Bayesian Networks
and Their Extensions in Modern Machine Learning

Marco Scutari
scutari@idsia.ch

Dalle Molle Institute for
Artificial Intelligence (IDSIA)

October 13, 2021

A Graph and a Probability Distribution

A Bayesian network (BN) [30] is defined by:

- a network structure, a directed acyclic graph $\mathcal{G}$, in which each node corresponds to a random variable X_i;
- a global probability distribution $\mathbf{X}$ with parameters Θ, which can be factorised into smaller local probability distributions according to the arcs present in $\mathcal{G}$.

The main role of the network structure is to express the conditional independence relationships among the variables in the model through graphical separation, thus specifying the factorisation of the global distribution:

$$\mathrm{P}(\mathbf{X}) = \prod_{i=1}^{p} \mathrm{P}(X_i \mid \Pi_{X_i}; \Theta_{X_i}) \quad \text{where} \quad \Pi_{X_i} = \{\text{parents of } X_i\}.$$

What About the Probability Distributions?

The choice of $\mathrm{P}(\mathbf{X})$ should such that the BN:

- can be learned efficiently from data;
- is flexible (distributional assumptions should not be too strict);
- is easy to query to perform inference.

The three most common choices in the literature (by far) [6, 17], are:

- discrete BNs (DBNs), in which $\mathbf{X}$ and the $X_i \mid \Pi_{X_i}$ are multinomial;
- Gaussian BNs (GBNs), in which $\mathbf{X}$ is multivariate normal and the $X_i \mid \Pi_{X_i}$ are univariate normal;
- conditional linear Gaussian BNs (CLGBNs), in which $\mathbf{X}$ is a mixture of multivariate normals and the $X_i \mid \Pi_{X_i}$ are either multinomial, univariate normal or mixtures of normals.

It has been proved in the literature that exact inference is possible in these three cases, hence their popularity.

Learning a BN $\mathcal{B} = (\mathcal{G}, \Theta)$ from a data set $\mathcal{D}$ is performed in two steps:

$$\underbrace{\mathrm{P}(\mathcal{B} \mid \mathcal{D}) = \mathrm{P}(\mathcal{G}, \Theta \mid \mathcal{D})}_{\text{learning}} = \underbrace{\mathrm{P}(\mathcal{G} \mid \mathcal{D})}_{\text{structure learning}} \cdot \underbrace{\mathrm{P}(\Theta \mid \mathcal{G}, \mathcal{D})}_{\text{parameter learning}}.$$

In a Bayesian setting structure learning consists in finding the DAG with the best $\mathrm{P}(\mathcal{G} \mid \mathcal{D})$. We can decompose $\mathrm{P}(\mathcal{G} \mid \mathcal{D})$ into

$$\mathrm{P}(\mathcal{G} \mid \mathcal{D}) \propto \mathrm{P}(\mathcal{G})\,\mathrm{P}(\mathcal{D} \mid \mathcal{G}) = \mathrm{P}(\mathcal{G}) \int \mathrm{P}(\mathcal{D} \mid \mathcal{G}, \Theta)\,\mathrm{P}(\Theta \mid \mathcal{G}) d\Theta$$

where $\mathrm{P}(\mathcal{G})$ is the prior distribution over the space of the DAGs and $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$ is the marginal likelihood of the data given $\mathcal{G}$ averaged over all possible parameter sets Θ; and then

$$\mathrm{P}(\mathcal{D} \mid \mathcal{G}) = \prod_{i=1}^{N} \left[\int \mathrm{P}(X_i \mid \Pi_{X_i}, \Theta_{X_i})\,\mathrm{P}(\Theta_{X_i} \mid \Pi_{X_i}) d\Theta_{X_i} \right].$$

Structure and Parameter Learning Algorithms

The most common structure learning algorithms are score-based algorithms, in which we are looking for the DAG that maximises a score such as the posterior $\mathrm{P}(\mathcal{G} \mid \mathcal{D})$ [16] or BIC [28].

As an alternative, constraint-based structure learning algorithms use statistical to learn which variables are conditionally independent from each other, and they assume that the corresponding the corresponding nodes are graphically separated.

Thorough reviews of structure learning algorithms: [31, 5].

Once the structure of the model is known, the problem of estimating the parameters of the global distribution can be solved by estimating the parameters of the local distributions, one at a time. We can use usual maximum likelihood, Bayesian or regularised estimators for linear models and contingency tables we have from classical statistics [17, 6].

Events, Evidence and Queries

A BN represents a working model of the world that a computer can understand: we can ask it questions and have algorithms perform probabilistic inference automatically to answer them.

Questions that can be asked are called queries and are typically about an event of interest given some evidence. The evidence is the input to the computer system and the event is the output. This is often called belief update: we observe some evidence and we update our beliefs before taking action.

The most common queries are conditional probability and most probable explanation queries. Both can be answered using a variety of exact or approximate algorithms [17, 4].

- ✔ An Introduction to Bayesian Networks
- → **Advances in Bayesian Networks**
- Summary
- References

The Classic Definition of Bayesian Networks

The classic definition of a BN [17] involves:

- a network structure, a directed acyclic graph $\mathcal{G}$, in which each node corresponds to a random variable X_i;
- a probability distribution $\mathbf{X}$ and its factorisation into $X_i \mid \Pi_{X_i}$.

What are we assuming when trying to learn a BN? Typically that:

- observations are independent and there are no missing values;
- all variables are observed, that is, no latent variables introducing confounding into the model;
- we measure probabilistic associations (or rather, independencies) and we cannot necessarily interpret them as causal.

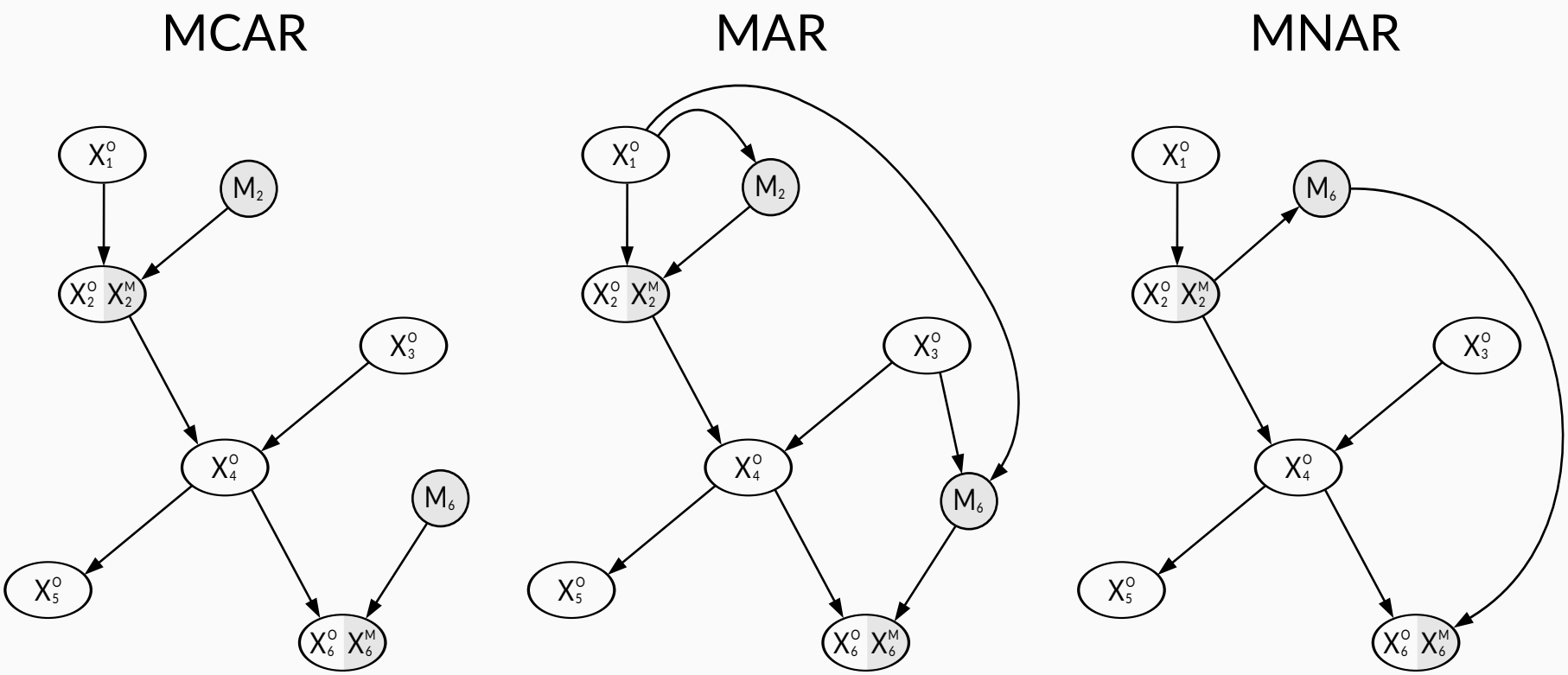
What happens if we relax those assumptions? Many extensions suddenly become possible, see [29] for a recent review.

Missing Data

Little and Rubin [21, 26] formalised three possible patterns of missingness:

- Missing completely at random (MCAR): complete samples are indistinguishable from incomplete ones. The probability that a value will be missing is independent from both observed and missing values.
- Missing at random (MAR): incomplete samples differ from complete ones, but the pattern of missingness is predictable from other observed variables. The probability that a value will be missing is a function of the observed values.
- Missing not at random (MNAR): the pattern of missingness is not random or it is not predictable from other observed variables; the probability that an entry will be missing depends on both observed and missing values. Common examples are variables that are missing systematically or for which the pattern of missingness depends on the missing values themselves.

Missing Data as Bayesian Networks



Ignorable Patterns of Missingness

MCAR and MAR are ignorable. If we denote with $\mathcal{D}^O$ and $\mathcal{D}^M$ the observed and unobserved portions of $\mathcal{D}$, and we group all the binary missingness indicators M_i in $\mathbf{M}$ with parameters Ξ, then we can write

$$\mathrm{P}\left(\mathcal{D}, \mathbf{M} \mid \mathcal{G}, \Theta, \Xi\right) = \mathrm{P}\left(\mathcal{D}^O, \mathcal{D}^M, \mathbf{M} \mid \mathcal{G}, \Theta, \Xi\right) = \int \mathrm{P}(\mathcal{D}^O, \mathcal{D}^M \mid \mathcal{G}, \Theta)\, \mathrm{P}\left(\mathbf{M} \mid \mathcal{D}^O, \mathcal{D}^M, \mathcal{G}, \Xi\right)\, d\mathcal{D}^M.$$

If the missing data are MAR then $\mathbf{M}$ only depends on $\mathcal{D}^O$,

$$\mathrm{P}\left(\mathbf{M} \mid \mathcal{D}^O, \mathcal{D}^M, \mathcal{G}, \Xi\right) = \mathrm{P}\left(\mathbf{M} \mid \mathcal{D}^O, \mathcal{G}, \Xi\right);$$

and if the missing data MCAR then $\mathbf{M}$ does not depend on $\mathcal{D}^O$ or $\mathcal{D}^M$,

$$\mathrm{P}\left(\mathbf{M} \mid \mathcal{D}^O, \mathcal{D}^M, \mathcal{G}, \Xi\right) = \mathrm{P}\left(\mathbf{M} \mid \mathcal{G}, \Xi\right).$$

In both cases it is possible to model $\mathbf{M}$ from the available data. However, this is not the case for MNAR since $\mathbf{M}$ depends on the unobserved $\mathcal{D}^M$.

Parameter Learning with Incomplete Data: EM

A classic approach to handle missing data is the Expectation-Maximisation (EM) algorithm [20]:

- the expectation (E) step consists in computing the expected values of the sufficient statistics (such as the counts n_{ijk} in discrete BNs, partial correlations in GBNs), using exact inference to make use of incomplete as well as complete samples;
- the maximisation (M) step takes the sufficient statistics from the E-step and estimates the parameters of the BN, either using maximum likelihood or Bayesian posterior estimators.

The parameter estimates are then used in the next E-step to update the expected values of the sufficient statistics. Repeated iterations of these two steps will in the limit return the maximum likelihood or maximum a posteriori estimates for the parameters.

The E-step is equivalent to computing $\mathrm{E}(\mathcal{D}^M, \mathcal{D}^O \mid \mathcal{G}, \Theta)$ and the M-step is equivalent to maximising $\mathrm{P}(\Theta \mid \mathcal{G}, \mathcal{D}^O, \mathcal{D}^M)$.

Data Augmentation (DA) [24, 32] is quite similar, but instead of converging iteratively to a single set of parameter estimates it uses Gibbs Sampling to generate values from the posterior distributions of both $\mathcal{D}^M$ and Θ. The two steps are as follows:

- in the imputation (I) step the data are completed with values drawn from the predictive distributions of the missing values;
- in the parameter (P) estimation step a parameter value is drawn from the posterior distribution of Θ conditional on the completed data from the I-step.

Formally, we define the augmented parameter vector $\{\mathcal{D}^M, \Theta\}$ containing both the missing values and the parameters of the BN. Given an initial set of values, DA updates each element of $\{\mathcal{D}^M, \Theta\}$ by sampling a new value for each missing value from $\mathrm{P}(\mathcal{D}_i^M \mid \mathcal{D}_{-i}^M, \Theta)$, and by sampling a new value for each parameter from $\mathrm{P}(\Theta_i \mid \Theta_{-i}, \mathcal{D}^M)$, each in turn.

Learning the structure of a BN from incomplete data is computationally unfeasible because we need to perform a joint optimisation over the missing values and the parameters to score each candidate network. The maximum a posteriori DAG maximises

$$\mathrm{P}\left(\mathcal{D} \mid \mathcal{G}\right) = \int \mathrm{P}\left(\mathcal{D}^O, \mathcal{D}^M \mid \mathcal{G}, \Theta\right) \mathrm{P}\left(\Theta \mid \mathcal{G}\right)\, d\Theta =$$

$$= \int \underbrace{\mathrm{P}\left(\mathcal{D}^M \mid \mathcal{D}^O, \mathcal{G}, \Theta\right)}_{\text{missing data}} \underbrace{\mathrm{P}\left(\mathcal{D}^O \mid \mathcal{G}, \Theta\right)}_{\text{observed data}} \underbrace{\mathrm{P}\left(\Theta \mid \mathcal{G}\right)\, d\Theta}_{\text{averaging over parameters}} .$$

A full Bayesian approach would require averaging over all the possible configurations of the missing data, leading to

$$\mathrm{P}\left(\mathcal{D} \mid \mathcal{G}\right) = \iint \mathrm{P}\left(\mathcal{D}^M \mid \mathcal{D}^O, \mathcal{G}, \Theta\right) \mathrm{P}\left(\mathcal{D}^O \mid \mathcal{G}, \Theta\right) \mathrm{P}\left(\Theta \mid \mathcal{G}\right)\, d\Theta\, d\mathcal{D}^M.$$

which has one one extra dimension for each missing value. An additional problem is that $\mathrm{P}(\mathcal{D}^M \mid \mathcal{D}^O, \mathcal{G}, \Theta)$ does not factorise in the general case.

Structure Learning from Incomplete Data: Structural EM

The Structural EM algorithm [9] makes structure learning computationally feasible by searching for the best structure inside of EM, instead of embedding EM inside a structure learning algorithm. It consists of two steps like the classic EM:

- in the E-step, we complete the data by computing the expected sufficient statistics using the current network structure;
- in the M-step, we find the structure that maximises the expected score function for the completed data.

Since the scoring in the M-step uses the completed data , structure learning can be implemented efficiently using standard algorithms. The original proposal by [9] used BIC and greedy search; [10] later extended SEM to a fully Bayesian approach based posterior scores, and proved the convergence of the resulting algorithm.

Structure Learning from Incomplete Data: Approximations

An alternative is to use scores that approximate $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$ and that are decomposable and efficient to compute even on incomplete data. Some options are:

- The variational-Bayesian EM from [1] that maximises a variational approximation of $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$, which in turn is a lower bound to the true $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$.
- Replacing BIC with the node-average penalised log-likelihood computed from locally-complete observations as suggested in [2].
- Using mixtures of truncated exponentials [7] to approximate the distributions of variables that are not completely observed.

What About Latent Variables?

Note that we can work on latent variables using similar approaches if $\mathcal{G}$ is fixed and we just need to learn Θ. A recent example is given in [34], who shows it is possible to learn the domain of a latent discrete variable as well as the associated parameters as long as it has observed parents and children. Several other examples are discussed in the context of dynamic BNs in [22].

If $\mathcal{G}$ is not fixed, we need to learn:

- how many latent variables we are dealing with;
- their domains;
- how they are connected to the observed variables

at the same time, which is not feasible in general.

Dynamic BNs (DBNs) [22] combine classic BNs and Markov processes to model dynamic data in which each individual is measured repeatedly over time, such as longitudinal or panel data.

Assume we have one set $\mathbf{X}^{(t)}$ of random variables for each of $t = 1, \ldots, T$ time points. We can model it as a DBN with a Markov process of the form

$$\mathrm{P}\left(\mathbf{X}^{(0)}, \ldots, \mathbf{X}^{(T)}\right) = \mathrm{P}\left(\mathbf{X}^{(0)}\right) \prod_{t=1}^{T} \mathrm{P}\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(t-1)}\right).$$

where $\mathrm{P}(\mathbf{X}^{(0)})$ gives the initial state of the process and $\mathrm{P}(\mathbf{X}^{(t)} \mid \mathbf{X}^{(t-1)})$ defines the transition between times $t-1$ and t. When modelling $\mathbf{X}^{(t)}$, the nodes in $\mathbf{X}^{(t-1)}$ only appear in the conditioning; we take them to be essentially fixed and to have no free parameters, so we leave them as root nodes.

We can model this transition with a 2-time BN (2TBN) defined over $(\mathbf{X}^{(t-1)}, \mathbf{X}^{(t)})$, in which we naturally assume that any arc between a node in $t-1$ and a node in t must necessarily be directed towards the node in t following the arrow of time.

We may also want to assume that there are no arcs connecting two nodes in the same t or, in other words, no instantaneous dependencies.

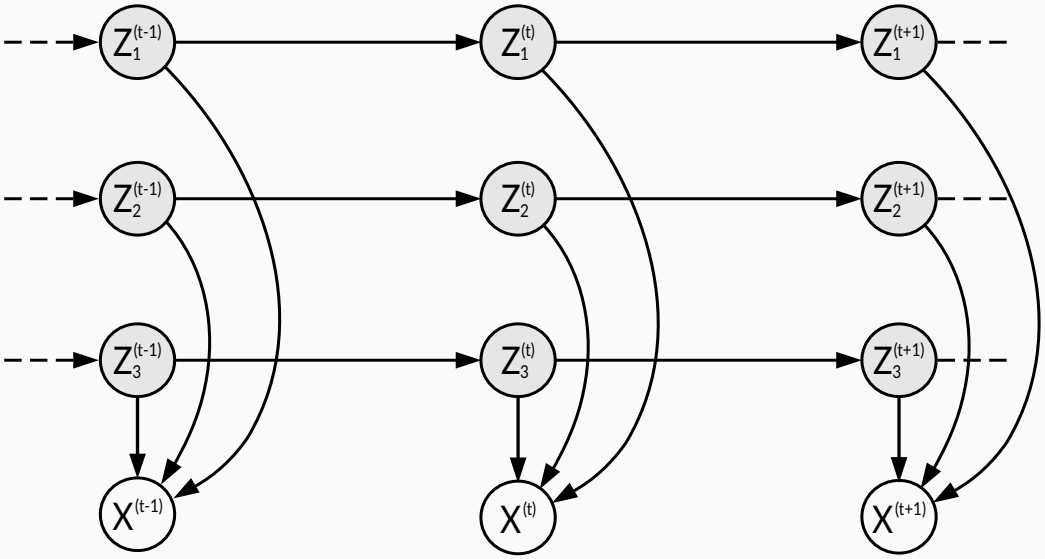
We can then write the decomposition into local distributions

$$\mathrm{P}\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(t-1)}\right) = \prod_{i=1}^{N} \mathrm{P}\left(X_i^{(t)} \mid \Pi_{X_i^{(t)}}\right),$$

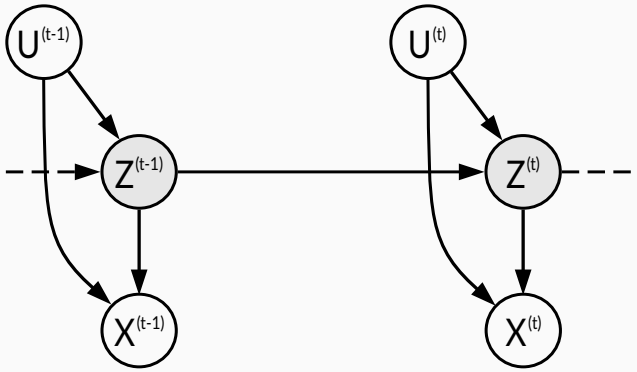
and we usually assume that the parameters associated with the local distributions do not change over time to make the process time-homogeneous.

Many Models are Dynamic Bayesian Networks

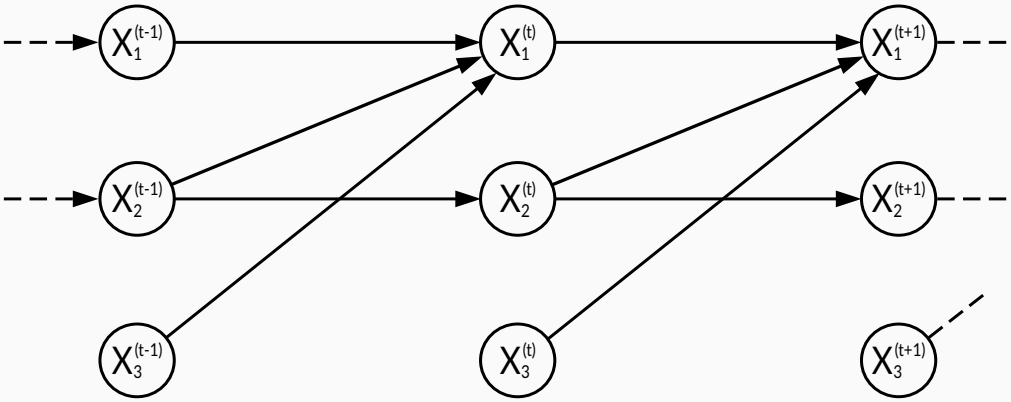
Factorial HMM



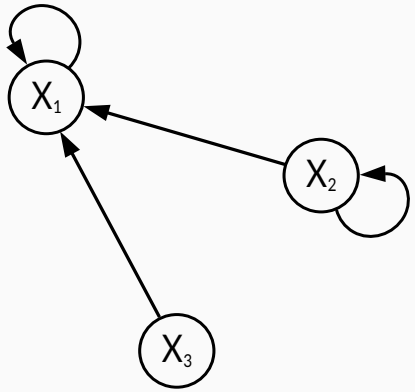
Kalman Filter



Unrolled VAR



Rolled VAR



Hidden Markov models (HMMs) [36] are one of the most widespread approaches to model phenomena with hidden state, that is, in which the behaviour of the observed variables $\mathbf{X}$ depends on that of one or more discrete latent variables $\mathbf{Z}$ as well as on other variables in $\mathbf{X}$.

In DBN terms, an HMM model with M latent variables can be written as

$$\mathrm{P}\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(t-1)}, \mathbf{Z}^{(t)}\right) = \prod_{i=1}^{N} \mathrm{P}\left(X_i^{(t)} \mid \Pi_{X_i^{(t)}}, \mathbf{Z}^{(t)}\right)$$

$$\mathrm{P}\left(\mathbf{Z}^{(t)}\right) = \prod_{j=1}^{M} \mathrm{P}\left(Z_j^{(t)} \mid \Pi_{Z_j^{(t)}}\right),$$

with the restriction that the parents of $Z_j^{(t)}$ can only be other latent variables. In the vast majority of the literature all variables are assumed to be discrete. Depending on the choice of $\Pi_{Z_j^{(t)}}$, we can obtain various HMM variants such as hierarchical HMMs [8] and factorial HMMs [12].

Many Models are Dynamic Bayesian Networks: VARs

Vector auto-regressive models (VARs) [3] are a straightforward multivariate extension of univariate auto-regressive time series for continuous variables.

VARs are defined as

$$\mathbf{X}^{(t)} = A_1\mathbf{X}^{(t-1)} + \ldots + A_L\mathbf{X}^{(t-L)} + \varepsilon_t, \qquad \varepsilon_t \sim N(0, \Sigma),$$

for some fixed Markov order L. We can rewrite that as

$$\mathbf{X}^{(t)} \mid \mathbf{X}^{(t-1)}, \ldots, \mathbf{X}^{(t-L)} \sim N\left(A_1\mathbf{X}^{(t-1)} + \ldots + A_L\mathbf{X}^{(t-L)}, \varepsilon_t\right)$$

and then restrict the parents of each $X_i^{(t)}$ to those for which the corresponding regression coefficients in $A_1, \ldots, A_L$ are different from zero using the one-to-one correspondence between regression coefficients and partial correlations [33]. Formally, $X_j^{(t-l)} \in \Pi_{X_i^{(t)}}$ if and only if $A_l[i, j] \neq 0$, which makes it possible to write a VAR as a Gaussian DBN.

Many Models are Dynamic Bayesian Networks: Kalman Filters

Kalman filters [15] combine traits of both HMMs and VARs, as discussed in depth in [11] and [25]: like VARs, they are linear Gaussian DBNs; but they also have latent variables like HMMs.

In their simplest form, Kalman Filters include a layer of one or more latent variables that model the unobservable part of the phenomenon,

$$\mathbf{Z}^{(t)} = A\mathbf{Z}^{(t-1)} + B\mathbf{U}^{(t)} + \zeta_t, \qquad \zeta_t \sim N(0, \Psi)$$

feeding into one or more observed variables

$$\mathbf{X}^{(t)} = C\mathbf{Z}^{(t)} + D\mathbf{U}^{(t)} + \varepsilon_t, \qquad \varepsilon_t \sim N(0, \Sigma)$$

with independent Gaussian noise added in both layers. Both layers often include additional (continuous) explanatory variables $\mathbf{U}$ and can also be augmented with (discrete) switching variables to allow for different regimes as in [13, 14]. If we exclude the latter, the assumption is that the system is jointly Gaussian: that makes it possible to frame Kalman filters as DBNs in the same way we did for VARs.

Learning Bayesian Networks as Causal Models

Learning causal models [23], especially from observational data, presents significant challenges. In particular, three additional assumptions are needed:

- Causal Markov assumption: each variable $X_i \in \mathbf{X}$ is conditionally independent of its non-effects, both direct and indirect, given its direct causes.
- Faithfulness: there must exist a DAG which is faithful to the probability distribution $\mathbf{P}$ of $\mathbf{X}$, so that the only dependencies in $\mathbf{P}$ are those arising from d-separation in the DAG.
- Causal Sufficiency: there must be no *latent variables* (unobserved variables influencing the variables in the network) acting as *confounding factors*. Such variables may induce spurious correlations between the observed variables, thus introducing bias in the causal network.

These assumptions are difficult to verify in real-world settings, as the set of the potential confounding factors is not usually known.

Why is Learning Causal Bayesian Networks Important?

From a practical perspective, ensuring that a BN we learn from data correctly models the underlying cause-effect relationships between the variables is important because:

- it improves our understanding of the underlying phenomenon;
- it allows us to target interventions to effect some desirable change to the underlying phenomenon;
- it allows us to reason about counterfactuals using the BN.

Identifying the correct targets for interventions is the foundation upon which quantitative policy decision making is built.

A counterfactual [23] is an "if" statement in which the "if" portion is untrue or unrealised. The "if" portion of a counterfactual is called the hypothetical condition, or more often, the antecedent. We use counterfactuals to emphasise our wish to compare two outcomes under the exact same conditions, differing only in one aspect: the antecedent.

Counterfactual Fairness: a sensitive attribute A a should not be a cause of a target variable in any (statistical) individual [27, 19]. In other words, changing A while holding things which are not causally dependent on A constant will not change the distribution of Y.

Formally: a predictor $\hat{Y}$ of Y is counterfactually fair given the sensitive attribute $A = a$ and any observed variables $\mathbf{X}$ if

$$\mathrm{P}(\hat{Y}_{A \leftarrow a} = y \mid \mathbf{X} = \mathbf{x}, A = a) = \mathrm{P}(\hat{Y}_{A \leftarrow a'} = y \mid \mathbf{X} = \mathbf{x}, A = a)$$

for all y and $a' \neq a$.

In graphical terms: $\hat{Y}$ is counterfactually fair if it is a function of the non-descendants of A in causal network.
This condition is too strong for most practical applications because it requires that the distributions are exactly the same for all values of A.

Approximate Counterfactual Fairness

Relaxing the definition of counterfactual fairness by introducing some tolerance in the comparison of the distributions of $\hat{Y}$ under different values of A gives:

Approximate Counterfactual Fairness: a predictor $f(\mathbf{X}, A)$ satisfies (ε, δ)-approximate counterfactual fairness if, given the sensitive attribute $A = a$ and any instantiation $\mathbf{X} = \mathbf{x}$, we have that:

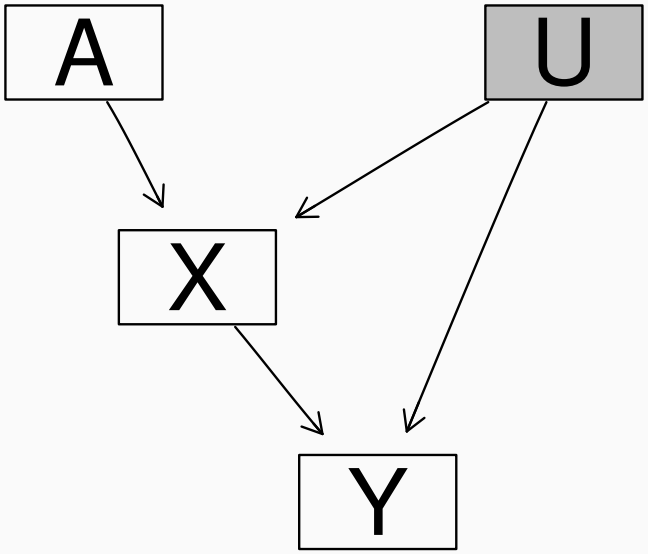
$$\mathrm{P}\left(|f(\mathbf{X}_{A \leftarrow a}, a) - f(\mathbf{X}_{A \leftarrow a'}, a')| \leqslant \varepsilon \mid \mathbf{X} = \mathbf{x}, A = a\right) > 1 - \delta$$

for all $a' \neq a$.

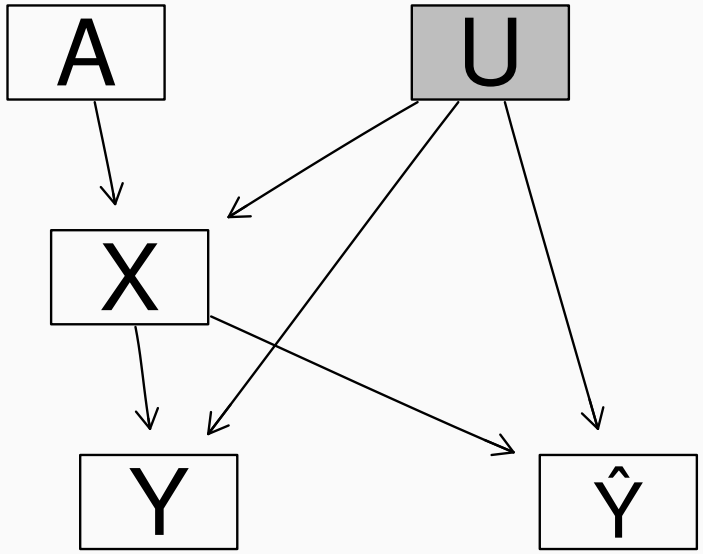
In other words, we allow some degree of unfairness but we assume that the predictor $f(\mathbf{X}, A)$ is mostly fair most of the time. This is common in all fairness literature [18, 35, etc.], regardless of the statistical model.

Consider: an Example

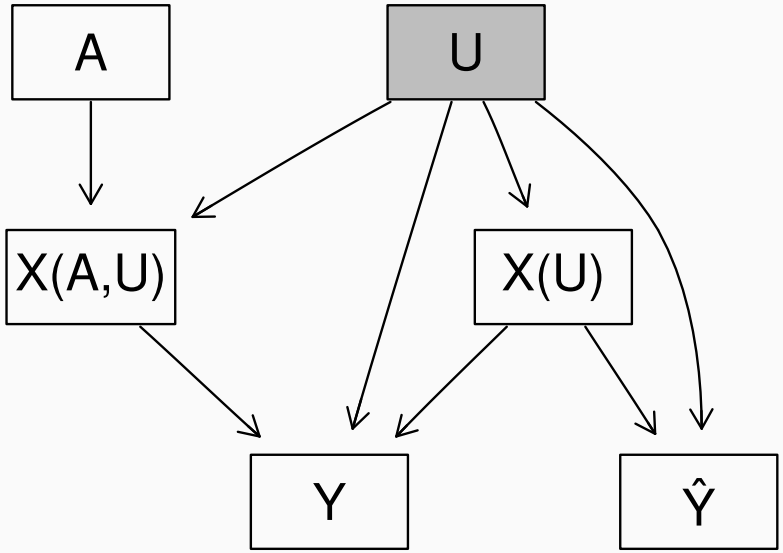
original model



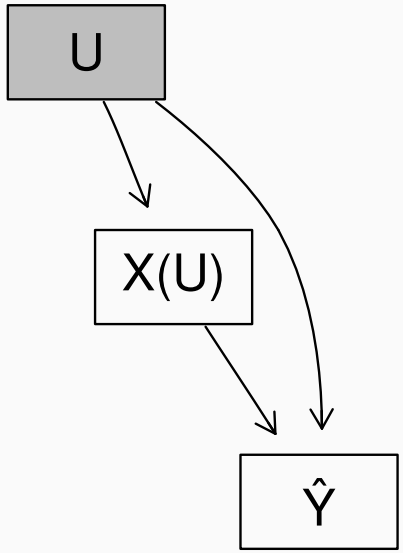
with predictions



split fair and unfair



fair prediction



- ✓ An Introduction to Bayesian Networks
- ✓ Advances in Bayesian Networks
- → Summary
- References

Bayesian Networks and Modern Machine Learning

- BNs provide an intuitive representation of the relationships linking heterogeneous sets of variables, which we can use for qualitative and causal reasoning.
- BNs subsume a large number of probabilistic models and thus can readily incorporate other techniques from statistics and computer science.
- BNs can be extended to handle data with complex structure such as multivariate time series and incomplete data, while taking advantage of standard learning and inference approaches.
- BNs also provide a rigorous solution to the problem of producing fair predictions that do not discriminate individuals based on sensitive attributes.

- ✔ An Introduction to Bayesian Networks
- ✔ Advances in Bayesian Networks
- ✔ Summary
- → References

References I

- M. J. Beal and Z. Ghahramani.
  The Variational Bayesian EM Algorithm for Incomplete Data: with Application to Scoring Graphical Model Structures.
  In *Proceedings of the 7th Valencia International Meeting*, pages 453–464, 2003.
- T. Bodewes and M. Scutari.
  Learning Bayesian Networks from Incomplete Data with the Node-Averaged Likelihood.
  International Journal of Approximate Reasoning, 138:145–160, 2021.
- G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung.
  Time Series Analysis: Forecasting and Control.
  Wiley, 5th edition, 2016.
- E. Castillo, J. M. Gutiérrez, and A. S. Hadi.
  Expert Systems and Probabilistic Network Models.
  Springer, 1997.
- A. C. Constantinou, Y.Liu, K. Chobtham, Z.Guo, and N. K. Kitsona.
  Large-scale Empirical Validation of Bayesian Network Structure Learning Algorithms with Noisy Data.
  International Journal of Approximate Reasoning, 131:151–188, 2021.
- D. I. Edwards.
  Introduction to Graphical Modelling.
  Springer, 2nd edition, 2000.
- A. Fernández, J. D. Nielsen, and A. Salmerón.
  Learning Bayesian Networks for Regression from Incomplete Databases.
  International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 18(01):69–86, 2010.
- S. Fine, Y. Singer, and N. Tibshy.
  The Hierarchical Hidden Markov Model: Analysis and Applications.
  Machine Laarning, 32(1):41–62, 1998.
- N. Friedman.
  Learning Belief Networks in the Presence of Missing Values and Hidden Variables.
  In *ICML*, pages 125–133, 1997.

References II

- N. Friedman.
  The Bayesian Structural EM Algorithm.
  In *UAI*, pages 129–138, 1998.
- Z. Ghahramani.
  An Introduction to Hidden Markov Models and Bayesian Networks.
  International Journal of Pattern Recognition and Artificial Intelligence, 15(1):9–42, 2001.
- Z. Ghahramani and M. Jordan.
  Factorial Hidden Markov Models.
  In *NIPS*, pages 472–278, 1996.
- M. Grzegorczyk and D. Husmeier.
  Non-Stationary Continuous Dynamic Bayesian Networks.
  In *NIPS*, pages 682–690, 2009.
- M. Grzegorczyk and D. Husmeier.
  Non-homogeneous Dynamic Bayesian Networks for Continuous Data.
  Machine Learning, 83(3):355–419, 2011.
- J. D. Hamilton.
  Time Series Analysis.
  Princeton University Press, 1994.
- D. Heckerman and D. Geiger.
  Learning Bayesian Networks: a Unification for Discrete and Gaussian Domains.
  In *UAI*, pages 274–284, 1995.
- D. Koller and N. Friedman.
  Probabilistic Graphical Models: Principles and Techniques.
  MIT Press, 2009.

References III

- J. Komiyama, A. Takeda, J. Honda, and H. Shimao.
  Nonconvex Optimization for Regression with Fairness Constraints.
  Proceedings of Machine Learning Research, 80:2737–2746, 2018.
  Proccedings of the of the 35th International Conference on Machine Learning (ICML).
- M. Kusner, J. Loftus, C. Russell, and R. Silva.
  Counterfactual Fairness.
  In *NIPS*, pages 4066–4076, 2017.
- S. L. Lauritzen.
  The EM Algorithm for Graphical Association Models with Missing Data.
  Computational Statistics and Data Analysis, 19(2):191–201, 1995.
- R. J. A. Little and D. B. Rubin.
  Statistical Analysis with Missing Data.
  Wiley, 1987.
- K. Murphy.
  Dynamic Bayesian Networks: Representation, Inference and Learning.
  PhD thesis, UC Berkeley, Computer Science Division, 2002.
- J. Pearl and M. Glymour and.
  Causal Inference in Statistics: a Primer.
  Wiley, 2016.
- C. Riggelsen.
  Learning Parameters of Bayesian Networks from Incomplete Data via Importance Sampling.
  International Journal of Approximate Reasoning, 42(1–2):69–83, 2006.
- S. Roweis and Z. Ghahramani.
  A Unifying Review of Linear Gaussian Models.
  Neural Computation, 11(2):305–345, 1999.

References IV

- D. B. Rubin.
  Inference and Missing Data.
  Biometrika, 63:581–592, 1976.
- C. Russell, M. J. Kusner, J. R. Loftus, and R. Silva.
  When Worlds Collide: Integrating Different Countefactual Assumptions in Fairness.
  In *NIPS*, volume 30, pages 6414–6423, 2017.
- G. Schwarz.
  Estimating the Dimension of a Model.
  The Annals of Statistics, 6(2):461–464, 1978.
- M. Scutari.
  Bayesian Network Models for Incomplete and Dynamic Data.
  Statistica Neerlandica, 74(3):397–419, 2020.
- M. Scutari and J.-B. Denis.
  Bayesian Networks with Examples in R.
  Chapman & Hall, 2nd edition, 2021.
- M. Scutari, C. E. Graafland, and J. M. Gutiérrez.
  Who Learns Better Bayesian Network Structures: Accuracy and Speed of Structure Learning Algorithms.
  International Journal of Approximate Reasoning, 115:235–253, 2019.
  This is an extended version of the “Who Learns Better Bayesian Network Structures: Constraint-Based, Score-Based or Hybrid Algorithms?” PMLR paper.
- M. Tanner and W. Wong.
  The Calculation of Posterior Distributions by Data Augmentation.
  Journal of the American Statistical Association, 82(398):528–540, 1987.
- C. E. Weatherburn.
  A First Course in Mathematical Statistics.
  Cambridge University Press, 1961.

References V

- K. Yamazaki and Y. Motomura.
  Hidden Node Detection between Observable Nodes Based on Bayesian Clustering.
  Entropy, 21(1):32, 2019.
- M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi.
  Fairness Constraints: a Flexible Approach for Fair Classification.
  Journal of Machine Learning Research, 20:1–42, 2019.
- W. Zucchini and I. L. MacDonald.
  Hidden Markov Models for Time Series: An Introduction Using R.
  Chapman & Hall, 2009.

Thanks!

Any questions?