Algorithms for learning Bayesian networks

James Cussens, University of Bristol

BIAS22, 2022-09-06

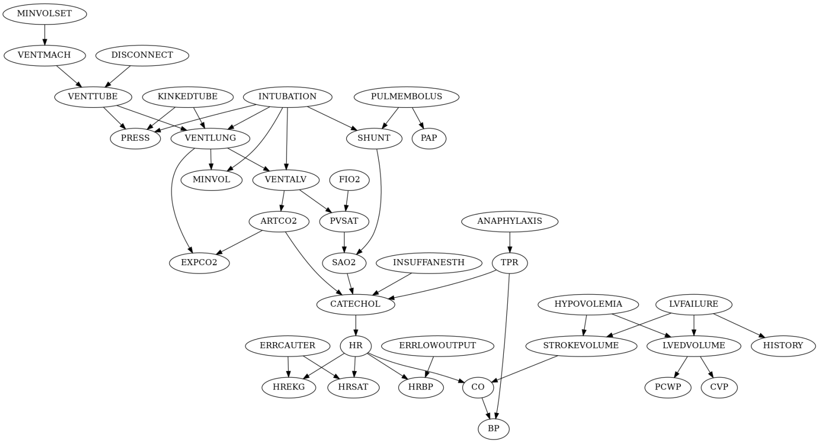
So many Bayesian network learning algorithms

Algorithms in the benchpress system

Agenda

1. Bayesian networks
2. Constraint-based learning of Bayesian networks
3. Causal models (estimating causal effects, adjustment sets)
4. Score-based learning of Bayesian networks
5. Evaluation

The Alarm Bayesian network



Bayesian networks define probability distributions

- Let's define a Bayesian network (BN) with 3 binary variables: X, Y and Z.
- We choose a *structure* which is a directed acyclic graph (DAG):

$$X \rightarrow Y \rightarrow Z$$

- and *parameters* which are a bunch of conditional probability distributions: $P(X)$, $P(Y|X)$, $P(Z|Y)$.
- Each variable gets a distribution conditional on its *parents*.
- The BN defines a (joint) probability distribution:

$$P(X = x, Y = y, Z = z) = \\ P(X = x)P(Y = y|X = x)P(Z = z|Y = y)$$

Sampling data from a BN

- Sampling a datapoint from a BN X → Y → Z is easy using *ancestral sampling*.
- Here we first sample from $P(X)$, suppose we get $X = 1$.
- Next we sample a value for Y from $P(Y|X = 1)$, suppose we get $Y = 0$.
- Finally sample a value for Z from $P(Z|Y = 0)$.
- In general: sample values for parents before children. This is always possible since we have a DAG.
- To generate a dataset with n datapoints, just repeat n times (giving an iid sample).
- (I have a demo, perhaps later . . .)

Bayesian network learning

- It's just this: given a dataset, estimate the BN it was generated from.
- So BN learning is a form of unsupervised learning.
- Sometimes we just want to guess the structure (DAG).
- But we can also estimate the parameters (typically after estimating the DAG) and so get an estimate of the data-generating probability distribution.
- OK, but why bother?

What's the point?

1. To estimate the data-generating distribution (density estimation): learn structure and parameters from observational data.
2. To estimate conditional independence relations between variables (model selection): learn structure from observational data.
3. To estimate a *causal* model (causal discovery): learn structure (and typically parameters) from observational data and/or experimental data.

We will focus on causal discovery since it's the most interesting.

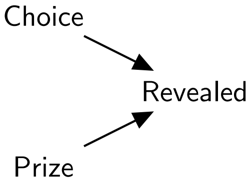
Conditional independence

- Suppose $P(X, Z|Y) = P(X|Y)P(Z|Y)$ for some joint distribution P over the random variables X, Y and Z.
- We say X and Z are independent conditional on Y (in distribution P).
- Notation: $(X \perp Z|Y)_P$ or just $X \perp Z|Y$ if it's obvious which P it is.
- Intuition: once we know the value of Y (whatever it might be) then knowing the value of X does not help us predict the value of Z (and vice-versa).
- In general, we deal with sets of random variables, e.g. $\{A, B\} \perp \{C\}|D, E$ or $\{B\} \perp \{C, E\}|\emptyset$.
- Every joint probability distribution P has a corresponding (finite) list of conditional independence statements associated with it.

Bayesian networks and conditional independence

- A BN structure (i.e. a DAG) can be seen as a compact way of encoding a set of conditional independence (CI) relations: the set of CI relations obeyed by all distributions which can be defined using that DAG.
- There are two methods for checking whether some CI relation (e.g. $X \perp Z|Y$) is implied by a DAG:
  1. *d*-separation in the DAG
  2. separation in moralised minimal ancestral graph

The (inevitable) Monty Hall example



- There is only one implied CI relation: $\text{Choice} \perp \text{Prize}$.
- In particular: $\text{Choice} \not\perp \text{Prize} | \text{Revealed}$.
- This is the only DAG with variables Choice, Prize and Revealed whose set of implied CI relations is $\{\text{Choice} \perp \text{Prize}\}$.

Constraint-based learning

- The connection between CI relations and DAGs leads to an 'obvious' method for DAG (i.e. BN structure) learning.
- Given some data on, say, variables X, Y and Z, do statistical tests on the data (e.g. chi-squared) to estimate which CI relations hold in the data-generating distribution.
- And then find a DAG which implies (only) the CI relations that hold according to these statistical tests.

A successful example of constraint based learning

$$G_1 = \begin{tikzcd} Y \arrow[dr] & \\ & Z \\ X \arrow[ur] & \end{tikzcd}$$

G_1 is the only DAG which implies this set of CI relations: $X \perp Y$, $X \not\perp Z$, $Y \not\perp Z$, $X \not\perp Y|Z$, $X \not\perp Z|Y$, $Y \not\perp Z|X$.

$$G_2 = \begin{tikzcd} Y \arrow[dr] \arrow[dd] & \\ & Z \\ X \arrow[ur] & \end{tikzcd}$$

There are P^* for G_2 where $(X \perp Y)_{P^*}$, but almost all distributions P for G_2 have $(X \not\perp Y)_P$. The assumption that the true distribution is not like P^* is known as *faithfulness*.

Problems for constraint-based learning

1. These 3 DAGs: $X \to Y \to Z$, $X \leftarrow Y \to Z$, $X \leftarrow Y \leftarrow Z$ are *Markov equivalent*.
   - That means they encode the same set of conditional independence relations, namely $\{X \perp Z | Y\}$.
   - The 3 DAGs represent different causal models, but observational data alone cannot pick out the right one.
2. Statistical tests, particularly with small datasets and/or large conditioning sets, don't always give the right answer. And the answer depends on some choice of confidence value.
3. Doing the tests may be time-consuming.

Constraint-based learning in practice

- Algorithms for constraint-based learning of BNs aim to do as few tests as possible to narrow down the set of BNs consistent with the test results.
- For example, the seminal PC algorithm first of all estimates the *undirected skeleton* of the DAG and then later attempts to orient the graph edges.
- There's an edge between X and Y if and only if there is some *separating set* S such that $X \perp Y | S$.

Inferring latent variables

- A nice thing about constraint-based learning is that it makes it possible to infer the existence of latent (i.e. hidden) variables.

$$G_3 = \begin{tikzcd} X_1 \arrow[d] & & X_4 \arrow[d] \\ X_2 & L \arrow[l] \arrow[r] & X_3 \end{tikzcd}$$

- Suppose the true data-generating DAG were G_3 but variable L was latent, so we only had observed data on X_1, X_2, X_3, X_4.
- The CI relations on the X_i are $X_1 \perp X_3, X_1 \perp X_4, X_2 \perp X_4, X_1 \perp X_3 | X_4, X_2 \perp X_4 | X_1, X_1 \perp X_4 | X_2, X_1 \perp X_4 | X_3$.
- There is no DAG on the X_i consistent with these CI relations so an algorithm like FCI (Fast Causal Inference) or RFCI (Really Fast Causal Inference) could infer the existence of a latent variable.

Causal Bayesian Networks

- In this section I will shamelessly pilfer material associated with the `dagitty` software for creating, drawing and analysing causal DAGs.
- `dagitty` is not concerned with learning DAGs!
- In fact, in applications of causal DAGs people do not use learning to get a DAG (Johannes Textor, Simons Institute talk). See, for example, Ferguson *et al.*[1]

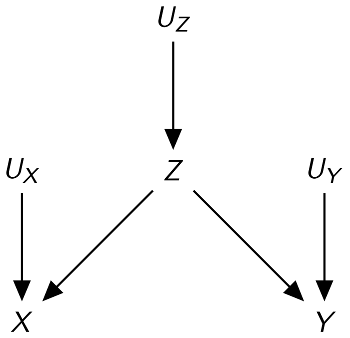
[1] Karl D Ferguson et al. "Evidence synthesis for constructing directed acyclic graphs (ESC-DAGs): a novel and systematic method for building directed acyclic graphs". In: *International Journal of Epidemiology* 49.1 (July 2019), pp. 322–329.

Causal DAGs in a nutshell

"In a nutshell, a DAG is a graphic model that depicts a set of hypotheses about the causal process that generates a set of variables of interest. An arrow $X \rightarrow Y$ is drawn if there is a direct causal effect of X on Y. Intuitively, this means that the natural process determining Y is directly influenced by the status of X, and that altering X via external intervention would also alter Y."[2]

[2] Johannes Textor. *Drawing and Analyzing Causal DAGs with DAGitty*. 2020.

Interventions and graph surgery

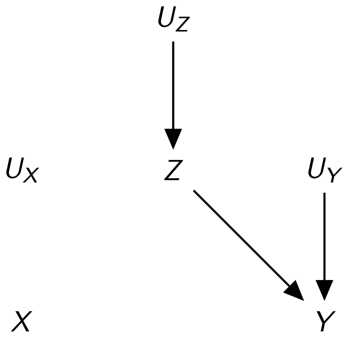


- X is ice cream sales
- Y is crime rates
- Z is temperature
- $P(Y = y|X = x)$

Example from Pearl *et al*[3]

[3] Judea Pearl, Madelyn Glymour, and Nicholas P. Jewell. *Causal Inference in Statistics: A Primer*. Wiley, 2016.

Interventions and graph surgery



- X is ice cream sales
- Y is crime rates
- Z is temperature
- $P(Y = y | \mathrm{do}(X = x))$

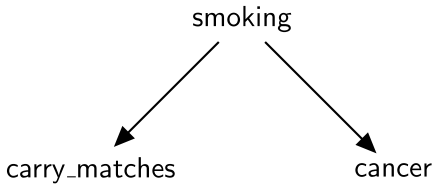
Example from Pearl *et al*[3]

[3] Judea Pearl, Madelyn Glymour, and Nicholas P. Jewell. *Causal Inference in Statistics: A Primer*. Wiley, 2016.

Identifying causal effects

"A key question in Epidemiology (and many other empirical sciences) is: how can we infer the causal effect of an exposure on an outcome of interest from an observational study? . . . If the assumptions encoded in a given diagram hold, then it is sometimes possible to devise an identification strategy from that diagram, by which it would be possible to devise an unbiased estimate of a causal effect from observed data." [ibid.]

Identifying causal effects



"[Assuming the above is the true causal model], would we adjust for smoking, e.g. by weighted averaging of separate effect estimates for smokers and non-smokers or by including smoking status as a covariate in a regression model, we would no longer find a correlation between carrying matches and lung cancer." [ibid]

Score-based learning of BNs

- In a Bayesian approach to learning BNs we have some prior $P(G)$ over possible DAGs.[4]
- And for each DAG we have some prior over parameter values $P(\theta|G)$.

$$P(G|D) \propto P(G)P(D|G) = P(G)\int_{\theta} P(D|\theta, G)P(\theta|G)d\theta$$

- Just find $\arg\max_G P(G|D)$, where D is the observed data.
- This is an example of *score-based learning*, where posterior probability is the score.
- Choose $P(\theta|G)$ so that $P(D|G)$ has a convenient closed-form. Can choose a uniform prior.

[4] There is nothing particularly Bayesian about BNs. Some people, particularly statisticians, prefer to call them *directed graphical models* or *recursive graphical models*.

Choices for score-based learning

- Which score?
  - If Bayesian, which prior?
  - If penalised likelihood, where we penalise for too many edges=parameters (e.g. ℓ_0, ℓ_1), which penalty?
- How/whether to find a score-optimal BN?
  - Finding a guaranteed optimal BN ('exact' learning) can be a slow (or practically impossible) task[5].
  - Heuristic algorithms: how to get a reasonably high-scoring BN reasonably quickly?
- If we have information additional to the data, how to use it?

[5] David M. Chickering, David Heckerman, and Christopher Meek. "Large-Sample Learning of Bayesian Networks is NP-Hard". In: *Journal of Machine Learning Research* 20 (Oct. 2004), pp. 1287–1330.

Strategies for exact learning

Use an existing solving strategy for discrete optimisation ...

- Dynamic programming[6]
- A^{*7}
- Weighted MAX-SAT[8]
- Integer linear programming[9]

---

[6] Tomi Silander and Petri Myllymäki. "A Simple Approach for Finding the Globally Optimal Bayesian Network Structure". In: *UAI*. 2006.

[7] Changhe Yuan and Brandon Malone. "Learning Optimal Bayesian Networks: A Shortest Path Perspective". In: *Journal of Artificial Intelligence Research* 48 (Oct. 2013), pp. 23–65.

[8] James Cussens. "Bayesian network learning by compiling to weighted MAX-SAT". In: *UAI 2008*.

[9] Tommi Jaakkola et al. "Learning Bayesian Network Structure using LP Relaxations". In: *AISTATS 2010*, James Cussens. "Bayesian network learning with cutting planes". In: *UAI 2011*.

Strategies for heuristic learning

Use an existing solving strategy for discrete optimisation . . .

- Local search e.g. hill climbing

Do continuous optimisation instead

- Inspired by lasso and glasso
- Use a suitable (ℓ_0 or ℓ_1) penalty
- Zero values correspond to non-edges.
- Might have to round down small values to zero to get enough sparsity.

Unsupervised learning as multiple coupled supervised learning

- Suppose we have to learn a BN with variables $X_1, \ldots X_p$.
- One option: view each X_i as a response variable and do, say lasso (i.e. ℓ_1) regression, using all other variables as predictors.
- Draw an edge from X_j to X_i iff X_j is chosen as a predictor for X_i.
- Can use, say a deep learning approach,[10] to predict each X_i from the others.
- Problem: this will typically lead to a cyclic graph.

[10] Sébastien Lachapelle et al. "Gradient-Based Neural DAG Learning". In: *ICLR 20*.

Smooth acyclicity constraint

From the NOTEARS paper[11]:

> *In order to make (3) amenable to black-box optimization, we propose to replace the combinatorial acyclicity constraint $G(W) \in \mathcal{D}$ in (3) with a single smooth equality constraint $h(W) = 0$. Ideally, we would like a function $h : \mathbb{R}^{d\times d} \to \mathbb{R}$ that satisfies the following desiderata:*
>
> 1. *$h(W) = 0$ if and only if W is acyclic;*
> 2. *The values of h quantify the "DAG-ness" of the graph;*
> 3. *h is smooth*
> 4. *h and its derivatives are easy to compute.*

[11] Xun Zheng et al. "DAGs with NO TEARS: Continuous Optimization for Structure Learning". In: *NeurIPS 2018*.

The need for empirical testing

- OK, so what actually works?
- Reading papers will not answer that question!
- Finding, installing and comparing all these algorithms is potentially a nightmare.
- Fortunately, the snakemake-based `benchpress` system makes this a whole lot easier...

benchpress

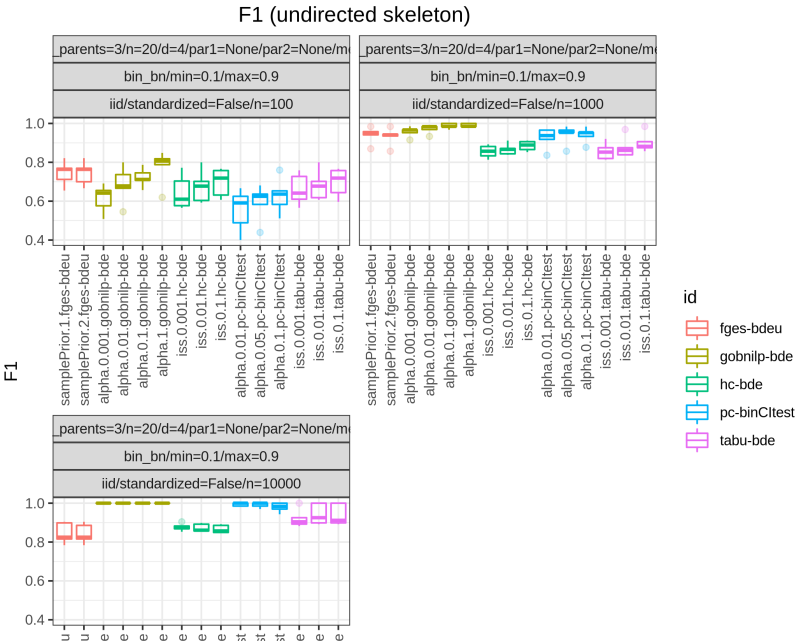
- Main developer of benchpress[12]: Felix Rios (formerly at Basel, now at KTH, Stockholm
- User writes config file (JSON format) which specifies:
  1. How to (randomly) generate 'true' DAGs (can be fixed)
  2. How to (randomly) parameterise the 'true' DAGs
  3. How to generate data from the parameterised true DAGs
  4. Which DAG learning algorithms to use (and with which hyperparameter settings)
  5. How to evaluate the learned DAGs.
- Learning algorithms typically run in a container using `singularity` (no installation required!)
- Snakemake works out how to organise the various jobs.
- Uses as many cores as you have available.

[12] Felix L. Rios, Giusi Moffa, and Jack Kuipers. *Benchpress: a scalable and platform-independent workflow for benchmarking structure learning algorithms for graphical models.* arXiv: 2107.03863. 2021.

Benchpress results (DAGs with many vertices)

Let's have a look at that paper.

Small discrete DAGs



Small continuous DAGs