
Efficient Algorithms for Bayesian Network Parameter Learning from Incomplete Data

Guy Van den Broeck* and Karthika Mohan* and Arthur Choi and Adnan Darwiche and Judea Pearl

Computer Science Department

University of California, Los Angeles

{guyvdb, karthika, aychoi, darwiche, judea}@cs.ucla.edu

Abstract

We propose a family of efficient algorithms for learning the parameters of a Bayesian network from incomplete data. Our approach is based on recent theoretical analyses of missing data problems, which utilize a graphical representation, called the missingness graph. In the case of MCAR and MAR data, this graph need not be explicit, and yet we can still obtain closed-form, asymptotically consistent parameter estimates, without the need for inference. When this missingness graph is explicated (based on background knowledge), even partially, we can obtain even more accurate estimates with less data. Empirically, we illustrate how we can learn the parameters of large networks from large datasets, which are beyond the scope of algorithms like EM (which require inference).

1 INTRODUCTION

When learning the parameters of a Bayesian network from data with missing values, the conventional wisdom among machine learning practitioners is that there are two options: use *expectation maximization* (EM) or *gradient methods* (to optimize the likelihood); see, e.g., Darwiche (2009), Koller and Friedman (2009), Murphy (2012), Barber (2012). Both of these approaches, however, suffer from the following disadvantages, which prevent them from scaling to large networks and datasets; see also Thiesson, Meek, and Heckerman (2001). First, they are *iterative*, and hence may need many passes over a potentially large dataset. Next, these algorithms may get stuck in *local optima*, which means that, in practice, one must run these algorithms multiple times with different initial seeds, and hope that one of them leads to a good optimum. Last, but not least, these methods *require inference* in the network, which places a hard limit

on the networks where EM and gradient methods can even be applied, namely for networks where exact inference is tractable, i.e., they have small enough treewidth, or sufficient local structure (Chavira & Darwiche, 2006, 2007).

Recently, Mohan, Pearl, and Tian (2013) showed that the joint distribution of a Bayesian network is recoverable from incomplete data, including data that falls under the classical *missing at random* assumption (MAR), but also for a broad class of data that is not MAR. Their analysis is based on a graphical representation for missing data problems, called the *missingness graph*, where one explicates the *causal mechanisms* that are responsible for the missingness in an incomplete dataset. Using this representation, they provide a way to decide whether a given query (e.g., a joint marginal) is recoverable, and if so, they provide a *closed-form* expression (in terms of the observables) for an asymptotically consistent estimate.

Building on the theoretical foundations set by Mohan et al. (2013), we propose a family of practical and efficient algorithms for estimating the parameters of a Bayesian network from incomplete data. For the cases of both MCAR and MAR data, where the missingness graph need not be explicit, we start by deriving the closed-form parameter estimates, as implied by Mohan et al. (2013). We next show how to obtain better estimates, by exploiting a factorized representation that allows us to aggregate distinct, yet asymptotically equivalent estimates, hence utilizing more of the data. We also show how to obtain improved estimates, when the missingness graph is only partially explicated (based on domain or expert knowledge). As in Mohan et al. (2013), all of our estimation algorithms are asymptotically consistent, i.e., they converge to the true parameters of a network, in the limit of infinite data.

As we show empirically, our parameter estimation algorithms make learning from incomplete data viable for larger Bayesian networks and larger datasets, that would otherwise be beyond the scope of algorithms such as EM and gradient methods. In particular, our algorithms (1) are non-iterative, requiring only a single pass over the data, (2) provide estimates in closed-form, and hence do not suffer from

*Both authors contributed equally to this work. GVdB is also affiliated with KU Leuven, Belgium.

$\begin{array}{cc} X & Y \\ \bar{x} & y \\ x & ? \\ x & \bar{y} \\ \bar{x} & ? \\ \dots & \dots \end{array}$	$\begin{array}{cc} X & Y Y^* & R_Y \\ \bar{x} & y & y & \text{ob} \\ x & ? & \text{mi} & \text{unob} \\ x & \bar{y} & \bar{y} & \text{ob} \\ \bar{x} & ? & \text{mi} & \text{unob} \\ \dots & \dots & \dots & \dots \end{array}$
(a) Dataset \mathcal{D} and DAG	(b) Missingness Dataset \mathcal{D} and DAG

Figure 1: Datasets and DAGs.

local optima, and (3) require no inference, which is the primary limiting factor for the scalability of algorithms such as EM. We note that these advantages are also available when learning Bayesian networks from *complete* data.

2 TECHNICAL PRELIMINARIES

In this paper, we use upper case letters (X) to denote variables and lower case letters (x) to denote their values. Variable sets are denoted by bold-face upper case letters (\mathbf{X}) and their instantiations by bold-face lower case letters (\mathbf{x}). Generally, we will use X to denote a variable in a Bayesian network and \mathbf{U} to denote its parents. A network parameter will therefore have the general form $\theta_{x|\mathbf{u}}$, representing the probability $\Pr(X=x|\mathbf{U}=\mathbf{u})$.

Given an incomplete dataset \mathcal{D} , we want to learn the parameters of the Bayesian network \mathcal{N} that the dataset originated from. This network induces a distribution $\Pr(\mathbf{X})$, which is in general unknown; instead, we only have access to the dataset \mathcal{D} .

2.1 MISSING DATA: AN EXAMPLE

As an illustrative example, consider Figure 1(a), depicting a dataset \mathcal{D} , and the directed acyclic graph (DAG) G of a Bayesian network, both over variables X and Y . Here, the value for variable X is always observed in the data, while the value for variable Y can be missing. In the graph, we denote a variable that is always observed with a double-circle. Now, if we happen to know the mechanism that causes the value of Y to become missing in the data, we can include it in our model, as in Figure 1(b). Here, we use a variable R_Y to represent the mechanism that controls whether the value of variable Y is missing or observed. Further, we witness the value of Y , or its missingness, through a proxy variable Y^* , as an observation. Such a graph, which explicates the missing data process, is called a *missingness graph*.

In our example, we augmented the dataset and graph with new variables R_Y , representing the causal mechanism that dictates the missingness of the value of Y . This mechanism can be active (Y is unobserved), denoted by $R_Y = \text{unob}$. Otherwise, the mechanism is passive (Y is observed), de-

noted by $R_Y = \text{ob}$. Variable Y^* acts as a proxy on the value of Y , which may be an observed value y , or a special value (mi) when the value of Y is missing. The value of Y^* thus depends functionally on variables R_Y and Y :

$$Y^* = f(R_Y, Y) = \begin{cases} \text{mi} & \text{if } R_Y = \text{unob} \\ Y & \text{if } R_Y = \text{ob} \end{cases}$$

That is, when $R_Y = \text{unob}$, then $Y^* = \text{mi}$; otherwise $R_Y = \text{ob}$ and the proxy Y^* assumes the observed value of Y .

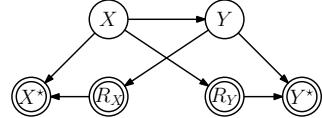


Figure 1: Datasets and DAGs.

Figure 2 highlights a more complex example of a missingness graph, with causal mechanisms R_X and R_Y that depend on other variables. In Section 2.3, we highlight how different missing data problems (such as MCAR and MAR) lead to different types of missingness graphs.

2.2 LEARNING WITH MISSINGNESS GRAPHS

Given a Bayesian network with DAG G , and an incomplete dataset \mathcal{D} , we can partition the variables \mathbf{X} into two sets: the fully-observed variables \mathbf{X}_o , and the partially-observed variables \mathbf{X}_m that have missing values in \mathcal{D} . As in our example above, one can take into account knowledge about the processes responsible for the missingness in \mathcal{D} . More specifically, we can incorporate the causal mechanisms that cause the variables \mathbf{X}_m to have missing values, by introducing (1) variables \mathbf{R} representing the *causal mechanisms* that are responsible for missingness in the data, and (2) variables \mathbf{X}_m^* that act as *proxies* to the variables \mathbf{X}_m . This augmented Bayesian network, which we refer to as the *missingness graph* \mathcal{N}^* , has variables $\mathbf{X}_o, \mathbf{X}_m^*, \mathbf{R}$ that are fully-observed, and variables \mathbf{X}_m that are only partially-observed. The missingness graph \mathcal{N}^* thus induces a distribution $\Pr(\mathbf{X}_o, \mathbf{X}_m, \mathbf{X}_m^*, \mathbf{R})$. Using the missingness graph, we want to draw conclusions about the partially-observed variables, by reasoning about the fully-observed ones.

Missingness graphs can serve as a powerful tool for analyzing missing data problems; see, e.g., Thoemmes and Mohan (2015), Francois and Leray (2007), Darwiche (2009), Koller and Friedman (2009). As Mohan et al. (2013) show, one can exploit the conditional independencies that missingness graphs encode, in order to extract *asymptotically consistent* estimates for missing data problems, including MNAR ones, whose underlying assumptions would put it out of the scope of existing techniques.¹ Mohan et al.

¹Note that maximum-likelihood estimation is asymptotically consistent, although a consistent estimator is not necessarily a maximum-likelihood estimator; see, e.g., Wasserman (2011).

(2013) identify conditions on \mathcal{N}^* that allow the original, partially-observed distribution $\Pr(\mathbf{X}_o, \mathbf{X}_m)$ to be identified from the fully-observed distribution $\Pr(\mathbf{X}_o, \mathbf{X}_m^*, \mathbf{R})$. However, in practice, we only have access to a dataset \mathcal{D} , and the corresponding *data distribution* that it induces:

$$\Pr_{\mathcal{D}}(\mathbf{x}_o, \mathbf{x}_m^*, \mathbf{r}) = \frac{1}{N} \mathcal{D}\#(\mathbf{x}_o, \mathbf{x}_m^*, \mathbf{r}),$$

where N is the number of instances in dataset \mathcal{D} , and where $\mathcal{D}\#(\mathbf{x})$ is the number of instances where instantiation \mathbf{x} appears in the data.² However, the data distribution $\Pr_{\mathcal{D}}$ tends to the true distribution \Pr (over the fully-observed variables), as N tends to infinity.

Building on the theoretical foundations set by Mohan et al. (2013), we shall propose a family of efficient and scalable parameter estimation algorithms from incomplete data. In essence, we will show how to query the observed data distribution $\Pr_{\mathcal{D}}$, in order to make inferences about the true, underlying distribution $\Pr(\mathbf{X}_o, \mathbf{X}_m)$ (in particular, we want the conditional probabilities that parameterize the given Bayesian network). As we shall discuss, in many cases the missingness graph need not be explicit. In other cases, when there is knowledge about the missingness graph, even just partial knowledge, we can exploit it, in order to obtain more accurate parameter estimates.

2.3 CATEGORIES OF MISSINGNESS

An incomplete dataset is categorized as *Missing Completely At Random* (MCAR) if all mechanisms \mathbf{R} that cause the values of variables \mathbf{X}_m to go missing, are marginally independent of \mathbf{X} , i.e., where $(\mathbf{X}_m, \mathbf{X}_o) \perp\!\!\!\perp \mathbf{R}$. This corresponds to a missingness graph where no variable in $\mathbf{X}_m \cup \mathbf{X}_o$ is a parent of any variable in \mathbf{R} . For example, if all mechanisms \mathbf{R} are root nodes, then the problem is MCAR. Note that the missingness graph of Figure 1(b) implies an MCAR dataset.

An incomplete dataset is categorized as *Missing At Random* (MAR) if missingness mechanisms are conditionally independent of the partially-observed variables given the fully-observed variables, i.e., if $\mathbf{X}_m \perp\!\!\!\perp \mathbf{R} | \mathbf{X}_o$. This corresponds to a missingness graph where variables \mathbf{R} are allowed to have parents, as long as none of them are partially-observed. In the example missingness graph of Figure 1(b), adding an edge $X \rightarrow R_Y$ results in a graph that yields MAR data. This is a stronger, variable-level definition of MAR, which has previously been used in the machine learning literature (Darwiche, 2009; Koller & Friedman, 2009), in contrast to the event-level definition of MAR that is prevalent in the statistics literature (Rubin, 1976).

²Note that the data distribution is well-defined over the variables $\mathbf{X}_o, \mathbf{X}_m^*$ and \mathbf{R} , as they are fully-observed in the augmented dataset, and that $\Pr_{\mathcal{D}}$ can be represented compactly in space linear in N , as we need not explicitly represent those instantiations \mathbf{x} that were not observed in the data.

Table 1: Summary of Estimation Algorithms

Algorithm	Description (Section Number)
D-MCAR	Direct Deletion for MCAR data (3.1)
D-MAR	Direct Deletion for MAR data (3.2)
F-MCAR	Factored Deletion for MCAR data (3.3)
F-MAR	Factored Deletion for MAR data (3.3)
I-MAR	Informed Deletion for MAR data (5.1)
IF-MAR	Informed Factored Deletion for MAR data (5.1)

An incomplete dataset is categorized as *Missing Not At Random* (MNAR) if it is not MAR (and thus not MCAR). For example, the DAG in Figure 2 corresponds to an MNAR missingness graph. This is because the mechanism R_X has a partially-observed variable as a parent; further, mechanism R_Y has a partially-observed parent X .

3 CLOSED-FORM LEARNING

We now present algorithms to learn the parameters of a Bayesian network \mathcal{N} from data \mathcal{D} . We first consider the classical missing data assumptions, with no further knowledge about the missingness graph that generated the data.

To estimate the conditional probabilities $\theta_{x|u}$ that parameterize a Bayesian network, we estimate the joint distributions $\Pr(X, U)$, which are subsequently normalized, as a conditional probability table. Hence, it suffices, for our discussion, to estimate marginal distributions $\Pr(\mathbf{Y})$ for families $\mathbf{Y} = \{X\} \cup U$. We let $\mathbf{Y}_o = \mathbf{Y} \cap \mathbf{X}_o$ denote the observed variables in \mathbf{Y} , and $\mathbf{Y}_m = \mathbf{Y} \cap \mathbf{X}_m$ denote the partially-observed variables. Further, we let $\mathbf{R}_Z \subseteq \mathbf{R}$ denote the missingness mechanisms for the partially-observed variables Z . Through \mathcal{D} , we have access to the data distribution $\Pr_{\mathcal{D}}$ over the variables in the missingness dataset. Appendix D illustrates our learning algorithms on a concrete dataset and Table 1 gives an overview of the different estimation algorithms in this paper.

3.1 DIRECT DELETION FOR MCAR

The statistical technique of *listwise deletion* is perhaps the simplest technique for performing estimation with MCAR data: we simply delete all instances in the dataset that contain missing values, and estimate our parameters from the remaining dataset, which is now complete. Of course, with this technique, we potentially ignore large parts of the dataset. The next simplest technique is perhaps pairwise deletion, or available-case analysis: when estimating a quantity over a pair of variables X and Y , we delete just those instances where variable X or variable Y is missing.

Consider now the following, more general, deletion technique, which is expressed in the terms of causal missingness mechanisms. In particular, to estimate the marginals $\Pr(\mathbf{Y})$ of a set of (family) variables \mathbf{Y} , from the data dis-

tribution $\Pr_{\mathcal{D}}$, we can use the estimate:

$$\begin{aligned}\Pr(\mathbf{Y}) &= \Pr(\mathbf{Y}_o, \mathbf{Y}_m | \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \quad \text{by } \mathbf{X}_o, \mathbf{X}_m \perp\!\!\!\perp \mathbf{R} \\ &= \Pr(\mathbf{Y}_o, \mathbf{Y}_m^* | \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \quad \text{by } \mathbf{X}_m = \mathbf{X}_m^* \text{ when } \mathbf{R} = \text{ob} \\ &\approx \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{Y}_m^* | \mathbf{R}_{\mathbf{Y}_m} = \text{ob})\end{aligned}$$

That is, we can estimate $\Pr(\mathbf{Y})$ by using the subset of the data where every variable in \mathbf{Y} is observed (which follows from the assumptions implied by MCAR data). Since the data distribution $\Pr_{\mathcal{D}}$ tends to the true distribution \Pr , this implies a consistent estimate for the marginals $\Pr(\mathbf{Y})$. In contrast, the technique of listwise deletion corresponds to the estimate $\Pr(\mathbf{Y}) \approx \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{Y}_m^* | \mathbf{R}_{\mathbf{X}_m} = \text{ob})$, and the technique of pairwise deletion corresponds to the above, when \mathbf{Y} contains two variables. To facilitate comparisons with more interesting estimation algorithms that we shall subsequently consider, we refer to the more general estimation approach above as *direct deletion*.

3.2 DIRECT DELETION FOR MAR

In the case of MAR data, we cannot use the simple deletion techniques that we just described for MCAR data—the resulting estimates would not be consistent. However, we show next that it is possible to obtain consistent estimates from MAR data, using a technique that is as simple and efficient as direct deletion. Roughly, we can view this technique as deleting certain instances from the dataset, but then re-weighting the remaining ones, so that a consistent estimate is obtained. We shall subsequently show how to obtain even better estimates by factorization.

Again, to estimate network parameters $\theta_{x|\mathbf{u}}$, it suffices to show how to estimate family marginals $\Pr(\mathbf{Y})$, now under the MAR assumption. Let $\mathbf{X}'_o = \mathbf{X}_o \setminus \mathbf{Y}_o$ denote the fully-observed variables outside of the family variables \mathbf{Y} (i.e., $\mathbf{X}_o = \mathbf{Y}_o \cup \mathbf{X}'_o$). We have

$$\begin{aligned}\Pr(\mathbf{Y}) &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_o, \mathbf{Y}_m, \mathbf{X}'_o) \\ &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o) \Pr(\mathbf{Y}_o, \mathbf{X}'_o)\end{aligned}$$

Hence, we reduced the problem to estimating two sets of probabilities. Estimating the probabilities $\Pr(\mathbf{Y}_o, \mathbf{X}'_o)$ is straightforward, as variables \mathbf{Y}_o and \mathbf{X}'_o are fully observed in the data. The conditional probabilities $\Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o)$ contain partially observed variables \mathbf{Y}_m , but they are conditioned on all fully observed variables $\mathbf{X}_o = \mathbf{Y}_o \cup \mathbf{X}'_o$. The MAR definition implies that each subset of the data that fixes a value for \mathbf{X}_o is locally MCAR. Like the MCAR case, we can estimate each conditional probability as

$$\Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o) = \Pr(\mathbf{Y}_m^* | \mathbf{Y}_o, \mathbf{X}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob}).$$

This leads to the following estimation algorithm,

$$\Pr(\mathbf{Y}) \approx \sum_{\mathbf{X}'_o} \Pr_{\mathcal{D}}(\mathbf{Y}_m^* | \mathbf{Y}_o, \mathbf{X}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{X}'_o)$$

Algorithm 1 F-MCAR(\mathbf{y}, \mathcal{D})

Input:

\mathbf{y} : A state of query variables \mathbf{Y}

\mathcal{D} : An incomplete dataset with data distribution $\Pr_{\mathcal{D}}$

Auxiliary:

CACHE: A global cache of estimated probabilities

Function:

```

1: if  $\mathbf{y} = \emptyset$  then return 1
2: if CACHE[ $\mathbf{y}$ ]  $\neq \text{nil}$  then return CACHE[ $\mathbf{y}$ ]
3:  $\mathcal{E} \leftarrow \emptyset$  // Initialize set of estimates
4: for each  $y \in \mathbf{y}$  do
5:    $\mathbf{u} \leftarrow \mathbf{y} \setminus \{y\}$  // Factorize with parents  $\mathbf{u}$ 
6:   add  $\Pr_{\mathcal{D}}(y | \mathbf{u}, \mathbf{R}_y = \text{ob}) \cdot \text{F-MCAR}(\mathbf{u}, \mathcal{D})$  to  $\mathcal{E}$ 
7: CACHE[ $\mathbf{y}$ ]  $\leftarrow$  Aggregate estimates in  $\mathcal{E}$  // E.g., mean
8: return CACHE[ $\mathbf{y}$ ]
```

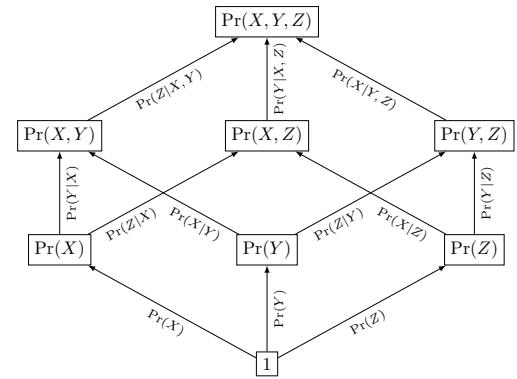


Figure 3: Factorization Lattice of $\Pr(X, Y, Z)$

which uses only the fully-observed variables of the data distribution $\Pr_{\mathcal{D}}$. Note that the summation requires only a single pass through the data, i.e., for only those instantiations of \mathbf{X}'_o that appear in it. Again, $\Pr_{\mathcal{D}}$ tends to the true distribution \Pr , as the dataset size tends to infinity, implying a consistent estimate of $\Pr(\mathbf{Y})$.

3.3 FACTORED DELETION

We now propose a class of deletion algorithms that exploit more data than direct deletion. In the first step, we generate multiple but consistent estimates for the query so that each estimate utilizes different parts of a dataset to estimate the query. In the second step, we aggregate these estimates to compute the final estimate and thus put to use almost all tuples in the dataset. Since this method exploits more data than direct deletion, it obtains a better estimate of the query.

Factored Deletion for MCAR Algorithm 1 implements factored deletion for MCAR. Let the query of interest be $\Pr(\mathbf{Y})$, and let Y^1, Y^2, \dots, Y^n be any ordering of the n

variables in \mathbf{Y} . Each ordering yields a unique factorization:

$$\Pr(\mathbf{Y}) = \prod_{i=1}^n \Pr(Y^i | Y^{i+1}, \dots, Y^n)$$

We can estimate each of these factors independently, on the subset of the data in which all of its variables are fully observed (as in direct deletion), i.e.,

$$\Pr(Y^i | Y^{i+1}, \dots, Y_m^n) = \Pr(Y^i | Y^{i+1}, \dots, Y_m^n, \mathbf{R}_{\mathbf{Z}^i} = \text{ob})$$

where \mathbf{Z}^i is the set of partially-observed variables in the factor. When $|\mathbf{Y}_m| > 1$, we can utilize much more data than direct deletion. See Appendix D, for an example.

So far, we have discussed how a consistent estimate of $\Pr(\mathbf{Y})$ may be computed given a factorization. Now we shall detail how estimates from each factorization can be aggregated to compute more accurate estimates of $\Pr(\mathbf{Y})$. Let k be the number of variables in a family \mathbf{Y} . The number of possible factorizations is $k!$. However, different factorizations share the same sub-factors, which we can estimate once, and reuse across factorizations. We can organize these computations using a lattice, as in Figure 3, which has only 2^k nodes and $k2^{k-1}$ edges. Our algorithm will compute as many estimates as there are edges in this lattice, which is only on the order of $O(n \log n)$, where n is the number of parameters being estimated for a family Y (which is also exponential in the number of variables k). To emphasize the distinction with direct deletion, which uses only those instances in the data where *all* variables in \mathbf{Y} are observed, factored deletion uses any instance in the data where *at least one* variable in \mathbf{Y} is observed.

More specifically, our factored deletion algorithm first estimates the conditional probabilities on the edges of the lattice, each estimate using the subset of the data where its variables are observed. Second, it propagate the estimates, bottom-up. For each node, there are several alternative estimates available, on its incoming edges. There are various ways of aggregating these estimates, such as mean, median, and propagating the lowest-variance estimate.³

Factored Deletion for MAR Algorithm 2 implements factored deletion for MAR. Let $Y_m^1, Y_m^2, \dots, Y_m^n$ be any ordering of the n partially observed variables $\mathbf{Y}_m \subseteq \mathbf{Y}$ and let $\mathbf{X}'_o = \mathbf{X}_o \setminus \mathbf{Y}_o$ denote the fully-observed variables outside of \mathbf{Y} . Given an ordering, we have the factorization:

$$\Pr(\mathbf{Y}) = \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_o, \mathbf{X}'_o) \prod_{i=1}^n \Pr(Y_m^i | \mathbf{Z}_m^{i+1}, \mathbf{X}_o)$$

where $\mathbf{Z}_m^i = \{Y_m^j | i \leq j \leq n\}$. We then proceed in a manner similar to factored deletion for MCAR to estimate individual factors and aggregate estimates to compute $\Pr(\mathbf{Y})$. For equations and derivations, please see Appendix A.

³In initial experiments, all aggregations performed similarly. Reported results use an inverse-variance weighting heuristic.

Algorithm 2 F-MAR(\mathbf{y}, \mathcal{D})

Input:

\mathbf{y} : A state of query variables \mathbf{Y} , consisting of \mathbf{y}_o and \mathbf{y}_m
 \mathcal{D} : An incomplete dataset with data distribution $\Pr_{\mathcal{D}}$

Function:

```

1:  $e \leftarrow 0$                                 // Estimated probability
2: for each  $\mathbf{x}_o$  appearing in  $\mathcal{D}$  that agrees with  $\mathbf{y}_o$  do
3:    $\mathcal{D}_{\mathbf{x}_o} \leftarrow$  subset of  $\mathcal{D}$  where  $\mathbf{x}_o$  holds
4:    $e \leftarrow e + \Pr_{\mathcal{D}}(\mathbf{x}_o) \cdot \text{F-MCAR}(\mathbf{y}_m, \mathcal{D}_{\mathbf{x}_o})$ 
5: return  $e$ 

```

4 EMPIRICAL EVALUATION

To evaluate the learning algorithms we proposed, we simulate partially observed datasets from Bayesian networks, and re-learn their parameters from the data.⁴

In our first sets of experiments, we compare our parameter estimation algorithms with EM, on relatively small networks for MCAR and MAR data. These experiments are intended to observe general trends in our algorithms, in terms of their computational efficiency, but also in terms of the quality of the parameter estimates obtained. Our main empirical contributions are presented in Section 4.3, where we demonstrate the scalability of our proposed estimation algorithms, to larger networks and datasets, compared to EM (even when using approximate inference algorithms).

We consider the following algorithms:

D-MCAR & F-MCAR: direct deletion and factored deletion for MCAR data.

D-MAR & F-MAR: direct deletion and factored deletion for MAR data.

EM- k -JT: EM with k random restarts, jointree inference.

F-MAR + EM-JT: EM seeded with F-MAR estimates, jointree inference.

Remember that D-MCAR and F-MCAR are consistent for MCAR data only, while D-MAR and F-MAR are consistent for general MAR data. EM is consistent for MAR data, but only if it converges to maximum-likelihood estimates.

We evaluate the learned parameters in terms of their *likelihood* on independently generated, fully-observed test data, and the *Kullback–Leibler divergence* (KLD) between the original and learned Bayesian networks. We report per-instance log-likelihoods (which are divided by dataset size). We evaluate the learned models on unseen data, so all learning algorithms assume a symmetric Dirichlet prior

⁴An implementation of our system is available at <http://reasoning.cs.ucla.edu/deletion>.

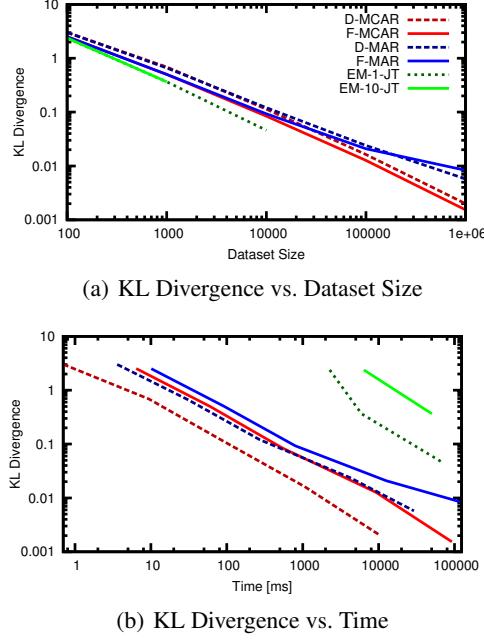


Figure 4: Learning the `alarm` network from MCAR data.

on the network parameters with a concentration parameter of 2 (which corresponds to Laplace smoothing).

4.1 MCAR DATA

First, we consider learning from *MCAR data*, evaluating the quality of the parameters learned by each algorithm. We simulate training sets of increasing size, from a given Bayesian network, selecting 30% of the variables to be partially observed, and removing 70% of their values completely at random. All reported numbers are averaged over 32 repetitions with different learning problems. When no number is reported, a 5 minute time limit was exceeded.

To illustrate the trade-off between data and computational resources, Figure 4 plots the KLDs as a function of dataset size and time; further results are provided in Table 5 of Appendix B. First, we note that in terms of the final estimates obtained, there is no advantage in running EM with restarts: EM-1-JT and EM-10-JT learn almost identical models. This indicates that the likelihood landscape for MCAR data has few local optima, and is easy to optimize. Hence, EM may be obtaining maximum-likelihood estimates in these cases. In general, maximum-likelihood estimators are more statistically efficient (asymptotically) than other estimators, i.e., they require fewer samples. However, other estimators (such as method-of-moments) can be more computationally efficient; see, e.g., Wasserman (2011). We also observe this trend here. EM obtains better estimates with smaller datasets, with smaller KLDs. However, direct and factored deletion (D-MCAR and F-MCAR) are both orders-of-magnitude faster, and can scale to much larger

datasets, than EM (which requires inference). Further, F-MCAR needs only a modest amount of additional data to obtain comparable estimates.

To compare our direct and factored methods, we see that F-MCAR is slower than D-MCAR, as it estimates more quantities (one for each lattice edge). F-MCAR learns better models, however, as it uses a larger part of the available data. Finally, D-MAR performs worse than F-MCAR and D-MCAR, as it assumes the weaker MAR assumption. All learners are consistent, as all KLDs converge to zero.

4.2 MAR DATA

Next, we consider the more challenging problem of learning from *MAR data*, which we generate as follows: (1) select an m -fraction of the variables to be partially observed, (2) add a missingness mechanism variable R_X for each partially-observed variable X , (3) assign p parents to each R_X , randomly selected from the set of observed variables, giving preference to neighbors of X in the network, (4) sample parameters for the missingness mechanism CPTs from a Beta distribution, (5) sample a complete dataset with R_X values, and (6) hide values of X accordingly.

For our first MAR experiment, we use a small network that is tractable enough for EM to scale to large dataset sizes, so that we can observe trends in this regime. Figure 5(a) shows KLD for the `fire alarm` network, which has only 6 variables (and hence, the complexity of inference is negligible). The missing data mechanisms were generated with $m = 0.3$, $p = 2$, and a Beta distribution with shape parameters 1.0 and 0.5. All numbers are averaged over 64 repetitions with different random learning problems.⁵

There is a significant difference between EM, with and without restarts, indicating that the likelihood landscape is challenging to optimize (compared to MCAR, which we just evaluated). EM-10-JT performs well for small dataset sizes, but stops converging after around 1,000 instances. This could be due to all restarts getting stuck in local optima. The KLD of F-MAR starts off between EM-1-JT and EM-10-JT for small sizes, but quickly outperforms EM. For the largest dataset sizes, it learns networks whose KLD is two orders of magnitude smaller than EM-10-JT. The KLD improves further when we use F-MAR estimates to seed EM. This approach is on par with EM-10 for small datasets, while still converging for large dataset sizes. However, note that using F-MAR to seed EM will not be practical for larger networks, where inference becomes a

⁵On our chosen parameters: (1) the number of repetitions was chosen to produce smooth learning curves; (2) a Beta distribution with shape parameter 1 is uniform, and with parameter 0.5, it is slightly biased (so that it acts more like an MAR, and less like an MCAR, mechanism); (3) $m = 0.3$ corresponds to a low amount of missing data, and later $m = 0.9$ corresponds to high amount; and (4) $p = 2$ encourages sparsity and keeps the CPTs small, although setting p to 1 or 3 does not change the results.

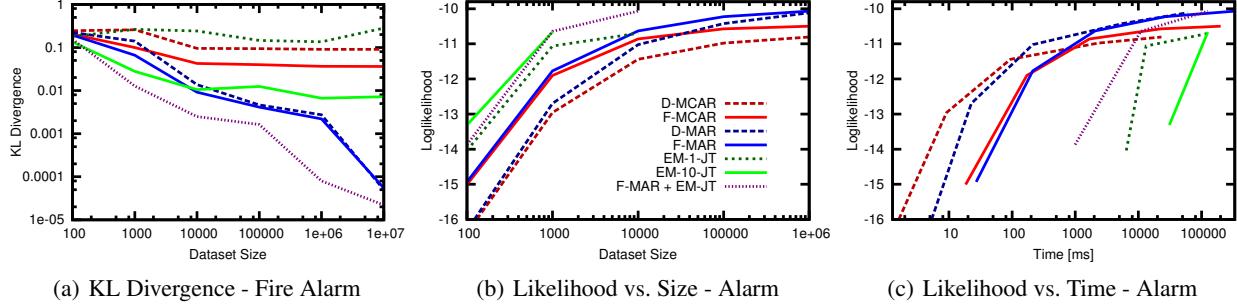


Figure 5: Learning small, tractable Bayesian networks from MAR data. The legend is given in sub-figure (b).

bottleneck. D-MCAR and F-MCAR are not consistent for MAR data, and indeed converge to a biased estimate with a KLD around 0.1. Finally, we observe that the factorized algorithms generally outperform their direct counterparts.

For our second MAR experiment, we work with the classical alarm network, which has 37 variables. The missing data mechanisms were generated with $m = 0.9$, $p = 2$, and a Beta distribution with shape parameters 0.5. All reported numbers are averaged over 32 repetitions, and when no number is reported, a 10 minute time limit was exceeded.

Figures 5(b) and 5(c) show test set likelihood as a function of dataset size and learning time. EM-10-JT performs well for very small dataset sizes, and again outperforms EM-1-JT. However, inference time is non-negligible and EM-10-JT fails to scale beyond 1,000 instances, whereas EM-1-JT scales to 10,000 (as one would expect). The closed-form learners dominate all versions of EM as a function of time, and scale to dataset sizes that are two orders of magnitude larger. EM seeded by F-MAR achieves similar quality to EM-10-JT, while being significantly faster than EM learners with random seeds. D-MAR and F-MAR are more computationally efficient, and can scale to much larger dataset sizes. Further, as seen in Figure 5(c), they can obtain good likelihoods even before the EM methods report their first likelihoods.

4.3 SCALING TO LARGER NETWORKS

In our last set of experiments of this section, we evaluate our algorithms on their ability to scale to larger networks, with higher treewidths, where exact inference is more challenging.⁶ Again, inference is the main factor that limits the scalability of algorithms such as EM, to larger networks and datasets (EM invokes inference as a sub-routine, once per data instance, per iteration). Tables 2 & 3 report results on four networks, where we simulated MAR datasets, as in the previous set of experiments. Each method is given a time limit of 5 or 25 minutes. Appendix C provides results on additional settings. We consider the following methods:

⁶The grid network has 400 variables, munin1 has 189 variables, water has 32 variables, and barley has 48 variables.

EM-JT The EM-10-JT algorithm used in anytime fashion, which returns, given a time limit, the best parameters found in any restart, even if EM did not converge.

EM-BP A variant of EM-JT that uses (loopy) belief propagation for (approximate) inference (in the E-step).

We see that EM-JT, which performs exact inference, does not scale well to these networks. This problem is mitigated by EM-BP, which performs *approximate* inference, yet we find that it also has difficulties scaling (dashed entries indicate that EM-JT and EM-BP did not finish 1 iteration of EM). In contrast, F-MAR, and particularly D-MAR, can scale to much larger datasets. This efficiency is due to the relative simplicity of the D-MAR and F-MAR estimation algorithms: they are not iterative and require only a single pass over the data. In contrast, with EM-BP, the EM algorithm is not only iterative, but the BP algorithm that EM-BP invokes as a sub-routine, is itself an iterative algorithm. As for accuracy, F-MAR typically obtains the best likelihoods (in bold) for larger datasets, while EM-BP can perform better on smaller datasets. We also evaluated D-MCAR and F-MCAR, although they are not in general consistent for MAR data. We find that they scale even further, and can also produce good estimates in terms of likelihood.

5 EXPLOITING MISSINGNESS GRAPHS

We have so far made very general assumptions about the structure of the missingness graph, capturing the MCAR and MAR assumptions. In this section, we show how to exploit additional knowledge about the missingness graph to further improve the quality of our estimates. Having deeper knowledge of the nature of the missingness mechanisms will even enable us to obtain consistent estimators for datasets that are not MAR (in some cases).

5.1 INFORMED DELETION FOR MAR

Consider any MAR dataset, and a missingness graph where each $R \in \mathbf{R}$ depends every observed variable in \mathbf{X}_o . This would be an MAR missingness graph that assumes

Table 2: Log-likelihoods of large networks, with higher treewidths, learned from MAR data (5 min. time limit).

Size		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR
10^2	Grid 90-20-1	-	-57.14	-80.92	-57.01	-80.80	-56.53	Water	-19.10	-18.76	-25.31	-21.76	-25.29	-21.81
10^3		-	-65.41	-38.54	-30.07	-38.27	-29.86		-	-14.73	-19.13	-16.45	-18.93	-16.36
10^4		-	-	-25.95	-23.30	-25.36	-22.88		-	-20.70	-16.66	-14.90	-16.33	-14.67
10^5		-	-	-22.74	-22.01	-21.60	-		-	-	-15.49	-	-14.90	-
10^2	Munin 1	-	-103.72	-115.50	-105.81	-115.41	-104.87	Barley	-	-89.22	-89.54	-89.26	-89.60	-89.14
10^3		-	-69.03	-71.01	-65.91	-70.61	-65.51		-	-74.26	-71.67	-70.46	-71.68	-70.18
10^4		-	-157.23	-56.07	-54.24	-55.46	-		-	-	-56.44	-55.12	-56.40	-
10^5		-	-	-52.00	-	-	-		-	-	-	-	-	-

Table 3: Log-likelihoods of large networks, with higher treewidths, learned from MAR data (25 min. time limit).

Size		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR
10^2	Grid 90-20-1	-	-49.15	-80.00	-56.45	-79.81	-55.94	Water	-18.88	-18.73	-25.84	-22.11	-25.87	-22.25
10^3		-	-53.64	-38.14	-29.32	-37.75	-29.09		-17.63	-14.41	-18.39	-15.95	-18.27	-15.79
10^4		-	-85.65	-26.21	-23.05	-25.45	-22.62		-	-14.52	-15.57	-14.07	-15.24	-13.92
10^5		-	-	-22.78	-21.54	-21.60	-20.79		-	-24.99	-14.17	-13.46	-13.71	-13.19
10^6	Munin 1	-	-	-	-	-	-		-	-	-13.73	-	-	-
10^2		-	-99.15	-114.76	-106.07	-114.66	-105.12	Barley	-89.05	-89.15	-89.57	-89.17	-89.62	-89.03
10^3		-	-67.85	-74.18	-67.81	-73.82	-67.39		-	-70.38	-71.86	-70.54	-71.87	-70.27
10^4		-	-66.62	-57.50	-54.94	-56.96	-54.64		-	-76.48	-56.37	-55.13	-56.33	-
10^5		-	-	-53.07	-51.66	-52.27	-		-	-	-51.31	-	-51.19	-

the least, in terms of conditional independencies, about the causal mechanisms \mathbf{R} . If we know more about the nature of the missingness (i.e., the variables that the \mathbf{R} depend on), we can exploit this to obtain more accurate estimates. Note that knowing the parents of an R is effectively equivalent to knowing the Markov blanket of R (Pearl, 1987), which can be learned from data (Tsamardinos, Aliferis, Statnikov, & Statnikov, 2003; Yaramakala & Margaritis, 2005). With sufficient domain knowledge, an expert may be able to specify the parents of the \mathbf{R} . It suffices even to identify a set of variables that just *contains* the Markov blanket.

Suppose that we have such knowledge of the missing data mechanisms of an MAR problem, namely that we know the subset \mathbf{W}_o of the observed variables \mathbf{X}_o that suffice to separate the missing values from their causal mechanisms, i.e., where $\mathbf{X}_m \perp\!\!\!\perp \mathbf{R} \mid \mathbf{W}_o$. We can exploit this knowledge in our direct deletion algorithm, to obtain improved parameter estimates. In particular, we can reduce the scope of the summation in our direct deletion algorithm from the variables \mathbf{X}'_o (the set of variables in \mathbf{X}_o that lie outside the family \mathbf{Y}), to the variables \mathbf{W}'_o (the set of variables in \mathbf{W}_o that lie outside the family \mathbf{Y}), yielding the algorithm:

$$\begin{aligned} & \Pr(\mathbf{Y}) \\ & \approx \sum_{\mathbf{W}'_o} \Pr_{\mathcal{D}}(\mathbf{Y}_m^* \mid \mathbf{Y}_o, \mathbf{W}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{W}'_o) \end{aligned}$$

Again, we need only consider, in the summation, the instantiations of \mathbf{W}'_o that appear in the dataset.

Table 4: alarm network with Informed MAR data

Size	F-MCAR	D-MAR	F-MAR	ID-MAR	IF-MAR
Kullback-Leibler Divergence					
10^2	1.921	2.365	2.364	2.021	2.011
10^3	0.380	0.454	0.452	0.399	0.375
10^4	0.073	0.071	0.072	0.059	0.053
10^5	0.041	0.021	0.022	0.011	0.010
10^6	0.040	0.006	0.008	0.001	0.001
Test Set Log-Likelihood (Fully Observed)					
10^2	-11.67	-12.13	-12.13	-11.77	-11.76
10^3	-10.40	-10.47	-10.47	-10.42	-10.40
10^4	-10.04	-10.04	-10.04	-10.02	-10.02
10^5	-10.00	-9.98	-9.98	-9.97	-9.97
10^6	-10.00	-9.97	-9.97	-9.96	-9.96

We refer to this algorithm as *informed direct deletion*. By reducing the scope of the summation, we need to estimate fewer sub-terms $\Pr_{\mathcal{D}}(\mathbf{Y}_m^* \mid \mathbf{Y}_o, \mathbf{W}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob})$. This results in a more efficient computation, but further, each individual sub-expression can be estimated on more data. Moreover, our estimates remain consistent. We can similarly replace \mathbf{X}_o by \mathbf{W}_o in the factored deletion algorithm, to obtain an *informed factored deletion* algorithm.

Empirical Evaluation Here, we evaluate the benefits of informed deletion. In addition to the MAR assumption, with this setting, we assume that we know the set of parents \mathbf{W}_o of the missingness mechanism variables. To gen-

erate data for such a mechanism, we select a random set of s variables to form \mathbf{W}_o . We further employ the sampling algorithm previously used for MAR data, but now insist that the parents of \mathbf{R} variables come from \mathbf{W}_o . Table 4 shows likelihoods and KLDs on the `alarm` network, for $s = 3$, and other settings as in the MAR experiments. Informed D-MAR (ID-MAR) and F-MAR (IF-MAR) consistently outperform their non-informed counterparts.

5.2 LEARNING FROM MNAR DATA

A missing data problem that is not MAR is classified as MNAR. Here, the parameters of a Bayesian network may not even be identifiable. Further, maximum-likelihood estimation is in general not consistent, so EM and gradient methods can yield biased estimates. However, if one knows the mechanisms that dictate missingness (in the form of a missingness graph), it becomes possible again to obtain consistent estimates, in some cases (Mohan et al., 2013).

For example, consider the missingness graph of Figure 2, which is an MNAR problem, where both variables X and Y are partially observed, and the missingness of each variable depends on the value of the other. Here, it is still possible to obtain consistent parameter estimates, as $\Pr(X, Y) =$

$$\frac{\Pr(R_X=\text{ob}, R_Y=\text{ob}) \Pr(X^*, Y^*|R_X=\text{ob}, R_Y=\text{ob})}{\Pr(R_X=\text{ob}|Y^*, R_Y=\text{ob}) \Pr(R_Y=\text{ob}|X^*, R_X=\text{ob})}$$

For a derivation, see Mohan et al. (2013). Such derivations for recovering queries under MNAR are extremely sensitive to the structure of the missingness graph. Indeed, the class of missingness graphs that admit consistent estimation has not yet been fully characterized.

6 RELATED WORK

When estimating the parameters of a Bayesian network, maximum-likelihood (ML) estimation is the typical approach, where for incomplete data, the common wisdom among machine learning practitioners is that one needs to use Expectation-Maximization (EM) or gradient methods (Dempster, Laird, & Rubin, 1977; Lauritzen, 1995); see also, e.g., Darwiche (2009), Koller and Friedman (2009), Murphy (2012), Barber (2012). Again, such methods do not scale to large datasets or large networks as (1) they are iterative, (2) they suffer from local optima, and most notably, (3) they require inference in a Bayesian network. Considerable effort has been expended in improving on EM across these dimensions, in order to, for example, (1) accelerate the convergence of EM, and to intelligently sample subsets of a dataset, e.g., Thiesson et al. (2001), (2) escape local optima, e.g., (Elidan, Ninio, Friedman, & Shuurmans, 2002), and (3) use approximate inference algorithms in lieu of exact ones when inference is intractable, e.g., Ghahramani and Jordan (1997), Caffo, Jank, and Jones

(2005). Further, while EM is suitable for data that is MAR (the typical assumption in practice), there are some exceptions, such as work on recommender systems that explicitly incorporate missing data mechanisms (Marlin & Zemel, 2009; Marlin, Zemel, Roweis, & Slaney, 2007, 2011).

In the case of complete data, the parameter estimation task simplifies considerably, in the case of Bayesian networks: maximum-likelihood estimates can be obtained inference-free and in closed-form, using just a single pass over the data: $\theta_{x|\mathbf{u}} = \Pr_{\mathcal{D}}(x|\mathbf{u})$. In fact, the estimation algorithms that we proposed in this paper also obtain the same parameter estimates in the case of complete data, although we are not concerned with maximum-likelihood estimation here—we simply want to obtain estimates that are consistent (as in estimation by the method of moments).

Other inference-free estimators have been proposed for other classes of graphical models. Abbeel, Koller, and Ng (2006) identified a method for closed-form, inference-free parameter estimation in factor graphs of bounded degree from complete data. More recently, Halpern and Sontag (2013) proposed an efficient, inference-free method for consistently estimating the parameters of noisy-or networks with latent variables, under certain structural assumptions. From the perspective of maximum-likelihood learning, where evaluating the likelihood (requiring inference) seems to be unavoidable, the ability to consistently estimate parameters—without the need for inference—greatly extends the accessibility and utility of such models. For example, it opens the door to practical structure learning algorithms, under incomplete data, which is a notoriously difficult problem in practice (Abbeel et al., 2006; Jernite, Halpern, & Sontag, 2013).

7 CONCLUSIONS

In summary, we proposed a family of efficient and scalable algorithms for learning the parameters of Bayesian networks, from MCAR and MAR datasets, and sometimes MNAR datasets. Our parameter estimates are asymptotically consistent, and further, they are obtained inference-free and in closed-form. We further introduced and discussed some improved approaches for parameter estimation, when given additional knowledge of the missingness mechanisms underlying an incomplete dataset. Empirically, we demonstrate the practicality of our method, showing that it can scale to much larger datasets, and much larger Bayesian networks, than EM.

Acknowledgments

This work was supported in part by ONR grants #N00014-10-1-0933, #N00014-12-1-0423 and #N00014-13-1-0153, by NSF grants #IIS-1118122 and #IIS-1302448, and the Research Foundation-Flanders (FWO-Vlaanderen).

References

- Abbeel, P., Koller, D., & Ng, A. Y. (2006). Learning factor graphs in polynomial time and sample complexity. *Journal of Machine Learning Research*, 7, 1743–1788.
- Barber, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.
- Caffo, B. S., Jank, W., & Jones, G. L. (2005). Ascent-based monte carlo expectation-maximization. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2), pp. 235–251.
- Chavira, M., & Darwiche, A. (2006). Encoding CNFs to empower component analysis. In *SAT*, pp. 61–74.
- Chavira, M., & Darwiche, A. (2007). Compiling Bayesian networks using variable elimination. In *Proceedings of IJCAI*, pp. 2443–2449.
- Darwiche, A. (2009). *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press.
- Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39, 1–38.
- Elidan, G., Ninio, M., Friedman, N., & Shuurmans, D. (2002). Data perturbation for escaping local maxima in learning. In *Proceedings of AAAI*, pp. 132–139.
- Francois, O., & Leray, P. (2007). Generation of incomplete test-data using Bayesian networks. In *IJCNN*, pp. 2391–2396.
- Ghahramani, Z., & Jordan, M. I. (1997). Factorial hidden markov models. *Machine Learning*, 29(2-3), 245–273.
- Halpern, Y., & Sontag, D. (2013). Unsupervised learning of noisy-or Bayesian networks. In *Proceedings of UAI*.
- Jernite, Y., Halpern, Y., & Sontag, D. (2013). Discovering hidden variables in noisy-or networks using quartet tests. In *NIPS*, pp. 2355–2363.
- Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Lauritzen, S. (1995). The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19, 191–201.
- Marlin, B., & Zemel, R. (2009). Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*, pp. 5–12. ACM.
- Marlin, B., Zemel, R., Roweis, S., & Slaney, M. (2007). Collaborative filtering and the missing at random assumption. In *UAI*.
- Marlin, B., Zemel, R., Roweis, S., & Slaney, M. (2011). Recommender systems: missing data and statistical model estimation. In *IJCAI*.
- Mohan, K., Pearl, J., & Tian, J. (2013). Graphical models for inference with missing data. In *Proceedings of NIPS*.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
- Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models. *AIJ*, 32(2), 245–257.
- Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3), 581–592.
- Thiesson, B., Meek, C., & Heckerman, D. (2001). Accelerating EM for large databases. *Machine Learning*, 45(3), 279–299.
- Thoemmes, F., & Mohan, K. (2015). Graphical representation of missing data problems. *Structural Equation Modeling: A Multidisciplinary Journal*. To appear.
- Tsamardinos, I., Aliferis, C. F., Statnikov, A. R., & Statnikov, E. (2003). Algorithms for large scale Markov blanket discovery.. In *Proceedings of FLAIRS*, Vol. 2003, pp. 376–381.
- Wasserman, L. (2011). *All of Statistics*. Springer Science & Business Media.
- Yaramakala, S., & Margaritis, D. (2005). Speculative markov blanket discovery for optimal feature selection. In *Proceedings of ICDM*.

Efficient Algorithms for Bayesian Network Parameter Learning from Incomplete Data

SUPPLEMENTARY MATERIAL

A FACTORED DELETION FOR MAR

We now give a more detailed derivation of the factored deletion algorithm for MAR data. Let the query of interest be $\Pr(\mathbf{Y})$, and let $\mathbf{X}'_o = \mathbf{X}_m \setminus \mathbf{Y}_m$ and $\mathbf{Z}'_m = \{Y_m^j | i \leq j \leq n\}$. We can then factorize the estimation of $\Pr(\mathbf{Y})$ as follows.

$$\begin{aligned} \Pr(\mathbf{Y}) &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_m, \mathbf{Y}_o, \mathbf{X}'_o) \\ &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_o, \mathbf{X}'_o) \Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o) \\ &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{X}_o) \Pr(\mathbf{Y}_m | \mathbf{X}_o) \\ &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{X}_o) \prod_{i=1}^n \Pr(Y_m^i | \mathbf{Z}_m^{i+1}, \mathbf{X}_o) \\ &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{X}_o) \prod_{i=1}^n \Pr(Y_m^i | \mathbf{Z}_m^{i+1}, \mathbf{X}_o, \mathbf{R}_{\mathbf{Z}_m^i} = \text{ob}) \end{aligned}$$

The last step makes use of the MAR assumption. This leads us to the following algorithm, based on the data distribution $\Pr_{\mathcal{D}}$, and the fully-observed proxy variables $Y_m^{i,*}$ and $\mathbf{Z}_m^{i+1,*}$.

$$\begin{aligned} \Pr(\mathbf{Y}) &\approx \sum_{\mathbf{X}'_o} \Pr_{\mathcal{D}}(\mathbf{X}_o) \prod_{i=1}^n \Pr_{\mathcal{D}}(Y_m^{i,*} | \mathbf{Z}_m^{i+1,*}, \mathbf{X}_o, \mathbf{R}_{\mathbf{Z}_m^i} = \text{ob}) \end{aligned}$$

B EXTENDED EMPIRICAL EVALUATION: MCAR

Table 5 shows additional results for the classical `alarm` Bayesian network, from Section 4.1.

C EXTENDED EMPIRICAL EVALUATION: MAR

In this Appendix, we expand on the empirical results of Section 4 w.r.t. learning from MAR data. Here, we provide

Table 5: `alarm` network with MCAR data

Size	EM-1-JT		EM-10-JT		D-MCAR	F-MCAR	D-MAR	F-MAR
					Runtime [s]			
10^2	2	6			0	0	0	0
10^3	6	50			0	0	0	0
10^4	69	-			0	1	0	1
10^5	-	-			1	9	4	13
10^6	-	-			11	92	29	124
Test Set Log-Likelihood								
10^2	-12.18	-12.18	-12.85	-12.33	-12.82	-12.32		
10^3	-10.41	-10.41	-10.73	-10.55	-10.69	-10.55		
10^4	-10.00	-	-10.07	-10.04	-10.07	-10.05		
10^5	-	-	-9.98	-9.98	-9.99	-9.98		
10^6	-	-	-9.96	-9.96	-9.97	-9.97		
Kullback-Leibler Divergence								
10^2	2.381	2.381	3.037	2.525	3.010	2.515		
10^3	0.365	0.365	0.688	0.502	0.659	0.502		
10^4	0.046	-	0.113	0.084	0.121	0.093		
10^5	-	-	0.016	0.013	0.024	0.021		
10^6	-	-	0.002	0.002	0.006	0.008		

additional empirical results on standard real-world networks where inference is challenging, as originally highlighted in Table 3.

We consider two settings of generating MAR data, as in Section 4. In the *first setting*, the missing data mechanisms were generated with $m = 0.3$, $p = 2$, and a Beta distribution with shape parameters 1.0 and 0.5. In the second setting, we have $m = 0.9$, $p = 2$, and a Beta distribution with shape parameters 0.5 (as in Section 4.3). We consider three time limits, of 1 minute, 5 minutes, and 25 minutes. For all combinations of these setting, test set log-likelihoods are shown in Table 3, and in Tables 6 to 9.

We repeat the observations from the main paper (cf. Section 4). The EM-JT learner, which performs exact inference, does not scale well to these networks. This problem is mitigated by EM-BP, which performs *approximate* inference, yet we find that it also has difficulties scaling (dashed entries indicate that EM-JT and EM-BP did not finish 1 iteration of EM). In contrast, F-MAR, and particularly D-MAR, can scale to much larger datasets. As for accuracy, the F-MAR method typically obtains the best likelihoods (in bold) for larger datasets, although EM-BP can perform better on small datasets. We further evaluated D-MCAR and F-MCAR, although they are not in general consistent

Table 6: Log-likelihoods of large networks learned from MAR data (1 min. time limit, 1st setting).

Size		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR
10^2	Grid 90-20-1	-	-62.38	-64.15	-50.78	-63.51	-50.24	Water	-	-19.50	-20.51	-19.37	-20.41	-19.35
10^3		-	-79.75	-38.96	-32.77	-38.26	-32.44		-	-16.11	-16.26	-15.27	-16.09	-15.23
10^4		-	-	-30.65	-28.61	-30.05	-28.34		-	-	-15.03	-14.22	-14.86	-14.14
10^5		-	-	-	-	-	-		-	-	-14.30	-	-	-
10^2		-	-98.95	-103.59	-98.68	-103.54	-98.49		-	-85.33	-85.84	-85.68	-86.13	-85.75
10^3	Munin 1	-	-79.83	-70.49	-67.27	-69.78	-66.97	Barley	-	-	-67.70	-67.18	-67.67	-67.13
10^4		-	-	-59.25	-57.11	-	-		-	-	-54.93	-	-	-

Table 7: Log-likelihoods of large networks learned from MAR data (5 min. time limit, 1st setting).

Size		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR
10^2	Grid 90-20-1	-	-56.23	-63.34	-50.55	-62.38	-50.06	Water	-18.84	-18.06	-21.23	-19.61	-21.07	-19.57
10^3		-	-55.04	-39.89	-33.34	-39.09	-33.01		-	-14.99	-16.47	-15.33	-16.24	-15.26
10^4		-	-98.20	-30.46	-27.26	-29.73	-26.98		-	-17.39	-15.59	-14.52	-15.26	-14.43
10^5		-	-	-28.63	-26.06	-27.89	-		-	-	-15.22	-	-	-
10^6		-	-	-	-	-	-		-	-	-15.09	-	-	-
10^2	Munin 1	-	-96.51	-102.51	-98.21	-102.40	-97.95	Barley	-	-85.59	-85.70	-85.60	-85.99	-85.66
10^3		-	-68.04	-67.82	-65.49	-67.21	-65.22		-	-67.07	-67.58	-66.97	-67.53	-66.91
10^4		-	-95.01	-57.68	-56.00	-57.05	-55.79		-	-	-55.04	-54.33	-54.78	-
10^5		-	-	-54.30	-	-	-		-	-	-	-	-	-

Table 8: Log-likelihoods of large networks learned from MAR data (25 min. time limit, 1st setting).

Size		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR
10^2	Grid 90-20-1	-	-47.66	-59.84	-48.34	-59.39	-47.88	Water	-21.30	-18.66	-21.58	-19.87	-21.36	-19.83
10^3		-	-46.53	-37.29	-31.60	-36.76	-31.28		-17.67	-17.10	-18.64	-15.95	-18.27	-15.86
10^4		-	-62.98	-28.74	-26.71	-28.26	-26.45		-	-14.83	-16.71	-14.58	-16.30	-14.44
10^5		-	-	-25.88	-24.97	-25.43	-24.75		-	-18.78	-16.31	-14.38	-15.62	-14.08
10^6		-	-	-25.27	-	-24.78	-		-	-	-15.25	-	-	-
10^7	Munin 1	-	-	-	-	-	-		-	-	-15.13	-	-	-
10^2		-	-90.79	-98.57	-94.50	-98.48	-94.28	Barley	-85.11	-85.53	-86.00	-85.74	-86.24	-85.80
10^3		-	-60.71	-66.06	-63.95	-65.45	-63.67		-	-65.96	-67.88	-67.23	-67.79	-67.15
10^4		-	-60.35	-56.57	-55.38	-55.95	-55.16		-	-57.21	-55.34	-54.56	-55.05	-54.43
10^5		-	-	-54.29	-53.38	-53.67	-		-	-	-51.09	-	-	-

Table 9: Log-likelihoods of large networks learned from MAR data (1 min. time limit, 2nd setting).

Size		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR		EM-JT	EM-BP	D-MCAR	F-MCAR	D-MAR	F-MAR
10^2	Grid 90-20-1	-	-62.25	-80.10	-56.59	-79.93	-56.07	Water	-	-20.15	-26.40	-22.85	-26.24	-22.88
10^3		-	-129.38	-38.74	-29.88	-38.51	-29.70		-	-17.76	-20.45	-17.80	-20.32	-17.64
10^4		-	-	-27.83	-24.30	-27.25	-23.97		-	-	-17.59	-15.40	-17.28	-15.29
10^5		-	-	-	-	-	-		-	-	-15.38	-	-	-
10^2	Munin 1	-	-99.49	-111.95	-104.07	-111.72	-103.10	Barley	-	-89.16	-89.63	-89.13	-89.66	-88.99
10^3		-	-99.56	-70.32	-66.08	-69.76	-65.57		-	-	-71.76	-70.50	-71.74	-
10^4		-	-	-56.25	-54.36	-	-		-	-	-56.59	-	-	-

for MAR data, and find that they scale even further, and can also produce relatively good estimates (in terms of likelihood).

D EXAMPLE: DATA EXPLOITATION BY CLOSED-FORM ESTIMATORS

This appendix demonstrates with an example how each learning algorithm exploits varied subsets of data to estimate marginal probability distributions, given the manifest (or data) distribution in Table 10 which consists of four variables, $\{X, Y, Z, W\}$ such that $\{X, Y\} \in \mathbf{X}_m$ and $\{Z, W\} \in \mathbf{X}_o$.

We will begin by examining the data usage by deletion algorithms while estimating $\Pr(x, w)$ under the MCAR assumption. All three deletion algorithms, namely listwise deletion, direct deletion and factored deletion guarantee consistent estimates when data are MCAR. Among these algorithms, listwise deletion utilizes the least amount of data (4 distinct tuples out of 36 available tuples, as shown in table 11) to compute $\Pr(xw)$ whereas factored deletion employs two thirds of the tuples (24 distinct tuples out of 36 available tuples as shown in table 11) for estimating $\Pr(xw)$.

Under MAR, no guarantees are available for listwise deletion. However the three algorithms, namely direct deletion, factored deletion and informed deletion, guarantee consistent estimates. While estimating $\Pr(x, y)$, all the three algorithms utilize every tuple in the manifest distribution at least once (see Table 12). Compared to the direct deletion algorithm, the factored deletion algorithm utilizes more data while computing $\Pr(x, y)$ since it has multiple factorizations with more than two factors in each of them; this allows more data to be used while computing each factor (see Table 11). In contrast to both direct and factored deletion, the informed deletion algorithm yields an estimator that involves factors with fewer elements in them ($\Pr(w)$ vs. $\Pr(zw)$) and hence can be computed using more data ($\Pr(w = 0)$ uses 18 tuples compared to $\Pr(z = 0, w = 0)$ that uses 9 tuples).

Precise information regarding the missingness process is required for estimation when dataset falls under the MNAR category. In particular, only algorithms that consult the missingness graph can answer questions about the estimability of queries.

Table 10: Manifest (Data) Distribution with $\{X, Y\} \in \mathbf{X}_m$ and $\{Z, W\} \in \mathbf{X}_o$.

#	X	Y	W	Z	R_X	R_Y	#	X	Y	W	Z	R_X	R_Y
1	0	0	0	0	ob	ob	19	0	?	1	0	ob	unob
2	0	0	0	1	ob	ob	20	0	?	1	1	ob	unob
3	0	0	1	0	ob	ob	21	1	?	0	0	ob	unob
4	0	0	1	1	ob	ob	22	1	?	0	1	ob	unob
5	0	1	0	0	ob	ob	23	1	?	1	0	ob	unob
6	0	1	0	1	ob	ob	24	1	?	1	1	ob	unob
7	0	1	1	0	ob	ob	25	?	0	0	0	unob	ob
8	0	1	1	1	ob	ob	26	?	0	0	1	unob	ob
9	1	0	0	0	ob	ob	27	?	0	1	0	unob	ob
10	1	0	0	1	ob	ob	28	?	0	1	1	unob	ob
11	1	0	1	0	ob	ob	29	?	1	0	0	unob	ob
12	1	0	1	1	ob	ob	30	?	1	0	1	unob	ob
13	1	1	0	0	ob	ob	31	?	1	1	0	unob	ob
14	1	1	0	1	ob	ob	32	?	1	1	1	unob	ob
15	1	1	1	0	ob	ob	33	?	?	0	0	unob	unob
16	1	1	1	1	ob	ob	34	?	?	0	1	unob	unob
17	0	?	0	0	ob	unob	35	?	?	1	0	unob	unob
18	0	?	0	1	ob	unob	36	?	?	1	1	unob	unob

Table 11: Enumeration of sample # used for computing $\Pr(x, w)$ by listwise deletion, direct deletion and factored deletion algorithms under MCAR assumptions.

Algorithm	Estimator and Sample #
Listwise	$\Pr(xw) = \Pr(xw R_X = \text{ob}, R_Y = \text{ob})$ 11,12,15,16
Direct	$\Pr(xw) = \Pr(xw R_X = \text{ob})$ 11,12,15,16,23,24
Factored	$\Pr(xw) = \Pr(x w, R_X = \text{ob}) \Pr(w)$ 3,4,7,8,11,12,15,16,19,20,23,24,27,28,31,32,35,36 $\Pr(xw) = \Pr(w x, R_X = \text{ob}) \Pr(x R_X = \text{ob})$ 9,10,11,12,13,14,15,16,21,22,23,24

Table 12: Enumeration of sample # used for computing $\Pr(x, y)$ by direct deletion, factored deletion and informed deletion algorithms under MAR assumption.

Algorithm	Estimator and Sample #
Direct	$\Pr(xy) = \sum_{z,w} \Pr(xy w, z, R_X = \text{ob}, R_Y = \text{ob}) \Pr(zw)$ 13, 14, 15, 16 for $\Pr(xy w, z, R_X = \text{ob}, R_Y = \text{ob})$ all tuples: [1,36] for $\Pr(z, w)$
Factored	$\Pr(xy) = \sum_{z,w} \Pr(x w, z, y, R_X = \text{ob}, R_Y = \text{ob}) \Pr(y z, w, R_Y = \text{ob}) \Pr(zw)$ 13, 14, 15, 16 for $\Pr(x y, w, z, R_X = \text{ob}, R_Y = \text{ob})$ 5, 6, 7, 8, 13, 14, 15, 16, 29, 30, 31, 32 for $\Pr(y w, z, R_Y = \text{ob})$ all tuples: [1,36] for $\Pr(z, w)$ $\Pr(xy) = \sum_{z,w} \Pr(y x, w, z, R_X = \text{ob}, R_Y = \text{ob}) \Pr(x z, w, R_X = \text{ob}) \Pr(zw)$ 13, 14, 15, 16 for $\Pr(y x, w, z, R_X = \text{ob}, R_Y = \text{ob})$ 9, 10, 11, 12, 13, 14, 15, 16, 21, 22, 23, 24 for $\Pr(x w, z, R_X = \text{ob})$ all tuples: [1,36] for $\Pr(z, w)$
Informed (direct)	$\Pr(xy) = \sum_w \Pr(xy w, R_X = \text{ob}, R_Y = \text{ob}) \Pr(w)$ 13, 14, 15, 16 for $\Pr(xy w, R_X = \text{ob}, R_Y = \text{ob})$ all tuples: [1,36] for $\Pr(w)$ 