use of truncated Gaussian distributions and also lets us stipulate a linear (rather than exponential) number of parameters to specify a conditional probability distribution.

Since these probabilities are rarely known with certainty, it might be worthwhile to ask experts for intervals rather than point estimates; that is, for worst-case and best-case values that could later be used to analyze various scenarios, according to their opinion or experience. This requires more work from the experts but it generates a more informative risk measure by giving an estimate of the uncertainty of the expert.

6.1.3 A Causal Model for Pharmaceutical Manufacturing Processes

This section describes the causal network that we derived using the previous methodology for pharmaceutical manufacturing processes. Since data was not available, the causal structure and the parameters were estimated by a pool of experts, by successive iterations until a consensus was reached. The section ends with an example on how to use the resulting causal network in two different scenarios.

Setup

To analyze the quality risk related to the pharmaceutical manufacturing process, we organized a series of workshops with industry experts. These workshops consisted of a mixture of brainstorming and training sessions and generally lasted two days. Two thirds of the team was composed of manufacturing process experts, the remainder being statisticians and specialists of causal networks. First, we built the structure of the causal network based on the Ishikawa diagram represented in Figure 6.1 (a). We added some nodes based on a study from the FDA Center for Drug Evaluation and Research (Tran et al., 2005). This study was also used to set the probabilities for some of the nodes of the causal network. Most of the probabilities, however, were set from expert knowledge. The resulting network is shown in Figure 6.1 (b), where only part of the network is depicted. Rounded rectangles are a convenient representation of subsets of the graph that will be detailed later.

Causal Network Structure

The first layer of the graph is structured as in U.S. Food and Drug Administration (2002) and decomposes the quality failure into three categories. First, the product-failure category represents any quality failure due to a defect in the drug such as contamination, instability of active ingredient, etc. Second, the technology-failure category represents any failure due to the technology used in the production line: a blender that does not work properly because it is too old, a technology that is not controlled because it is too new, etc. Last, the GMP system-failure category corresponds to any problems in the manufacturing process itself such as lack of documentation, process poorly controlled, quality assurance not implemented, etc. It is linked to the six GMP systems defined by the FDA (U.S. Food and Drug Administration, 2002) and which are targeted in the inspections.

In the graph, this decomposition is represented as three separate risk nodes. Additionally, we included a "risk summary" node that implements the following rule: there is a quality failure when at least one of the three parent failures occurs. These three categories can share the same set of root causes, as Figure 6.1 (b) shows. Besides the four submodels that are detailed later, four nodes appear at this level of description. The GEOGRAPHY/COUNTRY node encodes the dependency of the execution of some process on the geography. It is linked to ABILITY TO LEARN: countries have different strengths and weaknesses in skills and know-how. The three other nodes have been included at the top of the Ishikawa diagram to account for a recent FDA study on potential sources of process failure (Tran et al., 2005). These are: PRODUCT TYPE for the different types of products that are manufactured on the site under investigation, UNIT OF OPERATION for the different units that compose the site, and CONTAMINATION VULNERABILITY to assess the risk of contamination in the product. Each variable is modeled as having various discrete levels; e.g., low, medium, high.

We divided the details of the full causal network into four submodels, shown separately in Figure 6.2. The dosage-form variability can be a potential source of defects for a product and is the focus of the first submodel. Producing a product with many pharmaceutical ingredients with different dosage forms and possibly different packaging/labeling increases the chance of confusion and the chance of uncontrolled biological interactions is higher. Most of the nodes in this submodel relate to the ingredients and the way the product is manufactured.

Second, the **PROCESS CONTROL** submodel represents the set of variables that might be responsible for any loss of control in the manufacturing process. The INHERENT LOSS OF CONTROL node is

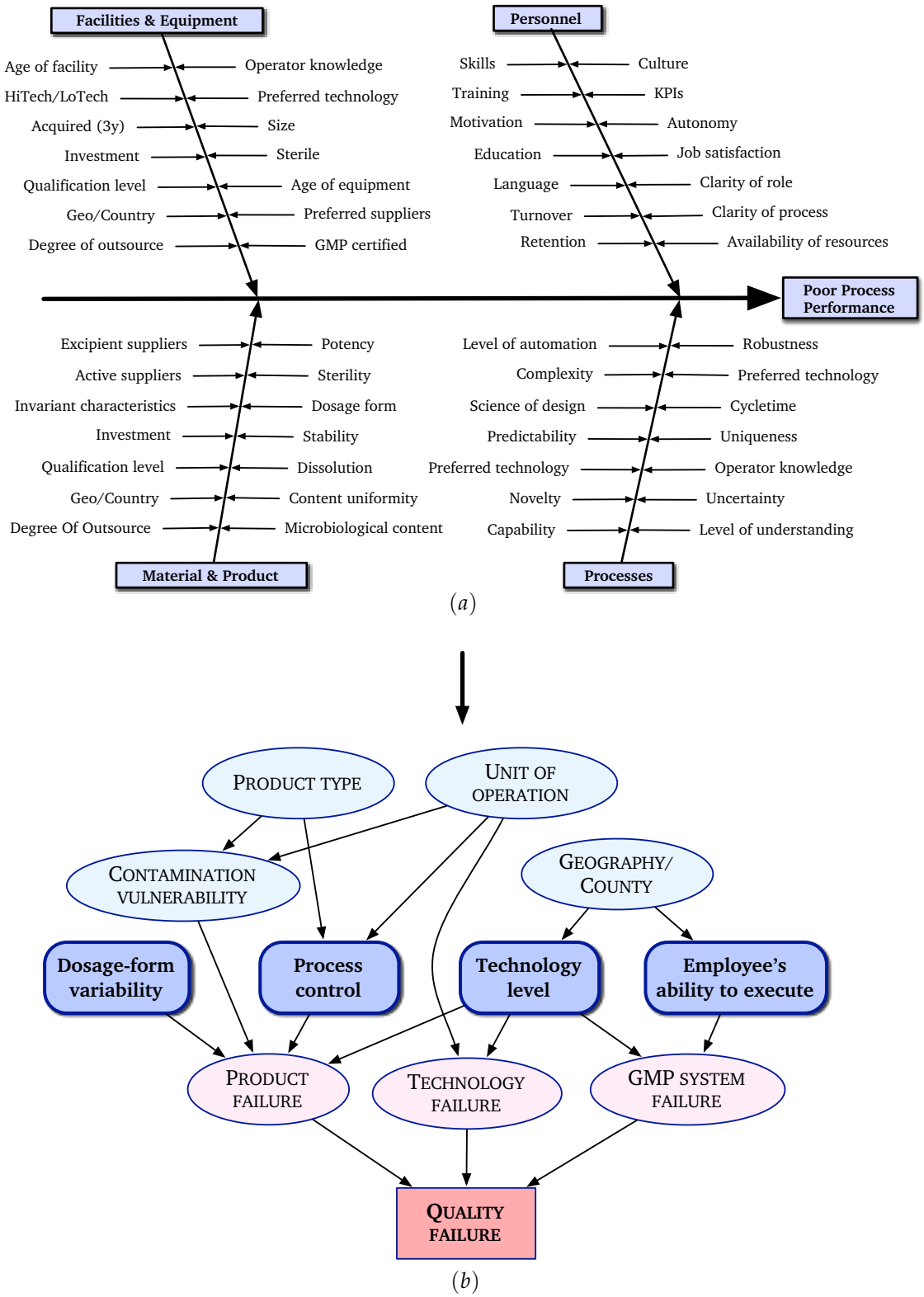


Figure 6.1: From an Ishikawa diagram (a) to a causal graph (b). We followed the methodology described in Section 6.1.2 for the conversion. The causal graph contains four submodels depicted on a dark blue background, detailed in Figure 6.2.

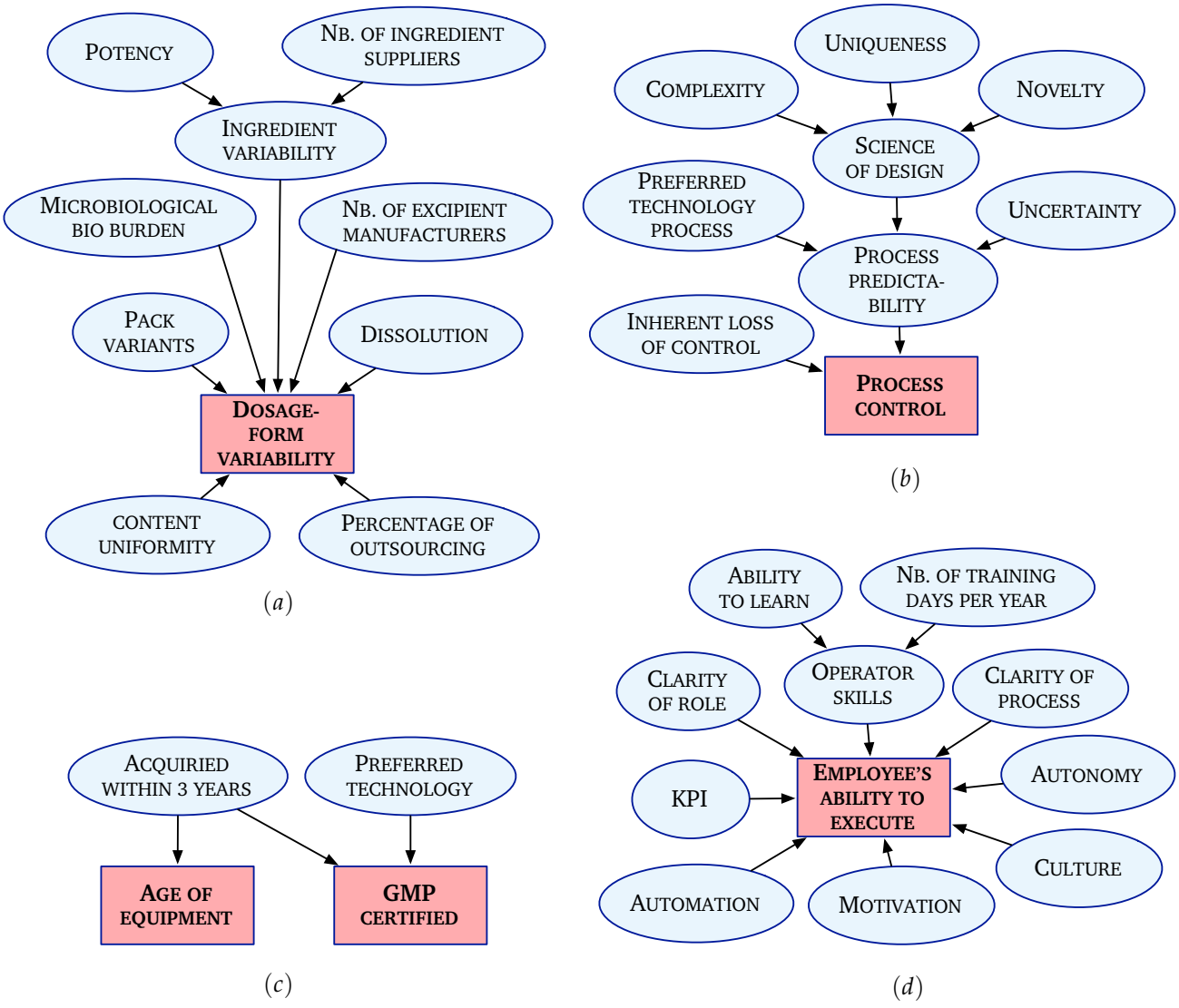


Figure 6.2: Details for the four submodels of the causal graph in Figure 6.1 (b): (a) **Dosage-form variability**, (b) **Process control**, (c) **Technology level**, and (d) **Employee's ability to execute**. The red nodes represent output for the arrows coming out of the submodels in Figure 6.1.

Name	Description
NB. OF INGREDIENT SUPPLIERS	*self explanatory*
POTENCY	Potency of the product defined as its power to induce a response from the patient
INGREDIENT VARIABILITY	Set to *yes* if the ingredient can be significantly different during the process
MICROBIOLOGICAL BIO BURDEN	Level of biological burden that should be handled by the process. In some drugs, microbial organisms are present and require an antimicrobial process to remove any contamination
NB. OF EXCIPIENT MANUFACTURERS	Number of excipient manufacturers, not distributors
PACK VARIANTS	Number of packages that a particular dosage could be packed into (plants supplying a multilingual packaging will have a higher value)
DISSOLUTION	Dissolution for products that are liquids (sprays, injections, etc.)
CONTENT UNIFORMITY	Uniformity of the drug before dosage is processed (e.g., uniformity after blending and before granulating to produce tablets)
PERCENTAGE OF OUTSOURCING	*self explanatory*
DOSAGE-FORM VARIABILITY	Variability in the final version of the drug product (excipient plus ingredient)

Table 6.1: Description of the nodes for the **DOSAGE-FORM VARIABILITY** submodel.

Name	Description
UNIQUENESS	Percentage of the processing steps that are unique within the process of interest
COMPLEXITY	Complexity of the processing steps
NOVELTY	Proportion of new processing steps within the process of interest
SCIENCE OF DESIGN	Amount of science of design that was involved in the process of interest. Science of design refers to activities relating to the validation and design of the original process. A simple process that is not unique and has been running in different plants for many years requires less science of design than a new and complex process
PREFERRED TECHNOLOGY PROCESS	Use of process-analytical tools for monitoring the process
UNCERTAINTY	Noise in the process that makes any prediction difficult
PROCESS PREDICTABILITY	Ability to predict the status of the process of interest
INHERENT LOSS OF CONTROL	Inherent loss of control independent of any detection. A process can become out of control because it manufactures complex products with possible contamination. However, if such abnormal behavior is detected, the process although inherently out of control is effectively in control
PROCESS CONTROL	Ability to control the process, either by sticking to the specifications or by detecting when the process gets out of control

Table 6.2: Description of the nodes for the **PROCESS CONTROL** submodel.

Name	Description
ACQUIRED WITHIN 3 YEARS	Three years generally corresponds to the time when employees of the acquired company can leave without any financial damages
PREFERRED TECHNOLOGY	Limited to only two values PAT or low technology
AGE OF EQUIPMENT	*self explanatory*
GMP CERTIFIED	Sets the level of certification regarding the current good manufacturing practice of the FDA

Table 6.3: Description of the nodes for the **TECHNOLOGY LEVEL** submodel.

Name	Description
ABILITY TO LEARN	Highest educational level
NUMBER OF TRAINING DAYS PER YEAR	*self explanatory*
OPERATOR SKILLS	*self explanatory*
CLARITY OF ROLE	Percentage of the operator's role that is clearly defined and understood by the operator
KPI	KPI on the process
AUTOMATION	Level of assistance that an employee gets through automation or mistake proofing or other enablers
CLARITY OF PROCESS	*self explanatory*
AUTONOMY	Empowerment and ability to work independently and make decisions independently; usually not so much autonomy
CULTURE	Propensity is defined by the regulatory and environment and structure of the process. Implications are high for those who are risk takers
MOTIVATION	Symptoms of lack of motivation are, for instance, high turnover, absenteeism, sloppy work, etc.
EMPLOYEE'S ABILITY TO EXECUTE	Ability to execute independently of the complexity of the process

Table 6.4: Description of the nodes for the **EMPLOYEE'S ABILITY TO EXECUTE** submodel.

linked to unit of operations and relates to a loss of control owing to properties of the drug or of the unit of operations. A complex biotech product for instance is generally considered more risky to manufacture than aspirin (Tran et al., 2005). On the other hand, the control variable depends on the intrinsic riskiness of the process and the ability of employees and systems to predict its status. An inherently risky process can present many defects but would still be under control if defects are predicted and detected.

Our third submodel, the **TECHNOLOGY LEVEL** submodel, relates to any failure that might be due to the aging of the equipment, the lack of advanced analytics to track process performance, and more generally to a defect in the tool to monitor the process or produce the drug. Such a failure is linked to product and technology failure via the AGE OF EQUIPMENT node. It is linked to the GMP system failure node via the GMP certified node. The PREFERRED TECHNOLOGY is influenced here by GEOGRAPHY/COUNTRY.

Finally, the **EMPLOYEE'S ABILITY TO EXECUTE** submodel is directly linked to GMP failures. It relates to events such as reporting not done properly, quality-control protocol not implemented correctly, etc. Many nodes in this part of the graph are about employee profiles and organization.

A short description of each variable can be found in Tables 6.1 through 6.4.

6.1.4 Experiments and Discussion

This part is divided into two: first, we examine an explanatory approach with causal-explanation trees, then, we evaluate scenarios with the help of the *do*-calculus. This corresponds to a real situation, where analysts first try to understand properties of the current situation and then try out various possible scenarios to fix problems identified in the explanatory phase, for example.

In both situations, we use the causal model we have built, shown in Figures 6.1 (b) and 6.2.

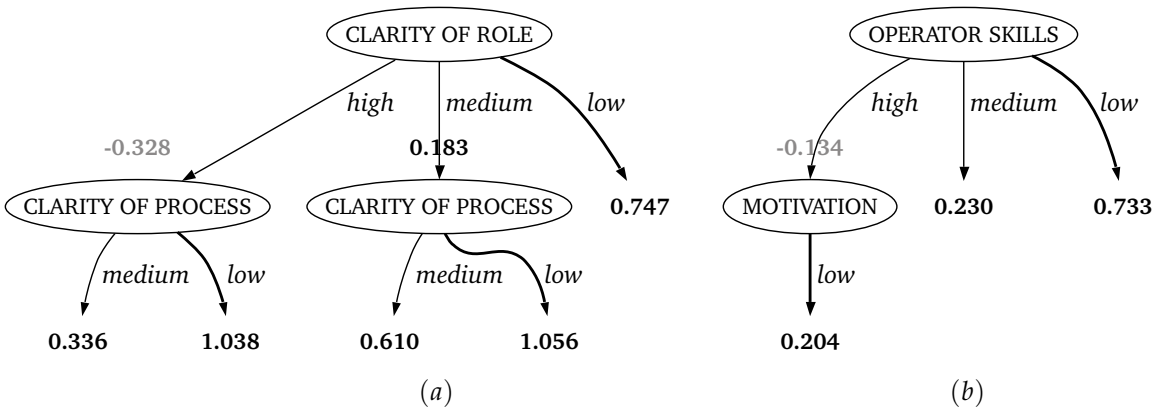


Figure 6.3: In the context of the **EMPLOYEE'S ABILITY TO EXECUTE** submodel in Figure 6.2 (d): (a) explains the low ability to execute either with a low clarity of role, or (if clarity of role is high or medium) with a low or medium clarity of process; (b) explains the low ability to execute given a high clarity of role and a high clarity of process. The proposed explanations are low or medium operator skills, or (with high operator skills) low motivation.

Explanatory Analysis

For various explanatory analysis cases, we used the causal-explanation tree (CET) method described in Chapter 5. We were careful to design the network causally and tried to be exhaustive in the description of the causal relationships in the graph: we can thus expect that CET will deliver sensible results. Let us recall that CET makes no additional assumption other than that the provided causal model is a perfect map and fully specified.

Our goal is to identify the problematic configurations likely to lead to quality failures; we show four example cases that were run in our submodels.

Looking at the **EMPLOYEE'S ABILITY TO EXECUTE** submodel in Figure 6.2 (d), suppose we wish to determine the likeliest causes for a low ability to execute. We ran CET with the explanandum EMPLOYEE'S ABILITY TO EXECUTE $= low$, and no additional observation, with the stopping criterion $\alpha = 0.005$. (This means that we stop growing the tree if any additional variable X to the explanation $\mathbf{P} = \mathbf{p}$ has a causal-information flow to the explanandum smaller than α; i.e., if $I(X \to e \,|\, \mathbf{o}, \hat{\mathbf{p}}) < \alpha$, with $\mathbf{o}$ being the additional optional observations about the system that we can provide as input to the algorithm.)

The result of this first explanation query is shown in Figure 6.3 (a). (It is worth noting that we ran CET as shown in Algorithm 5.2, including the pruning of bad explanation subtrees implemented by the condition at line 10. In the previous chapter, we still rendered those subtrees in gray.)

Let us recall that in an explanation tree, each path from the root of the tree to a leaf represents an explanation, i.e., a set of variables and of values. The leaf label represents the "goodness" of the explanation: the higher the number, the more likely the explanandum given the proposed explanation. Figure 6.3 (a) thus proposes 5 explanations for a low ability to execute, all involving lack of clarity in roles or in processes. This means that the lack of clarity is the main reason why employees perform badly: all other variables have a smaller expected causal influence.

Knowing that lack of clarity in the role and the process are primary reasons for a low-performance employee, we can ask what other variables are relevant when both CLARITY OF ROLE and CLARITY

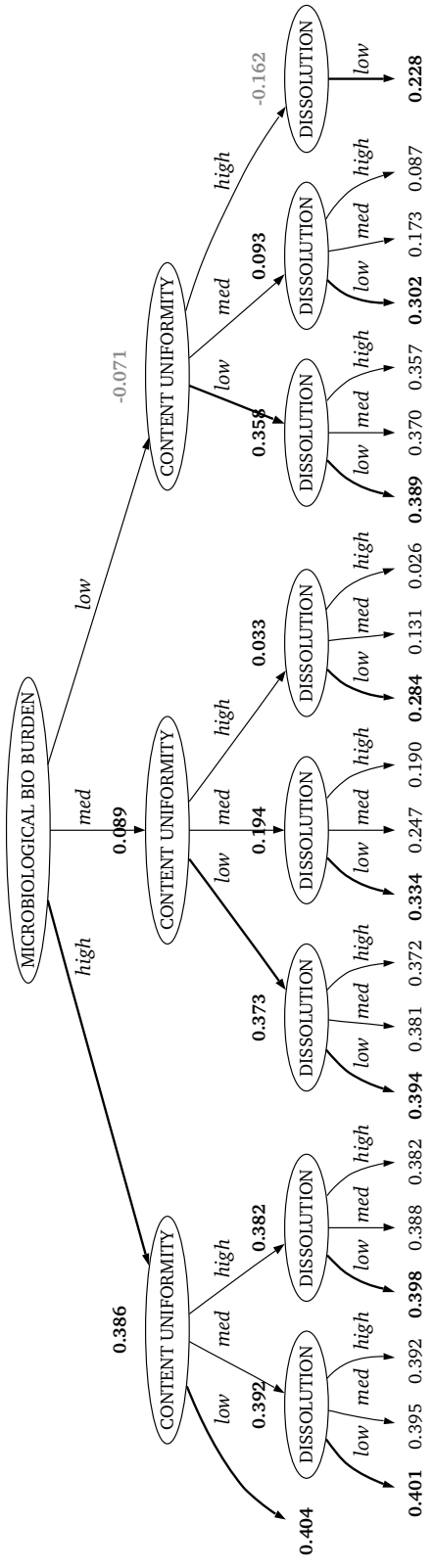


Figure 6.4: In the context of the **Dosage-form variability** submodel in Figure 6.2 (a), this causal-explanation tree explains a high variability in dosage form.

OF PROCESS are high. This is different from growing the tree in Figure 6.3 (a) further: now, we condition on the additional observations CLARITY OF ROLE $=$ *high* and CLARITY OF PROCESS $=$ *high* from the start, which "scales up" causal contributions of other variables that were too small to show up in the original tree. The new tree is shown in Figure 6.3 (b) and tells us that low or medium operator skills or low motivation are most relevant in this context.

We now examine a different submodel, the **DOSAGE-FORM VARIABILITY** submodel, as shown in Figure 6.2 (a). We wish to know what causes high variability in the dosage form, as this is detrimental to the overall quality risk. We ran CET with $\alpha = 0.05$ on this submodel, allowing as explanatory variables all parentless nodes. This excludes INGREDIENT VARIABILITY, which is itself the effect of two more fundamental, and thus more interesting, causes.

The resulting tree is shown in Figure 6.4. The tree is arranged so that the branches out of each node are arranged so that the state that favors the explanandum most is leftmost. A high microbiological burden is the primary explanation for high variability in the dosage form; adding to this a low content uniformity improves the explanation and is the best explanation returned by CET with a label of 0.404. All other proposed explanations containing other states of MICROBIOLOGICAL BIO BURDEN and CONTENT UNIFORMITY can usually be improved with a low or medium dissolution. DISSOLUTION is not really a node that we can intervene on, it is rather a property of the final pharmaceutical product, so we could have excluded it from the possible explanatory variables we give CET; leaving it in, however, teaches us that it is the third most important factor in our submodel to influence dosage-form variability.

Suppose that, for some reason, it is deemed too expensive to fix the problem with microbiological burden or the content uniformity, but we still need to improve the dosage-form variability, especially in the case where DISSOLUTION $=$ *low*. We also frequently observe INGREDIENT VARIABILITY $=$ *yes* and would like to take this into account to test its relevance. We can thus run CET with the sample explanandum, but with the additional observations DISSOLUTION $=$ *low* and INGREDIENT VARIABILITY $=$ *yes*. From the list of explanatory variables, we also exclude MICROBIOLOGICAL BIO BURDEN and DISSOLUTION as we found that we cannot influence them much anyway. The result, shown in Figure 6.5 (a), shows that the (observed) ingredient variability is still the primary explanation for high variability in the dosage form, and that additionally, low content uniformity makes things worse. Trying to fix other elements in this submodel is thus not likely to improve the dosage-form variability by much in this situation.

Finally, we propose to investigate the case where we are just interested in the general overall influence of one particular variable on the explanandum. Suppose we have an opportunity to reduce the number of excipient suppliers and are interested in how this would influence our problem of the high dosage-form variability. We can run CET again with only one explanatory X variable and see whether it is selected or not. If we get an empty, it means that X has too low an influence (according to the passed α, or no influence at all, if $\alpha = 0$) on the explanandum. If, on the contrary, X is selected for certain of its states, then it gives a precise indication that X in these states indeed has a causally relevant explanatory value. In other words, when we use CET like this, we ignore its search capacity on the explanatory variables themselves (because only X can be selected), but we use CET to search the states of X that contribute to the explanandum.

In this case, we can see in Figure 6.5 (b) that if we have more than 20 excipient suppliers, this can indeed make the dosage-form variability higher.[18] Consequently, if we have an opportunity to

[18] CET does not find the inequality "> 20" on its own; instead, *larger than twenty* is one of the discrete states of the variable NB. OF EXCIPIENT SUPPLIERS.

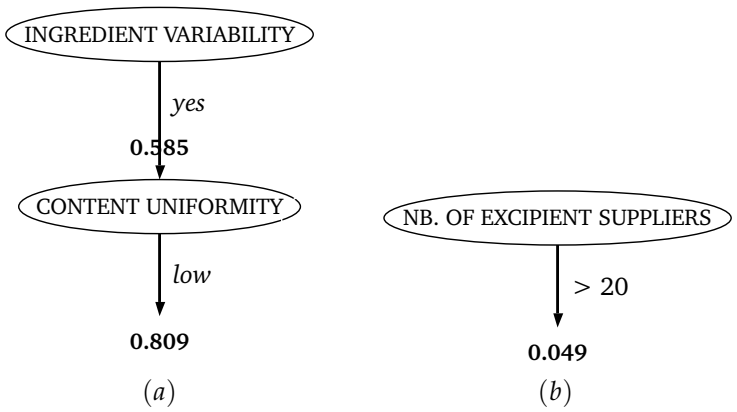


Figure 6.5: In the context of the **DOSAGE-FORM VARIABILITY** submodel in Figure 6.2 (a): (a) explains high dosage-form variability when DISSOLUTION $= high$ and INGREDIENT VARIABILITY $= yes$; (b) tests the relevance of the states of NB. OF EXCIPIENT SUPPLIERS towards a high dosage-form variability.

reduce it to below 20, we can expect a small improvement—small, because the label 0.049 tells us that, with respect to other labels in Figure 6.5 (b) or in Figure 6.4, NB. OF EXCIPIENT SUPPLIERS $>$ 20 has a comparatively small causal effect, too.

We see that explanatory analyses of this kind enable analysts to interpret the causal graph more easily by representing the information it contains with respect to a target variable given different contexts. Most of the time, reading an explanation tree is easier than interpreting the values of a conditional probability table, most of all for variables that are not directly connected to the target variable.

Exploratory Analysis

This kind of analysis can be seen as the converse of the explanatory approaches discussed in the previous paragraphs. Whereas in the context of an explanation, we observe variables and need to explain a subset of the observation, in the exploratory approach we first decide what to do, encode this as intervention, and use the *do*-calculus to see how this affects chosen target variables in their probability distribution.

Although our examples hereafter are unrelated to the explanation trees presented before, the explanatory and exploratory approaches can be used in close conjunction. In the context of a larger analysis, we could be interested to find an explanation for some undesirable observation $X = x$ somewhere in the network, and, looking at the proposed explanations, decide what to change in the system. We could represent this as the intervention $do(\mathbf{Z} = \mathbf{z})$. But before implementing the change, we can ask the causal model what this change would mean for a given variable Y somewhere else in the network, and compute $\mathrm{P}(Y \mid do(\mathbf{Z} = \mathbf{z}))$.

Exploratory analysis in the form of evaluation of a *do*-expression is the most straightforward application of the causal model, and is the equivalent of marginalization given some evidence in traditional Bayesian networks. With respect to a target Y, the result of the observation $\mathbf{Z} = \mathbf{z}$ and intervention $do(\mathbf{Z} = \mathbf{z})$ can coincide in certain cases, where we have $\mathrm{P}(Y \mid do(\mathbf{Z} = \mathbf{z})) = \mathrm{P}(Y \mid \mathbf{Z} = \mathbf{z})$. Intuitively, intervening on $\mathbf{Z}$ is the same as observing these variables in the corresponding states when there is no causal mechanism that generates, directly or indirectly, both certain variables in

QUALITY FAILURE	Current situation	Moving to another country	Increasing the turnover
death	0%	0%	0%
hospitalization	2%	5%	5%
acute illness	13%	14%	14%
benign aftereffect	14%	21%	22%
none	71%	59%	58%

Table 6.5: Overall quality risk in the example scenarios of the studied process.

$\mathbf{Z}$ and our target variable Y. The *do*-calculus tells us more formally when this is the case—we can extract the following implication as a special case of Rule 2 on page 26:

$$(Y \perp\!\!\!\perp \mathbf{Z})_{\mathcal{G}_{\underline{\mathbf{Z}}}} \implies \mathrm{P}(Y \mid do(\mathbf{Z})) = \mathrm{P}(Y \mid \mathbf{Z}). \tag{6.2}$$

What this tells us is that we can interchange intervention and observation in situations where the target variable Y is d-separated from the manipulated variables $\mathbf{Z}$ in $\mathcal{G}_{\underline{\mathbf{Z}}}$, where $\mathcal{G}_{\underline{\mathbf{Z}}}$ is the graph $\mathcal{G}$ representing the causal structure with all edges coming out of $\mathbf{Z}$ removed. This characterizes structures where $\mathbf{Z}$ and Y are not linked through a *back door* (Pearl, 2000, p. 79); i.e., through connecting paths that lie outside the direct causal effect of $\mathbf{Z}$ on Y. In circumstances where a back door exists, such as the Drug example we described in Section 2.1, we determined that we cannot use $\mathrm{P}(Y \mid \mathbf{Z} = \mathbf{z})$ to measure the causal effect of $\mathbf{Z}$ on Y, as this gives biased results and can lead to paradoxical interpretations.

We now describe two particular "what-if" scenarios that can be analyzed with the causal network. In the current situation, the process under investigation is the production of a low-active ointment. The units of operations are: emulsification, mixing, blending, de-aeration, heating, cooling, measuring and packaging. We assume that the process is running in Switzerland and that the pharmaceutical company has process-analytical tools in place. All variables related to employee's ability to execute are supposed to be at their most efficient values. We now consider two scenarios: moving to another country and increasing the turnover, and use the *do*-calculus as describe below to evaluate them in terms of the overall quality risk. We show the distribution of the quality risk for the two scenarios compared to the current situation in Table 6.5, where we indicate the probability for each discrete state: *death*, *hospitalization*, *acute illness*, *benign aftereffect*, and *none*.

In the "moving to another country" scenario, we propose to move the production to Puerto Rico. Most of the variables would stay the same except those regarding employees and GEOGRAPHY/COUNTRY. The latter is manipulated to be set to *PuertoRico*, and we evaluate the intervention $do(\text{GEOGRAPHY/COUNTRY} = \textit{PuertoRico})$.

The model tells us that there is a large degradation of the employee's ability to execute: 53% is rated *high* in the current situation, which drops to 15% in the new scenario. This is due to a decrease in the clarity of role and the autonomy, which are nearly perfect without moving. The impact on the quality risk is a decrease from 71% to 59%, less than the employee's ability to execute as the latter has only a partial influence on the quality risk. Process control and technology are also important and are supposed in this scenario to be performing well. The variables that would individually have the most impact on a risk reduction are OPERATOR SKILLS, MOTIVATION, CLARITY OF ROLE, and CULTURE. Obviously, the employee's ability to execute has a strong impact on the quality risk. Technology on the other hand is not as critical. This example shows that introducing a change in a manufacturing process implies an increase of the quality risk. This

seems natural and unavoidable since clarity of role and autonomy, for instance, are degraded. On the other hand, it is unclear which change would least degrade the quality risk.

In the "increasing the turnover" scenario, we assume that 50% of the employees decide to leave and are replaced by new hires. The **EMPLOYEE'S ABILITY TO EXECUTE** submodel will therefore be changed based on the new configuration. In accord with experts, we expect AUTONOMY and CLARITY OF ROLE to be reduced: new employees are supposed to be less autonomous than existing employees. Their culture has been changed as well: they are slightly less risk-averse. Assuming that new employees are eager to perform well, their motivation is higher than before. Based on these changes modeled as manipulations, we evaluate the corresponding manipulations on the causal model. (Alternatively, AUTONOMY, CLARITY OF ROLE, and similar nodes could all have been defined in terms of a common parent, say, NOVICITY. The extent to which such mechanisms are explicitly modeled or just appear as interventions in second-level effect nodes like these also depends on the personal taste of the analysts.)

Overall, employees would not perform as well as in the first scenario: the proportion of them rated as having low ability increases from 10% to 52%. This difference is smoothed in the final quality risk, as is shown in Table 6.5.

If we compare the two scenarios, we see that there is only at most 1% difference between the probabilities resulting from the manipulations. This shows that technology and product quality being well under control in both cases, moving to another country is, where quality failures are concerned, almost identical to renewing half of the production staff. The difference is very small and might change depending on the hypothesis, i.e., on the distributions that are assumed for some of the variables. However, by using such probabilistic tools, we have the assurance of deriving conclusions compatible with the knowledge put in during construction, and of being bias-free in the sense demonstrated by Simpson's paradox, as we are using a causal model.

6.1.5 Conclusion

Up to the present day, causal analysis, rendered accessible to artificial intelligence through mathematics, has been practically limited to the domains of medicine, sociology, economics and artificial intelligence: the field of processes, on the other hand, owing to their qualitative nature and the extra challenges they raise, has had to wait. Most business processes are exceedingly complex, and require a deep understanding of the industries in which they function. It is for these reasons that an uncommonly large gap exists between statisticians and industry experts. The latter, usually consultants, recognize the value of formalizing their knowledge; however, they also realize that it has been built up from experience and does not correspond to any tangible objects. Unlike the analysis of potential failures in a pacemaker, investigating the potential causes of low process performance concerns such qualities as motivation, culture or other concepts that are difficult even to define, let alone quantify. All of which makes the construction of a causal model more difficult and resource-consuming.

On the other hand, the gain could be worth the investment. First, building a causal network bring experts to agree on the right information to collect in order to understand and monitor a process. Not only do they consider the potential causes of problems that occur now, they also consider all source of failures that might explain poor process quality. Once the network is built, experts can query the model, explore its properties, and compare scenarios, constantly validating the

adequacy of the resulting probabilities with their own knowledge. In particular, they can ensure that the decision they make is compatible—using probability rules—with the knowledge they have formalized into the causal network. Ultimately, if data is collected properly, this knowledge can be enhanced by objective facts: most of the structure can be learned by algorithms. Finally, causal networks provide a natural tool for the sharing of knowledge and its storage in a form that can be used and queried by others.

Application Summary

We have described a methodology for building a causal model to compute process-quality risk and demonstrated it with pharmaceutical manufacturing processes. As data is scarce and unreliable, we do not use automatic structure-learning algorithms to build the causal graph, but rather a methodology based on the transformation of an Ishikawa diagram. Parametrization of the causal model is done with expert knowledge. The final model has allowed us to do explanatory- and exploratory-type analyses: this means that we have been able to both ask it to explain certain current mechanisms by extracting information as causal-explanation trees, and ask it to predict what alternative scenarios would look like according to given interventions. In this sense, we have argued that it was worth the effort to build a full causal model.

6.2 GRADUATE-PROGRAM ADMISSION

In this section, we look at a context where decision-support tools should be developed to help determine whether to accept international students into a university's master's program.

With the Bologna accords, the bachelor's/master's system has been making its way into the European higher-education area since 1999 (European Ministers of Education, 1999). One of the goals of the Bologna process is to create more comparable, compatible, and coherent systems across European higher education. In particular, the Bologna declaration specifies the "*adoption of a system essentially based on two main cycles, undergraduate and graduate. Access to the second cycle shall require successful completion of first-cycle studies, lasting a minimum of three years. The degree awarded after the first cycle shall also be relevant to the European labor market as an appropriate level of qualification. The second cycle should lead to the master's and/or doctorate degree as in many European countries.*"

One of the consequences of the Bologna accords is thus increased geographic mobility between the undergraduate and graduate cycles. In particular, university A needs to be able to handle applications of students who completed their bachelor's program at university B, i.e., decide whether to allow them into their master's program. As universities are usually required to choose students that stand a good chance of completing the master's program, they are confronted with the delicate problem of finding viable selection criteria.

The chosen criteria should be a good indicator of the probability of success at the master's program. It would be in both universities' and foreign students' best interest if there were a perfect predictor of whether student S would successfully complete the master's program at university A. Such a predictor, of course, does not exist, but universities usually still select students according to criteria of some kind, based on application information such curriculum (including former grades and past course projects and theses, if any), reputation of university B, reference-letter rating, work experience if any, difficulty of the program that they are applying to, etc.

This is a rather difficult causal situation, where traditional tools are not likely to be readily applicable. All typically observed variables—chiefly grades and scores—are basically effects of the real causes we are trying to measure: various aspects of a student's knowledge and capabilities, which are not directly measurable. In this section, we use causal modeling to represent knowledge and capabilities as hidden variables, and try to quantify their impact on graduate performance. This case study is intended both as a discussion around the use of causal methods where predictive power is arguably more important than explanatory power, and as an example of situation where we have to develop alternative approaches derived from causal methods.

We first briefly review the literature about student-performance prediction. As we describe the data gathered from the international student's application form, we discuss why this data is difficult to use, and why traditional approaches might be of little help—basically, the lack of comparable data. We thus decide to first work with another dataset and propose a way to cluster course or course types according to a causal criterion, which helps characterize the kind of knowledge and capabilities that each course requires and influences.

(The approach we describe here is part of ongoing work with Judith Zimmermann of the Machine Learning and Pattern Recognition group at ETH Zurich. Though we only describe the causal clustering of the courses, ongoing activities include the use of such methods in predictive models, which we do not detail here.)

6.2.1 Introduction and Context

There are a lot of studies that involve selecting students for certain programs or trying to predict college or graduate-school performance: these selection (and consequently, success-prediction) problems have interested scholars for centuries, and publications dating back to the first half of the 20th century can still easily be found. They are part of a domain known as *educational research*, a social science concerned with investigating behavior of pupils/students, teachers, and educational institutions (Gall et al., 2006).

In this subsection, we do not aim to provide an exhaustive literature review—the list of relevant publications would be prohibitively long. What we do instead is summarize the findings and methods of some relatively recent studies and key papers in this area.

Widespread interest in the analysis of academic performance can be elucidated both by explanatory and selectionary needs. On the explanatory side, analysts have been trying to determine the factors that lead to success, looking at why students succeed or fail. The selectionary needs arise from a desire to offer education to the students that are more likely to succeed.

A good many studies are concerned with selecting students' applications to a graduate program, especially in expensive domains such as medicine. Most of these note the high significance or good correlation of the undergraduate grade point average (GPA) with the score at the final exam. Beeson & Kissling (2001) examined predictors of success for baccalaureate nursing graduates on the National Council Licensure Examination–Registered Nurse, and found that students who passed the final exam had significantly higher undergraduate GPAs. Chesnut & Phillips (2000) looked into criteria used for entry-level pharmacy programs, and found that motivation, character, ethics, and oral communication skills, as measured by a composite interview–references score, completed the academic criterion measured by (variants of) GPAs. Actually, in the U.S., a legal document requires the consideration of non-academic criteria for admission into a program leading to the Doctor of Pharmacy degree (American Council on Pharmaceutical Education, 1997). Using multiple regression, Downey et al. (2002) showed that incoming college GPA and score on the standard Scholastic Aptitude Test were significant predictors for final exam GPA for dental hygiene graduates. Middleton (2008) conducted a similar study, examining GPA and score on another standard test known as the National Certification Examination, concluding that there were "practical significance or potentially significant correlations between the majority of the predictor measures." Similarly, Evans & Wen (2007) investigated the case of osteopathic medical students and the Medical College Admission Test, observing that GPA was a good predictor towards the measured academic performance. Similar conclusions in chiropractic education name GPA as best predictor of the score on a test of the National Board of Chiropractic Examiners (Zhang, 1999), but warns that one's learning strategies, behavior, and attitude are not factored into the equation and are significant hidden variables.

In many U.S. graduate schools, especially in more technical domains, passing the Graduate Record Examination (GRE) is a requirement for admission. The GRE is a commercially-run standard test focusing on abstract thinking in mathematics, vocabulary, and analytical writing; although it is quite widespread, it has earned a lot of criticism since its introduction in 1949 (Sedlacek, 2003). More than 40 years ago, Stricker & Huber (1967) warned against the GRE, preferring criteria derived from undergraduate GPA. In a more recent paper, Orlando (2005) discussed, in particular, the high variance in the correlation of the GRE with graduation GPA as reported by various studies, and criticizes its use as sole criterion for denial of admission. In the context of a Master

of Information Science, Agbonlaho & Offor (2008) also reported undergraduate performance, together with a short elapsed time between obtention of the undergraduate degree and application to the graduate program, as a significant predictor for program completion.

Armstrong (2000) investigated the generally low predictivity of tests given to college applicants in California. He underlined that test scores alone cannot reflect the abilities of students, and that various student "dispositional characteristics" explain the high variance of success, as well as instructors' grading practices. Livengood (1992) examined the importance of psychological factors that are indices of college success, conducted an experiment that revealed that "students' rules of reasoning about effort and ability, motivational goals, and confidence in their intelligence are strongly related" to their "participation in the college experience and their level of satisfaction."

Our case study starts with the problem of dealing with applications from international students. In many European countries, Master-of-Science programs are taught in English, so the score on a test of English can be relevant. The score on the Test of English as a Foreign Language (TOEFL), in particular, was investigated: Sharon (1972) showed that this increased the predictive power of a model based on the GRE if it was included; Winberley et al. (1992) likewise reported that it was also beneficial to a model considering the undergraduate GPA. Nelson et al. (2004), factoring in more variable, such as age, gender, geographic categories of native countries, native language, academic area of concentration, and admission status (regular or probationary), and found that although the TOEFL score could be of use to predict academic performance, it was not a good predictor of whether or not international students completed the graduate program. The reason cited for this was that, in their study, international students that lacked English skills were already sufficiently screened out by the selection based on high previous academic performance.

We see that finding out why people succeed academically is a hard task, influenced by many unquantifiable factors. Most true causes are volatile and difficult to quantify, in the sense that they can change quickly during the academic career of a given student. Predicting success or failure with a given set of predictors, on the other hand, can be seen as a problem more of a statistical nature: the task boils down to finding which variables are most predictive of eventual success, usually measured by graduation. Actually, an overwhelming majority of studies concerned with predicting success use multiple linear regression with statistical tests on the regression weights to determine the significance of a predictor, or examine large correlation matrices and test the significance of the coefficients, little more. This is in line with old model-selection studies: Remus & Wong (1982), for instance, showed that a regression model performs better than other simple thresholding models involving the undergraduate GPA and the GMAT score[19] (as used in some U.S. universities at the time) for the prediction of student performance in a graduate program. More recently, Wilson & Hardgrave (1995) suggested that using classification, logistic-regression, or neural-networks models may be more appropriate than linear regression, but that accurate success prediction remains difficult using only the typical data describing the students.

To the best of our knowledge, Johnson & Richardson (1986) were the only ones to explicitly discuss a causal model to look into factors affecting student persistence in undergraduate studies. Various manually designed structures with goodness-of-fit criteria were tested, including variables such as EDUCATIONAL ASPIRATION, ACADEMIC SELF-CONCEPT, INTENT, or ACADEMIC INTEGRATION. Data was gathered with a survey addressed to a random sample of 955 students, about 575 of which responded. However, The final model was geared towards fitting PERSISTENCE and not

[19] GMAT stands for "Graduate Management Admission Test" and is commonly used in business schools in preference to the GRE.

academic success and was not used for any further causal reasoning—at the time, causal tools like the *do*-calculus had not been clearly formulated. More recently, Ward (2006) used a model based on multiple regression with features derived from and combined with GPA and GRE scores. The result is a web page that, based on GPA, GRE, and other academic information, informs computer-science students about which schools are most likely to accept them for a graduate program. This can be seen as an indirect approach, considering that the predicted variable is not program completion but admission into the program.

An important problem that we have with international students' applications is the lack of comparable data, especially because of the different courses and grading systems. But in our approach we will not be directly concerned with a prediction-type study to select applications, which has been done numerous times already. Instead, we will be concerned with building a model that aims to make this data more usable for building predictive models and to understand how various kinds of courses influence the graduate GPA. As we have seen, a lot of previous work indicates undergraduate GPA as the best predictor for graduate-program completion, and we want to refine this: we will base our analysis on the individual grades and courses that make up the GPA. In order to compare courses from different universities and different grading systems, we need a model that tells us which courses or course types are most relevant. Eventually, the model can be further refined by taking into account the targeted specialization indicated by the applicant (as described in the next subsection): no doubt certain specializations require different skills than others.

In order to build this model, we will not work with the international student data, but use the target university's historical data, which contains complete and comparable information about undergraduate and graduate performance, to model the importance of the knowledge and capabilities acquired in the undergraduate courses. In the following subsections, we examine our actual case study, describe what kind of data is available, discuss the reliability of the data extracted from the application form sent by international students, and propose a way to group courses according to their causal relevance.

6.2.2 Task and Data Specification

The original task is as follows: given application data, decide whether or not to accept a student into the master's program of a computer-science faculty. The application itself contains information that can be classified into four groups:

1. Personal information: gender, age, country of origin;
2. Curriculum information: current university, courses taken with grades, title of previous bachelor's or semester theses, official transcript from university, targeted specialization for the master's program, targeted final program (master's or doctorate),;
3. Extracurricular information: work experience, internships, other relevant activities;
4. Extra information: letter of motivation and personal statement, letters of recommendation, and other free-text information the student would like to provide.

Note that applications explicitly target a given specialization within the master's program; i.e., a focal area that will be more thoroughly studied. We are interested in this because no two specializations require the same skills, and we suspect that we may find different good predictors depending on the specialization. A student could thus be deemed insufficiently qualified for specialization s_1, but could be incited to apply for specialization s_2 instead. Out of 60 credit points

(excluding the master's thesis) that must be earned during the master's program, 26 must come from focus courses in the chosen specialization. There are 7 specializations: Information Security, Information Systems, Distributed Systems, Visual Computing, Theoretical Computer Science, Software Engineering, and Computational Science.

Data from the application form is difficult to use as is. The major problems we have with it are:

- Heterogeneity of the undergraduate programs. Courses with similar content can have different names in various universities (or be named in the local language and not in English); conversely, courses with a similar name can have greatly different actual content. Another important factor is the educational style: in some countries, students are more directed, more groomed than in others and may expect more or less guidance in the master's program than they have been accustomed to.
- Very different grading systems. Some are letter-based, others are numeric, some are mixed; the passing grades differ; some have different grades for non-passing results whereas others do not. In general, it is difficult to find viable means of conversion from one system to another. In addition, within the same grading system, practices may vary: even within universities, barely obtaining the passing grade may be commonplace or rather exceptional, according to the instructor.
- Difficulty of using free-text variables. The personal statement, letter of motivation and letters of recommendation are all free text and it is not obvious how to quantitatively encode their contents. Actually, it is hard to interpret these texts even for admission officers because of cultural or domain-specific differences in how they are written.
- Scarcity of easily comparable applicants. While several students that come from the same university are easier to compare than two students who followed different undergraduate programs in different parts of the world, for the majority of the applicants, there are not yet enough other applicants or historical data to which they could be more easily compared.

Another obvious problem is that for most applications, including past applications, we miss the target variable indicating success or failure at the master's program; past applications for which we know the target variables can only be students that were previously accepted into the program by whatever selection mechanism was in place at the time. We cannot know if rejected students would have completed the program successfully or not, so we have a significant selection bias in the usable training data: cases who were least likely to succeed are underrepresented (assuming the admission officer was more likely to reject low-potential applications). Conversely, very good students that were actually accepted into the program might not have come because they were accepted in another program with a better reputation or because they won a scholarship for another university.

These problems with the international students' data motivated us to consider an alternative solution to try and understand the relationships between grades at various courses and successful program completion, in an effort to determine the relevant course categories. We thus tried to build a model using the comparable and reliable data from students of the target university itself. There are several reasons why this data is better:

- In principle, all students follow the same undergraduate program, with the same grade system and very similar grading policies. Inter-year differences are also very limited;
- In this country, an undergraduate degree is not yet considered quite adequate for the job market, so that most if not all students continue studying with the master's program. This is

Semester	**Courses**
First semester	Introduction to Programming Logic Calculus I Linear Algebra Probability & Statistics
Second semester	Data Structures & Algorithms Physics Calculus II Discrete Mathematics Digital Design
Third semester	Systems Programming Computer Architecture Information Theory Theoretical Computer Science Electronics & Communication Computational Science
Fourth semester	Operating Systems Computer Networks Software Architecture Formal Methods & Functional Programming Databases Scientific Computing
Fifth & sixth semesters	*Course groups:* Information Security Information Systems Distributed Systems Visual Computing Algorithms, Probability & Computing Software Engineering Models & Simulation

Table 6.6: The undergraduate courses and course groups at the target university.

practical for us because for most students, both undergraduate and graduate GPA and course details are known;

- Data is guaranteed to be correct because we are able to extract it from university records directly.

We wish to use this data to build a model pointing at which courses or course categories are most relevant to eventual success in the master's program. The model should then help assess the relevance of courses that international students cite on their applications by allowing the decision officer to relate it to a known course category.

Here is what the six-semester undergraduate program looks like at the target university: the courses of the first and second semesters are fixed, with an exam for each course at the end of the second semester. The average grade determines whether the first two semesters are passed or failed. In the third and fourth semesters, the courses are also fixed, but each of them must be passed independently. During the first year, if more than 8 credit points worth of courses are failed, the whole year must be repeated; otherwise, any failed courses must be repeated during the fifth or sixth semester. During these last two semesters of the undergraduate program, available courses (together with their credit points) are listed by topics, which correspond to the 7 specializationsand can be chosen freely, given that the selection fulfills some basic constraints (whose details are not relevant for this application). A semester thesis or an internship are part of the third-year program. Successful completion of the undergraduate program is determined by the obtention of a fixed number of credit points, including the semester thesis.

One important and central issue to keep in mind is that there is no one-to-one mapping for courses in two different universities. This is because the number of courses per year, the course contents and the degree of involvement required of students all vary considerably. If we observe capabilities at a coarser level, we get a more uniform view: in an undergraduate computer-science program, students will probably learn everywhere about programming languages and software design, computer architecture and digital design, theoretical aspects of computer science, the basics of calculus and physics, etc. If we consider course *categories* instead of courses, we can obtain an average grade which is more representative of the general type of skills required in certain courses, is more easily comparable, and smooths out differences among the instructors' grading policies.

We now move on to detailing how to build a model that learns the proposed course groups.

6.2.3 A Causal Model for Course Relevance

We expect to have problems building a perfect-map causal model to predict graduate success with the data described in Subsection 6.2.2. Arguably, no grade can actually *cause* another grade—except for average grades, which can effectively be seen as a deterministic effect of the grades they average. Instead, grades are manifestations of the capabilities of a student in the topics treated in the corresponding course. It is thus hardly possible to build a graph involving only grades and claim it is causal.

The actual simplified causal process could roughly be represented by a graph like the one shown in Figure 6.6. Before a study year, a student has various knowledge and capabilities, which are updated by the courses taken and by external factors. Performance in each course is then represented by a grade that depends on the updated capabilities (possibly also with knowledge gained

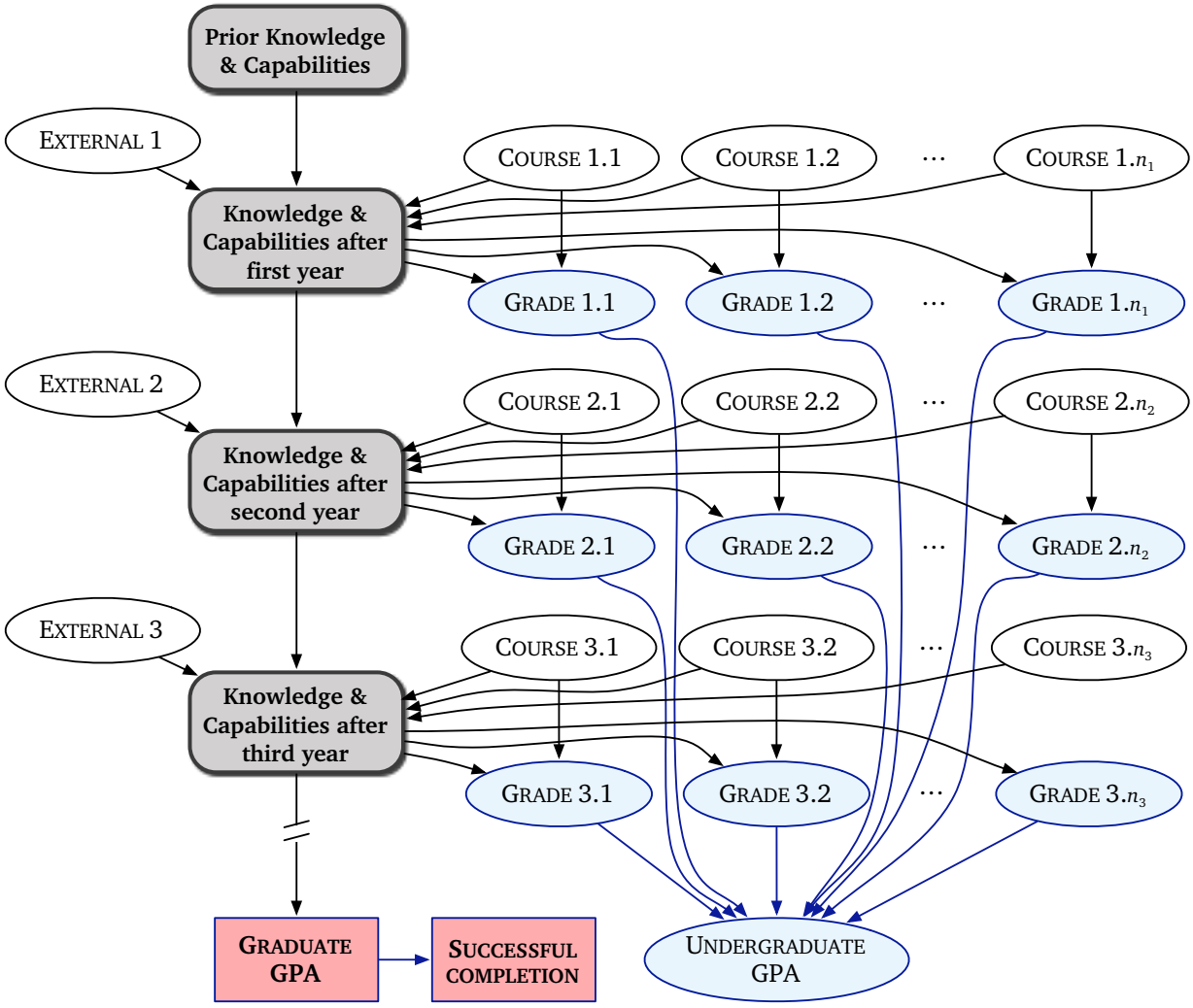


Figure 6.6: A plausible simplified causal model for grade obtention in an undergraduate program. Measurable variables are in blue, hidden variables or submodels are in white and gray, respectively. The two target variables are in red; deterministic links are shown in blue. Capabilities are shown as a rounded rectangle, denoting an unknown subgraph of variables.

from other simultaneous courses or other external factors) and on the course taken. This process is repeated every year till graduation. The only measurable variables are the grades, which are finally summarized in the GPA. It is very difficult to represent and to quantify the knowledge and capabilities themselves: these are indirectly measured by grades instead. Measurable variables are shown in blue in Figure 6.6. Eventually, what we would like to know is whether the knowledge and capabilities at the end of the undergraduate program are sufficient for passing all graduate courses and activities that lead to the graduate GPA, which in turn determines whether the student successfully completes the graduate program. Depending on the approach, we might wish to predict either a Boolean SUCCESSFUL COMPLETION of the program, or a continuous variable representing the GRADUATE GPA. The latter would be more useful to reveal borderline cases close to the passing grade.

Let us note that this is a vastly simplified model. To keep the graph readable, we assume that taking one course has no influence on another course taken at the same time; we also assume that previous grades do not influence the choice of courses in the third year. Both of these assumptions are probably wrong, but at this stage, finer-grained modeling would be mostly between hidden variables, which does not help gain much insight as far as the observed variables are concerned.

If we apply what we know from Section 4.1 and remove from the graph all hidden variables, we can build the structure of the PDAG that an algorithm like PC would construct, shown in Figure 6.7. This is almost a fully connected graph, and is thus not very informative in terms of the dependency structure: in particular, each grade is connected to every other grade.

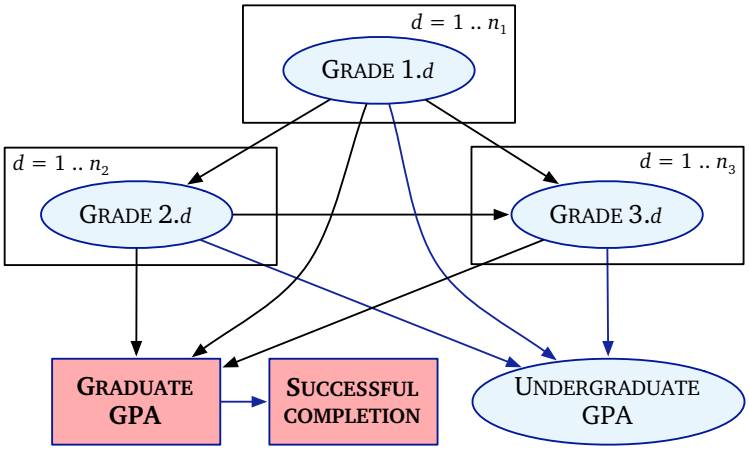


Figure 6.7: The graphical structure of Figure 6.6 if learned without the hidden variables. Boxes are used to avoid repetitions in the graph, as without them the almost-fully connected graph would be difficult to read. Links connected to boxed variables are actually connected to all variables represented by the box. Omitted are, for each box, (undirected) links between all pairs of variables that the box represents. Deterministic links are shown in blue.

The expected graph changes if we refine our model of the knowledge and capabilities of the students (shown as gray rounded rectangles in Figure 6.6). It seems reasonable to postulate that different courses emphasize different kind of capabilities: for instance, a course in formal logic may require capabilities similar to those required by a course on theorem proving and different from those required by software architecture. Let us then assume that we can roughly divide a student's knowledge and capabilities into k "subcapability groups," as shown in Figure 6.8. We model overlapping between the individual capabilities by allowing links between them. We

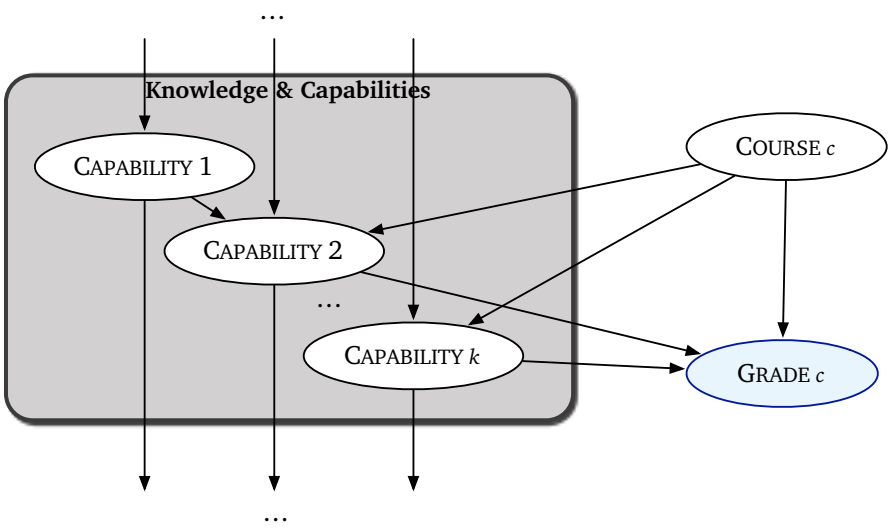


Figure 6.8: A refined version of the modeling of knowledge and capabilities in Figure 6.6.

assume that any given course can be mapped to a subset of these capabilities, which will in turn influence the grade obtained. We still assume, according to Figure 6.6, that the capabilities at the end of the undergraduate program are the last modeled variables which (eventually) causally influence the graduate GPA and which also causally influence some of our observed variables.

In our case study, we have a dataset of 144 students, where each data point describes the grades obtained during the undergraduate studies and the graduate GPA. We first tried running structure-learning algorithms described in Chapters 3 and 4. In the event, these algorithms turn out to be rather useless, as in this dataset, it occurs pretty frequently that two subsets of grades $\mathbf{G}$ and $\mathbf{G}'$ make each other independent of the target graduate GPA $G^\star$; i.e., we have:

$$\left(\mathbf{G} \perp\!\!\!\perp G^\star \,\middle|\, \mathbf{G}'\right) \quad \text{and} \quad \left(\mathbf{G}' \perp\!\!\!\perp G^\star \,\middle|\, \mathbf{G}\right),$$

as returned by the test of zero partial correlation (see Subsection 3.3.2). This can be a symptom for several reasons:

- The data-generating process that we are trying to understand cannot be represented by a DAG: it may for instance contain cycles, i.e., causal feedback loops;
- The joint distribution may not be DAG-isomorphic: faithfulness can be violated, and $G^\star$ may have several equivalent Markov blankets (Statnikov, 2008);
- The z-test of zero partial correlation may not be an appropriate measure of conditional independence, as it is known to be equivalent only in certain cases, including the case where the joint distribution is multivariate Gaussian (Baba et al., 2004);
- The dataset that we have is not large enough to enable the test to find enough evidence to refute the null hypothesis of independence (Tsamardinos et al., 2006).

We do not believe that the problem contains feedback loops, as we have "unrolled" the variables according to the times in the studies at which they are generated, and we have already ruled out direct causation between the observed variables. Lack of faithfulness can occur, but is untestable unless we know the true joint distribution. As for the chosen test of conditional independence, we found the same problem with Margaritis's (2005) Recursive Median test or a χ^2 test with

discretized data. Our belief is that there is not enough data available to make out the subtle influences of each of the grades on the final GPA; conditioning on large subsets is especially likely to render any statistical test useless because of their decreasing power as the size of the conditioning set grows (see Section 3.2).

In order to extract meaningful information from this data, we turned to more local independence properties, following our qualitative graphs. For instance, observing that the grades G_1 and G_2 for two courses are independent ($G_1 \perp\!\!\!\perp G_2$) means, assuming our graph is a perfect causal map, that they were caused by two non-overlapping subsets of capabilities that did not influence each other in the way that CAPABILITY 1 influences CAPABILITY 2 in Figure 6.8. Similarly, if a grade G_1 is independent of the graduate GPA $G^\star$ ($G_1 \perp\!\!\!\perp G^\star$), then the graduate GPA is determined by none of the capabilities that influence G_1.

In our case study, no two grades turn out to be independent, including the graduate GPA: unconditionally, everything is significantly correlated with everything else, as could have been expected. So instead of looking at the Boolean output of the conditional independence tests, we looked at the magnitude of the quantity used as input for the tests: in our case, the partial correlation.

What we are interested in for the creation of the course groups is how similarly two courses influence the graduate GPA, as this allow us to better understand the relevance of similar courses in international students' applications. We can thus build a distance measure between two courses as follows.

Take the mutual information $I(G_1; G^\star \mid G_2)$ between course a course G_1 and the GPA $G^\star$ given G_2. This measures the association between G_1 and $G^\star$, with the effect of G_2 removed. If we assume that G_2 shares full information with the capabilities that it is linked to (according to the model in Figure 6.8), then $I(G_1; G^\star \mid G_2)$ measures how G_1 affects $G^\star$ outside of the capabilities affected by G_2. If G_1 and G_2 are perfectly redundant, then $I(G_1; G^\star \mid G_2) = 0$; if, on the contrary, G_2 has nothing to do with G_1, then $I(G_1; G^\star \mid G_2) = I(G_1; G^\star)$, as information theory tells us. The intermediate values can be seen as quantifying the *independent causal contribution* of G_1 and G_2.

There is one hidden assumption that we need to verify to be able to claim that $I(G_1; G^\star \mid G_2)$ measures the causal contribution from G_1 to $G^\star$ that is independent of the contribution of G_2. Indeed, in general, nothing prevents a quantity like $I(G_1; G^\star \mid G_2)$ from being larger in magnitude than the unconditional mutual information $I(G_1; G^\star)$. A simple scenario illustrating this is the structure $G_1 \to G_2 \leftarrow G^\star$, where, for a linear model, we expect $I(G_1; G^\star \mid G_2) > I(G_1; G^\star) = 0$. In our case, we have a linear model where we can argue that all nonzero coefficients represented by the arrows in Figures 6.6 and 6.8 are positive: we normally expect capabilities to be reinforced by the courses taken over the years, and, likewise, the grades to be positively influenced with the increased capabilities. This, together with the knowledge that by construction, G_2 is never a collider for G_1 and $G^\star$, allows us to call this model *monotone DAG-faithful* (Cheng & Bell, 1997). In short, this property implies that the (conditional) mutual information between a pair of variables is a monotonic function of the set of active paths between those variables: the more active paths between the variables, the higher the mutual information. This property holds for linear models with all coefficients positive, but it is known that it does not hold in general for arbitrary parametrizations of the model (Chickering & Meek, 2006).

We thus built a distance measure based on the partial correlations of the type $I(G_1; G^\star \mid G_2)$, normalizing it with its maximum value $I(G_1; G^\star)$. As we deal with continuous variables, those quantities may be difficult to obtain, so we used, postulating a linear model, the correlation $\rho_{G_1 G^\star}$

instead of $I(G_1; G^\star)$, and the partial correction $\rho_{G_1 G^\star \cdot G_2}$ instead of $I(G_1; G^\star \mid G_2)$.[20] However, as the proposed expression is not symmetrical with respect to G_1 and G_2, we defined the distance between two grades G_1, G_2 as the average of the two independent contributions of G_1 and G_2:

$$\operatorname{dist}(G_1, G_2) = \frac{1}{2}\left(\frac{\rho_{G_1 G^\star \cdot G_2}}{\rho_{G_1 G^\star}} + \frac{\rho_{G_2 G^\star \cdot G_1}}{\rho_{G_2 G^\star}}\right). \tag{6.3}$$

This yields the *causal-influence distance matrix* for the grades: $\mathbf{D} = \big(\operatorname{dist}(G_i, G_j)\big)$. Given this distance matrix, we wish to examine and compare two ways of visualizing the information it contains: with a dendrogram representing a hierarchical cluster tree (see, e.g., Hastie et al., 2009, pp. 520–528), and with a scatter plot after multidimensional scaling (Cox & Cox, 2000).

Cluster trees are binary trees where the leaves are the clustered objects, and the branches link together objects or subclusters of objects. One of the advantages of cluster trees is that constructing them does not require as input the number of wanted clusters, as, for instance, k-means does (MacQueen, 1967): the tree allows the reading off of clusters at several granularity levels. Figure 6.9 shows a sample cluster tree, where objects A, B, C, D, and E appear. The tree can be read as showing two clusters $\{A, B\}$ and $\{C, D, E\}$ if we use a distance of 0.7 as cut-off threshold; alternatively, it can be read as grouping only D and E together, and assigning the remaining objects to clusters of their own with a cut-off threshold of 0.2. The inner nodes are positioned so that their y-coordinates reflect the distance between the two subclusters and the tree can be represented as a so-called *dendrogram*, which makes it easier to visualize the clusters at a given cut-off threshold.

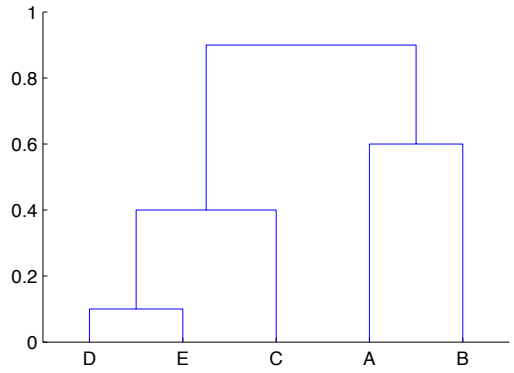


Figure 6.9: A sample hierarchical cluster tree represented as a dendrogram. The y-coordinate of a branch reflects the distance between its two subclusters.

Such trees can be constructed easily by iteratively grouping objects and subclusters, starting with the leaves and finishing with the root, in such a way that elements are at a smaller distance from elements of the same cluster than from elements of other clusters. The distance $\operatorname{dist}(A, B)$ between two objects A and B is directly given by the distance matrix provided as input, and the distance $\operatorname{dist}(\mathbf{A}, \mathbf{B})$ between two subclusters $\mathbf{A}, \mathbf{B}$ (two disjoint subsets of objects) can be defined in a variety of ways. The most popular ways are *single linkage*, *complete linkage*, and *group average*, in which inter-cluster distance is defined, respectively, as the minimum, maximum, and average distance over all object pairs taken from the two clusters (Hastie et al., 2009). Other cluster distances, such as Ward's (1963) criterion, can also be used when the input matrix represents the Euclidean distance. But as the distance we defined in (6.3) is non-Euclidean, we cannot use them.

[20] In our dataset, all these (partial) correlations were positive, so we can omit taking the absolute value.

The distance matrix **D** can alternatively be visualized in a scatter plot after multidimensional scaling (MDS). MDS regroups a set of techniques to assign a location in a d-dimensional space to each object given a distance matrix (see, e.g., Cox & Cox, 2000, for details on the various approaches to MDS). If $d = 2$ or 3, the result can be visualized in a scatter plot, and can give more information about the structure of the data, as it is easy to empirically estimate the distance between any two courses on the plot. Doing so from the dendrogram is not convenient, as only inter-cluster distances are directly represented.

The scatter plot allows an interesting follow-up analysis. The distance between courses represents how similarly they influence the GPA; i.e., how similar the capabilities linked with each course are. Each (x,y)-coordinate on the scatter plot can thus be interpreted as representing the knowledge and capabilities associated with the nearby courses. The whole plot can be seen as representing a two-dimensional *continuum of capabilities*, to which we can assign a causal-relevance score as well. A possibility to do so is to solve for each point (x,y) a regression problem for the graduate GPA, using as predictors noisy course grades, where the amount of noise added to a grade G is proportional to the distance between the corresponding course on the scatter plot and the point (x,y). The causal relevance of a point (x,y) is then higher as the residuals of the regression are lower; the resulting scores can be visualized in a contour plot or heat map.

Other distance measures could have been used instead of (6.3), but we ruled out, in particular, the following ones (and their derivatives):

- Distances based on the mutual information between two grades. Two grades may be strongly associated, but this says nothing of their relation to the graduate GPA, our target variable;
- Distances based on the conditional mutual information of the grades, given the graduate GPA. Although this factors in the graduate GPA, what this measures is the mutual information between the grades once we have removed from them the part that is associated with the graduate GPA. This does not suit us as we are interested in the part of the association that is shared with graduate GPA instead.

In essence, with our distance (6.3) we are trying to approximate the average fraction of the causal-information flow from some course to the graduate GPA that is independent of the causal information flow from another course (the causal-information flow was discussed in Subsection 5.3.1)—which is not per se computable given the hidden variables in the causal model.

In the next subsection, we present the results of experiments run to obtain the causal clustering.

6.2.4 Experiments and Discussion

We applied the causal-clustering technique to an early dataset comprising data about undergraduate grades and graduate GPA at the target university. The dataset contains information about the 21 undergraduate courses from the first four semesters (Calculus I and II were merged) and 7 course groups from the fifth and sixth semesters (see Table 6.6), plus the graduate GPA, for 144 students.

One issue with the optional courses is data missing for all courses that students did not take. Moreover, only one average grade for the whole course *group* is available, not for individual courses. So, for this experiment, in order to keep all courses comparable and obtain baseline results for the clustering, we left out the grading information about the course groups as focused

Cluster 1:	**Cluster 2:**	**Cluster 3:**
Programming	Logic	Calculus
Operating Systems	Linear Algebra	Physics
Data Structures & Algorithms	Digital Design	Electronics & Communication
Computer Networks	Information Theory	
Software Architecture	Theoretical Computer Science	**Cluster 4:**
Databases	Computational Science	Systems Programming
Discrete Mathematics	Formal Meth. & Funct. Progr.	Computer Architecture
Probability & Statistics	Scientific Computing	

Table 6.7: The four clusters obtained from the linkage shown in Figure 6.10 with a cut-off threshold $t = 0.8$. For $t = 0.7$, Clusters 1 and 2 would both be split into two, where each two subclusters would comprise the four upper and lower courses of Clusters 1 and 2, respectively.

on the 21 fully comparable mandatory courses instead. (Work is being carried out to obtain a larger dataset and address the issues of the optional courses.)

We followed (6.3) to compute the distance matrix, and then used complete linkage to construct the cluster tree. Single and average linkages were also tried, but exhibited the tendency to produce "chain" trees (Seifoddini & Djassemi, 1991). Complete linkage also tends to produce more compact clusters, although the built clusters can sometimes violate the "closeness" property and thus contain members that can be closer to members of other clusters than they are to some members of their own cluster (Hastie et al., 2009). We also used an MDS algorithm to create a scatter plot of the courses according to the distance matrix using Sammon's (1969) method. Finally, we computed the causal relevance of the continuum of capabilities represented by the scatter plot by solving a regression problem for each point in a 50×50 discrete grid over the area defined by the MDS. In order to average out the variations in the random noise added to the distant grades, each regression problem was solved 20 times, and the resulting scores in the 50×50 matrix were smoothed with simple a 5×5 mean filter.

The results are shown in Figure 6.10 as a dendrogram, and the MDS scatter plot is shown in Figure 6.11. The four different point shapes represent the four clusters that we can build from the linkage with a cut-off threshold $t = 0.8$, as listed vertically by cluster in Table 6.7. The heat map representing causal relevance of the course categories is shown in Figure 6.12.

Let us recall that the courses are clustered according to a distance measuring how similarly they influence the graduate GPA, and not according to how predictive they are of the GPA—their predictive power only shows up in the heat map. Following our causal reasoning in the previous subsection, two courses are thus grouped together (in Figure 6.10) or close to one another (in Figure 6.11) if they influence similar kinds of knowledge and capabilities. In the subsequent discussion, "close" should thus be read as "having a similar causal way of influencing the graduate GPA."

Choosing a cut-off threshold $t = 0.8$ in the cluster tree is arbitrary: it was chosen because it led to a reasonable number of clusters. We could instead have favored a threshold t around 0.72, which would maximize δ such that all thresholds $t' \in [t - \delta, t + \delta]$ would yield the same number of clusters; namely, in our case, 6 (excepting the less interesting two-cluster solution around $t = 0.9$). Doing so would split the first two clusters shown in Table 6.7 into two clusters of four courses each; the first subclusters would group the first four courses of Clusters 1 and 2, and the second subclusters would group the last four courses of Clusters 1 and 2, respectively.

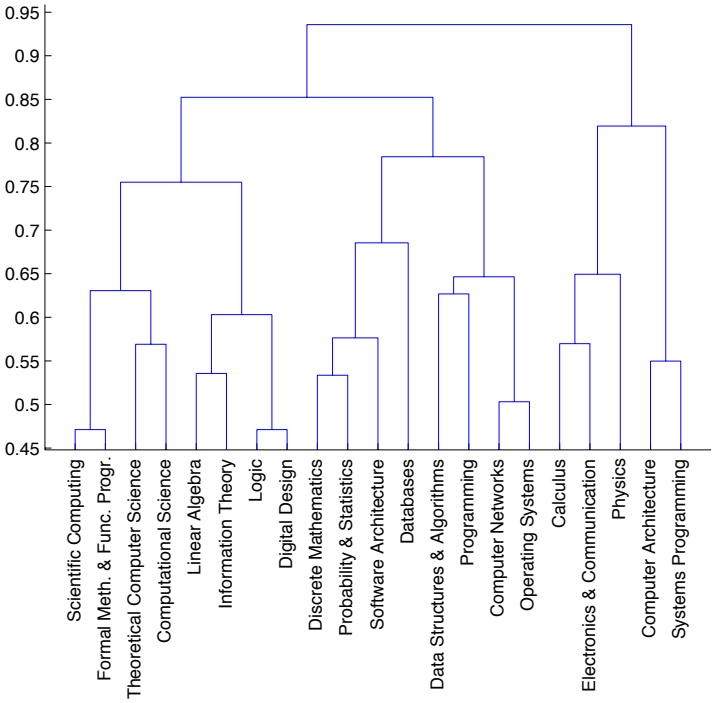


Figure 6.10: The dendrogram representing the course cluster tree obtained with complete linkage over the distance defined by (6.3).

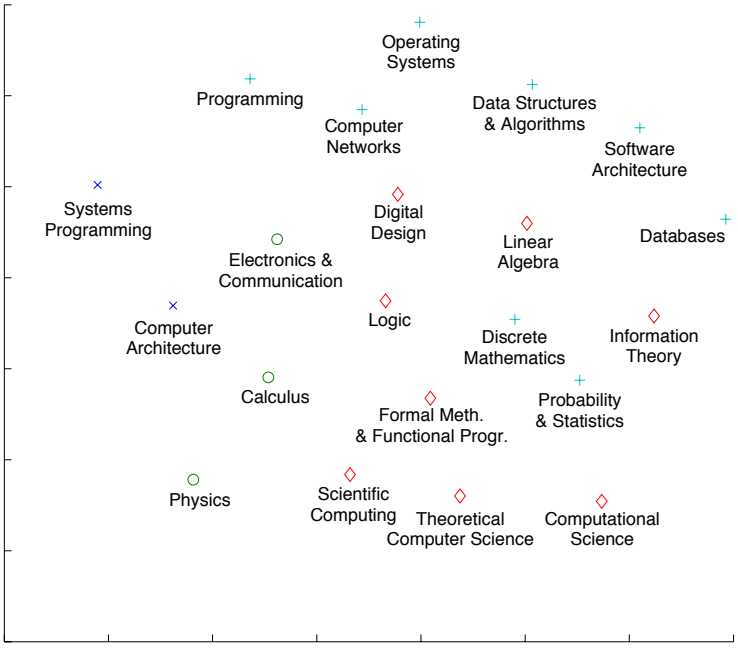


Figure 6.11: Scatter plot of the courses after applying Sammon's MDS method to the distance matrix defined by (6.3).

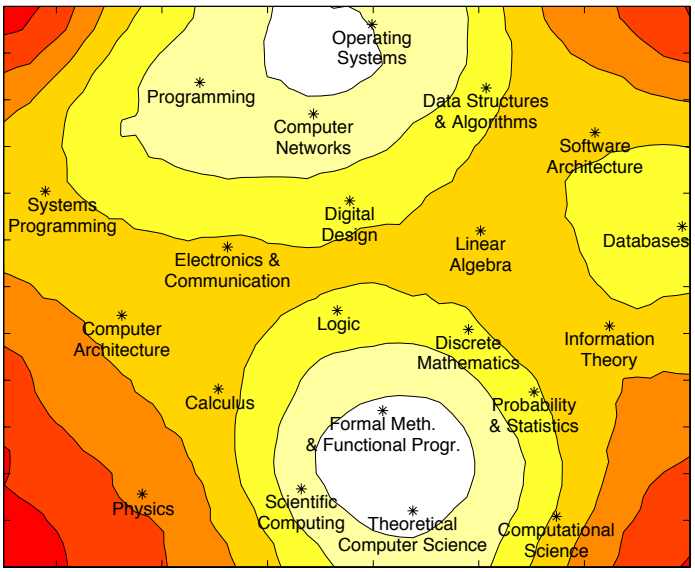


Figure 6.12: Heat map showing causal relevance of the capability regions defined by the MDS shown in Figure 6.11. Brighter colors denote lower regression residuals and thus greater causal relevance.

According to the clustering in Table 6.7 and the scatter plot in Figure 6.11, Physics and Calculus are close, which makes sense: a large part of the Physics course applies techniques and formulas learned in the Calculus course. Electronics & Communication is also in the same cluster, and the contents of the course can indeed easily be related to Calculus methods.

Cluster 1, if anything, could be called the "geek programmer cluster" as it contains many courses related to programming and to low-level hardware, especially the first four courses. Although Discrete Mathematics and Probability & Statistics belong to Cluster 1, they appear in the middle of Cluster 2 in Figure 6.11. The explanation of this is twofold: first, as mentioned before, complete linkage can break the closeness property and thus include members that can be closer to members of other clusters; second, a two-dimensional MDS might not be a good fit for our data. We tried a three-dimensional MDS where, indeed, Clusters 1 and 2 could be represented in a way such that their members could have been linearly separated. Notice that Formal Methods & Functional Programming, for instance, is not closely related to Programming, but is much closer to Logic and Theoretical Computer Science, which makes sense—functional programming is a programming paradigm different enough from the more traditional imperative programming to be assumed to require quite different capabilities.

One could also be surprised to find Computer Architecture and Systems Programming alone in Cluster 4. Computer Architecture, especially, could have been expected to be more closely related to Digital Design.

In Cluster 2, we find many of the more theoretical or abstract courses, like Logic, Theoretical Computer Science, Formal Methods & Functional Programming, or Information Theory. The contents of Scientific Computing build upon the bases acquired in the Computational Science course, so we do indeed expect them to be close.

Let us note how can read more information off the scatter plot than off the dendrogram: Electronics & Communication, Digital Design, and Computer Networks, for instance, are close to one another in Figure 6.11 but do not appear close in Figure 6.10. Another obvious example is the

pair Probability & Statistics and Information Theory: the latter strongly bases its principles on probability theory. They are correspondingly close on the scatter plot, but not on the dendrogram.

The heat map in Figure 6.12 is the only part of this analysis concerned with providing a measure of the association strength between the courses and GRADUATE GPA. As explained before, low regression residuals (represented with bright colors) result from a better predictive power of the nearby courses with respect to the graduate GPA, and thus by extension, a better causal relevance of the knowledge and capabilities they are associated with. This heat map shows three hot zones, which give superior causal relevance to the capabilities related to (i) Operating Systems, Computer Networks, and Programming; (ii) Theoretical Computer Science, Formal Methods & Functional Programming, and Scientific Computing; and (iii) (less intensely so) Databases. These courses all belong to the first two clusters. Although the other courses seem to address less important capabilities, let us note that all courses are positively correlated with the graduate GPA: darker zones in Figure 6.12 thus do not denote unimportant capabilities or irrelevant courses, but capabilities and courses that have a less strong association than those in brighter zones.

When more data becomes available, such heat maps can be generated for each specialization area of the graduate program, and the differences in the causal relevance of the capabilities can be compared across specializations. Eventually, this may help to better handle the application of international students in two ways: first, the relevance of unknown courses can be empirically assessed by relating them to similar courses on the heat map in Figure 6.12; second, specialization-specific heat maps can be used to determine the specializations where applicants are most likely to succeed, given their past performance. Methods to do so are currently being investigated.

6.2.5 Conclusion

We believe that we have found a convincing way to cluster courses according to their causal relevance and to model the underlying knowledge and capabilities. The obtained clustering, scatter plot, and heat map yield reasonable and interpretable results as provided by the trained model with information on 144 students. The information gathered from this analysis lays the foundation for more detailed kinds of both causal and predictive analyses.

On the causal side, we can use the results from the multidimensional scaling and clustering to try to explicitly model the hidden variables representing the knowledge and capabilities, shown in Figure 6.8. Several aspects must be determined: the number of distinctive variables to introduce in order to model these capabilities, their inter-influences (if any), and their relationships to the courses. Goodness-of-fit measures on MDS for various target dimensions could help determine the most appropriate number of hidden variables.

On the predictive side, the obtained clustering can be used to determine the predictive power not of individual courses, but of the modeled capabilities towards the final GPA. Eventually, this helps to assess the capabilities of the international students applying for the graduate program, especially in conditions where other courses with different syllabi are mentioned in the application. A more fine-grained analysis than a simple thresholding on the undergraduate GPA is then possible. A differentiated analysis on a specialization basis is also feasible: as the different specializations require different capabilities, international students with known capabilities can for instance be oriented towards the most suitable specialization, or discouraged from choosing one that obviously is not a good match. (We note that this does not help solve the problem of the very different grading systems that are used in various universities.)

It may be worthwhile to underline the parts of the proposed causal clustering which make it more of an ad hoc technique than a generally applicable methodology. The most important liberty we took was using the grade as a proxy for the capabilities it is related to. Causally, we treated the hidden capabilities and the grade as the same variable, instead of seeing the grade as an effect: this means that we neglected the effect of the course on the grade, as shown by the corresponding arrows in Figures 6.6 and 6.7 (this effect models the course-specific positive or negative bias given to the grade). Then, we used partial correlation to measure conditional association: this assumes a linear model with Gaussian noise. Finally, we only compared pairwise information while comparing courses, neglecting the higher-order effects that could, in principle, occur, if we conditioned on larger sets of variables.

Application Summary

In this application, we have examined a seemingly purely predictive task: given a matrix of predictors composed of personal and academic information, how to best predict success or failure in a graduate program. Instead of building a predictive model, we have postulated a causal model with hidden variables representing the knowledge and capabilities of the students. We have defined a distance measure for courses, based on how similarly they influenced various kinds of capabilities leading to graduation, and have used this distance measure to cluster courses using both a linkage method and multidimensional scaling. Using historical student-performance data, we have found convincing course groups.

CHAPTER

7

Conclusion

CAUSALITY IS THE FUNDAMENTAL TOOL we as humans use to interact with the rest of the world, and our very understanding of cause–effect relationships makes us more than observers in our environment. We can also intervene on it to change the way things work, and use our causal knowledge to predict the effect of our interventions. Whereas we would use traditional statistics for predicting what happens in a data-generating system based on historical data, only causal assumptions make it possible for us to determine what would happen if we actually intervened on key parts of this system. Cause–effect relationships represent the manipulation-invariant properties of the data-generating system, the ones that we know will still be valid after an intervention.

In order for causal analysis to make its way into the family of widespread tools that analysts understand and can use appropriately, we need adequate notation for causal assumptions and causal knowledge, algebraical tools for reasoning with causal models, and efficient algorithms to deal with the various problems pertaining to causal analysis. The graphical notation is particularly well-suited to the representation of cause–effect relationships. First, it is intuitive and understandable: lay people have a good chance to see what analysts are working with, to verify their claims, and to validate the model. Second, graphs encode the causal knowledge precisely: they are more than simple drawing tools, but formally represent the structure of the data. They can be read with *d*-separation and interpreted as depicting conditional independence, used as efficient inference tools if interpreted as Bayesian networks, and they serve as the basis for the *do*-calculus, the calculus of intervention that allows the qualitative and quantitative assessment of the effect of an intervention.

In this context, two main obstacles still stand in the way of an easy and practical causal analysis. One the one hand, approaches to the specification of causal models make stringent assumptions about the nature of the data they require: the range of problems where causal models can readily be built is still rather limited. On the other hand, causal-reasoning methods have still not convincingly evolved to the point where the added value of the causal analysis makes the overhead of building the full causal model clearly worthwhile.

In this thesis, we have contributed to making causal-model specification applicable to a wider range of problems through efficient structure-learning algorithms, and have proposed causal-

reasoning tools that are able to explain the state of variables intuitively with the causal model through causal-explanation trees.

⋆ ⋆ ⋆

The most important part in the specification of the causal model is the determination of the causal structure behind the variables of interest, which is the problem known as structure learning or causal-model selection. Structure-learning algorithms working with observational data exist: with conditional-independence tests, we can determine a graphical pattern with which the given data is compatible. However, we cannot fully identify the causal graph because of causal underdetermination: different causal structures can generate statistically equivalent data.

We can distinguish two main contexts in which structure learning can be applied: the causally sufficient case, where we assume that there are no hidden confounders, no hidden causes in the analysis; and the causally insufficient case, where we do not make this assumption. We have contributed algorithms in both contexts.

Lack of scalability (structure learning is NP-hard) and stringent assumptions about the input data, such as discreteness or Gaussianity, are two significant problems with current structure-learning algorithms. We have tried to address these problems with novel approaches that rely on first learning a local neighborhood around each variable, called a Markov blanket, and then making adjustments to transform this intermediate result into a causal graph. Stating the problem like this is beneficial: learning the Markov blankets, in principle, can be done with traditional consistent feature-selection algorithms. We have shown algorithms based on this approach that outperform baselines in execution speed by several orders of magnitude, being at least as accurate—as measured by the structure of the returned causal graph.

Generalizing the method applied in the causally sufficient case to the causally insufficient case is surprisingly easy. Whereas the generalization of other detailed algorithms like PC is not straightforward, the concept of Markov blankets seems to naturally account for hidden causes. We showed how the lack of causal sufficiency changed the rules of causal inference, and explained why learning Markov blankets still works very well in these cases.

Scalability of the proposed approaches is a key factor to their usability. As all provably correct structure-learning algorithms, our method exhibits an exponential worst-case complexity, in situations where all variables depend on all others. However, the more typical case of causal analysis involves graphs where the node degree is limited. If we assume it is constant throughout the network, we can show that algorithms based on our framework have polynomial complexity. This is not just a convenient theoretical result: using this new approach allows us to tackle in a matter of minutes problems that used to take days to solve with traditional algorithms.

In the future, the presented framework can be used to plug in optimized nonlinear feature-selection algorithms that help relax the assumptions on the input data. This has already been done by Zhang & Hyvärinen (2009), for instance, who use nonlinear regression and a kernel-based test of conditional-independence to create a new structure-learning algorithm based on our TC algorithm. The computational overhead is an issue with the proposed kernel-based test: the low number of tests that TC makes to recover the structure is thus directly beneficial.

Currently open problems in causal-structure learning are challenging. They include designing more scalable algorithms whose accuracy degrades gracefully if their assumptions are violated.

For instance, many algorithms deal badly with lack of faithfulness in the data, and return useless results. Usually, the returned graph for the XOR example (described on page 19) is empty: algorithms do not realize that two variables together completely determine the third one, and check for pairwise association instead.

Finding ways to mitigate the causal-underdetermination problem would be another challenging research direction. A simple instance of this problem is determining the causal mechanism between a pair of variables X and Y: what criteria could allow us to assert $X \dashrightarrow Y$ or $Y \dashrightarrow X$? In general, this is impossible to distinguish, but Hoyer et al. (2009) have shown that it can be done for certain distributions. From another standpoint, Sun et al. (2008) have proposed to examine the *shape* of the conditional distributions $\mathrm{P}(X \mid Y)$ and $\mathrm{P}(Y \mid X)$ and choose the "simpler one" as the one representing the causal process. If embedded in new algorithms, such techniques may allow structure-learning algorithms to return fully oriented DAGs rather than PDAGs, and thus to be greatly helpful to causal-model specification.

Other directions are worth investigating. We saw, for instance, how orientation rules that yield conflicting orientations can invalidate the assumption of causal sufficiency, and thus reveal that part of the causal structure is unobserved. The faithfulness assumption could, in principle, be tested as well. A problem where variables that are pairwise independent but where any two variables fully determine the third one (as illustrated by the XOR example) violate faithfulness. This violation could be identified by conditional-independence tests on *subsets* of variables, rather than on single variables only. If we find two conditional-independence statements that violate one of the graphoid axioms, we can mark the problem as unfaithful. Here again, the challenge would be to conduct a low number of tests with small variables sets, while maximizing the likelihood of identifying the unfaithful cases.

Yet another research area in causal-structure learning that we have not mentioned deals with situations where we have extra information on timing between the variables or the samples. For instance, we could know for sure that, in each data point, variable Y is always measured *after* variable X. As a cause must precede its effects in time, we can incorporate this extra constraint in the structure-learning problem. This time ordering could either simplify the task—because fewer tests would be required for a theoretically correct causal graph—or yield interesting structural contradictions, which, again, could show that some of the fundamental assumptions, such as causal sufficiency or faithfulness, do not hold.

Alternatively, X and Y could be time series. This is in contrast with the setting we have had in this thesis, where we were concerned with cross-sectional data, without any explicit time ordering of the samples. Having time series also supports more specialized tests, like tests of *Granger causality* (Granger, 1988). The causal structure of systems with time series has also been represented with the help of graphical models, using tests of Granger causality to build the model rather than conditional-independence tests (Arnold et al., 2007).

⋆ ⋆ ⋆

After structure learning, we turned our attention to causal reasoning. What interesting tasks made possible by a full causal model would clearly show how causal considerations allow a deeper understanding of the data than statistical associations? We examined the problem of evidence explanation: given some observation about the system, we extract information from the causal model to account for this observation by means of other variables. We have seen that traditional

approaches ignore causality altogether; argued that this did not lead to intuitive and interpretable explanations; and proposed a novel approach, causal-explanation trees.

Explanation trees represent, and assess the quantitative relevance of, several good explanations with an intuitive graphical notation. In our causal-explanation trees, relevance is determined by the causal-information flow, a causal version of mutual information. On several sample causal models, we have shown that causal-explanation trees are more intuitive and readable than other approaches proposed in the past, and successful at extracting the relevant information from the causal model.

The increased relevance of causal explanations is not only the result of some improved search algorithm or better use of the underlying probabilities: it is due to the causal model itself. The causal model tells us much more than the joint probability distribution used by other explanation techniques: it makes it possible to simulate interventions and to know how which variables can causally influence the explanandum. In this way, we can extract explanations that do not contradict human intuition about the causal context of the analysis.

In addition to our own examples, causal-explanation trees have also been used by Helldin (2009) and Helldin & Riveiro (2009) to explain anomalies in vessels given maritime traffic data, where the authors found that the causal explanations provided more sensible results than other methods.

Future work on causal explanation should focus on making it possible to handle a broader range of models. Causal-explanation trees as presented here handle discrete variables. This can be extended to continuous variables as well, where evaluating a given causal-information flow is a computationally intensive task, and appropriately partitioning the domains of the variables so as to maximize the causal-information flow is also a challenge.

Other causal-reasoning techniques are worth investigating. The information available in the causal model is only beginning to be used in analyses and can be employed to many purposes. Methods could be invented to construct scenarios in order to reach a desired state under given cost or feasibility constraints; to suggest structural changes so as to reduce the interdependencies of certain processes; to mark variables as causally critical or sensitive for some parts of the network; to propose interventions and manipulations whose measured outcome would make the model more robust; etc. Causal-reasoning techniques will certainly be a very active research domain, as they are the ones to deliver on the promises of causal analysis and to reap the benefits of the time and energy invested in building the causal model.

⋆ ⋆ ⋆

Finally, in the penultimate chapter, we reported two case studies in which we applied causal analysis. We have seen how, in practice, it is easy to stumble across the practical limits of algorithms, as the whole causal-analysis domain is still relatively young. We have shown how to use specialized techniques to solve the problems that arose in the causal-model specification for these two applications.

The first application was about quality-risk assessment in a pharmaceutical context. We built a causal model with experts and used causal explanations and the *do*-calculus to explore the properties of the pharmaceutical manufacturing process. The second application parted from the techniques proposed in the first part of the thesis: we wanted to show what causal modeling could bring in the context of a seemingly purely predictive task. In a causally insufficient context,

we have dealt with modeling and quantifying the knowledge and capabilities of a student using undergraduate performance.

We wish to reiterate that we do not consider these two studies as premier applications of the methods described in the preceding chapters, but rather as a complement, demonstrating that in practice, causal-modeling techniques must be tailored to each particular context.

⋆ ⋆ ⋆

We would like to conclude with a plea to the causal community to continue developing more flexible and more scalable approaches, and to continue investigating the theoretical aspects of causal inference. In particular, the design and validation of new causal algorithms that can tackle real problems is crucial to the effective development of causal analysis. For an overview of a large effort being made in this direction, see the Causality Workbench initiative:[21] a repository for causal problems, proposed solutions, and comparative studies around many aspects of causal problems, including model selection and predicting the effects of interventions. We believe that undertakings such as this will at last give causal analysis the visibility and attention it deserves, and, eventually, the immense practical value it promises.

[21] http://www.causality.inf.ethz.ch/

References

Abramson, **B.**, **J. Brown**, **A. Murphy**, **& R. L. Winkler** (1996): *Hailfinder: A Bayesian system for forecasting severe weather*. International Journal of Forecasting, vol. 12, pp. 57–71.

Agbonlaho, **R. O. & U. J. Offor** (2008): *Predicting success in a Master of Information Science degree programme*. Education for Information, vol. 26 (3–4), pp. 169–190.

Aliferis, **C. F.**, **A. Statnikov**, **I. Tsamardinos**, **S. Mani**, **& X. D. Koutsoukos** (2010*a*): *Local causal and Markov blanket induction for causal discovery and feature selection for classification, Part I: Algorithms and empirical evaluation*. Journal of Machine Learning Research, vol. 11, pp. 171–234.

Aliferis, **C. F.**, **A. Statnikov**, **I. Tsamardinos**, **S. Mani**, **& X. D. Koutsoukos** (2010*b*): *Local causal and Markov blanket induction for causal discovery and feature selection for classification, Part II: Analysis and extensions*. Journal of Machine Learning Research, vol. 11, pp. 235–284.

Aliferis, **C. F.**, **I. Tsamardinos**, **& A. Statnikov** (2003): *HITON, a novel Markov blanket algorithm for optimal variable selection*. In *Proceedings of the 2003 American Medical Informatics Association (AMIA) Annual Symposium*. pp. 21–25.

American Council on Pharmaceutical Education (1997): *Accreditation standards and guidelines for the professional program in pharmacy leading to the Doctor of Pharmacy degree*. ACPE, Chicago.

Andersson, **S.**, **D. Madigan**, **& M. Perlman** (1997): *A characterization of Markov equivalence classes for acyclic digraphs*. Annals of Statistics, vol. 25, pp. 505–541.

Andreassen, **S.**, **R. Hovorka**, **J. Benn**, **K. G. Olesen**, **& E. R. Carson** (1991): *A model-based approach to insulin adjustment*. In *Proceedings of the Third Conference on AI in Medicine*. Springer-Verlag, pp. 239–248.

Armstrong, **W. B.** (2000): *The association among student success in courses, placement test scores, student background data, and instruction grading practices*. Community College Journal of Research and Practice, vol. 24 (8), pp. 681–695.

Arnold, **A.**, **Y. Liu**, **& N. Abe** (2007): *Temporal causal modeling with graphical Granger methods*. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. pp. 66–75.

Ay, **N. & D. Polani** (2006): *Information flows in causal networks*. Tech. rep., Max Planck Institute for Mathematics in the Sciences.

Baba, **K.**, **R. Shibata**, **& M. Sibuya** (2004): *Partial correlation and conditional correlation as measures of conditional independence*. Australian & New Zealand Journal of Statistics, vol. 46 (4).

Bach, **F. R. & M. I. Jordan** (2003): *Learning graphical models with Mercer kernels*. In *Advances in Neural Information Processing Systems 15*.

Barney, **M. & T. McCarty** (2002): *The new Six Sigma: a leader's guide to achieving rapid business improvement and sustainable results*. Pearson Education.

Beeson, **S. A. & G. Kissling** (2001): *Predicting success for baccalaureate graduates on the NCLEX-RN*. Journal of Professional Nursing, vol. 17 (3), pp. 121–127.

Beinlich, **I.**, **H. J. Suermondt**, **R. M. Chavez**, **& G. F. Cooper** (1989): *The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks*. In *Proceedings of the Second European Conference on AI in Medicine*. pp. 247–256.

Binder, **J.**, **D. Koller**, **S. Russell**, **& K. Kanazawa** (1997): *Adaptive probabilistic networks with hidden variables*. Machine Learning, vol. 29.

Bishop, **C. M.** (2006): *Pattern Recognition and Machine Learning*. Springer.

Boyen, **X.**, **N. Friedman**, **& D. Koller** (1999): *Discovering the hidden structure of complex dynamic systems*. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*. pp. 91–100.

Bradford Hill, **A.** (1965): *The environment or disease: association or causation?* Proceedings of the Royal Society of Medicine, vol. 58, pp. 295–300.

Brodie, **M.**, **I. Rish**, **& S. Ma** (2002): *Intelligent probing: A cost-effective approach to fault diagnosis in computer networks*. IBM Systems Journal, vol. 41 (3), pp. 372–385.

Bruges, **C.** (1998): *A tutorial on support vector machines for pattern recognition*. Data Mining and Knowledge Discovery, vol. 2 (2), pp. 121–167.

Cartwright, **N.** (1994): *No causes in, no causes out*. In *Nature's Capacities and Their Measurement*, Chapter 2. Oxford University Press, pp. 39–91.

Chajewska, **U. & J. Y. Halpern** (1997): *Defining explanation in probabilistic systems*. In *Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence (UAI-97)*. Morgan Kaufmann, San Francisco, CA, pp. 62–71.

Chan, **H. & A. Darwiche** (2006): *On the robustness of most probable explanations*. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*. pp. 63–71.

Cheng, **J. & D. Bell** (1997): *Learning Bayesian networks from data: an efficient approach based on information theory*. In *Proceeding of the 6th ACM International Conference on Information and Knowledge Management*.

Chesnut, **R. J. & C. R. Phillips** (2000): *Current practices and anticipated changes in academic and nonacademic admission sources for entry-level pharmd programs*. American Journal of Pharmaceutical Education, vol. 64 (3), pp. 251–259.

Chickering, **D. M.**, **D. Geiger**, **& D. Heckerman** (1994): *Learning Bayesian networks is NP-hard*. Tech. Rep. MSR-TR-94-17, Microsoft Research.

Chickering, **D. M. & C. Meek** (2006): *On the incompatibility of faithfulness and monotone DAG faithfulness*. Artificial Intelligence, vol. 170 (8), pp. 653–666.

Cox, **T. F. & M. A. A. Cox** (2000): *Multidimensional Scaling*. Chapman and Hall, second edn.

de Campos, **L. M.**, **J. A. Gámez**, **& S. Moral** (2001): *Simplifying explanations in Bayesian belief networks*. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 9.

Dìez, **F. J.** (1993): *Parameter adjustment in Bayes networks. the generalized noisy OR-gate*. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA, USA, pp. 99–105.

Downey, **M. C.**, **M. A. Collins**, **& W. D. Browning** (2002): *Predictors of success in dental hygiene education: A six-year review*. Journal of Dental Education, vol. 66 (11), pp. 1269–1273.

Drton, **M. & T. Richardson** (2004): *Iterative conditional fitting for Gaussian ancestral graph models*. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. pp. 130–137.

Dummett, **M.** (1964): *Bringing about the past*. Philosophical Review, vol. 73, pp. 338–59.

Elidan, **G.** (2001): *Bayes net repository*. http://compbio.cs.huji.ac.il/Repository/.

Elidan, **G. & N. Friedman** (2001): *Learning the dimensionality of hidden variables*. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*. pp. 144–151.

Elidan, **G.**, **N. Lotner**, **N. Friedman**, **& D. Koller** (2001): *Discovering hidden variables: A structure-based approach*. In *Proceedings of the 13th Conference on Advances in Neural Information Processing Systems*. pp. 479–485.

Elisseeff, **A.**, **J.-P. Pellet**, **& E. Pratsini** (2010): *Causal networks for risk and compliance: Methodology and application*. IBM Journal of Research and Development, vol. 54 (3).

European Ministers of Education (1999): *The Bologna declaration*.

Evans, **P. & F. K. Wen** (2007): *Does the medical college admission test predict global academic performance in osteopathic medical school?* Journal of the American Osteopathic Association, vol. 107 (4), pp. 157–162.

Fenton, **N. E.**, **M. Neil**, **& J. Gallan** (2007*a*): *Using ranked nodes to model qualitative judgments in Bayesian networks*. IEEE Transactions on Knowledge and Data Engineering, vol. 19 (10), pp. 1420–1432.

Fenton, **N. E.**, **M. Neil**, **W. Marsh**, **P. Hearty**, **D. Marquez**, **P. Krause**, **& R. Mishra** (2007*b*): *Predicting software defects in varying development lifecycles using Bayesian nets*. In *Information & Software Technology*, vol. 49. pp. 32–43.

Fenton, **N. E.**, **M. Neil**, **W. Marsh**, **P. Hearty**, **L. Radlinski**, **& P. Krause** (2008): *On the effectiveness of early-life cycle defect prediction with Bayesian nets*. In *Empirical Software Engineering*. pp. 499–537.

Fine, **S. & A. Ziv** (2003): *Coverage directed test generation for functional verification using Bayesian networks*. In *Proceedings of the 40th conference on Design automation - DAC*. pp. 286–291.

Flores, **M. J.** (2005): *Bayesian networks Inference: Advanced algorithms for triangulation and partial abduction*. Ph.D. thesis, Universidad De Castilla-La Mancha.

Frey, **B. J.**, **F. R. Kschischang**, **& H.-A. Loeliger** (2001): *Factor graphs and the sum-product algorithm*. IEEE Transactions on Information Theory, vol. 47.

Friedman, **N.** (1998): *The Bayesian structural EM algorithm*. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*. pp. 129–138.

Friedman, **N.**, **M. Linial**, **I. Nachman**, **& D. Pe'er** (2000): *Using Bayesian networks to analyze expression data*. In *RECOMB*. pp. 127–135.

Fu, **L. D.** (2005): *A comparison of state-of-the-art algorithms for learning Bayesian network structures from continuous data*. Master's thesis, Vanderbilt University.

Gall, **M. D.**, **W. R. Borg**, **& J. P. Gall** (2006): *Educational Research: An Introduction*. Allyn & Bacon, eighth edn.

Galles, **D. & J. Pearl** (1995): *Testing identifiability of causal effects*. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*. pp. 185–190.

Geiger, **D.** (1987): *The non-axiomatizability of dependencies in directed acyclic graphs*. Tech. Rep. R-83, UCLA Cognitive Systems Laboratory.

Geisser, **S.** (1993): *Predictive Inference*. Chapman and Hall.

Glymour, **C.** (2001): *The Mind's Arrows*. MIT Press.

Glymour, **C. & G. F. Cooper** (1999): *Computation, Causation, & Discovery*. AAAI Press/MIT Press.

Granger, **C. W. J.** (1988): *Some recent developments in a concept of causality*. Journal of Econometrics, vol. 39 (1), pp. 199–211.

Guyon, **I.**, **C. Aliferis**, **G. Cooper**, **A. Elisseeff**, **J.-P. Pellet**, **P. Spirtes**, **& A. Statnikov** (2008): *Design and analysis of the causation and prediction challenge*. In *JMLR Workshop and Conference Proceedings*, vol. 3: WCCI 2008 causality challenge. Hong Kong.

Guyon, **I.**, **C. Aliferis**, **& A. Elisseeff** (2007): *Causal feature selection*. In H. Liu & H. Motoda (eds.), *Computational Methods of Feature Selection*. Chapman and Hall/CRC Press.

Guyon, **I. & A. Elisseeff** (2003): *An introduction to variable and feature selection*. Journal of Machine Learning Research, vol. 3, pp. 1157–1182.

Guyon, **I.**, **J. Weston**, **S. Barnhill**, **& V. Vapnik** (2002): *Gene selection for cancer classification using support vector machines*. Machine Learning, vol. 46 (1-3), pp. 389–422.

Hagmayer, **Y.**, **S. A. Sloman**, **D. A. Lagnado**, **& M. R. Waldmann** (2005): *Causal reasoning through intervention*. Tech. rep., University of Goettingen, Brown University, University College London.

Halpern, **J. Y. & J. Pearl** (2005): *Causes and explanations: A structural-model approach. Part II: Explanations*. The British Journal for the Philosophy of Science.

Hardin, **D.**, **I. Tsamardinos**, **& C. F. Aliferis** (2004): *A theoretical characterization of linear SVM-based feature selection*. In *Proceedings of the Twenty First International Conference on Machine Learning*.

Hastie, **T.**, **R. Tibshirani**, **& J. Friedman** (2009): *The Elements of Statistical Learning*. Springer, second edn.

Hausman, **D. M. & J. Woodward** (1999): *Independence, invariance and the causal Markov condition*. British Journal for the Philosophy of Science, vol. 50, pp. 521–583.

Heckerman, **D.**, **E. J. Horviz**, **& B. N. Nathwani** (1992): *Toward normative expert systems: Part I, the Pathfinder project*. Methods for Information in Medicine, vol. 32, pp. 90–105.

Heckerman, **D.**, **A. Mamdani**, **& M. P. Wellman** (1995): *Real-world applications of Bayesian networks*. Communications of the ACM, vol. 38 (3), pp. 24–26.

Heckerman, **D. & B. N. Nathwani** (1992): *Toward normative expert systems: Part II, probability-based representations for efficient knowledge acquisition and inference*. Methods for Information in Medicine, vol. 31, pp. 106–116.

Heckman, **J. J.** (1979): *Sample selection bias as a specification error*. Econometrica, vol. 47 (1), pp. 153–162.

Helldin, **T.** (2009): *Explanation methods for Bayesian networks*. Master's thesis, University of Skövde.

Helldin, **T. & M. Riveiro** (2009): *Explanation methods for Bayesian networks: Review and application to a maritime scenario*. In *Proceedings of the 3rd Annual Skövde Workshop on Information Fusion Topics*. pp. 28–32.

Henrion, **M. & M. J. Druzdzel** (1990): *Qualitative propagation and scenario-based scheme for exploiting probabilistic reasoning.* In *UAI*. pp. 17–32.

Hoyer, **P. O.**, **D. Janzing**, **J. Mooij**, **J. Peters**, **& B. Schölkopf** (2009): *Nonlinear causal discovery with additive noise models*. In *Advances in Neural Information Processing Systems 21*. pp. 689–696.

Hulten, **G.**, **D. M. Chickering**, **& D. Heckerman** (2003): *Learning Bayesian networks from dependency networks: A preliminary study*. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*.

Ishikawa, **K.** (1990): *Introduction to Quality Control*. Productivity Press.

Jeffreys, **H.** (1961): *Theory of Probability*. Oxford University Press.

Jensen, **F. V.** (2001): *Bayesian Networks and Decision Graphs*. Springer.

John, **G. H.**, **R. Kohavi**, **& K. Pfleger** (1994): *Irrelevant feature and the subset selection problem*. In *Proceedings of the Eleventh International Conference on Machine Learning*. pp. 121–129.

Johnson, **N. T. & R. C. Richardson**, **Jr.** (1986): *A causal model of academic factors affecting student persistence*. In *Annual Meeting of the American Educational Research Association*.

Judge, **G. G.**, **R. C. Hill**, **W. E. Griffiths**, **H. Lütkepohl**, **& T.-C. Lee** (1988): *Introduction to the Theory and Practice of Econometrics, 2nd Edition*. Wiley.

Kraaijeveld, **P. C. & M. J. Druzdzel** (2005): *Genierate: An interactive generator of diagnostic Bayesian network models*. In *16th International Workshop on Principles of Diagnosis*. Monterey, California, USA, pp. 175–180.

Lacave, **C. & F. J. Dìez** (2002): *A review of explanation methods of Bayesian networks*. Knowledge Engineering Review, vol. 17 (2), pp. 107–127.

Lacerda, **G.**, **P. Spirtes**, **J. Ramsey**, **& P. Hoyer** (2008): *Discovering cyclic causal models by independent components analysis*. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence*. pp. 366–374.

Lachin, **J. M.**, **J. P. Matts**, **& L. J. Wei** (1988): *Randomization in clinical trials: Conclusions and recommendations*. Controlled Clinical Trials, vol. 9 (4), pp. 365–74.

Lauritzen, **S. L.**, **A. P. Dawid**, **B. N. Larsen**, **& H.-G. Leimer** (1990): *Independence properties of directed Markov fields*. Networks, vol. 20, pp. 491–505.

Lauritzen, **S. L. & D. J. Spiegelhalter** (1988): *Local computations with probabilities on graphical structures and their application to expert systems*. Journal of the Royal Statistical Society, Series B, vol. 50, pp. 157–224.

Leray, **P. & O. François** (2004): *BNT structure learning package*. banquiseasi.insa-rouen.fr/projects/bnt-slp/.

Lewis, **D.** (1973): *Counterfactuals*. Blackwell Publishers.

Lewis, **D.** (2000): *Causation as influence*. The Journal of Philosophy, vol. 97, pp. 182–197.

Livengood, **J. M.** (1992): *Students' motivational goals and beliefs about effort and ability as they relate to college academic success*. Research in Higher Education, vol. 33 (2).

MacQueen, **J. B.** (1967): *Some methods for classification and analysis of multivariate observations*. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. pp. 281–297.

Margaritis, **D.** (2005): *Distribution-free learning of Bayesian network structure in continuous domains*. In *Proceedings of the 20th National Conference on AI*.

Margaritis, **D. & S. Thrun** (1999): *Bayesian network induction via local neighborhoods*. In *Advances in Neural Information Processing Systems 12*.

Marieb, **E. N. & K. Hoehn** (2006): *Human Anatomy & Physiology*. Pearson Benjamin Cummings, seventh edn.

Marquez, **D.**, **M. Neil**, **& N. E. Fenton** (2008): *Solving dynamic fault trees using a new hybrid Bayesian network inference algorithm*. In *16th Mediterranean Conference on Control and Automation*.

Marsh, **D. W. R. & G. Bearfield** (2008): *Generalising event trees using Bayesian networks with a case study of train derailment*. In *Proceedings of the Institution of Mechanical Engineers. Part O, Journal of Risk and Reliability*, vol. 222. pp. 105–114.

Marsh, **W. & G. Bearfield** (2004): *Using Bayesian networks to model accident causation in the UK railway industry*. In *Proceedings of the 7th International Conference on Probabilistic Safety Assessment and Management*.

Meek, **C.** (1995): *Causal inference and causal explanation with background knowledge*. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*. pp. 403–410.

Middleton, **D. E.** (2008): *A predictive valdity study: correlation of admission variables with program completion and student performance on the National Certification Examination in a physician assistant program*. Ph.D. thesis, University of Southern California.

Morgan, **S. L. & C. Winship** (2007): *Counterfactuals and Causal Inference: Methods and Principles for Social Research*. Cambridge University Press.

Neapolitan, **R. E.** (2003): *Learning Bayesian Networks*. Prentice Hall.

Neil, **M.**, **N. Fenton**, **& L. Nielsen** (2000): *Building large-scale Bayesian networks*. The Knowledge Engineering Review, vol. 15 (3), pp. 257–284.

Neil, **M.**, **N. E. Fenton**, **S. Forey**, **& R. Harris** (2001): *Using Bayesian belief networks to predict the reliability of military vehicles*. IEEE Computing and Control Engineering Journal, vol. 12 (1), pp. 11–20.

Neil, **M.**, **M. Tailor**, **D. Marquez**, **N. E. Fenton**, **& P. Hearty** (2008): *Modelling dependable systems using hybrid Bayesian networks*. Reliability Engineering and System Safety, vol. 93 (7), pp. 933–939.

Nelson, **C. V.**, **J. S. Nelson**, **& B. Malone** (2004): *Predicting success of international graduate students in an American university*. College and University, vol. 80 (1), pp. 19–27.

Nielsen, **U. H.**, **J.-P. Pellet**, **& A. Elisseeff** (2008): *Explanation trees for causal Bayesian networks*. In D. McAllester & P. Myllymäki (eds.), *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence*. AUAI Press, pp. 427–434.

Nilsson, **R.**, **J. M. Peña**, **J. Björkegren**, **& J. Tegnér** (2007): *Consistent feature selection for pattern recognition in polynomial time*. The Journal of Machine Learning Research, vol. 8, pp. 589–612.

Orlando, **J.** (2005): *The reliability of GRE scores in predicting graduate school success: A meta-analytic, cross-functional, regressive, unilateral, post-kantian, hyper-empirical, quadruple blind, verbiage-intensive and hemorrhoid-inducing study*. Ubiquity, vol. 6 (21).

Park, **J. D.** (2002): *Map complexity results and approximation methods*. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence*. pp. 388–396.

Peña, **J. M.**, **J. Björkegren**, **& J. Tegnér** (2005): *Scalable, efficient and correct learning of Markov boundaries under the faithfulness assumption*. In *Proceedings of the Eighth European Conference on Symbolic and Quantitative Approaches to Reasoning under Uncertainty*. pp. 136–147.

Pearl, **J.** (1982): *Reverend Bayes on inference engines: A distributed hierarchical approach*. In A. Press (ed.), *Proceedings of the Second National Conference on Artificial Intelligence*. pp. 133–136.

Pearl, **J.** (1986): *Fusion, propagation, and structuring in belief networks*. Artificial Intelligence, vol. 29 (3), pp. 241–288.

Pearl, **J.** (1988): *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, Los Altos.

Pearl, **J.** (1995): *Causal diagrams for empirical research*. Biometrika, vol. 82 (4), pp. 669–709.

Pearl, **J.** (2000): *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Pearl, **J.** (2008): *Causal inference as computational learning*. Talk at the NIPS 08 workshop on Causality: objectives and assessment.

Pearl, **J. & T. Verma** (1991): *A theory of inferred causation*. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann.

Pellet, **J.-P. & A. Elisseeff** (2007): *A partial correlation-based algorithm for causal structure discovery with continuous variables*. In *7th International Symposium on Intelligent Data Analysis*. pp. 229–239.

Pellet, **J.-P. & A. Elisseeff** (2008*a*): *Finding latent causes in causal networks: an efficient approach based on Markov blankets*. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems*.

Pellet, **J.-P. & A. Elisseeff** (2008*b*): *Using Markov blankets for causal structure learning*. Journal of Machine Learning Research, vol. 9, pp. 1295–1342.

Raveh, **A.** (1985): *On the use of the inverse of the correlation matrix in multivariate data analysis*. The American Statistician, vol. 39, pp. 39–42.

Remus, **W. & C. Wong** (1982): *An evaluation of five models for the admission decision*. College Student Journal, vol. 16 (1), pp. 53–59.

Rice, **J. A.** (1995): *Mathematical Statistics and Data Analysis*. Duxbury Press, second edn.

Richardson, **T.** (1996): *A polynomial-time algorithm for deciding equivalence of directed cyclic graphical models*. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*. pp. 462–469.

Sammon, **J. W., Jr.** (1969): *A nonlinear mapping for data structure analysis*. IEEE Transactions on Computers, vol. 18 (401–409).

Scheines, **R.** (2002): *Computation and causation*. MetaPhilosophy, vol. 33 (1).

Scheines, **R.**, **P. Spirtes**, **C. Glymour**, **C. Meek**, **& T. Richardson** (1995): *The TETRAD project: Constraint based aids to causal model specification*. Tech. rep., Carnegie Mellon University, Dpt. of Philosophy.

Sedlacek, **W. E.** (2003): *Alternative admissions and scholarship selection measures in higher education*. Measurement and Evaluation in Counseling and Development, vol. 35.

Seifoddini, **H. & M. Djassemi** (1991): *The production data-based similarity coefficient verses jaccard's similarity coefficient*. Computers and Industrial Engineering, vol. 21, pp. 263–266.

Sharon, **A. T.** (1972): *English proficiency, verbal aptitude, and foreign student success in american graduate schools*. Educational and Psychological Measurement, vol. 32 (2), pp. 425–431.

Shimodaira, **H.** (2000): *Improving predictive inference under covariate shift by weighting the log-likelihood function*. Journal of Statistical Planning and Inference, vol. 90 (2), pp. 227–244.

Shimony, **S. E.** (1991): *Explanation, irrelevance, and statistical independence*. In *AAAI*. pp. 482–487.

Shpitser, **I.** (2008): *Complete Identification Methods for Causal Inference*. Ph.D. thesis, UCLA Cognitive Systems Laboratory.

Silva, **R.**, **R. Scheines**, **C. Glymour**, **& P. Spirtes** (2006): *Learning the structure of linear latent variable models*. Journal of Machine Learning Research, vol. 7, pp. 191–246.

Simpson, **E. H.** (1951): *The interpretation of interaction in contingency tables*. Journal of the Royal Statistical Society, Ser. B, vol. 13, pp. 238–241.

Smola, **A. J. & B. Schölkopf** (1998): *A tutorial on support vector regression*. Tech. rep., NeuroCOLT2.

Sober, **E.** (1987): *Discussion: Parsimony, likelihood, and the principle of the common cause*. Philosophy of Science, vol. 54, pp. 465–469.

Spirtes, **P.**, **C. Glymour**, **& R. Scheines** (2001): *Causation, Prediction, and Search, Second Edition*. MIT Press, second edn.

Spirtes, **P.**, **C. Meek**, **& T. Richardson** (1995): *Causal inference in the presence of latent variables and selection bias*. In P. Besnard & S. Hanks (eds.), *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA, pp. 491–498.

Spirtes, **P.**, **T. Richardson**, **& C. Meek** (1996): *Heuristic greedy search algorithms for latent variable models*. In *Proceedings of the 6th International Workshop on Artificial Intelligence and Statistics*. pp. 481–488.

Stamelosa, **I.**, **L. Angelisa**, **P. Dimoua**, **& P. Sakellaris** (2005): *On the use of Bayesian belief networks for the prediction of software productivity*. Information and Software Technology, vol. 45 (1), pp. 51–60.

Statnikov, **A.** (2008): *Algorithms for Discovery of Multiple Markov Boundaries: Application to the Molecular Signature Problem*. Ph.D. thesis, Vanderbilt University.

Statnikov, **A.**, **D. Hardin**, **& C. F. Aliferis** (2006): *Using SVM weight-based methods to identify causally relevant and non-causally relevant variables*. Tech. rep., Vanderbilt University, USA.

Steel, **D.** (2005): *Homogeneity, selection, and the faithfulness condition*. Tech. rep., Michigan State University, Department of Philosophy.

Stricker, **G. & J. T. Huber** (1967): *The Graduate Record Examination and undergraduate grades as predictors of success in graduate school*. Journal of Educational Research, vol. 60 (10).

Sun, **X.**, **D. Janzing**, **& B. Schölkopf** (2008): *Causal reasoning by evaluating the complexity of conditional densities with kernel methods*. Neurocomputing, vol. 71 (7–9), pp. 1248–1256.

Talih, **M.** (2003): *Markov Random Fields on Time-Varying Graphs, with an Application to Portfolio Selection*. Ph.D. thesis, Hunter College.

Tibshirani, **R.** (1994): *Regression shrinkage and selection via the lasso*. Tech. rep., University of Toronto.

Tran, **N. L.**, **B. Hasselbalch**, **K. Morgan**, **& G. Claycamp** (2005): *Elicitation of expert knowledge about risks associated with pharmaceutical manufacturing processes*. Pharmaceutical Engineering, pp. 26–38.

Tsamardinos, **I. & C. Aliferis** (2003): *Towards principled feature selection: Relevancy*. Artificial Intelligence and Statistics.

Tsamardinos, **I.**, **C. F. Aliferis**, **& A. Statnikov** (2003): *Time and sample efficient discovery of Markov blankets and direct causal relations*. In A. Press (ed.), *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 673–678.

Tsamardinos, **I.**, **L. E. Brown**, **& C. F. Aliferis** (2006): *The max-min hill-climbing Bayesian network structure learning algorithm*. Machine Learning, vol. 65 (1).

Turoff, **M. & S. R. Hiltz** (1996): *Computer-based Delphi processes*. In M. Adler & E. Ziglio (eds.), *Gazing into the Oracle: The Delphi Method and its Application to Social Policy and Public Health*. Jessica Kingsley Publishers, Amsterdam, pp. 56–85.

U.S. Food and Drug Administration (2002): *Pharmaceutical cGMPs for the 21st century: A risk-based approach*. http://www.fda.gov/oc/guidance/gmp.html.

U.S. Food and Drug Administration (2004): *Risk-based method for prioritizing cGMP inspections of phar-*

maceutical manufacturing sites – a pilot risk ranking model. Tech. rep., Department of Health and Human Services.

Valorta, **M. & Y. Huang** (2006): *Pearl's calculus of intervention is complete*. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*. pp. 437–444.

Verma, **T.** (1993): *Graphical aspects of causal models*. Tech. Rep. R-191, Cognitive Systems Laboratory, UCLA.

Ward, **J. H.** (1963): *Hierarchical grouping to optimize an objective function*. Journal of the American Statistical Association, vol. 58 (301), pp. 236–244.

Ward, **N.** (2006): *Towards a model of computer science graduate admissions decisions*. Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 10 (3), pp. 372–383.

Wilson, **R. L. & B. C. Hardgrave** (1995): *Predicting graduate student success in an MBA program: Regression versus classification*. Educational and Psychological Measurement, vol. 55 (2), pp. 186–195.

Winberley, **D. W.**, **D. G. McCloud**, **& W. L. Finn** (1992): *Predicting success of indonesian graduate students in the united states*. Comparative Education Review, vol. 36 (4), pp. 487–508.

Wolbrecht, **E.**, **B. D'Ambroio**, **& R. P. D. Kirby** (2000): *Monitoring and diagnosis of a multi-stage manufacturing process using Bayesian networks*. Artificial Intelligence for Engineering, Design and Manufacturing, vol. 14, pp. 53–67.

Woodward, **J.** (2003): *Making things happen. A theory of causal explanation*. Oxford University Press.

Wright, **S. S.** (1921): *Correlation and causation*. Journal of Agricultural Research, vol. 20, pp. 557–85.

Yuan, **C. & T.-C. Lu** (2007): *Finding explanations in Bayesian networks*. In *The 18th International Workshop on Principles of Diagnosis*. pp. 414–419.

Yule, **G. U.** (1903): *Notes on the theory of association of attributes in statistics*. Biometrika, vol. 2, pp. 121–134.

Zhang, **J. Q.** (1999): *The correlation of students' entry-level GPA, academic performance, and the National Board Examination in all basic science subjects*. The Journal of Chiropractor Education, vol. 13 (2), pp. 91–99.

Zhang, **K. & A. Hyvärinen** (2009): *Causality discovery with additive disturbances: An information-theoretical perspective*. In *Proceedings of the 2009 European Conference on Machine Learning and Knowledge Discovery in Databases*. pp. 570–585.

Index

Curriculum Vitæ

Name:	Jean-Philippe Pellet
Citizenship:	Swiss
Date of birth:	November 12th, 1982
Place of origin:	St-Livres, VD, Switzerland

Education

1999–2001	High school (Morges, Switzerland)
2001–2003	Bachelor's studies in computer science at EPFL (Lausanne, Switzerland)
2003–2004	Exchange year at TU Berlin, Germany
2004–2006	Master's studies in computer science at EPFL Master's thesis at IBM Zurich Research Laboratory
2007–2010	Ph.D. studies in computer science at ETHZ (Zurich, Switzerland) Working at IBM Zurich Research Laboratory

Experience & Projects

1997–1999	Independent project: NetworkRun Development of a network-administration application
2000–2006	Independent project: QuizMaster Cross-platform e-learning software-solution development
2004	Internship at Condris Technologies (Ecublens, Switzerland) Security analysis of IP-based wireless networks
2005	Internship at CERN (Geneva, Switzerland) Co-development of a distributed process manager
2007–2009	Pre-doc at IBM Zurich Research Laboratory Ph.D. studies & research and development for IBM projects
2010	Research-staff member at IBM Zurich Research Laboratory
2010–present	Founder & software architect and developer at Wizzy Education Technologies SA