A tutorial on Bayesian Networks for psychopathology researchers

Giovanni Briganti
Harvard University
Université libre de Bruxelles

Marco Scutari
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)

Richard J. McNally
Harvard University

Bayesian Networks are probabilistic graphical models that represent conditional independence relationships among variables as a directed acyclic graph (DAG), where edges can be interpreted as causal effects connecting one causal symptom to an effect symptom. These models can help overcome one of the key limitations of partial correlation networks whose edges are undirected. This tutorial aims to introduce Bayesian Networks to identify admissible causal relationships in cross-sectional data, as well as how to estimate these models in R through three algorithm families with an empirical example data set of depressive symptoms. In addition, we discuss common problems and questions related to Bayesian networks. We recommend Bayesian networks be investigated to gain causal insight in psychological data.

Keywords: Bayesian Networks, Directed Acyclic Graphs, network modeling, causal inference, tutorial

This paper was accepted for publication in Psychological Methods

The last decade has seen the emergence of a new framework for the study of psychopathology: network theory, which conceptualizes an episode of disorder as issuing from interactions among its constituent symptoms (Borsboom, 2017; Borsboom & Cramer, 2013). Network theory comes with a set of statistical techniques called network analysis, where the networks that represent interactions among symptoms are composed of nodes (the symptoms themselves) and edges (connections among symptoms). In contrast to social networks, whose nodes (persons) and edges (e.g., friendships) are observable, psychopathology networks require statistical estimation to discern the unobservable connections between symptoms. Most network structures in the psychopathology field encode pairwise interactions among symptoms (Epskamp & Fried, 2018), and use a general class of models called pairwise Markov Random Fields.

Gaussian Graphical Models (GGMs), also called partial correlation networks, are the most common way of estimating network structures in many areas of psychopathology, such as posttraumatic stress disorder (Fried et al., 2018; McNally et al., 2015), depression (Briganti, Scutari, et al., 2021; Fried et al., 2016) and bipolar disorders (Curtiss et al., 2019). Other pairwise Markov Random Fields include Ising Models to estimate pairwise interactions from binary data (Briganti & Linkowski, 2020; Ising, 1925; Kruis & Maris, 2016; van Borkulo et al., 2014) and mixed graphical models to deal with mixed data (Haslbeck & Fried, 2017; Haslbeck & Waldorp, 2016). However, the causal interpretation of these models is limited: because their edges are undirected, it is impossible to tell whether symptom X is more likely to cause or be caused by symptom Y because edges have no direction and thus cannot encode this information. Furthermore, assuming a partial correlation network when the underlying model contains directed edges can lead to spurious causal connections. We can...
overcome these problems with the help of Bayesian networks (BNs), which are derived from the principles of causal reasoning (Pearl, 2009; Pearl & Mackenzie, 2018). That is, they can ascertain both the direction and the magnitude of causal effects (Maathuis et al., 2018). BNs are defined by a directed acyclic graph (DAG) and by the joint probability distribution of the variables under investigation. The role of the DAG is to express the conditional independence relationships between the variables (nodes) by using graphical separation (Koller & Friedman, 2009; Lauritzen, 1996; Pearl, 1988; Scutari & Denis, 2021). In other words, if two variables are separated in the DAG by some other variables, they are independent in probability conditional on (that is, after controlling for) those other variables. Hence, the role of the probability distribution is to express the magnitude of the causal effects linking the variables that are not graphically separated. Since BNs contain only directed edges, they are ideal for modeling the admissible causal relationships in observational data, thereby complementing insights provided by partial correlation networks. BNs should not be confused with pairwise Markov Random Fields estimated with Bayesian methods: these have been recently introduced in the network psychopathology literature in both methodological and applied research (Briganti, Williams, et al., 2020; Williams & Mulder, 2020a, 2020b).

Using BNs for rigorous causal inference requires several strong assumptions (Maathuis et al., 2018), such as the existence of a DAG underlying the data; sufficiency, that is, all the causes of a given variable are measured (which means that there are no latent variables and there is no selection bias); and faithfulness, that is, all the variables that are connected in a given way in the network are probabilistically dependent. To augment probabilistic inference with causal meaning the faithfulness and sufficiency assumptions need to be further strengthened into causal faithfulness and causal sufficiency, which will be discussed in more detail (see Section "Structure Learning", subsection "Assumption needed for structure learning"). These assumptions of having an underlying DAG and causal sufficiency are difficult to verify in psychological data: for instance, we do not know whether symptoms from a checklist or psychometric measure or structured diagnostic interview assessing a mental disorder interact as a DAG except in very specific cases, such as the relationship between a traumatic event (the cause) and PTSD symptoms (McNally, Heeren, et al., 2017). In most cases, such as depression, we do not know whether a given symptom unilaterally influences another (as in a DAG), whether two symptoms mutually influence each other (as in a partial correlation network), or whether there is one or more latent variables causing symptoms (as within the latent variable framework). The faithfulness assumption will likely be violated in a data set where the data generating process is such that $A \rightarrow B, B \rightarrow C, A \rightarrow C$, as the causal relationships are counterbalanced (Andersen, 2013; Cartwright & McMullin, 1984).

The insights that BNs can bring to the field of network psychopathology can be helpful in various situations, such as performing inference on the network structure based on its estimates. Edge weights are used, for instance, to identify the most central (i.e., interconnected) nodes in a network (Elliott et al., 2019): summing edge weights (strength centrality) or estimating the shared variance $R^2$ (predictability) are the two most common ways of gauging the importance of symptoms (Boccaletti et al., 2006; Haslbeck & Fried, 2017). Central symptoms are highly connected, with many connections or several strong ones. If such symptoms are the source of activation that maintains an episode of disorder, then successfully reducing their severity ought to cause a beneficial therapeutic cascade resulting in recovery from illness. Accordingly, they would be ideal therapeutic targets. Unfortunately, because the edges in partial correlation networks are undirected, it is impossible to tell whether a high-centrality symptom is the source of activation or the recipient of activation. Only if it is the source does it make sense to consider it a high-priority target for clinical intervention. Yet because they have often been theorized as potential targets for clinical intervention, centrality estimates are defined to operate on undirected networks and thus do not take edge directions into consideration. In other words, they do not ascertain whether central symptoms cause or are caused by many symptoms in the network; if a symptom is well-connected because it is highly controllable (that is, caused by several symptoms connected to it), acting on it is unlikely to affect other symptoms. Since interpreting network structures is an important part of any network study, complementing the study of interaction among symptoms with further causal insight is essential.

Only a handful of studies have applied BNs in psychopathology research. These have concerned dissociation (Cernis et al., 2021), self-harm (Hinze et al., 2021), paranoia (Bird et al., 2019), psychosis (Kuipers et al., 2019; Moffa et al., 2017), obsessive-compulsive disorder (Jones et al., 2018; McNally, Mair, et al., 2017), bipolar disorder (McNally et al., 2021), rumination (Bernstein et al., 2017), depression (Briganti, Scutari, et al., 2021), PTSD (McNally, Heeren, et al., 2017), and alexithymia (Briganti, Scutari, et al., 2020). In these studies, BNs illuminated potential causal relations among symptoms in cross-sectional data sets. However, the authors often varied in their computational methods, thereby underscoring the need for an accessible tutorial reference for researchers keen to apply BNs to study mental disorders. The present work aims to fill this gap by introducing BNs, by providing key commands in the R environment for statistical computing (R Core Team, 2020) to estimate BNs from data, and by addressing the main strengths and limitations of BNs in psychopathology.

This tutorial builds on previous empirical studies that applied the two main approaches for estimating BNs: constraint-based algorithms (Briganti, Scutari, et al., 2020, 2021) and
Bayesian Networks

Definition

BNs are defined as the combination of a network structure, specifically a DAG, and a probability distribution. The DAG provides a high-level abstraction useful for qualitative reasoning in the context of exploratory analysis and for testing hypotheses about how symptoms relate to one another. The probability distributions of the symptoms quantify those relationships in terms of their magnitude and direction.

Directed Acyclic Graphs

The network structure of a BN is a mathematical object \( G = (V, A) \) consisting of a set of nodes \( V = \{v_1, v_2, \ldots, v_N\} \) (where \( N \) is the number of nodes) and of a set of edges or arcs \( A = \{(v_1, v_2), (v_2, v_3), \ldots\} \) that represent all connections between two nodes. Given \( V \), a graph is uniquely identified by \( A \). In the case of a DAG, the edges can only be directed, that is, \( (v_i, v_j) \neq (v_j, v_i) \), with the assumption that two nodes are only connected by one edge. In each edge \( v_i \rightarrow v_j \), \( v_i \) is called the "parent" node and \( v_j \) is called the "child" node. DAGs are also acyclic: they do not contain any loops, that is, sequences of edges that start and end on the same node, such as \( v_i \rightarrow v_j \rightarrow \cdots \rightarrow v_k \rightarrow v_l \). The primary role of the DAG in a BN is to express the set of conditional independence of relationships among variables (that is, variables that do not cause each other).

Markov blankets

The DAG of a BN is an independence map of the probability distribution of the variables \( X_i \), which are represented in the networks by the nodes \( v_i \): retrieving such a map means determining which nodes are conditionally (in)dependent. This is achieved by using \( d \) separation, which defines graphical separation for DAGs and provides a way to algorithmically determine whether two nodes in a network are (in)dependent or conditionally (in)dependent (Geiger et al., 1990). Formally, two nodes \( v_i \) and \( v_j \) are \( d \)-separated by a conditioning set of nodes \( S \) if conditioning on all members of \( S \) block all paths (sequence of nodes and edges with \( v_i \) as starting node and \( v_j \) as ending node, that do not necessarily follow the direction indicated by edges) between \( v_i \) and \( v_j \). A collider does not generate an unconditional association between the variables that determine it (Greenland et al., 1999; Pearl, 1988). In Figure 1, three examples of a basic three-node DA G are illustrated, as well as how they differ in conditional independences.

The set \( S \) that makes a node \( v_i \) independent of all other nodes in \( G \) is known as the Markov blanket of \( v_i \). By the Markov property, the Markov blanket includes a node’s parents, children, and spouses, that is, children’s other parents. The Markov blanket is useful in investigating a target node of interest while ignoring the rest of the BN; all nodes outside of the Markov blanket are independent from the node of interest after controlling for those in the Markov blanket itself.

The Markov Property

If two nodes are not connected by an edge in a BN, then they are either independent or conditionally independent given some other nodes: this is called the local Markov property (Korb & Nicholson, 2010; Lauritzen, 1996). The global Markov property is a stronger version of the local property and generalizes it to sets of variables: any two subsets of variables are conditionally independent given a separating subset (Föllmer, 1980; Lauritzen, 1996). Graphical separation implies probabilistic independence

\[
\forall k \exists P(v_i \perp_G v_j | v_k \perp_P v_i \wedge v_j)
\]

(where \( \perp_G \) means graphical separation and \( \perp_P \) probabilistic independence), making the network itself a clear representation of the conditional independence relationships between nodes. For this reason, the DAG is called an independence map of the variables.

The Markov property makes it possible to write\[\]

\[
\Pr(X, \Theta) = \prod_{i=1}^{N} \Pr(X_i | \Pi_{X_i}; \Theta_{X_i})
\]

decomposing the larger model \( \Pr(X, \Theta) \) into a set of smaller models \( \Pr(X_i | \Pi_{X_i}; \Theta_{X_i}) \) one for each variable \( X_i \) conditional on its parents \( \Pi_{X_i} \) in the DAG (Korb & Nicholson, 2010), \( \Theta \) and \( \Theta_{X_i} \) denote the parameters of the respective

---

1If two nodes \( v_i \) and \( v_j \) are separated in the graph, then their probabilities are also independent.

2The joint probability of the set of variables \( X \) and its parameters is equal to the product of probabilities of each of the smaller models \( \Pr(X_i | \Pi_{X_i}; \Theta_{X_i}) \) for each variable \( X_i \) given its parents \( \Pi_{X_i} \) and its parameters \( \Theta_{X_i} \).
A and C are dependent, B and C are dependent, A and B

DAGs are tools that can help researchers identify instances of collider bias. DAGs are composed of conditional probability tables for each node given all other parent nodes, that is, the configuration of values of its parents (Spirtes & Meek, 1995). Learning the structure of a DAG is factorized (Briganti, Kornreich et al., 2021).

Figure 2 also shows a fundamental type of relationship in a BN, represented by the three nodes Increased Energy, Language-Thought Disorder, and Speech Disorder. Both Increased Energy and Language-Thought Disorder have an edge pointing to Speech Disorder and they are not adjacent, that is, they do not share an edge with one another. This pattern of edges is commonly known as a v-structure, or a collider, and it is one of the building blocks of BNs. In a collider (A → C ← B), four conditional independences are found: A and C are dependent, B and C are dependent, A and B are independent, and A and B are dependent conditional on C. This means that two independent variables, A and B, can become dependent when one conditions on C, because observing the common consequences of two independent causes can modify their probability (Pearl, 2009). The two causes (A and B) are therefore negatively partially correlated (if they are both positive or both negative), which is counter-intuitive; conditioning on the common effect (C) in the collider (that is, studying the associations while conditioning on the effect) leads to different estimates compared to studying the two causes individually. This phenomenon is known as collider bias (Berkson, 1946), and in network analysis it is known to induce spurious edges among nodes, and likely affects the generalizability of results when applied to the wrong population (De Ron et al., 2019; McNally, 2021). DAGs are tools that can help researchers identify instances of collider bias.

**Equivalence classes**

Although a factorization of the probability distribution of X is uniquely identified by a DAG, a DAG is not uniquely identified by a factorization of the probability distribution. For instance, patterns of edges like v_j → v_i → v_k, v_j ← v_i ← v_k and v_j ← v_i → v_k lead to equivalent probability distributions.

Hence several edges in a DAG can be reversed without changing the conditional (in)dependence relationships in X. In other words, if no new v-structure (or cycle) is created, we are just producing equivalent DAG representations of the same set of conditional (in)dependence relationships. All such DAGs are then part of an equivalence class that is uniquely identified by the undirected graph underlying these DAGs and by the v-structures. The direction of other edges is either uniquely identified because one of two directions would introduce a new v-structure or a cycle (these are sometimes called compelled edges), or completely undetermined. This results into a completed partially directed graph (Castelletti et al., 2018; Chickering, 2002a).

**Probability distributions for Bayesian Networks**

In this section, we introduce the most common probability distributions used in BNs as well as their limitations.

The three most common probability distributions for BNs are discrete, ordinal, Gaussian, and conditional linear Gaussian (Scutari & Denis, 2021). These probability distributions satisfy the following requirements: first, the structure of the BN should be learned efficiently from data by using statistical tests or goodness-of-fit measures; second, the BN should be flexible, that is, distributional assumptions should not be too strict; third, the BN should be easy to query to perform inference.

**Discrete and Ordinal Bayesian Networks**

In discrete BNs, local distributions of X_i | Π_{X_i} are composed of conditional probability tables for each node given all other parent nodes, that is, the configuration of values of its parents (Spirtes & Meek, 1995). Learning the structure
A Bayesian network of four nodes representing manic symptoms from the Young Mania Rating Scale, namely Elevated Mood, Increased Energy, Language-Thought Disorder, Speech Disorder. Elevated Mood has two directed edges towards Increased Energy and Language-Thought Disorder, and Speech Disorder receives two directed edges from Increased Energy and Language-Thought Disorder. Increased Energy, Language-Thought Disorder, and Speech Disorder are d-separated given Elevated Mood and Speech Disorder. Elevated Mood and Speech Disorder constitute the Markov blanket of both Increased Energy and Language-Thought Disorder. Increased Energy, Language-Thought Disorder, and Speech Disorder constitute a v-structure (collider).

Gaussian Bayesian Networks

Gaussian BNs (which are different from Bayesian GGMs) follow a multivariate normal distribution with mean $\mu$ and a variance-covariance matrix $\Sigma, X \sim N(\mu, \Sigma)$, with a precision matrix $\Omega$ computed as the inverse of the variance-covariance matrix $\Omega = \Sigma^{-1}$. The partial correlation coefficients $\rho_{ij}$, which model the residual correlation between two nodes after controlling all other nodes, are computed from the precision matrix as $\rho_{ij} = -\Omega_{ij}/\sqrt{\Omega_{ii}\Omega_{jj}}$, where $\Omega_{ij}$ is the element of $\Omega$. If $v_i$ and $v_j$ are d-separated in the DAG (or in the case of violation of the faithfulness assumption), the absence of an edge $v_i \rightarrow v_j$ in the DAG between the two nodes by definition implies the conditional independence $\rho_{ij} = 0$ between the
two variables \(X_i\) and \(X_j\), given all subsets of the remaining variables (we refer the reader to the section regarding the Inductive Causation Algorithm to have a better understanding on why going through all subsets of variables is necessary). From this, it follows that most edges of a Gaussian BN belong to a sub-space of a GGM (a partial correlation network), except specific independence relationships, such as a collider structure. On the other hand, cycles can be represented in a GGM but not a Gaussian BN.

As for the local distributions, \(X_i \mid \Pi_X\) are linear regression models with \(\Pi_X\) acting as regressors. This assumes that all dependencies are linear (Grzegorczyk, 2010), a requisite for the distribution of \(X\) to be multivariate normal. This may be advantageous in modeling Likert scales, but only if these scales are designed so that the items on the scale increase in linear increments (that is, item 2 implies twice the intensity of item 1, item 3 implies three times the intensity of item 1, and so forth). When this is not the case, the numbering of the items on the Likert scale leads the BN to incorrectly interpret the items as equidistant from each other and the resulting estimates of the \(\rho_{ij}\) will be biased.

Gaussian BNs have two main limitations. First, few real-world multivariate data sets follow a multivariate Gaussian distribution (the same applies to partial correlation networks since the probability distribution is the same). Second, computing partial correlations can be challenging in data sets with a large number of variables because of singularities, that is, perfect linear relationships between two variables (Lin et al., 2014). Although exact linear correspondence rarely occurs, relationships that are almost singular may result in numerical precision issues and may make BNs difficult and time-consuming to learn.

**Conditional Linear Gaussian Bayesian Networks**

Conditional Linear Gaussian BNs contain both discrete and continuous nodes and subsume both Discrete BNs and Gaussian BNs. In terms of local distributions, discrete nodes have conditional probability tables, whereas continuous nodes have a set of linear regression models, one for each configuration of the discrete parents, and with the continuous children as regressors (Cowell, 2005). In such networks, continuous nodes cannot be parents of discrete nodes, which is a strong limitation for treating mixed data: only a given set of edges can exist in the DAG, which is an important limitation on what models can be encoded by the network structure.

**Structure learning of Bayesian Networks**

Structure learning algorithms learn the structure of BNs from data, possibly integrating external expert knowledge if available. Constraint-based algorithms use statistical tests to learn conditional independence relationships (the constraints themselves); score-based algorithms rank candidate DAGs based on some goodness-of-fit criterion; hybrid algorithms use conditional independence tests to exclude most candidate DAGs, and then perform a score-based search on those that are still under consideration. In addition, specific edges can be whitelisted (included) or blacklisted (excluded) in candidate DA Gs by default based on prior knowledge: this helps the structure learning algorithm in its choice of good candidate models (Scutari & Denis, 2021).

All algorithms in this section are implemented in the R package bnlearn (Scutari, 2010), version 4.7. Two other packages that implement some algorithms for BNs are pcalg (Kalisch et al., 2012; Kalisch & Bühlmann, 2007) and BIDAG (Suter et al., 2021). To showcase the functions and their output, we use a data set consisting of seven key items from the 20 item Zung Self-Rating Depression Scale (Zung, 1965) from 1090 subjects. A detailed description of the data appears elsewhere (Briganti, Scutari, et al., 2021). In this case, we selected only seven nodes out of the 20 nodes in the scale to simplify the data visualization in the tutorial. The data and the code accompanying this tutorial are available in our repository on the Open Science Framework (https://osf.io/fine9m/), with information related to the package versions used to perform the different analyses (Briganti, 2021).

```r
library("readr")
data <- read_table("data.csv", sep= ";", header=TRUE)
# this loads the data
```

**Assumptions in structure learning**

Most structure learning algorithms make several fundamental assumptions (Dawid 2010; De Campos & Ji, 2011; Maathuis et al., 2018):

- The underlying structure that is to be learned is indeed a DAG.
- There is no selection bias and there are no latent or confounding variables, that is, all the common causes of all measured variables have been measured: this assumption is also known as causal sufficiency.
- Only the variables that are d-separated in a DAG will be independent, the others will be dependent: this assumption is also known as causal faithfulness, which is the converse of the Markov property.
- The only relationships between the random variables in \(X\) should be conditional independencies, the only kind of relationships that can be encoded by BNs. - Each node in the network must represent one random variable in \(X\): there should not be multiple nodes which are deterministic functions of the same random variable in \(X\) (such as a sum score of two variables). This is usually the case in psychological networks since separate symptoms are used as input.
- Observations must be independent realizations; if not, the network should be otherwise defined, such as a dynamic BN.
in the case of temporal dependence, since observations are not independent as a single subject is measured over multiple occasions (Song et al. 2009).

The global probability distribution \( P(X) \) must be strictly positive: that is, every combination of every possible value of the variables in \( X \) must represent an event that is observable in principle; this is needed to have uniquely determined Markov blankets and therefore an identifiable model. This is always the case in psychological data, since we measure events that patients report (e.g., fatigue) or that we observe (e.g., crying).

Most, but not all, algorithms require satisfaction of these assumptions, and some are incapable of testing whether the assumptions hold. Algorithms in the Fast Causal Inference (FCI) family are consistent when selection bias occurs or when a latent common cause is present. Using this kind of approach has multiple implications for drawing inferences. For instance, a directed edge from node A to node B in the presence of latent variables entails that A is a cause of B, but does not entail that A is a direct cause of B (Maathuis et al., 2018).

For examples of data sets where the assumption of independent realizations is violated due to temporal dependencies among variables, see studies on time-series networks (Bringmann et al., 2013; Epkskamp, 2020) or panel data networks (Briganti, Kornreich, et al., 2021).

### Stability in structure learning algorithms

The structure of BNs may vary when performing a learning algorithm several times.

For cross-sectional studies, it is desirable to study a "stable" network structure, that is, a set of edges and directions that is unlikely to vary. Stability is an important topic in network analysis: researchers need to study stable networks whose structure is likely to be influenced by factors such as sample size. For partial correlation networks, one can use bootstrapping methods to evaluate the stability of network estimates (Epkskamp et al., 2018). It is possible to directly account for stability in structure learning in a similar way: the same algorithm learns a sufficiently large number of BNs from bootstrap samples and we only consider edges that appear in a proportion of BNs higher than some threshold. Previous empirical papers (Briganti, Scutari, et al., 2020, 2021) only included edges that appeared in more than 85% of networks (this is called strength), and whose direction appeared in more than 50% of networks (this is called minimum direction). We recommend that researchers only report stable BNs obtained in this way, and that they should use 100-200 bootstrap samples to ensure the proportion of BNs in which each edge appears is estimated accurately. In addition, the proportion of times each edge was included in the stable BNs can be reported.

### Constraint-based algorithms

Constraint-based algorithms are based on Pearl and Verma’s (1991) work on causal graphical models. This class of algorithms identifies conditional independences (the “constraints”) with statistical tests and connects nodes that are not independent. The algorithms determine the sequence of tests that will be applied to the data, typically starting from marginal tests (that is, not conditioning on any variable) and gradually increasing their complexity (adding conditioning variables) to maximize speed and statistical power while ensuring the accuracy of the learned networks.

#### Tests for Conditional Independence.

Two random variables \( X \) and \( Y \) are conditionally independent given a third random variable \( Z \) if, given knowledge of \( Z \), knowledge of whether \( X \) occurs provides no information on the likelihood of \( Y \) occurring, and knowledge of whether \( Y \) occurs provides no information on the likelihood of \( X \) occurring (Dawid, 1979). Testing for conditional independence is a necessary step for constraint-based algorithms because conditional independence relationships themselves constitute the constraint. We hereby cite three default conditional independence test in bnlearn; further details appear in comprehensive reviews of the topic (Edwards, 2012). The asymptotic \( \chi^2 \) mutual information test is the default for categorical and mixed variables: it is an information-theoretic distance measure related to the deviance of the tested models and proportional to the log-likelihood ratio (the two differ by a factor), with the degrees of freedom equal to the number of free parameters in the two models (under the assumption that all parameters can be estimated). The Hotelling’s exact test is the default for continuous variables: it is the multivariate version of the Student’s \( t \) for testing the null hypothesis that the correlation is null. The Jonckheere-Terpstra test is the default for ordinal variables: it is a trend test that generalizes the Wilcoxon test (which only tolerates one ordinal variable compared to a binary factor) to two ordinal variables. A comprehensive review of methods to test for conditional independence exceeds the scope of this tutorial, but appears in the bnlearn manual (Scutari, 2010).

### The Inductive Causation algorithm.

The Inductive Causation (IC) algorithm (Pearl & Verma, 1991) was the first and simplest structure learning constraint-based algorithm. Although more modern algorithms are preferable for practical applications, all evolved from IC and thus share the same fundamental steps; hence we will describe IC in detail for illustrative purposes. The algorithm starts from a complete graph in which each variable is connected to all other variables by an undirected edge. First, for each pair of variables \( A \) and \( B \) in \( X \), the algorithm searches for a set \( S_{AB} \) such that \( A \) and \( B \) are independent given \( S_{AB} \) and such that \( A \) and \( B \) are not part of it \((A, B \notin S_{AB})\). Edges between all
such pair of variables are removed. The algorithm composes $S_{AB}$ starting from a list of no variables (with an empty $S_{AB}$) then each single variable other than $A$ and $B$ in $X$, then with a combination of other variables, until all combinations have been tried to remove the edge. Because of this, algorithms derived from the IC have one weakness: the complexity in the number of nodes in a graph (Li et al., 2019). Second, for each pair of variables that are not connected by an edge but are both connected to a common neighbor $C$, the algorithm checks whether $C \in S_{AB}$: if $C < S_{AB}$, then the direction of edge $A \rightarrow C$ becomes $A \rightarrow C$ and that of edge $C \rightarrow B$ becomes $C \leftarrow B$. Thirdly, the direction of the edges that are still undirected is set following two rules: if $A$ is adjacent to $B$ and there is a strictly directed path from $A$ to $B$ (that is, all edges in the path are directed), then $A \rightarrow B$ becomes $A \rightarrow B$; if $A$ and $B$ are not adjacent but $A \rightarrow C$ and $C \rightarrow B$, then $C \rightarrow B$ becomes $C \rightarrow B$. This step ends the algorithm which then returns the completed partially directed graph in which only those edge directions that can be uniquely identified from the data are represented.

The IC algorithm is implemented by the Peter & Clark (PC) algorithm (Spirtes et al., 1993b), which starts from a saturated network and then performs tests that gradually increase the number of conditioning nodes. Constraint based algorithms treat BNs as perfect maps, that is,

$$A \perp_P B | C \Leftrightarrow A \perp_G B | C$$

This assumption is also called faithfulness, and it is analogous to causal faithfulness from a purely probabilistic perspective. This assumption cannot be empirically verified. However, the assumption has the advantage of not requiring a strictly positive $P(X)$ to have uniquely defined Markov blankets.

In the R package bnlearn version 4.7 (Scutari, 2010), the PC algorithm is implemented in the pc.stable() function. To plot the graph in a consistent fashion with all other network analyses in psychological literature, we use the qgraph package version 1.9 (Epskamp et al., 2012).

First, we load the necessary packages, that is, bnlearn and qgraph.

```r
library("bnlearn")
# this loads the bnlearn package
library("qgraph")
# this loads the qgraph package
```

Second, we perform the PC algorithm to learn the structure of the BN.

```r
BNpc <- pc.stable(data, 
  # this includes data
  R = 200, 
  # number of boots
  algorithm = "pc.stable")
# this sets the algorithm
# note that boot.strength may give warnings
```

Fourth, we include the edges that appear in 85% of networks and whose direction appears in more than 50% of networks, and we compute the averaged network.

```r
bst1 <- BST[BST$strength > 0.85 & 
  # this sets the strength
  BST$direction > 0.5, ]
# this sets the minimum direction
avgnet1 <- averaged.network(BST, 
  # compute the average network
  threshold = 0.85)
```

Finally, we plot the averaged network with qgraph.

```r
qgraph(avgnet1, layout="circle")
# this plots the network with qgraph
```

The resulting network can be visualized in Figure 3.

### Other constraint-based algorithms.

Other well-known constraint-based algorithms include the Grow-Shrink (GS) algorithm (Margaritis, 2003) and the Incremental Association (IAMB) sets of algorithms (Tsamardinos et al., 2003), which first learn the Markov blanket for each node in the network to reduce the number of tests required by the IC algorithm. The initial network is assumed not to have any edges.

```r
BNgs <- gs(data) 
# this performs the GS algorithm
BNiamb <- iamb(data) 
# this performs the IAMB algorithm
```

### Score-based algorithms

Score-based structure learning algorithms assign a score to each candidate BN.

Network scores focus on the DAG as a whole (as opposed to individual constraints) and are goodness-of-fit statistics that assess how well the DAG mirrors the dependence structure in the data (Scutari & Denis, 2021). Two methods for scoring

3The probabilistic independence of $A$ and $B$ given $C$ is true if and only if the graphical independence of $A$ and $B$ is true.
are the Bayesian Information Criterion (BIC) and posterior probabilities from flat priors (Daly et al., 2011; Friedman & Koller, 2003; Geiger & Heckerman, 2002; Goudie & Mukherjee, 2016; Grzegorczyk & Husmeier, 2008; Heckerman & Geiger, 1995; Kuipers et al., 2018b; Madigan et al., 1995; Pensar et al., 2020). Score-based algorithms are typically applications of heuristic search algorithms, which are not guaranteed to return the highest-scoring DAG. Exact search algorithms, which do provide this guarantee, are markedly slower than heuristic algorithms and are not feasible for many practical applications (Cussens, 2012; Koivisto & Sood, 2004; Scanagatta et al., 2015; Suzuki, 2017). Hence, we will not consider them further.

The Hill-Climbing (HC) algorithm.

The Hill-Climbing (HC) algorithm (Russell & Norvig, 2002) is a greedy search algorithm that explores DAGs by single-edge additions, removals, and reversals. First, the algorithm chooses an initial candidate network structure $G$ (it chooses by default the empty DAG) and computes its score $\text{Score}(G)$. Second, it sets maxscore $= \text{Score}(G)$. Third, the algorithm computes the score of the modified networks $G^*$ obtained by every possible edge addition, deletion or reversal that does not result in a cyclic network. If any $G^*$ has $\text{Score}(G^*) > \text{Score}(G)$, the new score is updated in maxscore $= \text{Score}(G^*)$ and $G^*$ becomes the new candidate network. Finally, the algorithm returns the final network when no $G^*$ has a higher score than $G$.

\[
\text{BNhc } \leftarrow \text{hc(data)}
\]

# this performs the HC algorithm

### Other score-based algorithms.

Three well-known score-based algorithms include the Greedy Equivalence Search (Chickering, 2002b), a HC algorithm over equivalence classes instead of graphs (which minimizes the search space), the Tabu Search (Glover, 1990), a modified HC algorithm that does not stop at the first DAG for which every possible edge addition, deletion, or reversal does not improve the score, but further explores the space of DAGs to find a better DAG; and genetic algorithms (inspired by evolutionary biology) that perturb (mutation) and combine (crossover) features through several generations of structures while retaining those yielding better scores (Davis, 1991).

\[
\text{maxscore represents the maximum score computed from a network structure; the goal of the algorithm is to find a modified network structure that will have a higher score than maxscore, then updates maxscore as it goes, until it finds the highest score possible.}
\]
A comprehensive simulation study showed that was the first example of this approach in the literature. It is work by learning some conditional independence constraints to reduce the number of candidate networks prior to finding the network that maximizes a given score function (Kuipers et al., 2018a; Tsamardinos et al., 2006).

Hybrid algorithms

Hybrid algorithms combine the two previous classes of constraint-based and score-based algorithms: they are the state of the art for many problems (Scutari & Denis, 2021). They work by learning some conditional independence constraints to find the number of candidate networks prior to finding the network that maximizes a given score function (Scutari et al., 2018). If we consider individual algorithms, none consistently outperforms the others discussed here. This is also the case in our own example data set: the three classes of algorithms retrieve very similar structures, albeit with some differences in edge directions (see Figure 4). A nother comprehensive comparison of different structure learning algorithms showed that choosing an algorithm also depends on its reliability and resilience, and that synthetic performance may overestimate real-world performance (Constantinou et al., 2021).

The Sparse Candidate algorithm.

The Sparse Candidate algorithm (Friedman et al., 1999) was the first example of this approach in the literature. It is an iterative algorithm that restricts the parents of each variable to belong to a small subset of candidates. First, in a network $G$, the algorithm restricts the candidate parents for a node $X_i \in X$ to the parents of $X_i$ in $G$ in a set $C_i$. Second, the algorithm finds the network structure $G^*$ that maximizes Score$(G^*)$ among the networks in which the parents of each node $X_i$ are included in the corresponding set $C_i$, then sets the new network $G^* = G$. After the steps are repeated, the algorithm returns the graph $G$.

The Sparse Candidate algorithm is implemented in the rsmax2() function, which performs the step of restricting and maximizing only once as further steps rarely yield better DAGs (Friedman et al., 2013).

Common issues to consider when learning BNs

Choice of algorithm

Which approach should psychopathologists choose to learn the structure of a BN? The number of possible configurations of algorithms, scores, conditional independence tests, and all the tuning parameters that accompany them is indeed overwhelming. A comprehensive simulation study showed that there were no systematic differences in performance or sensitivity to error in real-world data when comparing the three classes of learning algorithms (Scutari et al., 2018). If we consider individual algorithms, none consistently outperforms the others discussed here. This is also the case in our own example data set: the three classes of algorithms retrieve very similar structures, albeit with some differences in edge directions (see Figure 4). A nother comprehensive comparison of different structure learning algorithms showed that choosing an algorithm also depends on its reliability and resilience, and that synthetic performance may overestimate real-world performance (Constantinou et al., 2021).

Although researchers should clearly present the algorithm chosen for structure learning and report the stable network structures, we recommend that they also perform the two other classes of structure learning algorithms as robustness checks. This would disclose the edge directions that vary among algorithms.

Assumptions needed for causal inference

BNs whose DAGs are in the same equivalence class are probabilistically indistinguishable: it is impossible to choose one over the other from data without resorting to additional expert knowledge. Therefore, inferring causal relationships from observational data should be done very carefully: the causal effects whose direction is supported by the data are those that correspond to edges with a defined direction in the completed partial DAG for the equivalence class. Furthermore, to defend a causal interpretation of a BN we need to satisfy three assumptions.

First, each variable $X_i$ is conditionally independent of its direct and indirect effects and from other variables given its direct causes: this is known as the causal Markov assumption.

Second, there is a DAG faithful to the probability distribution of $X$ such that the dependencies arising from d-separations in the DAG are the only ones in the probability distribution. The faithfulness assumption for causal inference implies that any population produced by the DAG has the same conditional independence relationships obtained by applying d-separation to it. However, this assumption can be challenged by selection bias; that is, the population can be "unfaithful" to the DAG. When estimating DAGs representing the causal relationships among symptoms of mental disorders or psychological constructs, the population could be unfaithful to the DAG if it were estimated from a specific subsample. For example, a network structure of depressive symptoms in manic patients will show different conditional independence relationships than a network structure estimated from unipolar depressive patients. For further details concerning selection bias in network psychopathology, we refer to more complete overviews of the topic (De Ron et al., 2019; McNally, 2021).

Third, there must be no latent variable that acts as a confounding factor thus introducing spurious dependencies among the observed variables, which translate to spurious arcs in the BN A confounding variable A is such that, in a path $A \rightarrow B \rightarrow C, A \rightarrow C$: this is also known as a backdoor path from A to C. In psychological data sets, confounding variables are an important issue since in many cases all relevant variables are not measured or included in a model. This assumption also concerns many studies designed from a latent variable point of view whereby items or symptoms presumptively reflect unobserved factors that cause their emergence and covariance. A possible solution to help overcome this issue is eliminating topological overlap among nodes (Fried & Cramer, 2017). That is, we can create composite variables to

```r
BNtabu <- tabu(data)
# this performs the Tabu algorithm

BNrs <- rsmax2(data)
# this computes the rsmax2 algorithm
```
ensure that items tapping the same construct do not appear as individual, near-synonymous nodes. Hence, each node would constitute a composite variable representing a domain. This has been done with both Gaussian Graphical Models and BNs (Briganti et al. 2019; Briganti, Scutari, et al., 2020; Briganti & Linkowski, 2019). Another way of reducing the number of network components is by using the goldbricker method in the networktools package (Jones, 2017) that identifies colinear network components (i.e., topologically overlapping items).

These assumptions are difficult to verify in real-world settings, and the best solution to tackle them is by using blocking in a carefully planned experimental design to screen out confounding factors via randomization, which severs any incoming causal link between the randomized variables and possible exogenous effects, but it is never applied to cross-sectional network studies. In the best-case scenario, scientific experiments can identify a small set of BNs that plausibly fit the data from a causal point of view.

What kind of data are needed for learning?

The data used to learn the structure of a BN and to estimate its parameters are a key driver of the quality of the models we can obtain.

For structure learning, it is important that we have enough data to rank networks accurately using scores and to ensure that the conditional independence tests we use to assess constraints have sufficiently low rates of type-I and type-II errors. The ability of structure learning algorithms to recover from errors (that is, incorrectly including or excluding edges in the DAG) is limited, which makes it important to limit the possibility of such errors occurring. For parameter learning, it is important to have enough data to detect the effect of the parents of each child and to estimate them with sufficient precision. In both cases, we may draw from existing literature on power and sample size calculations for linear and logistic regressions (Demidenko, 2007; Dupont & Plummer Jr, 1998). When the sign of the effect associated with each edge is important, we may complement traditional power calculations by assessing type-S errors (Gelman & Carlin, 2014). In both cases, it is important to remember that both structure and parameter learning operate at the level of the local distributions $X_i | \Pi_{X_i}$; we only need a sample size large enough to handle the most complex of them. If we assume that the network we are estimating is sparse, so that all nodes have a limited number of parents (say, less than 5), then the sample size required to learn a BN does not necessarily scale with the number of variables.

The number of variables will still determine the computational speed of estimating the model. Different structure learning algorithms will entail different trade-offs between speed and accuracy (Scutari et al., 2018). However, in general the complexity of most structure learning is quadratic in the number of variables if we assume that the network we are trying to learn is sparse. In other words, heuristic score-based algorithms with compute $O(N^2)$ network scores, and constraint-based algorithms will perform $O(N^2)$ conditional independence tests. If we do not assume the sparsity of the network, then the complexity becomes larger $O(e^N)$ making it difficult to perform learning in the first place. Similar considerations pertain to parameter learning: if nodes have few parents, then local distributions will have few parameters and it is possible to estimate them efficiently. Interestingly, the

![Figure 4](image_url)

**Figure 4**

BNs learned with the constraint-based PC algorithm (left), the score-based HC algorithm (middle), and the hybrid RS algorithm (right)
increase in the number of network scores or conditional independence tests evaluated during structure learning does not increase the likelihood of errors, as the literature on multiple error adjusting would suggest. It is widely recognized that all structure learning algorithms in this paper are empirically self-adjusting for multiplicity, even though no theoretical characterization of this phenomenon is available in the literature.

The quality of the data is also important. For instance, it is possible to learn BNs from incomplete data (i.e., not all values are observed for all variables in all observations) by incorporating Expectation-Maximization and other classic statistical techniques in structure and parameter learning (Scutari, 2020). It is also possible to learn BNs from the heterogeneous data that arise from trials in which observations are collected under different conditions or with different protocols (Azzimonti et al., 2020). However, handling such issues requires additional assumptions on how the data deviate from the aforementioned assumptions and it reduces power in both structure and parameter learning.

**Bayesian networks or pairwise Markov Random Fields?**

Empirical researchers may ask whether it is better to perform a network analysis using BNs or pairwise Markov Random fields, and how to compare the results from each approach. The difference between the two models from the example data set is shown in Figure 3.

Although pairwise Markov Random Fields allow for studying the interplay among different symptoms, and identify the most interconnected symptoms (Haslbeck & Fried, 2017), they cannot distinguish whether the interconnected symptoms are more likely to cause or be caused by surrounding variables: BNs can identify which causal relationships are admissible in the data set, and therefore complement the information given by current network inference measures estimated from pairwise Markov Random Fields, given the several critical assumptions, in particular the acyclicity and the absence of latent variables.

The same applies to colliders (\(A \rightarrow C \leftarrow B\)), which are important structures albeit scarcely discussed in the network literature: in pairwise Markov Random Fields, underlying collider structures can introduce spurious negative edges among variables (De Ron et al., 2019). This means researchers may conclude from a pairwise Markov Random Field that two symptoms \(A\) and \(B\) are negatively connected (that is, when symptom \(A\) is strong, symptom \(B\) is more likely to be weak, and vice versa) when in reality a BN may show an underlying collider structure that has \(A\) and \(B\) as unconnected parents of a third node \(C\), if a BN is the true underlying structure. Finally, researchers should choose the model (pairwise or directed) that fits best their study goal and design (Borsboom et al., 2021): for instance, researchers may prefer BNs for studying the effect of intervention as nodes in a network (Blanken et al., 2019; Kossakowski et al., 2019; Mooij et al., 2020).

### Assessing goodness of fit

It is common practice to assess how well a statistical model fits the data, as well as how well it predicts new data, to establish its practical relevance in the literature. This is more complicated for BNs than it is for classic statistical models for two reasons: their use of DAGs to model a multivariate distribution and the lack of a single variable of interest in learning.

First, it is important to note that a network score assigned to specific DAG is not itself useful for assessing the goodness of fit of a single model. It is not defined on a normalized scale: it becomes larger and larger in magnitude as the sample size increases and can become smaller and smaller in magnitude the more values the variables can take. Network scores are only useful to compare different DAGs, which is their task in structure learning. Reporting network scores relative to a pre-determined reference network can only be useful when such a network can be established in a rigorous way, which is not usually possible without substantial expert knowledge.

Bootstrap resampling and model averaging can reliably assess whether the edges in the BN are reliably supported by the data, rather than the result of statistical noise or measurement error. At a higher level, we can also check that most edges make sense to experts in the substantive domain (e.g., PTSD), or blacklisting clinically implausible edges and compare the scores of BNs with and without such expert knowledge (McNally, 2016).

```r
BST <- boot.strength(data, R = 200, 
algorithm = "pc.stable", 
depth = TRUE, 
cdag = TRUE)

# compute the average network strengths
avgt = averaged.network(BST, 
threshold = 0.85)

# compute the average network strengths
astring = arc.strength(average, 
data, 
threshold = "bic-g")

# compute edge strengths

Checking the parameters of the BN is more straightforward, in the sense that each 
\(Pr(X_i \mid |\Pi_{X_i})\) is either a conditional probability table in discrete BNs (Spirtes & Meek, 1995),
a linear model in Gaussian BNs (Grzegorczyk, 2010) or a collection of linear models in conditional Gaussian BNs (Cowell, 2005), which have been all studied in detail in the statistics literature (Koller & Friedman, 2009; Maathuis et al., 2018; Neapolitan, 2004; Pearl 2009; Scutari & Denis, 2021).

There are substantial practical guidelines on how to assess goodness of fit in such cases. In a Gaussian BN, each parent is associated with a regression coefficient that expresses how
its effect on the child increases or decreases for a unit change in that parent. Furthermore, the standard error associated with the node describes its noisiness in terms of the amount of variance that is not explained by the parents. In a discrete BN, conditional probabilities are assigned to each value that the node takes given the configuration of the values of its parents. In a conditional Gaussian BN, discrete nodes have the same parameters as the nodes of the discrete BN and continuous nodes have the same parameters as the nodes in a Gaussian BN, possibly in a mixture: hence their interpretation is analogous. This does not, unfortunately, apply to the entire BN.

Finally, we can use BNs as expert systems and run queries on various combinations of nodes and check whether their answers make sense to domain experts. In this context, a query consists in computing the probability of some event involving some variable of interest taking place (for instance, a given score on a symptom X) given some evidence on other variables (for instance, given scores on other symptoms Y and Z). If experts can quantify their confidence in observing events under a set of specific circumstances, based on their domain knowledge, we can measure the discrepancy between their confidence and the BN’s confidence as a measure of goodness-of-fit. This approach rests, of course, on the availability of reliable expert knowledge on the phenomenon being modeled since we are treating it as a surrogate of ground truth.

Conclusion

This work contains an introduction to BNs as well as a tutorial on estimating them in R using the three main families of structure-learning algorithms: constraint-based, score-based, and hybrid algorithms. The network obtained contains directed edges, which can be interpreted as admissible causal relationships among symptoms of mental disorders under several assumptions. BNs are models that can complement the popular partial correlation networks in the broad framework proposed by the network theory of mental disorders (Borsboom, 2017).

The field of network analysis is new and uniform methods for investigating DAGs remain unestablished. Our tutorial aims to offer an introduction to the topic as well as accessible methods to conduct these analyses.

References


Fried, E. I., & Cramer, A. O. J. (2017). Moving Forward: Challenges and Directions for Psy-


Kuipers, J., Moffa, G., Kuipers, E., Freeman, D., &


Scutari, M., Graafland, C. E., & Gutiérrez, J. M. (2018). Who learns better bayesian network structures: Constraint-based, score-based or hybrid algorithms? International Con-

dynamic bayesian networks. Advances in Neural Information
Processing Systems, 1732–1740.

Spirtes, P., Glymour, C., & Scheines, R. (1993a). Dis-
cover Algorithms for Causally Sufficient Structures. In
P. Spirtes, C. Glymour, & R. Scheines (Eds.), Causation,
Prediction, and Search (pp. 103–162). Springer New York.
https://doi.org/10.1007/978-1-4612-2748-9_5

Spirtes, P., Glymour, C., & Scheines, R. (1993b). Causa-

works with discrete variables from data. KDD, 1, 294–299.

Bayesian structure learning and sampling of Bayesian
networks with the R package BiDAG. ArXiv:2105.00488

Suzuki, J. (2017). An Efficient Bayesian Network Struc-
105–124.

Tsagris, M., Borboudakis, G., Lagani, V., & Tsamardinos,
International Journal of Data Science and Analytics, 6(1),
19–30.

Tsamardinos, I., Aliferis, C. F., Statnikov, A. R., & Stat-

max-min hill-climbing Bayesian network structure learning

van Borkulo, C. D., Borsboom, D., Epskamp, S., Blanken,
A new method for constructing networks from binary data.
Scientific Reports, 4, 5918. https://doi.org/10.1038/srep05918

Williams, D. R., & Mulder, J. (2020a). Bayesian hypothe-
sis testing for Gaussian graphical models: Conditional inde-
dependence and order constraints. Journal of Mathematical
Psychology, 99, 102441.

Williams, D. R., & Mulder, J. (2020b). BGGM: Bayesian
Gaussian graphical models in R. Journal of Open Source
Software, 5(51), 2111.

Young, R. C., Biggs, J. T., Zeigler, V. E., & Meyer, D. A.
(1978). A rating scale for mania: Reliability, validity and
https://doi.org/10.1192/bjp.133.5.429

chives of General Psychiatry, 12, 63–70.

Appendix A

Patterns of edges like $v_j \rightarrow v_i \rightarrow v_k$, $v_j \leftarrow v_i \leftarrow v_k$ and
$v_j \leftarrow v_i \rightarrow v_k$ lead to equivalent probability distributions.

The product of the probability of $v_i$, the probability of $v_j$
given $v_i$ and the probability of $v_k$ given $v_i$, which describes
the probability of $v_j \rightarrow v_i \rightarrow v_k$ is equal to: 1) the product
of the probability of $v_j$, the joint probability of $v_i$ and $v_j$
divided by the probability of $v_j$, and the joint probability of $v_i$ and $v_k$
divided by the probability of $v_k$; 2) the product of the joint
probability of $v_i$ and $v_j$ divided by the probability of $v_i$, and the
joint probability of $v_i$ and $v_k$; 3) the product of the probability
of $v_i$, the probability of $v_j$ given $v_i$, and the probability of $v_k$
given $v_i$, which describes the probability of $v_j \leftarrow v_i \rightarrow v_k$; 4) the product of the probability of $v_k$, the probability of $v_j$
given $v_i$, and the probability of $v_i$ given $v_k$, which describes
the probability of $v_j \leftarrow v_i \leftarrow v_k$

\[
\Pr (v_j) \Pr (v_i | v_j) \Pr (v_k | v_i) = \\
\Pr (v_j) \frac{\Pr (v_i, v_j)}{\Pr (v_i)} = \\
\frac{\Pr (v_i, v_j)}{\Pr (v_i)} \Pr (v_i, v_k) = \\
\Pr (v_j) \Pr (v_i | v_j) \Pr (v_k | v_i) = \\
\Pr (v_k) \Pr (v_j | v_i) \Pr (v_i | v_k)
\]