Published in final edited form as:
J Mach Learn Res. 2016 April ; 17(30): 1–39.

A Gibbs Sampler for Learning DAGs

Robert J. B. Goudie and
Medical Research Council Biostatistics Unit Cambridge CB2 0SR, UK

Sach Mukherjee
German Centre for Neurodegenerative Diseases (DZNE) Bonn 53175, Germany

Peter Spirtes [Editor]

Abstract

We propose a Gibbs sampler for structure learning in directed acyclic graph (DAG) models. The standard Markov chain Monte Carlo algorithms used for learning DAGs are random-walk Metropolis-Hastings samplers. These samplers are guaranteed to converge asymptotically but often mix slowly when exploring the large graph spaces that arise in structure learning. In each step, the sampler we propose draws entire sets of parents for multiple nodes from the appropriate conditional distribution. This provides an efficient way to make large moves in graph space, permitting faster mixing whilst retaining asymptotic guarantees of convergence. The conditional distribution is related to variable selection with candidate parents playing the role of covariates or inputs. We empirically examine the performance of the sampler using several simulated and real data examples. The proposed method gives robust results in diverse settings, outperforming several existing Bayesian and frequentist methods. In addition, our empirical results shed some light on the relative merits of Bayesian and constraint-based methods for structure learning.

Keywords

structure learning; DAGs; Bayesian networks; Gibbs sampling; Markov chain Monte Carlo; variable selection

1 Introduction

We consider structure learning for graphical models based on directed acyclic graphs (DAGs). The basic structure learning task can be stated simply: given data $\mathbf{X}$ on p variables, assumed to be generated from a graphical model based on an unknown DAG G, the goal is to make inferences concerning the edge set of G (the vertex set is treated as fixed and known). Structure learning appears in a variety of applications (for an overview see Korb and Nicholson, 2011) and is a key subtask in many analyses involving graphical models, including causal inference.

In a Bayesian framework, structure learning is based on the posterior distribution $P(G \mid \mathbf{X})$ (Koller and Friedman, 2009). The domain of the distribution is the space $\mathcal{G}$ of all DAGs with p vertices. The size of the space grows super-exponentially with p, precluding exhaustive enumeration for all but the smallest problems. Markov chain Monte Carlo (MCMC) methods are widely used to sample DAGs and thereby approximate the posterior $P(G \mid \mathbf{X})$.

Europe PMC Funders Author Manuscripts

Available methods include MC^3 (Madigan and York, 1995) and related samplers (Giudici and Castelo, 2003; Grzegorczyk and Husmeier, 2008) and algorithms that sample from the space of total orders (Friedman and Koller, 2003). Order-space sampling entails some restrictions on graph priors (Eaton and Murphy, 2007; Ellis and Wong, 2008), because the number of total orders with which a DAG is consistent is not constant. Order-space approaches have more recently led to exact methods based on dynamic programming (Koivisto and Sood, 2004; Parviainen and Koivisto, 2009; Tian and He, 2009; Tamada et al., 2011; Parviainen and Koivisto, 2013; Nikolova et al., 2013). These are currently feasible only in small domain settings (typically $p < 20$ but $p < 30$ is feasible on large cluster computers) and usually share the same restrictions on graph priors as order-space samplers. Frequentist constraint-based methods such as the PC-algorithm (Spirtes et al., 2000; Colombo and Maathuis, 2014) and the approach of Xie and Geng (2008) are an alternative. These methods make firm decisions about the DAG structure via a series of tests of conditional independence.

In this paper we propose a Gibbs sampler for structure learning of DAGs that ameliorates key deficiencies in existing samplers. Random-walk Metropolis-Hastings algorithms currently used for structure learning propose 'small' changes at each iteration, which can leave the algorithms unable to escape local modes. The Gibbs sampler proposed here considers the parents of a set of nodes as a single component (a so-called 'block'), sampling an entire parent set for each node in the block in one step. These 'large' moves are sampled from the appropriate conditional posterior distribution. This enables the sampler to efficiently locate and explore areas of significant posterior mass. The method is based on the simple heuristic that the parents of a node are similar to the covariates or inputs in variable selection, with the node as the output variable, but accounts for acyclicity exactly so that the equilibrium distribution is indeed the correct posterior distribution over DAGs. The sampler does not impose restrictions on priors or graph space beyond those (maximum in-degree, modular prior) common to most samplers for structure learning (e.g. Friedman and Koller, 2003; Ellis and Wong, 2008; Grzegorczyk and Husmeier, 2008). The maximum in-degree restriction formally precludes large-sample consistency in the general case, but facilitates effective and robust inference by reducing the size of the model space (see Discussion).

2 Notation and Model

Let G denote a DAG with vertex set $V(G) = \{1, \dots, p\}$, and directed edge set $E(G) \subset V(G) \times V(G)$; often we will refer to vertex and edge sets simply as V and E respectively, leaving G implicit. The binary adjacency matrix corresponding to G is denoted A^G, with entries specified as $A^G_{uv}=1 \iff (u,v) \in E(G)$ and diagonal entries equal to zero. For inference, G is treated as a latent graph in the space $\mathscr{G}$ of all possible DAGs with p vertices. When a DAG is used to define a probabilistic graphical model (a Bayesian network), each vertex (or node; we use both terms interchangeably) is associated with a component of a p-dimensional random vector $X = (X_1 \dots X_p)^{\mathrm{T}}$. X_Z denotes the random variables (RVs) corresponding to variable indices $Z \subseteq \{1, \dots, p\}$. We use bold type for the corresponding data; n samples are collected in the $n \times p$ matrix $\mathbf{X}$, with $\mathbf{X}_u$ denoting the column corresponding to variable u

and $\mathbf{X}_Z$ the submatrix corresponding to variable index set $Z \subseteq \{1, \ldots, p\}$ (for notational simplicity we assume columns are ordered by increasing variable index).

The proposed Gibbs sampler changes entire parent sets *en masse*, not just for one node but for a set of nodes, and so it will often be convenient and natural to specify a DAG G in terms of the parents of each node. Let $\mathrm{pa}_v^G = \{u : (u, v) \in E(G)\}$ denote the set of nodes that are parents of node v in G. A tuple of p parent sets $\left\langle \mathrm{pa}_1 = \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p = \mathrm{pa}_p^G \right\rangle$ fully specifies the edge set E, since $u \in \mathrm{pa}_v^G \iff (u, v) \in E(G)$. Thus, structure learning can be viewed as inference of parent sets pa_v. The parent set pa_v takes values in the power set $\mathscr{P}(V \setminus \{v\})$ of nodes excluding node v, subject to acyclicity. We will usually suppress the labels, and simply write $\left\langle \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p^G \right\rangle$ when specifying parent sets. Since pa_v^G is a subset of the variable indices, we can use $\mathbf{X}_{\mathrm{pa}_v^G}$ to denote the corresponding data submatrix.

We denote the tuple of parent sets of a set of nodes $Z = \{z_1, \ldots, z_s\} \subseteq V$ by $\mathrm{pa}_Z^G = (\mathrm{pa}_z^G)_{z \in Z}$. This is a tuple of s components (for notational convenience we take these to be ordered by increasing vertex index), each of which is the parent set for the corresponding node in G. We wish to make inferences about pa_Z, which takes values in $\mathscr{P}(V \setminus \{z_1\}) \times \ldots \times \mathscr{P}(V \setminus \{z_s\})$, subject to acyclicity. The tuple of parent sets for the complement $Z^{\mathrm{C}} = V \setminus Z$ is denoted $\mathrm{pa}_{-Z}^G = (\mathrm{pa}_z^G)_{z \in Z^{\mathrm{c}}}$. Clearly, a tuple $\left\langle \mathrm{pa}_{Z_1}^G, \ldots, \mathrm{pa}_{Z_s}^G \right\rangle$ of tuples of parent sets specifies the parents of *every* node in a graph whenever $\{Z_1, \ldots, Z_s\}$ forms a partition of V; the entire edge set can thus be specified by such a tuple. In particular, note that any graph G can be specified as $\left\langle \mathrm{pa}_Z^G, \mathrm{pa}_{-Z}^G \right\rangle$ for any $Z \subset V$. Some of the notation we use is illustrated in Figure 1.

Our statistical formulation for Bayesian networks is standard. We briefly summarize the main points and refer the reader to the references below for details. Given a DAG G, the joint distribution of X is $p(X \mid G, \{\theta_v\}) = \prod_{v \in V} p\left(X_v \mid X_{\mathrm{pa}_v^G}, \theta_v\right)$, i.e. the joint distribution factors over nodes, and each node is conditioned on its parents in G, parameterized by θ_v. For structural inference, interest focuses on the posterior distribution $P(G \mid \mathbf{X})$. This is proportional to the product of the marginal likelihood $p(\mathbf{X} \mid G)$ and a graph prior $\pi(G)$. Our sampler is compatible with essentially any specific model, but inherits computational costs associated with evaluation of the relevant quantities. In all examples we assume conjugate priors for θ_v, as well as local parameter independence and modularity; this leads to a closed-form marginal likelihood in both multinomial (Heckerman et al., 1995) and Gaussian (Geiger and Heckerman, 1994) cases. Computations are simplified if the graph prior is modular (Friedman and Koller, 2003), meaning it factorises as $\pi(G) = \prod_{v \in V} \pi_v\left(\mathrm{pa}_v^G\right)$; we assume modular priors in all examples. Note that the prior is not specified over the space of orders and so is not subject to the restrictions this entails (Ellis and Wong, 2008; Eaton and Murphy, 2007). Under these assumptions, the posterior factorises across nodes as

$$P\left(G|\mathbf{X}\right) = P(\mathrm{pa}_1 = \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p = \mathrm{pa}_p^G|\mathbf{X}) \propto \prod_{v \in V} p(\mathbf{X}_v|\mathbf{X}_{\mathrm{pa}_v^G})\pi_v(\mathrm{pa}_v^G),$$

where $p\left(\mathbf{X}_v|\mathbf{X}_{\mathrm{pa}_v^G}\right)$ is the marginal likelihood for node v given the graph $G = \left\langle \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p^G \right\rangle$. The distribution $P(G \mid \mathbf{X})$ is the target distribution for our sampler.

3 A Gibbs Sampler for Structure Learning

In this section we provide a high-level description of the Gibbs sampler for structure learning. We first recall the standard random-walk Metropolis-Hastings sampler (known as MC^3) and then describe a naïve Gibbs sampler, which offers no gains over MC^3, but prepares the ground for the introduction of ideas from the Gibbs sampling literature. Specifically we discuss a strategy known as blocking that can improve mixing in Gibbs samplers and show how to use blocking for structure learning of DAGs. Several points relating to computation are central to developing a practical Gibbs sampler in this setting, but for clarity of exposition we defer discussion of computational aspects to Section 4.

3.1 MC^3 Sampler

The standard sampler for structure learning of DAGs is MC^3 (Madigan and York, 1995). This is a classical Metropolis-Hastings sampler with proposal G' drawn uniformly at random from the set neigh(G) of DAGs that differ from the current DAG G by the addition or removal of a single edge. The acceptance probability is min(1, $r(G', G)$), where

$$r(G', G) = min\left\{1, \frac{P(G'|\mathbf{X})|\,\mathrm{neigh}(G')|^{-1}}{P(G|\mathbf{X})|\,\mathrm{neigh}(G)|^{-1}}\right\}.$$

Variants of MC^3 include single-edge direction reversal proposals (Giudici and Castelo, 2003).

3.2 A Naïve (and Inefficient) Gibbs Sampler

Constructing a Gibbs sampler that is analogous to MC^3 is straightforward. The posterior distribution on DAGs is a joint distribution over the off-diagonal entries in the adjacency matrix A^G, i.e. over the $p(p-1)$ binary RVs $A_{uv}^G, u \neq v$. At each iteration, MC^3 can be thought of as proposing to toggle the value A_{uv}^G for some $u \neq v$, subject to the restriction that the resulting graph must be acyclic.

A simple Gibbs sampler works in a similar way. At each step, a move from graph G to a new graph G' is chosen by sampling from the conditional distribution of $A_{uv}^{G'}$ (for some $u \neq v$) given the rest of the graph. If $(u, v) \in E(G)$, define the graph $G^{(u,v)}$ to be identical to G, and $G^{-(u,v)}$ to be the graph that differs from G only in lacking an edge from u to v; conversely, if $(u, v) \notin E(G)$, define $G^{-(u,v)}$ to be identical to G, and $G^{(u,v)}$ to be the graph that differs from

G only in including an edge from u to v. If $G^{(u,v)}$ is cyclic, $G^{-(u,v)}$ is sampled as the new graph G' with probability 1. If $G^{(u,v)}$ is acyclic, the conditional distribution of $A_{uv}^{G'}$ is Bernoulli, i.e.

$$P(A_{uv}^{G'}=a \mid A_{-uv}^{G})=\begin{cases} \frac{P(G^{-(u,v)}\mid \mathbf{X})}{P(G^{-(u,v)}\mid \mathbf{X})+P(G^{(u,v)}\mid \mathbf{X})} & a=0, \\ \frac{P(G^{(u,v)}\mid \mathbf{X})}{P(G^{-(u,v)}\mid \mathbf{X})+P(G^{(u,v)}\mid \mathbf{X})} & a=1. \end{cases}$$

The choice of u and v can either be made sequentially (systematically) or randomly (Roberts and Sahu, 1997); in this paper, random-scan Gibbs samplers are used throughout. We prove convergence of the Gibbs sampler to the target distribution in Appendix A.

3.3 Inefficiency in Gibbs Sampling

The mixing of Metropolis-Hastings samplers depends upon the proposal distribution, which for convenience is often chosen as a random-walk. In contrast, Gibbs samplers make moves according to conditional distributions that reflect local structure of the target distribution. Nonetheless, Gibbs sampling is not always efficient. In particular, correlation between the components being sampled can lead to inefficiency. To see this, consider a Gibbs sampler for a multivariate continuous distribution with highly correlated components. At each step, a single component is sampled according to its conditional distribution, but since it is strongly correlated with other component(s), the conditional is concentrated on only a small part of the support. This means the sampler is likely to make only small moves. Analogous issues arise with discrete distributions.

For graphical models based on DAGs, there may be strong dependence between the edge indicators A_{uv}^{G}, particularly for the collections of RVs corresponding to parent sets. For example, there may be RVs X_u and X_v that in combination score highly as parents of node w, but not individually. Then, A_{uw}^{G} and A_{vw}^{G} will be correlated. In addition, the acyclicity restriction may induce strong dependence. For example, suppose two RVs X_u and X_v are strongly correlated and both edges (u, v) and its reverse (v, u) have high posterior probability. If the edge (u, v) is present, the probability of it being removed is low, but its presence precludes the reversed edge (v, u) from being added. This possibility motivates the edge reversal move used in variants of MC^3 (Giudici and Castelo, 2003). Figure 2 shows a three node scenario. Here, both graphs (a) and (b) have high probability. Since reversing the edge (w, x) forms a cycle in (a), moves that consider the parents of only the pair w and x at the same time will not move between graphs (a) and (b) easily. Samplers that alter only a single edge indicator, such as MC^3, will also fail. In contrast, sampling the parents of all three nodes jointly would make it easy to move between graphs (a) and (b). Decorrelating transformations could alleviate such problems, but in general finding a suitable transformation is difficult, and for DAGs must of course encapsulate the requirement for acyclicity.

3.4 A Gibbs Sampler with Blocking

We address the deficiencies of the naïve Gibbs sampler by grouping a number of the components together and sampling from their joint conditional distribution. In Gibbs sampling, this is known as *blocking*. Sampling from such a joint conditional can ameliorate difficulties caused by correlations between components because the joint conditional naturally accounts for the correlation structure. In the multivariate normal case, Roberts and Sahu (1997) have shown that blocking improves convergence for random-scan Gibbs sampling.

In principle, any group of components can be taken as a block, but for the algorithm to be useful in practice, sampling from a block's joint conditional distribution must be feasible, and ideally simple. The blocks that we consider correspond to groups (specifically tuples) of parent sets, so that the parent sets of several nodes are considered simultaneously. It is natural that each block is a tuple of parent sets because, as described in Section 2, we can specify a graph G using a tuple $\left\langle \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p^G \right\rangle$ of parent sets. It is therefore convenient to describe the sampler using this alternative graph specification, in which $G = \left\langle \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p^G \right\rangle$.

We denote the set of q nodes whose parent sets will be sampled together as a block by $W = \{w_1, \ldots, w_q\} \subseteq V$. Suppose the current graph is $G = \left\langle \mathrm{pa}_W^G, \mathrm{pa}_{-W}^G \right\rangle$ (recall that any graph can be written in this way with respect to any partition of the node set). A move to a new graph $G' = \left\langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G'} \right\rangle$ is formed by changing the parents of the nodes in W from $\mathrm{pa}_W^G = (\mathrm{pa}_{w_1}^G, \ldots, \mathrm{pa}_{w_q}^G)$ to $\mathrm{pa}_W^{G'} = (\mathrm{pa}_{w_1}^{G'}, \ldots, \mathrm{pa}_{w_q}^{G'})$ and setting $\mathrm{pa}_{-W}^{G'} = \mathrm{pa}_{-W}^G$ (i.e. leaving the parents of nodes not in W unchanged). We sample the tuple $\mathrm{pa}_W^{G'}$ of parent sets jointly, conditional on the tuple pa_{-W}^G of parent sets of nodes not in W (that remain unchanged). In terms of the adjacency matrix A^G, each block consists of the indicators specifying the parents of the nodes in W, i.e. A_{vw}^G for $v \in V$, $w \in W$, $v \neq w$.

To construct a Gibbs sampler using these blocks, we need to find the conditional posterior distribution on the tuple pa_W of parent sets, given that the remaining tuple pa_{-W} of parent sets is set to pa_{-W}^G. The conditional distribution depends on whether the graph G' formed using the proposed parent sets is acyclic. We therefore introduce Pa_W^G to denote the set of permissible tuples, i.e. tuples $\mathrm{pa}_W^{G'}$ of parent sets such that $G' = \left\langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^G \right\rangle$ is acyclic. For tuples $\mathrm{pa}_W^{G'} \notin \mathrm{Pa}_W^G$, the conditional probability is 0. For $\mathrm{pa}_W^{G'} \in \mathrm{Pa}_W^G$, the conditional probability is

$$P(\mathrm{pa}_W = \mathrm{pa}_W^{G'} | \mathrm{pa}_{-W} = \mathrm{pa}_{-W}^{G}, \mathbf{X}) = \frac{P(\langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G} \rangle | \mathbf{X})}{P(\mathrm{pa}_{-W}^{G} | \mathbf{X})} = \frac{P(G' | \mathbf{X})}{\sum_{\mathrm{pa}_W^{G''} \in Pa_W^G} P(\mathrm{pa}_W^{G''}, \mathrm{pa}_{-W}^{G} | \mathbf{X})}. \quad (1)$$

This is the conditional distribution needed to specify the blocked Gibbs sampler; Algorithm 1 outlines the procedure at a high level, with the set W of nodes chosen uniformly at random at each step. Asymptotic convergence follows from the argument in Appendix A (in fact the requirements on the graph prior will be weaker than in the naïve case).

Algorithm 1 A Gibbs sampler for learning DAGs, with blocks

Initialise starting point $G_0 = \langle \mathrm{pa}_1^{G_0}, \ldots, \mathrm{pa}_p^{G_0} \rangle$

for t in 1 to N **do**

Sample q nodes uniformly at random from V, and call this set of nodes W

Sample $\mathrm{pa}_W^{G'}$ from $P(\mathrm{pa}_W = \mathrm{pa}_W^{G'} | \mathrm{pa}_{-W}^{G_{t-1}}, \mathrm{X})$

Set $G_t \leftarrow G = \langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G_{t-1}} \rangle$

end for

3.5 Tuning Parameters

To reduce the computational costs of structure learning it is common to set a maximum in-degree (e.g. Friedman and Koller, 2003; Grzegorczyk and Husmeier, 2008). We set a maximum in-degree $\kappa = 3$ in all empirical examples, except where stated otherwise (Section 5). This facilitates sampling from the conditional distribution in Equation 1 by reducing the computational cost of evaluating the normalising constant. We set the block size $q = |W| = 3$. While q can be chosen freely in principle, evaluating the conditional distribution in Equation 1 becomes unmanageable when q is large. Thus, both κ and q act as tuning parameters (and are in addition to any hyper-parameters in the Bayesian formulation).

3.6 Structure Learning and Variable Selection

It is interesting to note that when $q = 1$ (and thus $W = w$ for some $w \in V$) if no choice of parent set induces a cycle then $Pa_W^G = \mathscr{P}(V \setminus \{w\})$, the power set of the remaining nodes. In this case, the conditional distribution in Equation 1 can be viewed as the posterior distribution of a variable selection problem with response or output variable w, and the other variables as covariates or inputs. If the addition of particular nodes introduces a cycle, we have a constrained problem. In particular, suppose adding any of the nodes $Z \subset V$ as parents

Europe PMC Funders Author Manuscripts

of node w would create a cycle. Then $Pa_W^G = \mathscr{P}(V \setminus \{w\} \cup Z))$ and the conditional distribution in Equation 1 is a constrained variable selection problem in which the nodes Z are excluded from the set of candidate inputs.

4 Computational Aspects

Designing a computationally efficient sampler is not straightforward in this setting. To see why, note that to sample from the conditional in Equation 1 we need to be able to identify the set Pa_W^G of tuples of parent sets that is permissible (in the sense of maintaining acyclicity). The cardinality of this set is typically large, and the interdependence between the parent sets of each node in W makes decomposing the problem into subproblems non-trivial. A simple but naïve approach would list all possible parent sets for each node in W and check each such combination for cyclicity, but this approach is slow and cumbersome.

We propose a partitioning scheme that leads to a two-stage sampler. The key idea is to choose the partition of Pa_W^G so that, conditional on a component of the partition, the parents of each node are independent. This enables an efficient two-stage sampling method that we describe in Section 4.3. Acyclicity constraints are met by an efficient dynamic algorithm.

4.1 A Partition on Permissible Tuples of Parent Sets

We partition the set Pa_W^G of permissible tuples of parent sets as $Pa_W^G = \left\{ Pa_W^{G,H_1}, \ldots, Pa_W^{G,H_\eta} \right\}$. Each component Pa_W^{G,H_h} is a set of permissible tuples of parent sets for nodes in W. It is convenient to label the partition components using secondary (unrelated to G) DAGs $H_h \in \mathscr{H}$, where $\mathscr{H}$ is the space of DAGs with $q = |W|$ vertices; η is the cardinality of $\mathscr{H}$. We describe the relationship between each Pa_W^{G,H_h} and DAG H_h using the following elements, illustrated in Figure 3 and Table 1:

- The reduced graph $\overline{G} = \left\langle \mathrm{pa}_1^{\overline{G}}, \ldots, \mathrm{pa}_p^{\overline{G}} \right\rangle$, which is a function of the graph G, and is identical to G except that edges directed into nodes in W are removed.

$$\mathrm{pa}_w^{\overline{G}} = \begin{cases} \varnothing & w \in W, \\ \mathrm{pa}_w^G & w \notin W \end{cases}$$

- The (reflexive) transitive closure, which is the directed graph T^G on nodes V with edges E^{T}, where $(u, v) \in E^{\mathrm{T}}$ if and only if $u = v$ or a (directed) path from u to v exists in G (i.e. there exists a sequence of nodes $z_1, \ldots, z_s \in V(G)$, with $z_1 = u$ and $z_s = v$, such that $(z_1, z_2), (z_2, z_3), \ldots, (z_{s-1}, z_s) \in E(G)$).

- The descendant nodes $\mathrm{de}_w^{\overline{G}} = \left\{ v \in V : \mathrm{T}_{wv}^{\overline{G}} = 1 \right\}$ and the non-descendant nodes $\mathrm{nd}_w^{\overline{G}} = \left\{ v \in V : \mathrm{T}_{wv}^{\overline{G}} = 0 \right\}$ of $w \in W$ in the reduced graph $\overline{G}$. Note that $w \in \mathrm{de}_w^{\overline{G}}$ and $w \notin \mathrm{nd}_w^{\overline{G}}$ by definition.

- The nodes $\mathrm{nd}^{\overline{G}}=\cap_{w\in W}\mathrm{nd}_w^{\overline{G}}$ that are not descendants in the reduced graph $\overline{G}$ of any node in W.
- The nodes $\mathrm{de}_w^{\overline{G},H}=\cup_{x\in \mathrm{pa}_w^H}\mathrm{de}_x^{\overline{G}}$ that are descendants in the reduced graph $\overline{G}$ of any node $x\in \mathrm{pa}_w^H$, for a given node $w\in W$. Each node $x\in \mathrm{pa}_w^H$ is a parent node of the node w in the graph $H=\left\langle \mathrm{pa}_{w_1}^H,\ldots,\mathrm{pa}_{w_q}^H\right\rangle$.
- The nodes $\mathrm{de}_{-w}^{\overline{G},H}=\cup_{x\in W\setminus \mathrm{pa}_w^H}\mathrm{de}_x^{\overline{G}}$ that are descenda nts in the reduced graph $\overline{G}$ of any node $x\in W\setminus \mathrm{pa}_w^H$, for a given node $w\in W$. Each node $x\in W\setminus \mathrm{pa}_w^H$ is not a parent node of w in the graph $H=\left\langle \mathrm{pa}_{w_1}^H,\ldots,\mathrm{pa}_{w_q}^H\right\rangle$.

Using this notation, we define $P\mathrm{a}_w^{G,H}$, for given $G\in\mathcal{G}$, $W\subseteq V$, $w\in W$ and $H=\left\langle \mathrm{pa}_{w_1}^H,\ldots,\mathrm{pa}_{w_q}^H\right\rangle\in\mathcal{H}$, as the set of parent sets $\mathrm{pa}_w^{G'}$ that satisfy the following conditions.

(A) $\mathrm{pa}_w^{G'}\subseteq Q_w^{\overline{G},H}=\left(\mathrm{nd}^{\overline{G}}\cup\mathrm{de}_w^{\overline{G},H}\right)\setminus\mathrm{de}_{-w}^{\overline{G},H}$

(B) $\mathrm{pa}_w^{G'}\cap R_{w,x}^{\overline{G},H}\neq\varnothing$ for all nodes $x\in\mathrm{pa}_w^H$, with $R_{w,x}^{\overline{G},H}=\mathrm{de}_x^{\overline{G}}\setminus\mathrm{de}_{-w}^{\overline{G},H}$.

Note that (B) depends on $\mathrm{de}_x^{\overline{G}}$ not $\mathrm{de}_x^{\overline{G},H}$. We define $P\mathrm{a}_W^{G,H}$ as the set of tuples formed by the Cartesian product of the sets $P\mathrm{a}_W^{G,H}$ of parent sets for $w\in W$; in other words $\mathrm{pa}_W^{G'}=(\mathrm{pa}_{w_1}^{G'},\ldots,\mathrm{pa}_{w_q}^{G'})\in P\mathrm{a}_W^{G,H}$ if and only if $\mathrm{pa}_{w_1}^{G'}\in P\mathrm{a}_{w_1}^{G,H},\ldots,\mathrm{pa}_{w_q}^{G'}\in P\mathrm{a}_{w_q}^{G,H}$.

Lemma 1 $\{P\mathrm{a}_W^{G,H_1},\ldots,P\mathrm{a}_W^{G,H_\eta}\}$ *is a partition of* $P\mathrm{a}_W^G$.

Proof See Appendix B.

Condition (A) ensures all graphs $G'=\left\langle \mathrm{pa}_W^{G'},\mathrm{pa}_{-W}^G\right\rangle$, with parents $\mathrm{pa}_W^{G'}\in P\mathrm{a}_W^{G,H}$, are acyclic. It requires that all parents of $w\in W$ are either not descendants in the reduced graph $\overline{G}$ of any node in W, or a node whose ancestors in the reduced graph $\overline{G}$ are all parents in the graph H of node w. In particular, no descendant of w is added as an ancestor of w.

Condition (B) ensures each DAG G' is a member of $P\mathrm{a}_W^{G,H}$ for only a single $H\in\mathcal{H}$ for any given $G\in\mathcal{G}$ and thus that $\left\{P\mathrm{a}_W^{G,H_1},\ldots,P\mathrm{a}_W^{G,H_\eta}\right\}$ is a partition of the set of permissible tuples of parent sets. The condition checks that each edge in the graph H is 'used' i.e. that $\mathrm{pa}_W^{G'}=\left(\mathrm{pa}_{w_1}^{G'},\ldots,\mathrm{pa}_{w_q}^{G'}\right)\in P\mathrm{a}_W^{G,H}$ would not be allowed under any $H'\neq H$, $H'\in\mathcal{H}$. Specifically, it ensures that each parent set $\mathrm{pa}_w^{G'}$ contains at least one descendant node $v\in V$ of each parent pa_w^H of node w in the graph H, and that v is not a descendant in $\overline{G}$ of any node $x\in W$ that is not a parent in H of $w\in W$. To see why this condition is required, consider a

graph G' in which all nodes in W have no parents. Then $\mathrm{pa}_w^{G'}=\varnothing$ for all nodes $w\in W$ and so condition (A) holds for all $H\in\mathcal{H}$. Thus if condition (B) is disregarded, $\mathrm{pa}_W^{G'}\in Pa_W^{G,H}$ for all $H\in\mathcal{H}$, implying that $\left\{Pa_W^{G,H_1},\ldots,Pa_W^{G,H_\eta}\right\}$ is not a partition of Pa_W^G.

4.2 Fast Identification of Partition Components

Parent sets in each set $Pa_w^{G,H}$ can be identified easily using simple set operations, and consequently the partition components $Pa_w^{G,H}$ can be easily identified via Cartesian products of these sets. Let $Pa_w^{\mathrm{all}}=\mathscr{P}(V\setminus\{w\})$ denote the set of all possible parent sets of node w, subject to maximum in-degree κ, and $Pa_w^{\mathrm{all}}(v)=Pa_w^{\mathrm{all}}\setminus\mathscr{P}(V\setminus\{w,v\})$ denote the subset all Pa_w^{all} containing only parent sets that contain node $v\in V$. Parent sets in $Pa_w^{G,H}$ must (A) not include any nodes *not* in $Q_w^{\overline{G},H}$, and (B) include at least one node in $R_{w,x}^{\overline{G},H}$ for each $x\in\mathrm{pa}_w^H$, and thus

$$Pa_w^{G,H}=\begin{cases} Pa_w^{(B)}\setminus Pa_w^{(A)} & \text{if } \mathrm{pa}_w^H\neq\varnothing \\ Pa_w^{\mathrm{all}}\setminus Pa_w^{(A)} & \text{if } \mathrm{pa}_w^H=\varnothing, \end{cases}$$

where $Pa_w^{(A)}=\cup_{v\notin Q_w^{\overline{G},H}}Pa_w^{\mathrm{all}}(v)$ and $Pa_w^{(B)}=\cap_{x\in\mathrm{pa}_w^H}\cup_{r\in R_{w,x}^{\overline{G},H}}Pa_w^{\mathrm{all}}(r)$.

We fix $q=|W|$ as a small constant for all p and set a maximum in-degree κ. This means permissible tuples of parents sets can be identified in $\mathcal{O}(p^{\kappa+1})$ time by storing the 'lookup tables' $Pa_w^{\mathrm{all}}(v)$ as bit maps, assuming that $\mathcal{O}(1)$ querying of the descendants and non-descendants is available.

This can be achieved using a dynamic algorithm that provides access to the transitive closure. Giudici and Castelo (2003) previously used a similar approach. For a graph with transitive closure T^G with edges E^{T}, the descendants and non-descendants of node u are $\{v:(u,v)\in E^{\mathrm{T}}\}$ and $\{v:(u,v)\notin E^{\mathrm{T}}\}$ respectively. Thus, the descendants and non-descendants of a node can be identified in $\mathcal{O}(1)$ time when the transitive closure is known. The transitive closure for an arbitrary directed graph with p vertices can be determined in $\mathcal{O}(p^\omega)$ time (Munro, 1971), where ω is the best known exponent for matrix multiplication (Coppersmith and Winograd, 1990, show $\omega<2.376$). However, in the present setting only incremental changes are made to the current state G of the sampler, so it is not necessary to use an offline algorithm at each iteration. Instead, we can use a fully dynamic transitive closure algorithm (Demetrescu et al., 2010) that provides procedures both for querying the transitive closure, and for incrementally updating it when an edge is added or removed from the graph. We choose to implement the algorithm introduced by King and Sagert (2002), which performs queries in $\mathcal{O}(1)$ time, and updates in $\mathcal{O}(p^2)$ worst-case time (see reference for details of required assumptions). A trade-off exists between the performance of these two operations, but this bound for updates is thought to be the best possible whilst retaining $\mathcal{O}(1)$ queries (Demetrescu and Italiano, 2005) and the algorithm is simple to implement.

The algorithm maintains a $p \times p$ path count matrix C^G whose elements C^G_{uv} are the number of distinct paths from node u to node v in the graph G for $u \neq v$, and $C^G_{uv}=1$ for $u = v$. The transitive closure can be derived from the path count matrix by noting that $T^G_{uv}=1$ if and only if $C^G_{uv}>0$. Thus query operations are performed in $\mathcal{O}(1)$ by simply checking whether the relevant component of C^G is positive. When an edge is added or removed C^G is updated as follows. First consider adding an edge (u, v) to a graph G to form a new graph G'. The increase in the number of distinct paths between any two nodes x and y is given by the (x, y) element of $C^G_{*u} \otimes C^G_{v*}$, where C^G_{*u} denotes the u^{th} column of C^G, C^G_{v*} denotes the v^{th} row and $\otimes$ denotes the outer product. Thus, the path count matrix is updated simply as $C^{G'}=C^G+C^G_{*u} \otimes C^G_{v*}$. Similarly, when an edge (u, v) is removed from the graph the update is $C^{G'}=C^G - C^G_{*u} \otimes C^G_{v*}$.

4.3 Two-stage Sampling Method

We draw a new graph $G'=\left\langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \right\rangle$ starting from a graph G using a two-stage method: we sample first a component $Pa^{G,H'}_W$ of the partition of permissible tuples of parent sets and then a tuple $\mathrm{pa}^{G'}_W$ of parent sets from the selected partition component.

In the first stage, $Pa^{G,H'}_W$ is drawn from the conditional distribution, given the tuple of parent sets of nodes not in W.

$$P(Pa^{G,H'}_W|\mathrm{pa}^G_{-W}, \mathbf{X})=\frac{P\left(Pa^{G,H'}_W,\mathrm{pa}^G_{-W}|\mathbf{X}\right)}{P\left(\mathrm{pa}^G_{-W}|\mathbf{X}\right)}$$
$$=\frac{\sum_{\mathrm{pa}^{G'}_W \in Pa^{G,H'}_W}\prod_{w\in W} p\left(\mathbf{X}_w|\mathbf{X}_{\mathrm{pa}^{G'}_w}\right)\pi_w\left(\mathrm{pa}^{G'}_w\right)}{\sum_{H''\in\mathcal{H}}\sum_{\mathrm{pa}^{G''}_W \in \mathrm{Pa}^{G,H''}_W}\prod_{w\in W} p\left(\mathbf{X}_w|\mathbf{X}_{\mathrm{pa}^{G''}_w}\right)\pi_w\left(\mathrm{pa}^{G''}_w\right)}$$
$$=\frac{\prod_{w\in W}\sum_{\mathrm{pa}^{G'}_w \in Pa^{G,H'}_w} p\left(\mathbf{X}_w|\mathbf{X}_{\mathrm{pa}^{G'}_w}\right)\pi_w\left(\mathrm{pa}^{G'}_w\right)}{\sum_{H''\in\mathcal{H}}\prod_{w\in W}\sum_{\mathrm{pa}^{G''}_w \in Pa^{G,H''}_w} p\left(\mathbf{X}_w|\mathbf{X}_{\mathrm{pa}^{G''}_w}\right)\pi_w\left(\mathrm{pa}^{G''}_w\right)}$$

The final equality follows by an interchange of sum and product that is proved in Lemma 2. This makes evaluation more efficient by allowing the sums to be evaluated separately for each node. Friedman and Koller (2003) used a similar interchange.

Lemma 2 *The following identity holds for any $H\in\mathcal{H}$, $W\subseteq V$ and $G\in\mathcal{G}$.*

$$\sum_{\mathrm{pa}_W \in Pa^{G,H}_W}\prod_{w\in W} p\left(\mathbf{X}_w|\mathbf{X}_{\mathrm{pa}_w}\right)\pi_w\left(\mathrm{pa}_w\right)=\prod_{w\in W}\sum_{\mathrm{pa}_w \in Pa^{G,H}_w} p\left(\mathbf{X}_w|\mathbf{X}_{\mathrm{pa}_w}\right)\pi_w\left(\mathrm{pa}_w\right)$$

Proof See Appendix C.

Algorithm 2 A Gibbs sampler for learning DAGs, with general blocks

Initialise starting point $G_0 = \left\langle \mathrm{pa}_1^{G_0}, \ldots, \mathrm{pa}_p^{G_0} \right\rangle$

Compute initial path count matrix C^{G_0}

for t in 1 to N **do**

Sample q nodes uniformly at random from V, and call this set of nodes W

Let $G = G_{t-1}$

Form $\overline{G}$ as defined in Section 4.1

Evaluate $C^{\overline{G}}$ from C^G as described in Section 4.2

for $w \in W$ **do**

Evaluate $\mathrm{de}_w^{\overline{G}} = \left\{ v : C_{wv}^{\overline{G}} \geq 1 \right\}$

Evaluate $\mathrm{nd}_w^{\overline{G}} = \left\{ v : C_{wv}^{\overline{G}} = 0 \right\}$

end for

for $H \in \mathcal{H}$ **do**

for $w \in W$ **do**

Evaluate $Pa_w^{G,H}$, as described in Section 4.2

Let $K_w^H = \sum_{\mathrm{pa}_w^{G'} \in Pa_w^{G,H}} p\left(\mathbf{X}_w | \mathbf{X}_{\mathrm{pa}_w^{G'}}\right) \pi_w\left(\mathrm{pa}_w^{G'}\right)$

end for

Let $K^H = \prod_{w \in W} K_w^H$

end for

Sample $Pa_W^{G,H'}$ according to $P\left(Pa_W^{G,H'} | \mathrm{pa}_{-W}^G, \mathbf{X}\right) = \frac{K^{H'}}{\sum_{H'' \in \mathcal{H}} K^{H''}}$

for $w \in W$ **do**

Sample $\mathrm{pa}_W^{G'}$ according to $P\left(\mathrm{pa}_w^{G'} | Pa_W^{G,H'}, \mathrm{pa}_{-W}^G, \mathbf{X}\right)$

end for

Set $G_t \leftarrow G = \left\langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G_{t-1}} \right\rangle$

Update C^{G_t}

Europe PMC Funders Author Manuscripts

end for

In the second stage, we sample new parents $\mathrm{pa}_W^{G'}$ from the selected partition component, and form the new graph $G' = \left\langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G} \right\rangle$. The parents of each node $w \in W$, conditional on H', are independent, and so can be sampled separately from the following conditional distribution:

$$P\left(\mathrm{pa}_w^{G'} | P\mathrm{a}_W^{G,H'}, \mathrm{pa}_{-W}^{G}, \mathbf{X}\right) = \frac{p\left(\mathbf{X}_w | \mathbf{X}_{\mathrm{pa}_w^{G'}}\right) \pi_w\left(\mathrm{pa}_w^{G'}\right)}{\sum_{\mathrm{pa}_w^{G''} \in P\mathrm{a}_w^{G,H'}} p\left(\mathbf{X}_w | \mathbf{X}_{\mathrm{pa}_w^{G''}}\right) \pi_w\left(\mathrm{pa}_w^{G''}\right)}.$$

This step is straightforward because this distribution is simply the posterior distribution of a constrained variable selection with response w and $P\mathrm{a}_w^{G,H'}$ as the set of possible active sets (i.e. selected covariates).

Algorithm 2 outlines the complete algorithm. The methods in Sections 4.2 enable fast identification of each partition component. Run-time is a function of p, maximum in-degree κ, and the number q of nodes in the block. We choose a small, fixed q for all p, so the run-time is determined by the evaluation of $P\mathrm{a}_w^{G,H}$, which is $\mathcal{O}(p^{\kappa+1})$.

5 Results

In this section, we empirically assess the performance of the proposed sampler, comparing it with existing samplers as well as frequentist methods for structure learning of DAGs.

5.1 Evaluation Setup

We compare the Gibbs sampler with the MC^3 sampler (Madigan and York, 1995) and the REV sampler (Grzegorczyk and Husmeier, 2008), a variant of MC^3 that uses a more extensive edge reversal move. We also compare with two frequentist constraint-based methods: the PC-algorithm (Spirtes et al., 2000), and the Xie and Geng (2008) method, that is shown by its authors to outperform the PC-algorithm in some settings.

Tuning parameters for each method were set as follows. For the constraint-based methods, the significance level was $\alpha = 0.05$ by default, but we also show some results for $\alpha =$ 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1. The Gibbs sampler we use is a random-scan sampler, with $q = 3$ (i.e. the parent sets of three nodes are sampled jointly at each iteration). To permit a fair comparison, for MC^3 we use the same fast online transitive closure algorithm (Section 4.2), and pre-computation and caching of local marginal likelihoods used in the Gibbs sampler (REV uses a similar pre-computation and caching scheme). We constrain all of the samplers to maximum in-degree $\kappa = 3$ and set the graph prior as $\pi(G) \propto 1$. We use conjugate formulations throughout, specifically multinomial-

Europe PMC Funders Author Manuscripts

Dirichlet (Heckerman et al., 1995) for discrete data and Normal with a g-prior (with $g = n$) (Geiger and Heckerman, 1994; Zellner, 1986) for continuous data.

We consider six examples: a small domain example, where comparison with the exact posterior (Tian and He, 2009) is possible; data simulated from the ALARM network and from randomly-generated networks of varying sparsity; data sets from social science and biology; and a pathological 4-node example designed to highlight a failure case for our method.

In practical use samplers can only be run for a finite number of iterations (depending on available time and computational resources). We set the maximum number of iterations as follows. In total, we drew 10^6 iterations of REV (retaining only every 10^{th} iteration to reduce storage requirements). Following Grzegorczyk and Husmeier (2008), 85% of proposals within REV were MC^3 proposals (without MC^3 proposals, the REV sampler is not irreducible). In our implementation, the computational costs of the Gibbs sampler are an order of magnitude lower than REV's (accounting for MC^3 moves), but we nevertheless treated the computational costs as the same for the purposes of comparison and drew 10^6 iterations of the Gibbs sampler (again retaining every 10^{th} iteration). The computational costs of our implementation of MC^3 are roughly 1/10 of a Gibbs iteration, and so we performed 10^7 iterations of MC^3 (retaining every 100^{th}).

For each sampler, 10 independent runs starting from different initial graphs were performed. We discarded the first 1/4 of samples, but as an additional filter we considered trace plots (log marginal likelihood versus iteration) for each run of each sampler and discarded further samples if they appeared to be from a pre-convergent phase. Specifically, if there was a clear 'step change' in the trace for a certain run we discarded all samples in that run drawn prior to the highest 'step' being reached. We note that this simple filter cannot be regarded as detecting convergence because the highest 'step' may correspond to a region near a local rather than a global mode. We therefore also study convergence in detail via other metrics.

5.2 Evaluation Metrics

Each sampler gives an estimated posterior distribution over DAGs. From this we obtain a point estimate either as the **maximum a posteriori** (MAP) graph G^{MAP}, or as a **thesholded graph** G_{τ} formed by including all directed edges whose marginal posterior edge probabilities are at least τ (note that G_{τ} need not be acyclic). When $\tau = 0.5$ we get the **median probability model** G^{med} (for normal linear models Barbieri and Berger, 2004, show that in some settings this is the optimal model for prediction). We also consider thresholding to match sparsity levels seen in frequentist point estimates, i.e. setting τ such that $\left|E(G_{\tau})\right| = \left|E(\hat{G})\right|$ for a point estimate $\hat{G}$. By G_{τ}^{PC} and G_{τ}^{Xie} we denote thresholded graphs whose sparsity is matched to the PC and Xie-Geng estimates respectively.

We use the structural Hamming distance (SHD) to quantify differences between DAGs. This is the minimum number of edge insertions and deletions needed to transform one graph into another. Comparisons are made using completed partially directed acyclic graphs (Chickering, 2002a). To show the behaviour of the samplers at shorter MCMC run lengths

some metrics are shown against iteration t. These are based on the final 3/4 of samples drawn up to t (except for log marginal likelihoods), and so may incorporate some samples drawn before convergence.

We assess convergence and stability via the following metrics:

- **Trace plots** of log marginal likelihood against iteration t.
- Between-run agreement between **posterior edge probabilities**. These are visualized as scatter plots. When edge probabilities agree between a pair of runs, all points in the plot will lie close to the $y = x$ line. We show two variants: a hexagonally-binned version, to avoid over-plotting (Carr et al., 1987); and a panelled plot, in which each panel shows one pair of runs. We also consider the number of edges with estimated posterior probability greater than 0.9 in one run, and less than 0.1 in another run. Such **'major discrepancies'** represent serious Monte Carlo artefacts.
- **Potential scale reduction factor (PSRF)** is a multiple-chain convergence diagnostic, developed by Gelman and Rubin (1992), that compares within- and between-chain means and variances of a suitable summary statistic. PSRF < 1.1 is often taken to indicate that a sampler has run sufficiently long. The natural summary statistics in this context are the posterior edge probabilities. However, we find that the resulting PSRF is not a reliable indicator of convergence. In simulations (not shown) using a 20 node network for which we could calculate true posterior edge probabilities (Tian and He, 2009), we found little association between single-edge PSRF and the (absolute) error in posterior edge probability. In particular, the diagnostic appeared to be very conservative for edges with true posterior probability near 0 or 1. We therefore suggest that a reasonable use of PSRF in this context may be to assess relative convergence between different samplers, rather than as an absolute indicator of convergence. Below, we compare samplers in terms of the proportion of edges with PSRF < 1.1 (a larger proportion suggests better mixing). To calculate PSRF we consider the 10 runs of each sampler as a collection of 5 pairs of runs and calculate PSRF separately for each edge using the final three quarters of the samples drawn up to that point.
- For the real data, we assess **stability under resampling** ("shaking the data") by comparing estimates across bootstrap samples.

For experiments in which the true data-generating graph G^* is known we use the following metrics to assess accuracy:

- **Structural Hamming distance** (SHD) between G^* and the estimated graphs.
- **Receiver-operating characteristic (ROC) curves**. These show agreement between G^* and an estimate G_τ by plotting true positive against false positive rates parameterized by threshold τ. We consider also the **area under the ROC curve** (AUROC), focusing in particular on the small false positive rate region of the curve that is often of interest in applications.

Europe PMC Funders Author Manuscripts

Finally, when the posterior edge probabilities can be computed exactly (Tian and He, 2009) (i.e. in small p settings) we also consider the **maximum and average absolute error in posterior edge probability**, calculated across the set of all possible edges.

We note that while SHD and ROC scores are useful in assessing performance, they are not convergence measures, as they do not assess accuracy of the posterior distribution *per se* (to see why, consider a degenerate sampler that samples only G^*, giving perfect scores on SHD and ROC, but an incorrect posterior distribution).

5.3 Small Domain Comparison to Exact Posterior

We applied the methods to the Zoo data set (Newman et al., 1998) that records $p = 17$ (discrete) characteristics of $n = 101$ animals. The maximum log marginal likelihood found by the Gibbs and REV samplers was consistent across runs, but less so for the MC^3 sampler, and the Gibbs sampler reached a plateau of high probability after the fewest iterations (Figure A13a). REV required about ten times as many iterations as Gibbs to achieve the same proportion of edges with PSRF < 1.1 while MC^3 needed about 100 times as many (Figure A14a). The estimated edge probabilities given by the Gibbs sampler were stable between runs (Figure A15a). The results from the REV sampler were also stable, but MC^3 less so, with major disparities between some runs. Figure 4 shows the error in posterior edge probabilities as a function of iterations. Convergence was quickest for the Gibbs sampler, followed by REV and then MC^3. All runs of the Gibbs sampler reached a point at which the maximum absolute error was 0.05 (after 67,000 iterations on average). In contrast, only 5/10 runs of REV and 6/10 runs of MC^3 reached the same level of accuracy in their complete run. Similarly, the Gibbs sampler achieved an average absolute error of less than 0.01 in the fewest iterations.

5.4 The ALARM Network

The ALARM network (Beinlich et al., 1989) consists of 37 discrete nodes and 46 edges and has been widely used in studying structure learning (e.g. Friedman and Koller, 2003; Grzegorczyk and Husmeier, 2008). We simulated data sets from ALARM with sample sizes $n = 100, 500, 1000, 2500, 5000$. Figure 5 shows trace plots for the $n = 1000$ case as an illustrative example. The maximum log marginal likelihood found by the Gibbs sampler across runs was $-10{,}608.20 \pm 0.8$ (mean ± standard deviation), whereas for REV it was $-10{,}627.2 \pm 40.4$ and for MC^3 $-11{,}176.9 \pm 273.7$. The highest scoring graph discovered by any of the 10 runs of the MC^3 sampler had log marginal likelihood $-10{,}856.6$, far below the Gibbs maximum, suggesting non-convergence in all 10 runs. REV appeared to reach convergence in all but two runs (runs 9 and 10). The Gibbs sampler consistently (and rapidly) reached a high scoring plateau and appeared to have converged in all 10 runs. PSRF results were in line with these findings (Figure A14b); more than 10 times as many iterations of REV were needed to give the same proportion of edges with PSRF < 1.1 as the Gibbs sampler.

Figure 6 compares pairs of runs for $n = 1000$; this pair of runs is typical of all 10 runs (except for runs 9 and 10 of REV which disagree considerably; all runs shown in Figure A15b). There were no major between-run discrepancies (as defined in Section 5.2) for the

Gibbs sampler at any sample size. The mean number of major discrepancies (across pairs of runs) increased from 0 ($n = 100$) to 8 and 91 for MC^3 and REV respectively (when $n = 5000$).

Figure 7 shows ROC curves for false positive rates < 0.05. The Gibbs sampler performs better and with less variability than the other methods. Sample size is influential. For small n the Bayesian methods outperform the constraint-based methods. However, counter to the increase in statistical information, REV and MC^3 perform less well with larger n: e.g. at $n = 100$ for a false positive rate (FPR) of 0, the Gibbs sampler found 21.9 ± 1.9 (mean ± standard deviation) true edges; REV 20.1 ± 1.4; and MC^3 21.7 ± 2.1. But at $n = 5000$, for the same FPR, Gibbs found 43.0 ± 0.0 true edges; REV 13.2 ± 21.3; and MC^3 did not find any true edges (the true graph has 46 edges). The constraint-based methods performed well for large sample sizes, as anticipated by the asymptotic consistency of the PC-algorithm (Kalisch and Bühlmann, 2007). The Xie-Geng method performed particularly well for $n = 5000$. Figure 8 shows AUROC as a function of iterations; the Gibbs sampler performed best after all run lengths and sample sizes considered. These results are supported by SHDs between point estimates and the true graph (Table A2).

5.5 Networks of Varying Sparseness

Sparsity can be used to statistical and computational advantage but in practice it may be hard to know what level of sparsity is reasonable for a given application. We therefore sought to investigate the effect of varying network sparsity, including scenarios where the data-generating graph can violate the in-degree constraint we impose. We simulated data following a procedure described in Kalisch and Bühlmann (2007) that we outline below. We first generated a DAG G via its adjacency matrix A^G, by drawing entries as $A^G_{uv} \sim \text{Bernoulli}(\rho)$, where $\rho \in (0, 1)$ is a parameter controlling sparsity. Entries were drawn independently for each $u < v$, with $A^G_{uv}=0$ otherwise. Larger ρ gives a denser graph and the expected number of neighbours (parents or immediate children) of a node is $\rho(p-1)$. We set $p = 25$, $n = 1000$ and considered 5 values of the sparseness parameter ρ (corresponding to expected neighbourhood sizes 2, 3, 4, 5 and 6). For each ρ we drew 10 DAGs, simulating a data set from each DAG by ancestral sampling using a Normal linear model (see Kalisch and Bühlmann, 2007, for full details).

Figure 9 shows boxplots over SHDs. We see that accuracy decreases with increasing density, echoing results in Kalisch and Bühlmann (2007) for the PC algorithm. As throughout, all the samplers had an in-degree constraint (maximum in-degree $\kappa = 3$), while the frequentist methods did not. Nevertheless, the performance of the Bayesian methods did not appear to deteriorate any more rapidly than the frequentist methods. At all ρ's, the median probability graph G^{med} outperformed the frequentist methods which in turn outperformed the MAP graph. In this context, we draw attention to a difference between the median probability and MAP graphs: the former, although obtained by averaging over DAGs satisfying the in-degree constraint, may itself have in-degree greater than κ, while the latter is necessarily subject to the constraint.

5.6 Survey Data

The publicly available Behavioral Risk Factor Surveillance System Survey (BRFSS) (Centers for Disease Control and Prevention, 2008) is a household-level random-digit telephone survey, collected by the U.S. National Center for Chronic Disease Prevention and Health, that has been conducted throughout the United States since 1984. We considered (discrete) responses to $p = 24$ questions (see Appendix D), spanning most of the topics covered in the survey. We considered the responses from New York in the 2008 survey, removing samples for respondents who refused or were unsure of their response, or whose response was missing, to any of the 24 questions. The resulting sample size was $n = 4, 197$. The median probability graph G^{med} estimated by the Gibbs sampler is shown in Figure A17a.

Figure 10 shows between-run agreement. The Gibbs runs showed better agreement than the REV runs and the MC^3 runs disagreed considerably (these pairs of runs were typical; all runs shown in Figure A18a). Indeed there were no major between-run discrepancies (in the sense of Section 5.2) for the Gibbs sampler, whereas there were on average 2.4 major discrepancies for REV and 10.9 for MC^3. The Gibbs sampler also had the highest proportion of edges with PSRF < 1.1 (Figure A14c).

The maximum log marginal likelihoods found by each of the three samplers differed considerably as did the number of iterations needed to reach a plateau (Figure A13b). The Gibbs sampler typically reached a plateau after around 500 samples (although in one run 10^3 samples were needed). REV took longer to reach a (usually lower) plateau. MC^3 appeared to become stuck in a region with even lower log marginal likelihood.

To investigate stability under resampling of the data we applied the methods to 10 bootstrap resamples of the data set. The Gibbs and REV samplers were more stable than the frequentist methods as well as than MC^3. For example, using G_τ^{PC} as a point estimate for for the Bayesian methods (i.e. thresholding to give the same number of edges as PC), the mean SHD between results from pairs of bootstrap data sets was 21.7 (Gibbs), 22.3 (REV), 33.4 (MC^3) and 30.8 (PC-algorithm). Results for G^{MAP} and G_τ^{Xie} are shown in Figure A19a.

5.7 Large-sample, Single-cell Molecular Data

We used single-cell molecular data from Bendall et al. (2011) with $p = 34$ continuous variables (see Appendix D) and $n = 21, 691$. The median probability graph G^{med} estimated by the Gibbs sampler is shown in Figure A17b.

Figure 11 shows between-run agreement for the samplers for a typical pair of runs; the Gibbs sampler shows better agreement than REV or MC^3. All but one of the 10 runs of the Gibbs sampler were consistent with each other (Figure A18b) and the one inconsistent run nonetheless showed better agreement with the other Gibbs runs than any pair of runs of the other samplers. The Gibbs sampler had no major discrepancies (as defined in Section 5.2) between any pairs of runs, while there were on average 26 and 87 major discrepancies for REV and MC^3 respectively. Smaller discrepancies followed a similar pattern: the average number of edges differing by 0.1 or more in posterior probability between runs were 6.4

(Gibbs sampler), 56.6 (REV), and 151.8 (MC^3). The Gibbs sampler also had the highest proportion of edges with PSRF < 1.1 after any run length (Figure A14d). Trace plots are shown in Figure A13c. The Gibbs sampler found a region of higher log marginal likelihood than the other samplers and did so consistently. Indeed, the maximum log marginal likelihood achieved across all REV runs was exceeded in every Gibbs run and after only around 5000 iterations. Finally, we considered stability under resampling of the data (Figure A19b), including also the frequentist point estimators for comparison. The mean SHD between PC estimates from pairs of bootstrap samples was 214.6; the corresponding mean SHDs for the samplers (using G_τ^{PC} point estimates) were 128.8 (Gibbs), 140.1 (REV) and 225.2 (MC^3). We note that the SHDs are particularly high here because the PC-algorithm estimates a network with a high density (recall that under G_τ^{PC} all methods choose the same number of edges as PC).

5.8 A Pathological 4-node Example

Our final example demonstrates the potential sensitivity of the Gibbs sampler to choice of $q = |W|$ (the number of nodes whose parent sets are sampled together in a single iteration). The example was constructed to highlight a situation in which the sampler can become stuck in a local mode unless q is large enough, potentially leading to extremely slow convergence.

We used as data 10^5 simulated samples from a 4-node network in which the parents of node 2 were nodes 1 and 4; the parents of node 3 were nodes 1 and 2; and nodes 1 and 4 had no parents. Each node was Bernoulli distributed. The probability of success was 0.6 for nodes 1 and 4, while nodes 2 and 3 were noisy XORs with probability of success 0.9 when either parent (but not both) was ‘true’ and 0.1 otherwise.

For the purposes of demonstration, we set $q = 2$. As shown in Figure 12, after 10^6 iterations the maximum error in edge probability for the Gibbs sampler was 0.016 ± 0.015 (mean $\pm$ standard deviation across 10 runs). Given that there are only 543 DAGs with $p = 4$ nodes, this magnitude of error is unexpectedly large. The slow convergence is due to the concentration of posterior mass on two graphs that the sampler cannot easily move between. The MAP graph (posterior probability 0.61) is the graph in which the parents of node 2 are nodes 1 and 3; the parents of node 4 are nodes 1 and 2; and nodes 1 and 3 have no parents. The data-generating graph has posterior probability 0.38. The graphs differ in the parents of 3 nodes, and so with $q = 2$ the sampler cannot move between them in a single step. At the same time, the large sample size leads to all other graphs having low posterior probability (< 0.002), making multi-step transitions between the two graphs unlikely. Thus, the sampler becomes stuck on one of the two graphs. In this example, setting $q = 3$ improved convergence (maximum error 0.0011 ± 0.0009 after 10^6 iterations), but in real-world examples it is difficult to rule out the possibility that analogous issues may arise.

6 Discussion

We introduced a Gibbs sampler for structure learning of DAGs that can converge more easily than existing samplers due to its ability to efficiently make large moves in DAG space.

We showed that it provides often substantial gains in accuracy and stability in comparison with existing (Bayesian and frequentist) methods in a range of settings.

The formulation of the sampler develops and exploits the connection between variable selection and structure learning. This connection has been widely studied for undirected graphs (e.g. Meinshausen and Bühlmann, 2006), but for DAGs is complicated by the acyclicity requirement. Our approach accounts for acyclicity but further work in this area may ease the adaptation of results and methods from Bayesian variable selection to the case of DAGs. In the proposed sampler, existing variable selection methods could be of direct utility in sampling from the conditional distribution $P(\mathrm{pa}_W|\mathrm{pa}_{-W}^G, \mathbf{X})$. When this sampling step is difficult, a Metropolis-within-Gibbs approach (i.e. substituting a Metropolis step in place of the Gibbs step) could be considered. With $q = |W| = 1$, the conditional $P(\mathrm{pa}_W|\mathrm{pa}_{-W}^G, \mathbf{X})$ is identical to the posterior of the corresponding variable selection problem. Then, a Metropolis-within-Gibbs move could directly exploit existing variable selection methods.

In common with most of the structure learning literature, we used an in-degree constraint. This gives a smaller DAG space and in addition controls the number of parameters needed to specify conditional distributions. However, it is a strong constraint and a natural concern is whether it excludes higher in-degree models that could be appropriate for the data. This possibility cannot be ruled out, but the use of model averaging and marginal posterior summaries may ameliorate the concern to some extent, because even when the true number of parents of a node exceeds the restriction, important candidate parents may still have high edge scores (and appear in a summary graph). For example, suppose the true in-degree of a node is 4, but at most 3 incoming edges are permitted. In this case, although no model including all 4 parents can be considered, provided the signal can be detected considering only 3 nodes at a time, the posterior probability of all 4 edges may nonetheless be high.

In our empirical examples the REV sampler of Grzegorczyk and Husmeier (2008) showed impressive gains over MC^3. The Gibbs sampler generally seemed to outperform both. The use of conditional distributions in REV is a point of similarity with the Gibbs sampler proposed here. The performance gains of Gibbs could be explained by two key differences from REV. First, REV does not use the natural conditional distribution and requires an accept-reject step. Second, REV must include at least some MC^3 proposals (otherwise the sampler is not irreducible), and these steps are not tailored to the shape of the posterior distribution (Grzegorczyk and Husmeier, 2008, make REV proposals with probability 1/15 and so most steps are based on MC^3 proposals).

If there is strong correlation between the parent sets of more than $q = |W|$ nodes, the Gibbs sampler may not mix well. In this situation, constraint-based methods may be useful. Alternatively, q could be chosen at each step according to some distribution, so that a mixture of different block sizes is used. This would in particular allow larger blocks to be used without increasing the computational demands of the algorithm excessively. In this paper we fixed $q = 3$, and found this simple choice gave a well-behaved and effective

sampler. But there is a trade-off: increasing q increases the time taken to evaluate $P(\mathrm{pa}_W|\mathrm{pa}^G_{-W}, \mathbf{X})$, but also increases move size, with the potential to improve convergence.

Practical use of the Gibbs sampler in the form described here requires exact sampling from the conditional $P(\mathrm{pa}_W|\mathrm{pa}^G_{-W}, \mathbf{X})$ and there are situations related to this requirement in which other methods may be more suitable. First, when an appropriate maximum in-degree cannot be set, MC^3 or a variant could be more appropriate (although convergence could be very slow). Alternatively, search procedures such as GES (Chickering, 2002b) or HCMC (Castelo and Kočka, 2003) could be used. Second, exact sampling from the conditional distribution is challenging in settings with thousands of nodes. In this case, efficient constraint-based methods (such as the PC-algorithm) may be a better choice, particularly in the large sample setting. As noted by a referee, an interesting area for future research would be combining the Gibbs sampler with some aspects of other methods—such as PC—that are relatively well suited to the truly high-dimensional setting.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

The authors are grateful to Karen Sachs for providing the single-cell molecular data; to Marco Grzegorczyk and Dirk Husmeier for their inspiring work in this area and for providing their implementation of the REV sampler; to the referees for numerous suggestions that improved the article; and to Frances Griffiths, David Lunn and Paul Newcombe for helpful discussions. Part of this work was carried out while the authors were at the University of Warwick—whose excellent scientific environment the authors warmly acknowledge—and supported by the Economic and Social Research Council (ESRC) and Engineering and Physical Sciences Research Council (EPSRC).

References

Barbieri MM, Berger JO. Optimal predictive model selection. Annals of Statistics. 2004; 32(3):870–897.

Beinlich, IA., Suermondt, HJ., Chavez, RM., Cooper, GF. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. Second European Conference on Artificial Intelligence in Medicine. Springer-Verlag; Berlin: 1989. p. 247-256.

Bendall SC, Simonds EF, Qiu P, Amir ED, Krutzik PO, Finck R, Bruggner RV, Melamed R, Trejo A, Ornatsky OI, Balderas RS, et al. Single-cell mass cytometry of differential immune and drug responses across a human hematopoietic continuum. Science. 2011; 332(6030):687–696. [PubMed: 21551058]

Besag JE. Spatial interaction and the statistical analysis of lattice systems. Journal of the Royal Statistical Society: Series B (Methodological). 1974; 36(2):192–236.

Besag JE. Discussion of “Markov chains for exploring posterior distributions”. Annals of Statistics. 1994; 22(4):1734–1741.

Carr DB, Littlefield RJ, Nicholson WL, Littlefield JS. Scatterplot matrix techniques for large N. Journal of the American Statistical Association. 1987; 82(398):424–436.

Castelo R, Kočka T. On inclusion-driven learning of Bayesian networks. Journal of Machine Learning Research. 2003; 4:527–574.

Centers for Disease Control and Prevention. Behavioral Risk Factor Surveillance System Survey Data. U.S. Department of Health and Human Services; Atlanta, Georgia: 2008.

Chickering DM. Learning equivalence classes of Bayesian-network structures. Journal of Machine Learning Research. 2002a; 2:445–498.

Chickering DM. Optimal structure identification with greedy search. Journal of Machine Learning Research. 2002b; 3:507–554.

Colombo D, Maathuis MH. Order-independent constraint-based causal structure learning. Journal of Machine Learning Research. 2014; 15:3921–3962.

Coppersmith D, Winograd S. Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation. 1990; 9(3):251–280.

Demetrescu C, Italiano GF. Trade-offs for fully dynamic transitive closure on DAGs: breaking through the $O(n^2)$ barrier. Journal of the ACM. 2005; 52(2):147–156.

Demetrescu, C., Eppstein, D., Galil, Z., Italiano, GF. Dynamic graph algorithms. Algorithms and Theory of Computation Handbook: General Concepts and Techniques. Atallah, MJ., Blanton, M., editors. Chapman and Hall/CRC; Boca Raton, FL: 2010. p. 9.1-9.28.

Eaton, D., Murphy, K. Bayesian structure learning using dynamic programming and MCMC. Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence; Corvallis, Oregon: AUAI Press; 2007. p. 101-108.

Ellis B, Wong WH. Learning causal Bayesian network structures from experimental data. Journal of the American Statistical Association. 2008; 103(482):778–789.

Friedman N, Koller D. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. Machine Learning. 2003; 50(1–2):95–125.

Geiger, D., Heckerman, D. Learning Gaussian networks. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence; San Francisco, CA: Morgan Kaufmann; 1994. p. 235-240.

Gelman A, Rubin DB. Inference from iterative simulation using multiple sequences. Statistical Science. 1992; 7(4):457–472.

Giudici P, Castelo R. Improving Markov chain Monte Carlo model search for data mining. Machine Learning. 2003; 50(1–2):127–158.

Grzegorczyk M, Husmeier D. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. Machine Learning. 2008; 71(2–3):265–305.

Heckerman D, Geiger D, Chickering DM. Learning Bayesian networks: the combination of knowledge and statistical data. Machine Learning. 1995; 20(3):197–243.

Hobert JP, Robert CP, Goutis C. Connectedness conditions for the convergence of the Gibbs sampler. Statistics & Probability Letters. 1997; 33(3):235–240.

Kalisch M, Bühlmann P. Estimating high-dimensional directed acyclic graphs with the PC-Algorithm. Journal of Machine Learning Research. 2007; 8:613–636.

King V, Sagert G. A fully dynamic algorithm for maintaining the transitive closure. Journal of Computer and System Sciences. 2002; 65(1):150–167.

Koivisto M, Sood K. Exact Bayesian structure discovery in Bayesian networks. Journal of Machine Learning Research. 2004; 5:549–573.

Koller, D., Friedman, N. Probabilistic Graphical Models: Principles and Techniques. MIT Press; Cambridge, MA: 2009.

Korb, KB., Nicholson, AE. Bayesian Artificial Intelligence. Chapman & Hall/CRC Press; Boca Raton, FL: 2011.

Madigan D, York JC. Bayesian graphical models for discrete data. International Statistical Review / Revue Internationale de Statistique. 1995; 63(2):215–232.

Meinshausen N, Bühlmann P. High-dimensional graphs and variable selection with the Lasso. Annals of Statistics. 2006; 34(3):1436–1462.

Munro I. Efficient determination of the transitive closure of a directed graph. Information Processing Letters. 1971; 1(2):56–58.

Newman, D., Hettich, S., Blake, C., Merz, C. UCI Repository of Machine Learning Databases. University of California, Department of Information and Computer Science; Irvine, CA: 1998. URL http://www.ics.uci.edu/~mlearn/MLRepository.html

Nikolova O, Zola J, Aluru S. Parallel globally optimal structure learning of Bayesian networks. Journal of Parallel and Distributed Computing. 2013; 73(8):1039–1048.

Parviainen, P., Koivisto, M. Exact structure discovery in Bayesian networks with less space. Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence; Corvallis, Oregon: AUAI Press; 2009. p. 436-443.

Parviainen P, Koivisto M. Finding optimal Bayesian networks using precedence constraints. Journal of Machine Learning Research. 2013; 14:1387–1415.

Roberts GO, Sahu SK. Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler. Journal of the Royal Statistical Society: Series B (Methodological). 1997; 59(2): 291–317.

Spirtes, P., Glymour, C., Scheines, R. Causation, Prediction, and Search. The MIT Press; Cambridge, MA: 2000.

Tamada Y, Imoto S, Miyano S. Parallel algorithm for learning optimal Bayesian network structure. Journal of Machine Learning Research. 2011; 12:2437–2459.

Tian, J., He, R. Computing posterior probabilities of structural features in Bayesian networks. Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence; Corvallis, Oregon: AUAI Press; 2009. p. 538-547.

Xie X, Geng Z. A recursive method for structural learning of directed acyclic graphs. Journal of Machine Learning Research. 2008; 9:459–483.

Zellner, A. On assessing prior distributions and Bayesian regression analysis with g-prior distributions. Bayesian Inference and Decision Techniques: Essays in Honour of Bruno de Finetti. Goel, PK., Zellner, A., editors. North-Holland; Amsterdam: 1986. p. 233-243.

Europe PMC Funders Author Manuscripts

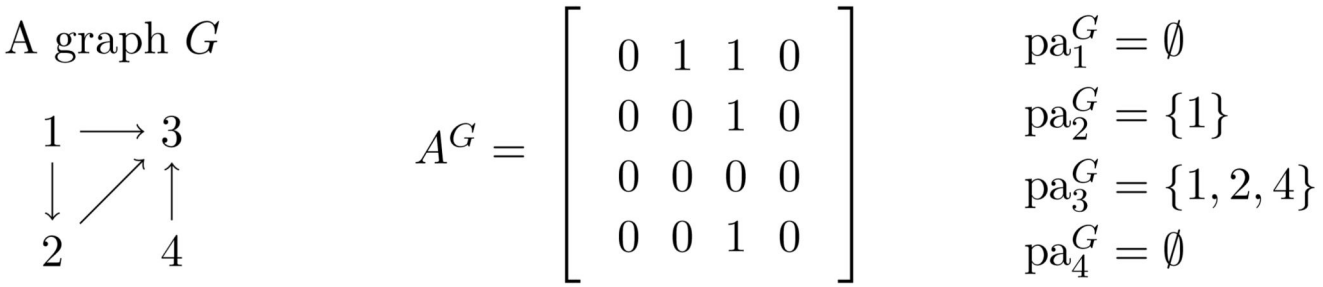

Figure 1.

Illustration of notation. A graph G, with vertex set $V = \{1, 2, 3, 4\}$, edge set $E = \{(1, 2), (1, 3), (2, 3), (4, 3)\}$ and adjacency matrix A^G as shown, can be specified using parent sets as $G = \langle \mathrm{pa}_1 = \emptyset, \{1\}, \mathrm{pa}_2 = \{1\}, \mathrm{pa}_3 = \{1,2,4\}, \mathrm{pa}_4 = \emptyset \rangle$ or abbreviated to $G = \langle \emptyset, \{1\}, \{1, 2, 4\}, \emptyset \rangle$ (see text for description of notation). For the vertex subset $Z = \{2, 4\}$, $\mathrm{pa}_Z^G = (\{1\}, \emptyset)$ and $\mathrm{pa}_{-Z}^G = (\emptyset, \{1, 2, 4\})$ and thus the graph can be specifed as $G = \left\langle \mathrm{pa}_Z = \mathrm{pa}_Z^G, \mathrm{pa}_{-Z} = \mathrm{pa}_{-Z}^G \right\rangle$. We abbreviate this as $G = \left\langle \mathrm{pa}_Z^G, \mathrm{pa}_{-Z}^G \right\rangle$.

Europe PMC Funders Author Manuscripts

Figure 2.
Illustrative scenario in which small local moves limit transitions between two regions of high probability. If both graphs (a) and (b) have high probability, the near-cyclic nature of the graphs makes transitions between (a) and (b) difficult.

Europe PMC Funders Author Manuscripts

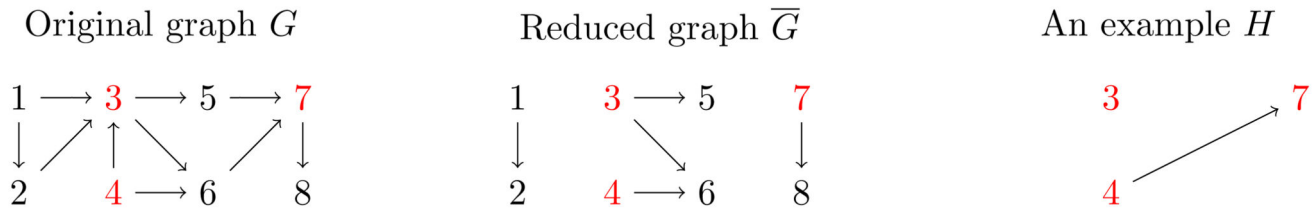

Figure 3.

An illustrative example of the relevant graphs and sets, with $W = \{3, 4, 7\}$ shown in red. The edges into W are removed from the original graph G, to form $\overline{G}$. Here $\mathscr{H}$ is the set of all DAGs on the nodes $\{3, 4, 7\}$ and thus $\eta = |\mathscr{H}| = 25$. For H as shown, $_{\mathrm{nd}}{}^{\overline{G}} = \{1, 2\}$.

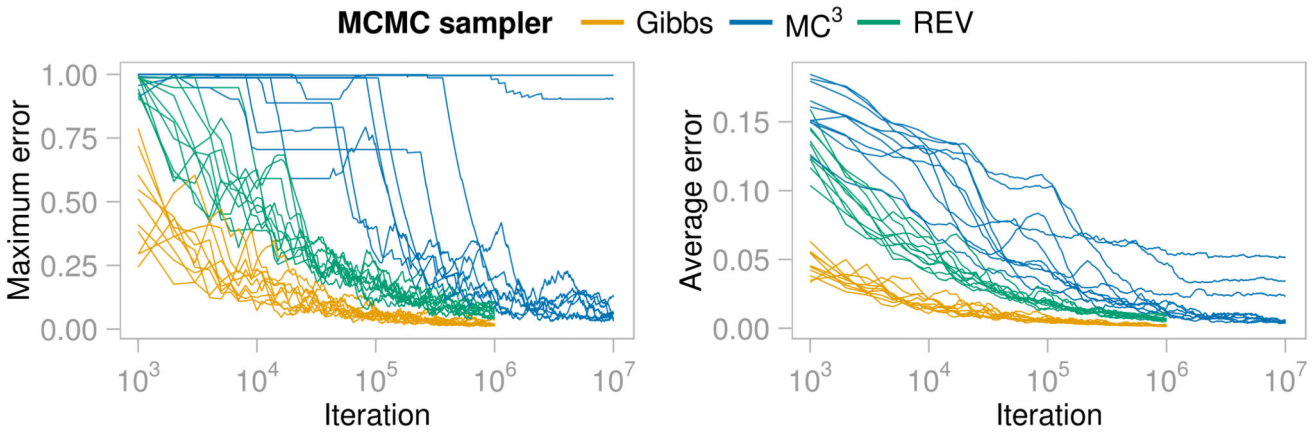

Figure 4.

For the Zoo data set, maximum and average (across all possible 272 edges) error in posterior edge probabilities versus iteration number (on a $\log_{10}$ scale). For each MCMC algorithm, 10 independent runs, initialised at disparate initial values, are shown.

Europe PMC Funders Author Manuscripts

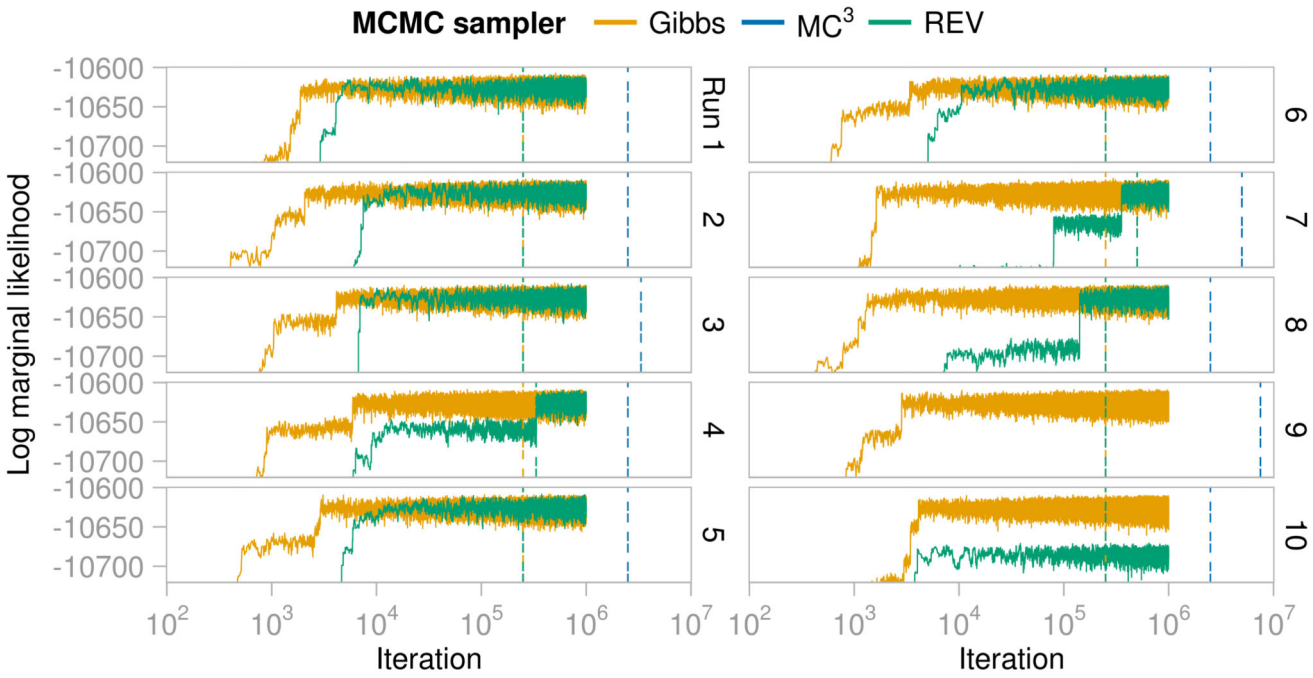

Figure 5.

Log marginal likelihood of the graphs visited by the three MCMC samplers in 10 independent runs, initialised at disparate initial conditions, on the ALARM data, with data sample size $n = 1000$. Iteration number is displayed on a $\log_{10}$ scale. The dashed lines indicate where the burn-in phase ended. In all cases the log marginal likelihood of the graphs reached by the MC^3 sampler are below the displayed range.

Europe PMC Funders Author Manuscripts

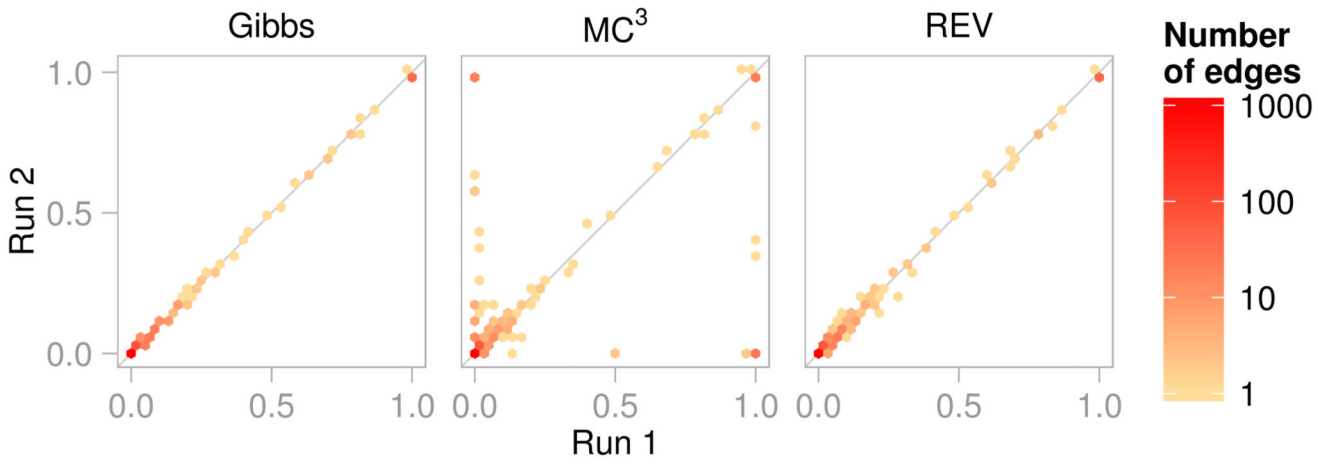

Figure 6.

Convergence diagnostic plots for each MCMC sampler for the ALARM data, with $n = 1000$. The posterior edge probabilities given by two independent MCMC runs are plotted against each other, binned into hexagonal areas to avoid over-plotting. When the edge probabilities of the two runs agree, all of the points in the plot will lie close to the $y = x$ line; strongly off-diagonal points indicate extreme discrepancies between the runs.

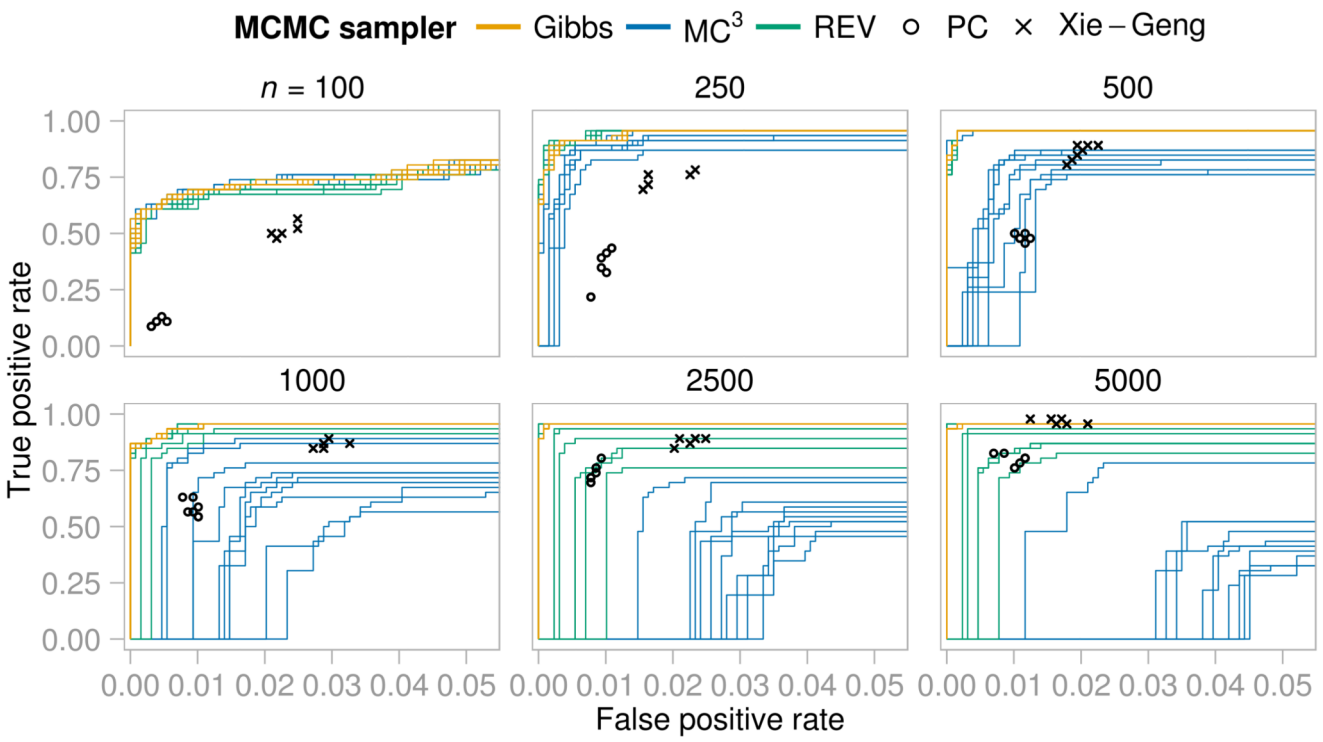

Figure 7.

ALARM data, receiver operating characteristic (ROC) curves for each of 10 independent MCMC runs for each MCMC sampler, and point estimates for the constraint-based methods. Note that the horizontal axis shows only false positives rates < 0.05, corresponding to the case in which interest focuses on high-ranked edges (the complete curves are shown in Figures A16). Point estimates from Xie-Geng's constraint-based method and the PC-algorithm are also shown for all 8 significance levels considered.

Europe PMC Funders Author Manuscripts

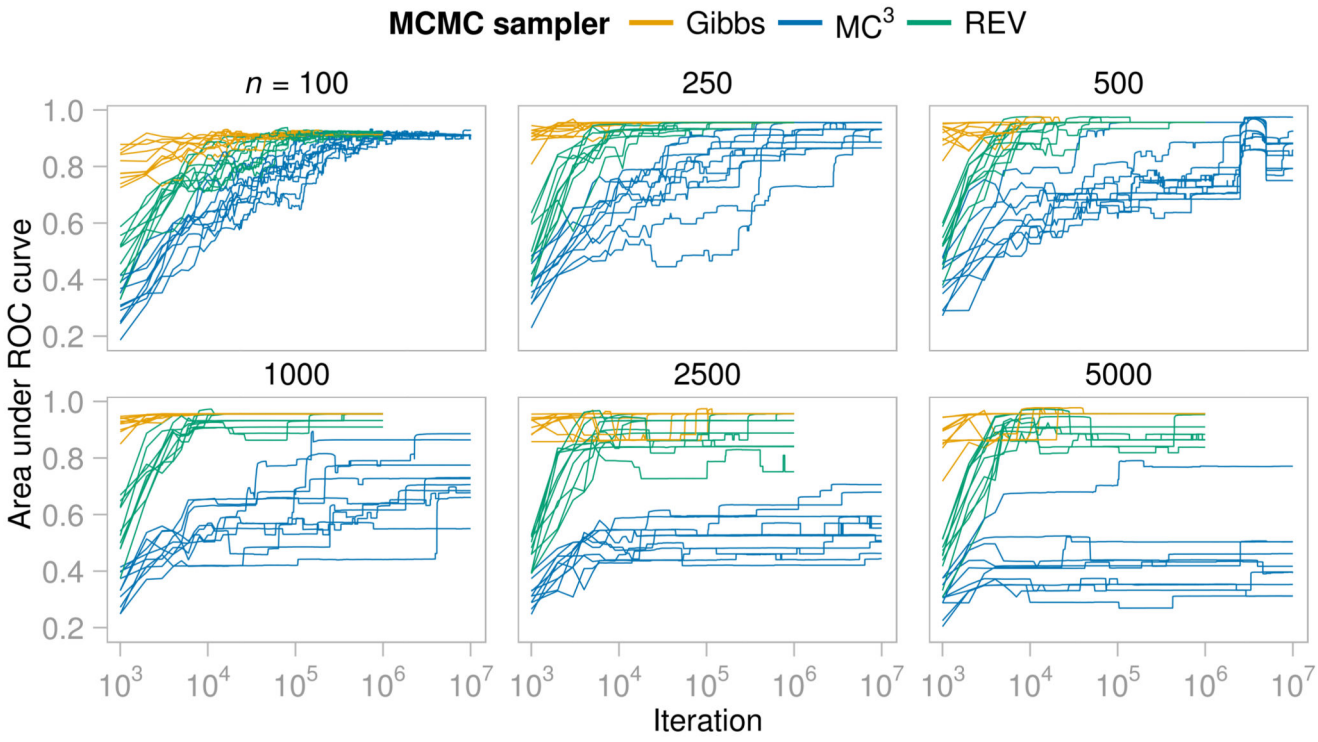

Figure 8.

ALARM data, area under the receiver operating characteristic curve (AUROC) against iteration number ($\log_{10}$ scale) for each of the 10 independent runs for each MCMC algorithm. Each panel shows results for a particular sample size $n = 100, \ldots, 5000$.

Europe PMC Funders Author Manuscripts

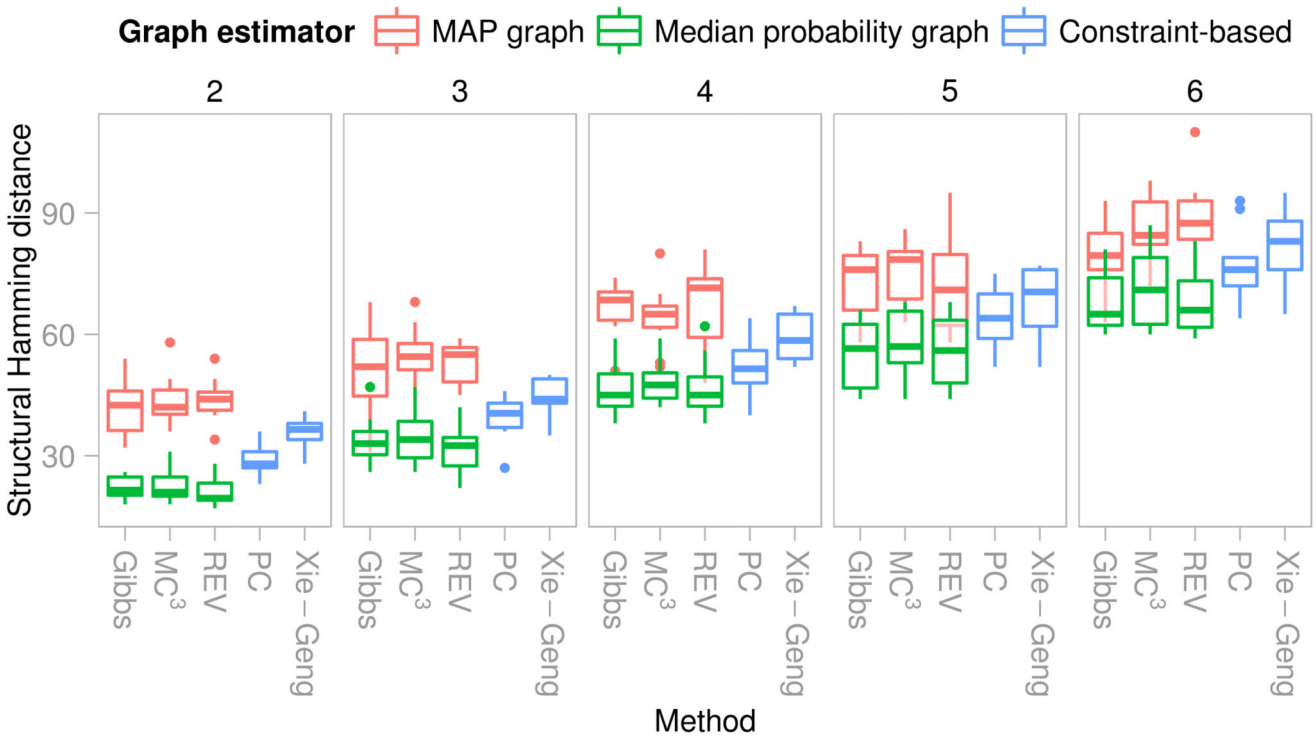

Figure 9.

Synthetic data, accuracy of estimation for networks with different levels of sparsity. The panels correspond to simulations in which the expected number of neighbours for each node in the data-generating graph is respectively 2, 3, 4, 5, and 6. Accuracy is quantified by structural Hamming distance (SHD) between estimates and data generating graphs (smaller SHDs correspond to more accurate estimates). Box plots are over the 10 independent networks/data sets simulated for each sparsity level. For MCMC methods, results from the MAP graph G^{MAP} and median probability model G^{med} are shown.

Europe PMC Funders Author Manuscripts

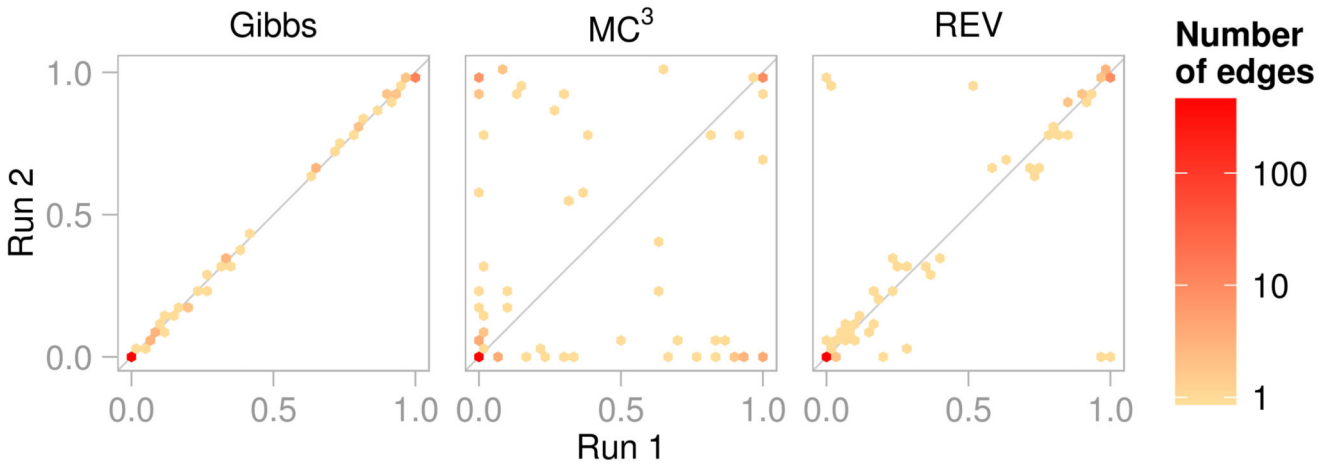

Figure 10.

Convergence diagnostic plots for each MCMC sampler for the survey data. The posterior edge probabilities given by two independent MCMC runs are plotted against each other, binned into hexagonal areas to avoid over-plotting. When the two runs give the same estimates of the posterior edge probabilities all of the points appear on the line $y = x$; strongly off-diagonal points indicate extreme discrepancies between the runs. This pair of runs was typical of all pairs.

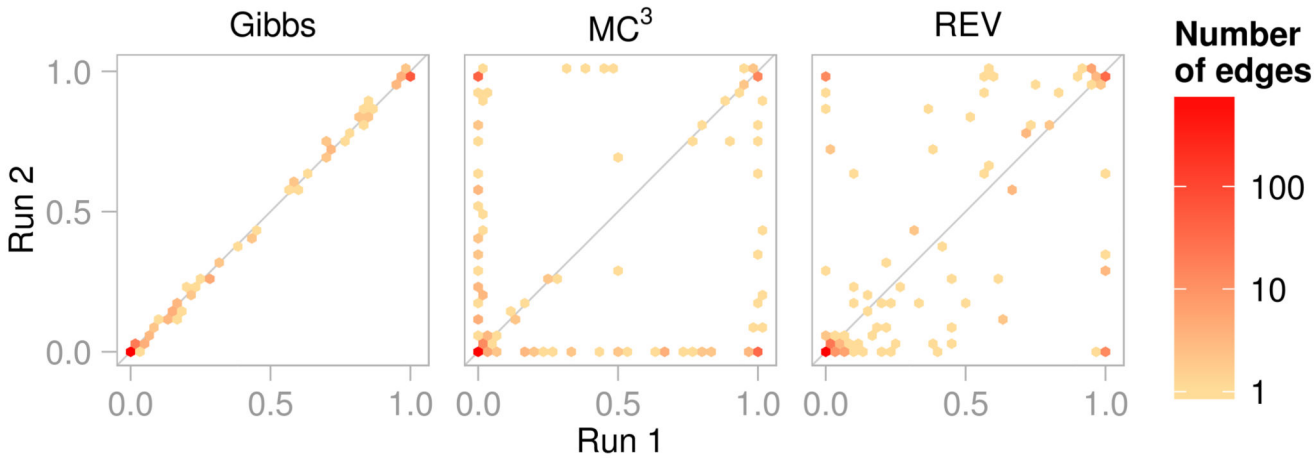

Figure 11.

Convergence diagnostic plots for each MCMC sampler for the single-cell molecular data. The posterior edge probabilities given by two independent MCMC runs are plotted against each other, binned into hexagonal areas to avoid overplotting. When the two runs give the same estimates of the posterior edge probabilities all of the points appear on the line $y = x$; strongly off-diagonal points indicate extreme discrepancies between the runs. This pair of runs was typical of all pairs of runs.

Europe PMC Funders Author Manuscripts

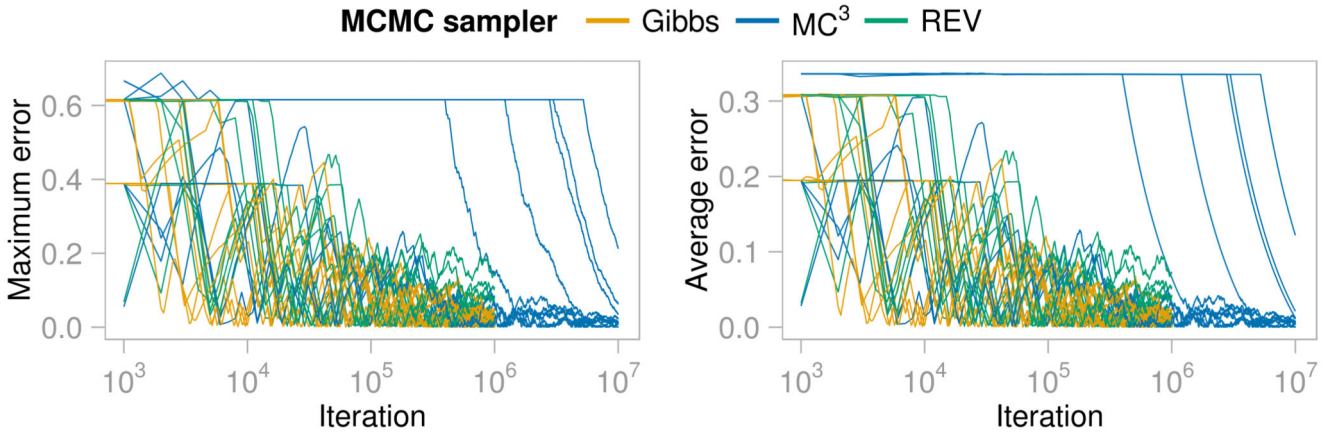

Figure 12.

For the pathological 4-node example, maximum and average (across all possible 12 edges) error in posterior edge probabilities by iteration number (on a $\log_{10}$ scale). For each MCMC algorithm, 10 independent runs, initialised at disparate initial values, are shown.

Europe PMC Funders Author Manuscripts

Table 1

The relevant sets and requirements of conditions (A) and (B) for the illustrative graph G shown in Figure 3, with H as shown. Recall $\mathrm{nd}^{\overline{G}} = \{1, 2\}$. Condition (B) depends on $R_{7,4}^{\overline{G},H} = \{4\}$, but it imposes no restriction when $w \in \{3, 4\}$ because $\mathrm{pa}_3^H = \mathrm{pa}_4^H = \varnothing$. We find that $Pa_W^{G,H}$ contains all tuples of parent sets for which $\mathrm{pa}_3^{G'} \subseteq \{1, 2\}$, $\mathrm{pa}_4^{G'} \subseteq \{1, 2\}$ and $\mathrm{pa}_7^{G'} = \{4\} \cap Z$ where $Z \subseteq \{1, 2\}$.

	$\mathrm{de}_w^{\overline{G}}$	$\mathrm{nd}_w^{\overline{G}}$	$\mathrm{de}_w^{\overline{G},H}$	$\mathrm{de}_{-w}^{\overline{G},H}$	$Q_w^{\overline{G},H}$	**Condition (B)**
$w = 3$	{3, 5, 6}	{1, 2, 4, 7, 8}	∅	{3, 4, 5, 6, 7, 8}	{1, 2}	No restriction
$w = 4$	{4, 6}	{1, 2, 3, 5, 7, 8}	∅	{3, 4, 5, 6, 7, 8}	{1, 2}	No restriction
$w = 7$	{7, 8}	{1, 2, 3, 4, 5, 6}	{4, 6}	{3, 5, 6, 7, 8}	{1, 2, 4}	$\mathrm{pa}_7^{G'} \cap \{4\} \neq \varnothing$