Presented at the Causal Modeling and Machine Learning Workshop at ICML-2014.

TECHNICAL REPORT
R-441
November 2014

Efficient Algorithms for Bayesian Network Parameter Learning from Incomplete Data

Guy Van den Broeck[*] and **Karthika Mohan**[*] and **Arthur Choi** and **Judea Pearl**
University of California, Los Angeles
Los Angeles, CA 90095
{guyvdb,karthika,aychoi,judea}@cs.ucla.edu

Abstract

We propose an efficient family of algorithms to learn the parameters of a Bayesian network from incomplete data. In contrast to textbook approaches such as EM and the gradient method, our approach is non-iterative, yields closed form parameter estimates, and eliminates the need for inference in a Bayesian network. Our approach provides consistent parameter estimates for missing data problems that are MCAR, MAR, and in some cases, MNAR. Empirically, our approach is orders of magnitude faster than EM (as our approach requires no inference). Given sufficient data, we learn parameters that can be orders of magnitude more accurate.

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: either use *expectation maximization* (EM) or use likelihood *optimization* with a gradient method; see, e.g., (Darwiche 2009; Koller and Friedman 2009; Murphy 2012; Barber 2012). These two approaches are known to consistently estimate the parameters when values in the data are *missing at random* (MAR). However, these two standard approaches suffer from the following disadvantages. First, they are *iterative*, and hence they may require many passes over a potentially large dataset. Next, they *require inference* in the Bayesian network, which is by itself already intractable (for high-treewidth networks with little local structure (Chavira and Darwiche 2006; 2007)). Finally, 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 keep those parameter estimates that obtained the best likelihood.

Recently, Mohan, Pearl, and Tian (2013) showed that the joint distribution of a Bayesian network can be recovered consistently from incomplete data, for all MCAR and MAR problems as well as a major subset of MNAR problems, when given access to a missingness graph. This graph is a formal representation of the *causal mechanisms* responsible for missingness in an incomplete dataset. Using this representation, they are able to decide whether there exists a consistent estimator for a given query Q (e.g., a joint or conditional distribution). If the answer is affirmative, they identify a *closed-form* expression to estimate Q in terms of the observed data, which is asymptotically consistent.

Based on this framework, we contribute a new and *practical family of parameter learning algorithms* for Bayesian networks. The key insight of our work is the following. There exists a most-general, least-committed missingness graph that captures the MCAR or MAR assumption, but invokes no additional independencies. Although this is a minor technical observation, it has far-reaching consequences. It enables the techniques of Mohan, Pearl, and Tian to be applied directly to MCAR or MAR data, without requiring the user to provide a more specific missingness graph. Hence, it enables our new algorithms to serve as drop-in replacements for the already influential EM algorithm in existing applications. It results in practical algorithms for learning the parameters of a Bayesian network from an incomplete dataset that have the following advantages:

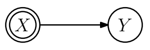
1. the parameter estimates are efficiently computable in *closed-form*, requiring only a *single pass over the data*, as if no data was missing,
2. the parameter estimates are obtained, *inference-free*, in the Bayesian network, and
3. the parameter estimates are *consistent* when the values of a dataset are MCAR or MAR, i.e., we recover the true parameters as the dataset size approaches infinity.

Advantages (1) and (2) are significant computational advantages over EM, in particular, when the dataset size is very large (cf., the Big Data paradigm), or for Bayesian networks that are intractable for exact inference. Moreover, because of advantage (1), we do not use iterative optimization, and our estimates do not suffer from local optima. Note further that all these advantages are already available to us when learning Bayesian networks from *complete* datasets, properties which certainly contributed to the popularity of Bayesian networks today, as probabilistic models.

As secondary contributions, we show how to factorize estimates to extract more information from the data, and how to use additional information about the missingness mechanism to improve the convergence of our algorithms. Moreover, we present an initial experimental evaluation of the

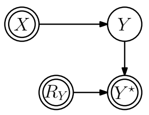
[*]Both authors contributed equally to this work. GVdB is also affiliated with KU Leuven, Belgium.

X	Y
$\bar{x}$	y
x	?
x	$\bar{y}$
$\bar{x}$	?
…	



(a) Dataset $\mathfrak{D}$ and DAG

X	Y	$Y^\star$	R_Y
$\bar{x}$	y	y	ob
x	?	mi	unob
x	$\bar{y}$	$\bar{y}$	ob
$\bar{x}$	?	mi	unob
…		…	



(b) Missingness Dataset $\mathfrak{D}$ and DAG

Figure 1: Datasets and DAGs.

proposed algorithms, illustrating their key properties.

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 $Pr(X{=}x|\mathbf{U}{=}\mathbf{u})$.

As an illustrative example, consider Figure 1(a), depicting a dataset $\mathfrak{D}$, and the directed acyclic graph (DAG) $\mathcal{G}$ 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 depicted in Figure 1(b). Here, the missingness of variable Y, which is reported by variable $Y^\star$, depends on the value of another variable R_Y. This graph is called a *missingness* graph, and can serve as a useful tool for analyzing missing data problems (Mohan, Pearl, and Tian 2013; Darwiche 2009; Koller and Friedman 2009).

In our example, we have augmented the dataset and graph with new variables. Variable R_Y represents the causal mechanism that dictates the missingness of the value of Y. This mechanism can be active (Y is unobserved), which we denote by $R_Y{=}\mathsf{unob}$. Otherwise, the mechanism is passive (Y is observed), which we denote by $R_Y{=}\mathsf{ob}$. Variable $Y^\star$ acts as a proxy on the value of Y in the data, which may be an observed value y, or a special value (mi) when the value of Y is missing. The value of $Y^\star$ thus depends functionally on variables R_Y and Y, with a corresponding CPT:

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

$$\Pr(Y^\star|Y, R_Y) = \begin{cases} 1 & \text{if } R_Y{=}\mathsf{unob} \text{ and } Y^\star = \mathsf{mi} \\ 1 & \text{if } R_Y{=}\mathsf{ob} \text{ and } Y^\star = Y \\ 0 & \text{otherwise.} \end{cases}$$

That is, when $R_Y{=}\mathsf{unob}$, then $Y^\star = \mathsf{mi}$; otherwise $R_Y{=}\mathsf{ob}$ and the proxy $Y^\star$ assumes the observed value of variable Y. Using missingness graphs, one can analyze the distribution over the observed and missing values, and relate it to the underlying distribution that we seek to estimate from data. As Mohan, Pearl, and Tian (2013) further show, one can exploit the conditional independencies that these graphs encode, in order to extract *consistent* estimates from missing data problems, including MNAR ones, whose underlying assumptions would put itself out of the scope of existing techniques.

We now more formally define the missing data problems that we consider in this paper. When learning a Bayesian network $\mathcal{N}$ from an incomplete dataset $\mathfrak{D}$, there is an underlying but unknown distribution $\Pr(\mathbf{X})$ that is induced by the network $\mathcal{N}$ that we want to learn. The variables $\mathbf{X}$ are partitioned into two sets: the fully-observed variables $\mathbf{X}_o$, and the partially-observed variables $\mathbf{X}_m$ that have missing values in the data. We can take into account the mechanisms that cause the values of variables $\mathbf{X}_m$ to go missing, as in our example above, by introducing variables $\mathbf{R}$ representing the *causal mechanisms* themselves, and variables $\mathbf{X}^\star_m$ that act as *proxies* to the variables $\mathbf{X}_m$. This augmented Bayesian network, which we refer to as the missingness graph $\mathcal{N}^\star$, now has variables $\mathbf{X}_o, \mathbf{X}^\star_m, \mathbf{R}$ that are fully-observed, and variables $\mathbf{X}_m$ that are only partially-observed. Moreover, network $\mathcal{N}^\star$ induces a distribution $\Pr(\mathbf{X}_o, \mathbf{X}_m, \mathbf{X}^\star_m, \mathbf{R})$ which now embeds the original distribution $\Pr(\mathbf{X}_o, \mathbf{X}_m)$ of network $\mathcal{N}$ as a marginal distribution.

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

$$\Pr_{\mathfrak{D}}(\mathbf{x}_o, \mathbf{x}^\star_m, \mathbf{r}) = \tfrac{1}{N}\mathfrak{D}\#(\mathbf{x}_o, \mathbf{x}^\star_m, \mathbf{r}),$$

where N is the number of instances in dataset $\mathfrak{D}$, and where $\mathfrak{D}\#(\mathbf{x})$ is the number of instances where instantiation $\mathbf{x}$ appears in the data.[1] However, the data distribution $\Pr_{\mathfrak{D}}$ tends to the true distribution $\Pr$ (over the fully-observed variables), as N tends to infinity.

In this paper, we show how we can leverage the results of Mohan, Pearl, and Tian (2013), even when we do not have access to the complete missingness graph that specifies the direct causes, or parents, of the missingness mechanisms $\mathbf{R}$. We identify practical and efficient algorithms for the consistent estimation of Bayesian network parameters. First, we only assume general conditions that hold in broad classes of missingness graphs, which further characterize commonly-used assumptions on missing data. Subsequently, we show how to exploit more specific knowledge of the underlying missingness graph that is available (say, from a domain expert), to obtain improved parameter estimates.

Missingness Categories 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

[1]Note that the data distribution is well-defined over the variables $\mathbf{X}_o, \mathbf{X}^\star_m$ and $\mathbf{R}$, as they are fully-observed in the augmented dataset, and that $\Pr_{\mathfrak{D}}$ 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.

$(\mathbf{X}_m, \mathbf{X}_o) \perp\!\!\!\perp \mathbf{R}$. This corresponds to a missingness graph where no variable in $\mathbf{X}_m \cup \mathbf{X}_o$ is a parent of any variable in $\mathbf{R}$. Note that the example graph that we started with in Section 2 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 $\mathbf{X}_m \perp\!\!\!\perp \mathbf{R} \mid \mathbf{X}_o$. 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 Section 2, 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 and Friedman 2009), in contrast to the event-level definition of MAR, that is prevalent in the statistics literature (Rubin 1976).

An incomplete dataset is categorized as *Missing Not At Random* (MNAR) if it is neither MCAR nor MAR. In the example graph of Section 2, adding an edge $Y \rightarrow R_Y$ yields an MNAR assumption.

3 Closed-Form Learning Algorithms

We now present a set of algorithms to learn the parameters $\theta_{x|\mathbf{u}}$ of a Bayesian network $\mathcal{N}$ from the data distribution $\Pr_{\mathfrak{D}}$ over the fully-observed variables in the augmented dataset. We do so for different missing data assumptions, but without knowing the missingness graph that generated the data. To estimate the conditional probabilities $\theta_{x|\mathbf{u}}$ that parameterize a Bayesian network, we estimate the joint distributions $\Pr(X, \mathbf{U})$, which are subsequently normalized. Hence, it suffices, for our discussion, to estimate marginal distributions $\Pr(\mathbf{Y})$, for families $\mathbf{Y} = \{X\} \cup \mathbf{U}$. Here, 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 $\mathbf{R}_\mathbf{Z} \subseteq \mathbf{R}$ denote the missingness mechanisms for a set of partially-observed variables $\mathbf{Z}$. Appendix E illustrates our learning algorithms on a concrete dataset.

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 deletion technique, which is expressed in the terms of causal missingness mechanisms, which we reviewed in the previous section. In particular, to estimate the marginals $\Pr(\mathbf{Y})$ of a set of (family) variables $\mathbf{Y}$, from the data distribution $\Pr_{\mathfrak{D}}$, we can use the estimate:

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

That is, we can estimate $\Pr(\mathbf{Y})$ by simply using the subset of the data where every variable in $\mathbf{Y}$ is observed (which follows from the assumptions implied by MCAR data). Because the data distribution $\Pr_{\mathfrak{D}}$ 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 $\Pr(\mathbf{Y}) \approx \Pr_{\mathfrak{D}}(\mathbf{Y}_o, \mathbf{Y}^\star_m | \mathbf{R}_{\mathbf{X}_m} = \mathsf{ob})$, 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*.

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. This is the key contribution of this paper, which provides a new algorithm with desirable properties (compared to EM), as described in the introduction. We shall subsequently show how to obtain even better estimates, later.

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. Analogous to the MCAR case, we can estimate each conditional probability as

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

This leads to the following algorithm,

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

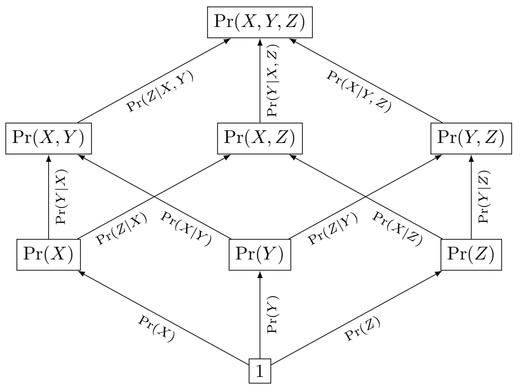


Figure 2: Factorization Lattice of $\Pr(X, Y, Z)$

which uses only the fully-observed variables of the data distribution $\Pr_{\mathfrak{D}}$.[2] Again, $\Pr_{\mathfrak{D}}$ tends to the true distribution $\Pr$, as the dataset size tends to infinity, implying a consistent estimate of $\Pr(\mathbf{Y})$.

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 estimands for the query so that each estimand 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 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, i.e., $\Pr(\mathbf{Y}) = \prod_{i=1}^{n} \Pr\left(Y^i \middle| Y^{i+1}, \dots, Y^n\right)$. 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., $\Pr\left(Y^i \middle| Y^{i+1}, \dots, Y^n_m\right) = \Pr\left(Y^i \middle| Y^{i+1}, \dots, Y^n_m, \mathbf{R}_{Z^i}{=}\mathsf{ob}\right)$ where $\mathbf{Z}^i$ is the set of partially observed variables in the factor. When $|\mathbf{Y} \cap \mathbf{X}_m| > 1$, we can utilize much more data compared to direct deletion. We refer to Appendix E 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 2, 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).

More specifically, our factored deletion algorithm works as follows. First, we estimate the conditional probabilities on the edges of the lattice, each estimate using the subset of the data where its variables are observed. Second, we propagate the estimates, bottom-up. For each node, there may be 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] Whereas direct deletion 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.

Factored Deletion for MAR Let $Y^1_m, Y^2_m, \dots, Y^n_m$ 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 can factorize $\Pr(\mathbf{Y})$ as $\sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_o, \mathbf{X}'_o) \prod_{i=1}^{n} \Pr\left(Y^i_m \middle| \mathbf{Z}^{i+1}_m, \mathbf{X}_o\right)$ where $\mathbf{Z}^i_m = \left\{Y^j_m \middle| i \leq j \leq n\right\}$. 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.

Learning with a Missingness Graph

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.

Informed Deletion for MAR Suppose now that we have more in-depth 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 such 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 $\mathbf{Y}$),[4] yielding the algorithm:

$$\Pr(\mathbf{Y}) \approx \sum_{\mathbf{W}'_o} \Pr_{\mathfrak{D}}(\mathbf{Y}^\star_m | \mathbf{Y}_o, \mathbf{W}'_o, \mathbf{R}_{\mathbf{Y}_m}{=}\mathsf{ob}) \Pr_{\mathfrak{D}}(\mathbf{Y}_o, \mathbf{W}'_o)$$

We refer to this algorithm as *informed direct deletion*. By reducing the scope of the summation, we need to estimate fewer sub-expressions $\Pr_{\mathfrak{D}}(\mathbf{Y}^\star_m | \mathbf{Y}_o, \mathbf{W}'_o, \mathbf{R}_{\mathbf{Y}_m}{=}\mathsf{ob})$. This results in a more efficient computation, but further, each individual sub-expression can be estimated on more data.

[2] 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.

[3] We use an inverse-variance weighting heuristic, which was somewhat better in our experiments.

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

Moreover, our estimates remain consistent. We similarly replace $\mathbf{X}_o$ by $\mathbf{W}_o$ in the factored deletion algorithm, yielding the *informed factored deletion* algorithm. Appendix B presents an empirical evaluation of informed deletion and exemplifies cases where knowing the missingness graph lets us consistently learn from MNAR data, which is beyond the capabilities of maximum-likelihood learners.

4 Empirical Evaluation

To evaluate the proposed learning algorithms, we simulate partially observed datasets from Bayesian networks, and relearn their parameters from the data.[5] We consider the following algorithms:

D-MCAR & F-MCAR direct/factored deletion for MCAR data.

D-MAR & F-MAR direct/factored deletion for MAR data.

EM-k-JT EM with k random restarts and using the jointree inference algorithm.

F-MAR + EM-JT EM using the jointree inference algorithm, seeded by the F-MAR estimates.

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 on the network parameters with a concentration parameter of 2.

Appendix C provides empirical results on the simpler case of learning from *MCAR data*, where all algorithms are consistent. As expected, all produce increasingly accurate estimates as more data becomes available. Compared to EM, where the complexity of inference prevents scaling to large datasets, D-MCAR and F-MCAR can obtain more accurate estimates orders-of-magnitude faster.

In this section, we investigate the more challenging problem of learning from *MAR data*, which are generated as follows: (a) select an m-fraction of the variables to be partially observed, (b) introduce a missingness mechanism variable R_X for each partially observed variable X, (c) assign p parents to each R_X, that are randomly selected from the set of observed variables, giving preference to neighbors of X in the network, (d) sample parameters for the missingness mechanism CPTs from a Beta distribution, (e) sample a complete dataset with R_X values, and (f) hide X values accordingly.

For our first MAR experiment, we work with a small network that is tractable enough for EM to scale to large dataset sizes. Figure 3(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.

There is a significant difference between EM, with and without restarts, indicating that the likelihood landscape is challenging to optimize (compared to MCAR in Appendix C). 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 several orders of magnitude smaller than EM. The KLD improves further when we use F-MAR estimates to seed EM, although EM will not scale to larger, intractable networks. D-MCAR and F-MCAR are not consistent for MAR data, and indeed converge to a biased estimate with a KLD around 0.1. The factorized algorithms 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 3(b) and 3(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. 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 other EM learners.

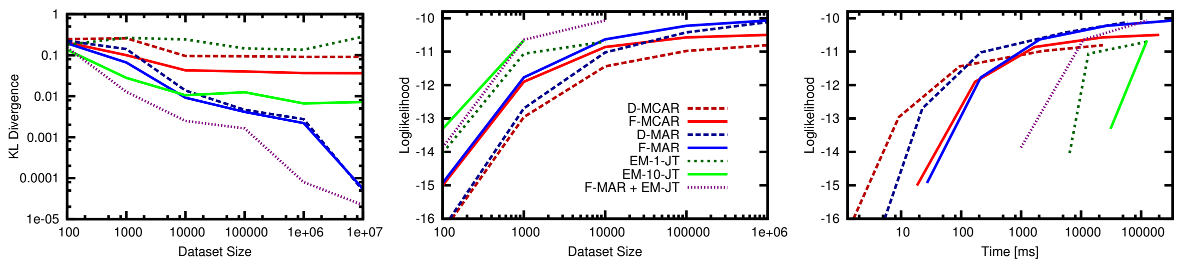
For our third MAR experiment, Table 1 reports results on four larger networks where exact inference is challenging. Each method is given a time limit of 25 minutes, and data is generated as above. Appendix D provides further results, on more settings. We consider the following algorithms.

EM-JT The EM-10-JT algorithm used in anytime fashion, which returns, given a time limit, the best parameters found so far 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. 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 for MAR data, and find that they scale even further, and can also produce relatively good estimates (in terms of likelihood).

[5]The implementation and experimental setup is available at `http://reasoning.cs.ucla.edu/deletion`



(a) KL Divergence - Fire Alarm (b) Likelihood vs. Size - Alarm (c) Likelihood vs. Time - Alarm

Figure 3: Learning small, tractable Bayesian networks from MAR data (legend in (b)).

Table 1: Log-likelihoods of large, intractable networks 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		-	-	-	-	-	-		-	-	**-13.73**	-	-	-
10^2	Munin 1	-	**-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**	-

5 Conclusions and Related Work

For estimating parameters in Bayesian networks, maximum likelihood (ML) estimation is the typical approach used, 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, and Rubin 1977; Lauritzen 1995) (see also, e.g., Darwiche; Koller and Friedman; Murphy; Barber (2009; 2009; 2012; 2012)). As we discussed, such methods do not scale well to large datasets or complex Bayesian networks as they (1) are iterative, (2) require inference in a Bayesian network, and (3) suffer from local optima. 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, Meek, and Heckerman (2001), (2) use approximate inference algorithms in lieu of exact ones when inference is intractable, e.g., Ghahramani and Jordan; Caffo, Jank, and Jones (1997; 2005), and (3) escape local optima, e.g., Elidan et al. (2002). While EM is suitable for data that is MAR (the typical assumption, in practice), there are some exceptions, such as recent work on recommender systems that explicitly incorporate missing data mechanisms (Marlin and Zemel 2009; Marlin et al. 2011; 2007).

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).

In summary, we proposed an inference-free, closed-form method for consistently learning Bayesian network parameters, from MCAR and MAR datasets (and sometimes MNAR datasets, as in Appendix B). Empirically, we demonstrate the practicality of our method, showing that it is orders-of-magnitude more efficient than EM, allowing it to scale to much larger datasets. Further, given access to enough data, we show that our method can learn much more accurate Bayesian networks as well.

Other inference-free estimators have been proposed for other classes of probabilistic graphical models. For example, 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. We note that inference-free learning of the parameters of: Bayesian networks under MAR data (this paper), factor graphs of bounded degree, under complete data (Abbeel, Koller, and Ng 2006), and structured noisy-or Bayesian networks with latent variables (Halpern and Sontag 2013), are all surprising results. 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 potential 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, Koller, and Ng 2006; Jernite, Halpern, and Sontag 2013).

References

Abbeel, P.; Koller, D.; and 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.; and 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., and Darwiche, A. 2006. Encoding CNFs to empower component analysis. In *Proceedings of SAT*, 61–74.

Chavira, M., and Darwiche, A. 2007. Compiling Bayesian networks using variable elimination. In *Proceedings of IJCAI*, 2443–2449.

Darwiche, A. 2009. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press.

Dempster, A.; Laird, N.; and 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.; and Shuurmans, D. 2002. Data perturbation for escaping local maxima in learning. In *Proceedings of AAAI*, 132–139.

Ghahramani, Z., and Jordan, M. I. 1997. Factorial hidden markov models. *Machine Learning* 29(2-3):245–273.

Halpern, Y., and Sontag, D. 2013. Unsupervised learning of noisy-or Bayesian networks. In *Proceedings of UAI*.

Jernite, Y.; Halpern, Y.; and Sontag, D. 2013. Discovering hidden variables in noisy-or networks using quartet tests. In *Proceedings of NIPS*, 2355–2363.

Koller, D., and 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., and Zemel, R. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*, 5–12. ACM.

Marlin, B.; Zemel, R.; Roweis, S.; and Slaney, M. 2007. Collaborative filtering and the missing at random assumption. In *Proceedings of UAI*.

Marlin, B.; Zemel, R.; Roweis, S.; and Slaney, M. 2011. Recommender systems: missing data and statistical model estimation. In *Proceedings of IJCAI*.

Mohan, K.; Pearl, J.; and 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. *Artificial Intelligence* 32(2):245–257.

Rubin, D. B. 1976. Inference and missing data. *Biometrika* 63(3):581–592.

Thiesson, B.; Meek, C.; and Heckerman, D. 2001. Accelerating EM for large databases. *Machine Learning* 45(3):279–299.

Tsamardinos, I.; Aliferis, C. F.; Statnikov, A. R.; and Statnikov, E. 2003. Algorithms for large scale Markov blanket discovery. In *Proceedings of FLAIRS*, volume 2003, 376–381.

Yaramakala, S., and Margaritis, D. 2005. Speculative markov blanket discovery for optimal feature selection. In *Proceedings of ICDM*.

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 $\mathbf{Z}^i_m = \{Y^j_m | i \le j \le n\}$. We can then factorize the estimation of $\Pr(\mathbf{Y})$ as follows.

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

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 $Y^{i,\star}_m$ and $\mathbf{Z}^{i+1,\star}_m$.

$$\Pr(\mathbf{Y}) \approx \sum_{\mathbf{X}'_o} \Pr_{\mathcal{D}}(\mathbf{X}_o) \prod_{i=1}^{n} \Pr_{\mathcal{D}}\left(Y^{i,\star}_m | \mathbf{Z}^{i+1,\star}_m, \mathbf{X}_o, \mathbf{R}_{\mathbf{Z}^i_m} = \mathsf{ob}\right)$$

B Learning with a Missingness Graph

Note that knowing the parents of a mechanism variable $\mathbf{R}$ is effectively equivalent, for the purposes of informed deletion, to knowing the Markov blanket of the variables in $\mathbf{R}$ (Pearl 1987), which can be learned from data (Tsamardinos et al. 2003; Yaramakala and Margaritis 2005). With sufficient domain knowledge, an expert may be able to specify the parents of the mechanism variables. It suffices even to identify a subset of the observed variables that just *contains* the Markov blanket; this knowledge can still be exploited to reduce the scope of the summation. As we discuss next, having deeper knowledge of the nature of the missingness mechanisms, will enable us to obtain consistent estimators, even for datasets that are not MAR (in some cases).

Empirical Evaluation of Informed Deletion

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 generate 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 2 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.

Missing Not at Random (MNAR) Data

A missing data problem that is neither MCAR nor MAR is classified as *Missing Not at Random* (MNAR). Here, the parameters of a Bayesian network may not even be identifiable. Further, maximum likelihood estimation is in general not consistent, so the EM algorithm and gradient methods are expected to yield biased estimates. However, if one knows the interactions of the mechanisms that dictate the missingness of a dataset (in the form of a missingness graph), then it becomes possible again to obtain consistent estimates, at least in some cases (Mohan, Pearl, and Tian 2013).

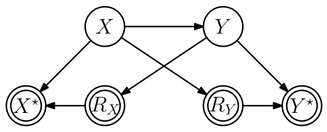


Figure 4: An MNAR missingness graph.

For example, consider the missingness graph of Figure 4, 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. In this case, it is still possible to obtain consistent parameter estimates, as

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

For a derivation, see Mohan, Pearl, and Tian (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. We view, as interesting future work, the identification of missingness graph structures that guarantee consistent estimators (beyond MCAR and MAR), under minimal assumptions (such as the ones we exploited for informed deletion).

C Extended Empirical Evaluation: MCAR

In this Appendix, we expand on the empirical results of Section 4. Here, we investigate learning from *MCAR data*, by generating MCAR datasets of increasing size, and evaluating the quality of the learned parameters for each algorithm.

Table 3: 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

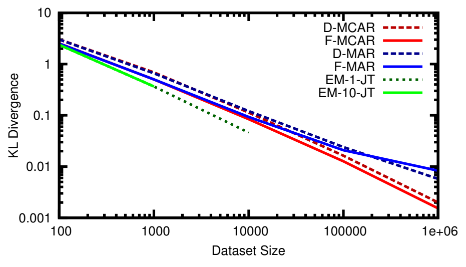
Table 3 shows results for the "Alarm" Bayesian network. Each training set is simulated from the original Bayesian network, selecting 30% of the variables to be partially observed, and removing 70% of their values completely at random. All reported numbers are averaged over 32 repetitions with different learning problems. When no number is reported, a 5 minute time limit was exceeded.

We first note that 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. Direct and factored deletion are orders of magnitude faster than EM, which needs to repeatedly run inference for every instance in the dataset. Even though EM outperforms F-MCAR in terms of KLD and likelihood, the difference is negligible, in the sense that only a small difference in the amount of available data makes F-MCAR outperform EM. F-MCAR is slower than D-MCAR, because it requires estimating more probabilities (one for each lattice edge). F-MCAR does learn better models, because it can use a larger portion of the available data. Finally, D-MAR performs worse than F-MCAR and D-MCAR, as it is operating on the weaker MAR assumption. All learners are consistent, as can be seen from the KLD converging to zero.

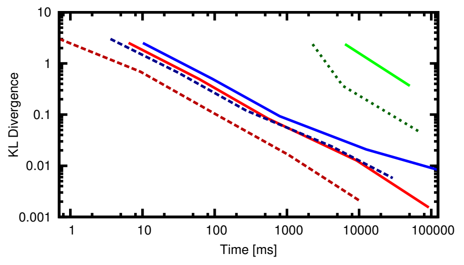
To illustrate the trade-off between data and computational resources, Figure 5 shows the KLDs from Table 3 as a function of dataset size and time. When data is limited, and computation power is abundant, it is clear that EM

Table 2: Alarm network with Informed MAR data

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



(a) KL Divergence vs. Dataset Size



(b) KL Divergence vs. Time

Figure 5: Learning the "Alarm" network from MCAR data, as show in Table 3.

is the algorithm of choice, even though the differences are small. When computation power is limited (e.g., when the Bayesian network is highly intractable), and data is abundant (e.g., the online learning or big data setting), the differences are marked. EM is several orders of magnitudes slower than D-MCAR at learning a model of similar quality. F-MCAR may provide a good trade-off.

D 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 additional empirical results on standard real-world networks where inference is challenging, as originally highlighted in Table 1.

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. 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 1, and in Tables 4 to 8.

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

E Data Exploitation by Closed-Form Parameter Learners: Example

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 9 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 10) to compute $\Pr(xw)$ where as factored deletion employs two thirds of the tuples (24 distinct tuples out of 36 available tuples as shown in table 10) 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 11). Compared to 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 10). In contrast to both direct and factored deletion, the informed deletion algorithm yields an estimand that involves factors with fewer elements in them ($\Pr(w)$ vs. $\Pr(zw)$) and hence

Table 4: 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	Munin 1	-	-98.95	-103.59	-98.68	-103.54	**-98.49**	Barley	-	**-85.33**	-85.84	-85.68	-86.13	-85.75
10^3		-	-79.83	-70.49	-67.27	-69.78	**-66.97**		-	-	-67.70	-67.18	-67.67	**-67.13**
10^4		-	-	-59.25	**-57.11**	-	-		-	-	**-54.93**	-	-	-

Table 5: 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 6: 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		-	-	-	-	-	-		-	-	**-15.13**	-	-	-
10^2	Munin 1	-	**-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 7: 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**	-	-	-

Table 8: Log-likelihoods of large networks learned from MAR data (5 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	-	-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**	-	-	-		-	-	-	-	-	-

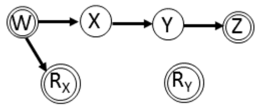
Table 9: Manifest (Data) Distribution with $\{X, Y\} \in \mathbf{X}_m$ and $\{Z, W\} \in \mathbf{X}_o$.

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

Table 10: Enumeration of sample # used for computing $\Pr(x, w)$ by listwise deletion, direct deletion and factored deletion algorithms under MCAR assumptions.

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

Table 11: Enumeration of sample # used for computing $\Pr(x, y)$ by direct deletion, factored deletion and informed deletion algorithms under MAR assumption.

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

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 estimability of queries.