Structure Learning with Adaptive Random Neighborhood Informed MCMC (Supplementary Material)

Anonymous Author(s)
Affiliation
Address
email

A Derivation of the posterior model probability of DAG model

In this section, we will introduce the companion form for the prior on W_γ and derive the marginal likelihood of γ starting from a spike-and-slab prior, as described in Section 2. In addition, under the *Markov property*, we show that the posterior distribution $\pi(\cdot)$ can be factorised into a product form [Koller and Friedman, 2009].

Let γ_j be the j-th column of of γ, γ_{ij} is the i-th component of γ_j (equivalent to the i, j-th entry of indicator matrix γ). Let $\gamma_{-i,j}$ be γ_j but excluding the i-th component γ_{ij}. Let X_{γ_j} denote the "active" variables in γ_j (those variables such that γ_{ij} = 1). The variables X_{γ_j} are the parents of X_j implied by γ. Let $W_{\gamma,j}$ be the coefficients of j-th column in W_γ selected by γ_j. In addition, let $\boldsymbol{X}_i = \{X_{j,i}\}_{j=1}^n$ be a vector of nodes for a fixed i-th observation, with X_j being the j-th node, while let $x_j = \{X_{j,i}\}_{i=1}^N$ be the vector of the N observations corresponding to the j-th node. The matrix x_{γ_j} is the collection of observations on the set of variables defined by γ_j.

Under γ, the parent set of X_j (Pa(X_j)) is equivalent to set of nodes collected in γ_j (i.e., X_{γ_j}). The joint prior distribution on W_γ then is

$$p(W_\gamma|\gamma, \{\sigma_j^2\}) = \prod_{j=1}^n p(W_{\gamma,j}|\gamma_j, \sigma_j^2) \tag{S.1}$$

where

$$W_{\gamma,j}|\ \gamma_j, \sigma_j^2 \sim N_{q_j}(0, g\sigma_j^2 I_{q_j}) \tag{S.2}$$

and q_j is the the parent size of X_j.

Remark A.1. *The identity matrix I_{q_j} in the prior (S.2) can be replaced by the inverse of the Gram matrix defined as $(x_{\gamma_j}^T x_{\gamma_j})^{-1}$. The resulting prior is then an analog, in the context of DAG sampling, to g-prior in the Bayesian variable selection context.*

Note that in a DAG model, each node X_j is independent of its non-descendants given its parents X_{γ_j}. This property implies a factorisation of the joint likelihood over data $\mathcal{D}$ given by

$$p(\mathcal{D}|W_\gamma, \{\sigma_j^2\}, \gamma) = \prod_{j=1}^n p(X_j|X_{\gamma_j}, W_{\gamma,j}, \sigma_j^2, \gamma_j),$$

which translates into

$$p(\mathcal{D}|W_\gamma, \{\sigma_j^2\}, \gamma) = \prod_{j=1}^n N(x_j|x_{\gamma_j}^T W_{\gamma,j}, \sigma_j^2 I_N)$$

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

in the context of Gaussian graphical models. Due to the factorisation above, it is possible to exchange the integral of W_γ and $\{\sigma_j^2\}$ with the product in j, and integrate out $W_{\gamma,j}$ and σ_j^2 from the joint probabilities

$$p(X_j, W_{\gamma,j}, \sigma_j^2 | X_{\gamma_j}, \gamma_j) = p(X_j | X_{\gamma_j}, W_{\gamma,j}, \sigma_j^2, \gamma_j) p(W_{\gamma,j} | \sigma_j^2, \gamma_j) p(\sigma_j^2) \tag{S.3}$$

for each individual node X_j. In what follows, we will show how to derive $p(X_j | X_{\gamma_j}, \gamma_j)$ from the above, by solving these integrals.

We start from the integration of the coefficients $W_{\gamma,j}$. We can marginalise out $W_{\gamma,j}$ simply due to conjugacy as below

$$\begin{aligned}
p(X_j | X_{\gamma_j}, \sigma_j^2, \gamma_j) &= \int p(X_j | X_{\gamma_j}, W_{\gamma,j}, \sigma_j^2, \gamma_j) p(W_{\gamma,j} | \sigma_j^2, \gamma_j) dW_{\gamma,j} \\
&= \int N(x_j | x_{\gamma_j}^T W_{\gamma,j}, \sigma_j^2 I_n) N(W_{\gamma,j} | 0, g\sigma_j^2 I_{q_j}) dW_{\gamma,j} \\
&= (2\pi\sigma_j^2)^{-\frac{N}{2}} g^{-\frac{q_j}{2}} \det(\Sigma_j)^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2\sigma_j^2} (x_j^T x_j - x_j^T x_{\gamma_j} \Sigma_j^{-1} x_{\gamma_j}^T x_j) \right\}
\end{aligned}$$

where $\Sigma_j = x_{\gamma_j}^T x_{\gamma_j} + \frac{1}{g} I_{q_j}$. Recall that the prior for σ_j^2 is specified by $p(\sigma_j^2) \propto \sigma_j^{-2}$, and thus we can marginalise out σ_j^2 in a similar way

$$\begin{aligned}
p(X_j | X_{\gamma_j}, \gamma_j) &= \int p(X_j | X_{\gamma_j}, \sigma_j^2, \gamma_j) p(\sigma_j^2) d\sigma_j^2 \\
&= \int_{\mathbb{R}^+} (2\pi\sigma_j^2)^{-\frac{N}{2}} g^{-\frac{q_j}{2}} \det(\Sigma_j)^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2\sigma_j^2} (x_j^T x_j - x_j^T x_{\gamma_j} \Sigma_j^{-1} x_{\gamma_j}^T x_j) \right\} \cdot \sigma_j^{-2} d\sigma_j^2 \\
&= (2\pi)^{-\frac{N}{2}} g^{-\frac{q_j}{2}} \det(\Sigma_j)^{-\frac{1}{2}} \Gamma\left(\frac{N}{2}\right) \left[\frac{1}{2} (x_j^T x_j - x_j^T x_{\gamma_j} \Sigma_j^{-1} x_{\gamma_j}^T x_j) \right]^{-\frac{N}{2}}
\end{aligned} \tag{S.4}$$

where $\Sigma_j = x_{\gamma_j}^T x_{\gamma_j} + \frac{1}{g} I_{p_j}$ and $\Gamma(\cdot)$ is the Gamma function. Eventually, putting all the steps above together, we can write down the model's marginal likelihood, conditional on γ only, as follows

$$\begin{aligned}
p(\mathcal{D} | \gamma) &= \int \int p(\mathcal{D} | W_\gamma, \{\sigma_j^2\}, \gamma) dW_\gamma d\{\sigma_j^2\} \\
&= \prod_{j=1}^n \int \int p(X_j | X_{\gamma_j}, \beta_{j,\gamma_j}, \sigma_j^2) dW_{\gamma,j} d\sigma_j^2 \\
&= \prod_{j=1}^n p(X_j | X_{\gamma_j}, \gamma_j) \\
&\propto \prod_{j=1}^n g^{-\frac{q_j}{2}} \det(\Sigma_j)^{-\frac{1}{2}} \left[\frac{1}{2} (x_j^T x_j - x_j^T x_{\gamma_j} \Sigma_j^{-1} x_{\gamma_j}^T x_j) \right]^{-\frac{N}{2}}
\end{aligned} \tag{S.5}$$

where $\Sigma_j = x_{\gamma_j}^T x_{\gamma_j} + \frac{1}{g} I_{p_j}$. From the above derivation, we can also conclude the following property for the posterior distribution $\pi(\cdot)$:

Property 1 (Markov property.)**.** *The posterior distribution of indicator variable can be factorised as the following*

$$\pi(\gamma) \propto \prod_{j=1}^n p(\gamma_j | \mathcal{D}) \times \mathbb{I}\{\mathcal{G}_\gamma \textit{ is a DAG}\} \tag{S.6}$$

where $p(\gamma_j | \mathcal{D}) = p(X_j | X_{\gamma_j}, \gamma_j) p^u(\gamma_j)$ and $p^u(\cdot)$ is the unconstrained prior

$$p^u(\gamma_j) = \left(\frac{h}{1-h} \right)^{d_{\gamma_j}}. \tag{S.7}$$

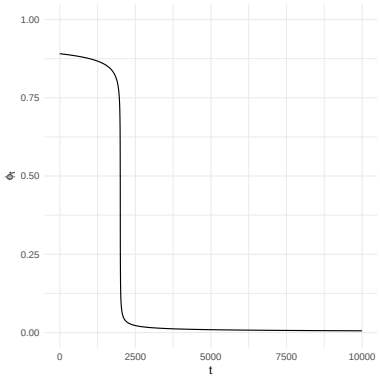


Figure S.1: The plot of diminishing sequence ϕ_t for 10,000 iterations with burn-in period of 2,000 iterations.

B The adaptive ϕ_t parameter

The decaying parameter ϕ_t is used for the adaptation of the algorithmic tuning parameter η as given in (9), and controls the trade-off between the pre-processing approximation $\tilde{\pi}$ and the ergodic average $\hat{\pi}^{(t)}$. Intuitively, one should rely more on the approximation $\tilde{\pi}$ during the burn-in period, since the ergodic average $\hat{\pi}^{(t)}$ has not fully converged. After the burn-in period though, the proposal should rely more heavily on the ergodic average. Based on the arguments, we use the following sequence of ϕ_t to adapt η:

$$\phi_t = \begin{cases} 1 - \frac{1}{2}\left(\frac{1}{N_b - t + 1}\right)^{0.2} & \text{if } t \leq N_b \\ \frac{1}{2}\left(\frac{1}{t - N_b}\right)^{0.5} & \text{if } t > N_b. \end{cases} \tag{S.8}$$

In particular, we have $\phi_t = 1/2$ when $t = N_b$. An example plot of decaying ϕ_t is given in Figure S.1, where we considered 10,000 iterations, with the first 2,000 iterations discarded as burn-in.

In addition, in the following lemma we derive the decaying rate of ϕ_t, and thus of $\eta^{(t)}$ as well:

Lemma S1. *After the burn-in period, the diminishing rate of ϕ_t defined by (S.8) is $\mathcal{O}(t^{-1})$. And thus the diminishing rate of tuning parameter $\hat{\eta}^{(t)}$ in (9) is $\mathcal{O}(t^{-1})$.*

Proof. Suppose $t > N_b$, we start from the definition of η_t and consider the diminishing rate

$$\begin{aligned} |\phi_{t+1} - \phi_t| &= \left| \frac{1}{2}\left(\frac{1}{t + 1 - N_b}\right)^{0.5} - \frac{1}{2}\left(\frac{1}{t - N_b}\right)^{0.5} \right| \\ &\leq \frac{1}{2}\left(\frac{(t + 1 - N_b)^{0.5} - (t - N_b)^{0.5}}{(t + 1 - N_b)^{0.5}(t - N_b)^{0.5}}\right) \\ &\leq \frac{1}{2}\frac{1}{t - N_b} \\ &= \mathcal{O}(t^{-1}) \end{aligned}$$

as required.

We can also show that the diminishing rate of the ergodic average $\hat{\pi}_{ij}^{t}$ has the same decaying rate

$$\begin{aligned} |\hat{\pi}_{ij}^{(t+1)} - \hat{\pi}_{ij}^{(t)}| &= \left| \frac{t}{t+1}\hat{\pi}_{ij}^{(t)} + \frac{1}{t+1}\gamma_{ij}^{(t+1)} - \hat{\pi}_{ij}^{(t)} \right| \\ &\leq \frac{1}{t+1}\hat{\eta}_{ij}^{(t)} + \frac{1}{t+1}\gamma_{ij}^{(t+1)} \\ &\leq \frac{2}{t+1} = \mathcal{O}(t^{-1}) \end{aligned}$$

as both $\gamma_{ij}^{(t)}$ and $\gamma_{ij}^{(t+1)}$ are bounded between 0 and 1.

We complete the proof by showing that also the tuning parameter $\hat{\eta}_{ij}^{(t)}$ has the same decaying rate

$$\begin{aligned}|\hat{\eta}_{ij}^{(t+1)} - \hat{\eta}_{ij}^{(t)}| &= |(\phi_t - \phi_{t+1})\tilde{\pi}_{ij} + (\phi_{t+1} - \phi_t)\hat{\pi}_{ij}^{(t+1)}| \\ &\leq |\phi_t - \phi_{t+1}|\tilde{\pi}_{ij} + |\phi_{t+1} - \phi_t|\hat{\pi}_{ij}^{(t+1)} \\ &= \mathcal{O}(t^{-1}) + \mathcal{O}(t^{-1}) = \mathcal{O}(t^{-1})\end{aligned}$$

as $\tilde{\pi}_{ij}$ is between 0 and 1.

□

Given the above lemma, and combining Lemmas 1 and 2 in Section 4.2.3 of Liang et al. [2022], we can establish the similar ergodicity and strong law of large numbers as in Theorem 2 of Liang et al. [2022].

C The warm-start approximation $\tilde{\pi}_{ij}$

The main idea of how to compute the warm-start approximation $\tilde{\pi}_{ij}$ is explained in Section 3.2 of the main body. In this section, we will fill in with more details on calculation of $\tilde{\pi}_{ij}$.

Consider the extended collection of permissible parent set $\mathbf{h}_+^j$, formally defined as

$$\mathbf{h}_+^j = \mathbf{h}^j \cup \left\{m \cup \{j\} \mid m \in \mathbf{h}^j,\ j \in (h^j)^c \setminus \{j\}\right\}. \tag{S.9}$$

The extended set $\mathbf{h}_+^j$ includes the parent set implied by $\mathbf{h}^j$, but also the additional parent sets obtained by including at least one more parent not included in h^j. Trivially thus the extended set $\mathbf{h}_+^j$ contains more permissible parents and minimises the risk of leaving out important parents when reducing the search space to a skeleton graph for scalability reasons [Kuipers et al., 2022].

We will show now how we calculate the marginal probability π_{ij}^u under the unconstrained posterior π^u and the extended sets $\{\mathbf{h}_+^j\}$. We define $\Gamma_j \subset \{0,1\}^p$ to be the sub-space of Γ such that $\gamma_{jj} = 0$ if $\gamma_j \in \Gamma_j$. Given Markov property, thanks to which the columns of γ can be factorised, we can define the unconstrained posterior distribution as

$$\pi^u(\gamma) \propto \prod_{j=1}^{n} p(\gamma_j|\mathcal{D}), \tag{S.10}$$

the marginal distribution of γ_j under the unconstrained posterior π^u is

$$\pi^u(\gamma_j) = \frac{p(\gamma_j|\mathcal{D})}{\sum\limits_{\gamma_j \in \Gamma_j} p(\gamma_j|\mathcal{D})}. \tag{S.11}$$

However, the sub-space Γ_j contains 2^{n-1} candidate parent sets and the full enumeration is computationally infeasible when n is big. So we replace the subspace by the extended collection of parent sets $\mathbf{h}_+^j$ which contains much less models then Γ_j. So this approximate the above as

$$\pi^u(\gamma_j) \approx \frac{p(\gamma_j|\mathcal{D})}{\sum\limits_{m \in \mathbf{h}_+^j} p(\rho(m)|\mathcal{D})}. \tag{S.12}$$

where $\rho : [p] \to \{0,1\}^p$ is a one-to-one mapping that converts the parent set m to γ_j which lies in the binary space Γ_j. Then we approximate the PEP with the following formula:

$$\pi_{ij}^u \stackrel{\text{def}}{=} \pi^u(\gamma_{ij} = 1) \approx \sum_{m \in \mathbf{h}_+^j:\ i \in m} \pi^u(\rho(m)). \tag{S.13}$$

The set $\{m \in \mathbf{h}_+^j \mid i \in m\}$ is equivalent to the set $\{\gamma_j \in \Gamma_j \mid \gamma_{ij} = 1\}$ of which X_i must be a parent of X_j from all these permissible parent sets. We also guarantee $\pi_{jj}^u = 0$ since j is not one of the permissible parents of j and not included in $\mathbf{h}_+^j$.

After the approximations, the π^u_{ij} are calculated and stored, then we can compute the warm-start approximations $\tilde{\pi}_{ij}$, followed by the equations (11) and (12) described in the main body. The calculations of $\tilde{\pi}_{ij}$ are equivalent to

$$\tilde{\pi}_{ij} = \frac{\pi^u_{ij}(1-\pi^u_{ji})}{\pi^u_{ij}(1-\pi^u_{ji}) + (1-\pi^u_{ij})\pi^u_{ji} + (1-\pi^u_{ij})(1-\pi^u_{ji})}. \tag{S.14}$$

The last step combines $\tilde{\pi}_{ij}$ with the ergodic average $\hat{\pi}_{ij}$ to form the algorithmic tuning parameters $\hat{\eta}_{ij}$ used in the PARNI-DAG proposal.

D The full PARNI-DAG proposal

We first recall the random neighbourhood construction specified in Section 3.1 of the main body. The neighbourhood indicator $k \in \mathcal{K} = \{0,1\}^{n\times n}$ indicates the positions that can be potentially flipped in the current DAG γ, and k follows the conditional distribution as given in (3) of the main body. The neighbourhood $\mathcal{N}(\gamma, k)$ given in (4) of the main body consists of 2^{d_k} DAG models, and the full enumeration over it is computationally efficient (and sometimes infeasible). For this reason we consider a pointwise implementation consisting in smaller sub-proposals. The first step in the pointwise implementation is to convert k into a set of variables $K = \{K_r\}_{r=1}^R$, which can take two categories:

- if $k_{ij} = 1$ but $k_{ij} = 0$, K_r only consists of one position $K_r = \{(i,j)\}$;
- if $k_{ij} = 1$ and $k_{ji} = 1$, K_r then contains two positions $K_r = \{(i,j),\ (j,i)\}$.

The order of K_r is random. To fully specify the sub-neighbourhoods $\mathcal{N}(\gamma(r-1), K_r)$ at each time r, we first define a new mapping that "flips" the components of γ according to the set K,

$$\gamma' = \text{flip}(\gamma, K)\ , \tag{S.15}$$

with $\gamma'_{ij} = 1 - \gamma_{ij}$ if $(i,j) \in K$ and $\gamma'_{ij} = \gamma_{ij}$ otherwise. The resulting neighbourhoods $N(\gamma(r-1), K_r)$ are defined as follows:

- If K only contains one position (i,j), then
$$\mathcal{N}(\gamma(r-1), K_r) = \{\gamma(r-1),\ \text{flip}(\gamma(r-1), K_r)\}\,. \tag{S.16}$$
- If K contains two positions (i,j) and (j,i), we construct a neighbourhood with reversal move given by
$$\begin{aligned}\mathcal{N}(\gamma(r-1), K_r) = \{&\gamma(r-1),\ \text{flip}(\gamma(r-1), (i,j)),\\ &\text{flip}(\gamma(r-1), (j,i)),\ \text{flip}(\gamma(r-1), K_r)\}.\end{aligned} \tag{S.17}$$

The next components in PARNI-DAG are the normalising constants of the sub-proposals q_{g,K_r} and of the reversal sub-proposals q_{g,K'_r}, used to facilitate the calculations of Metropolis-Hastings acceptance probability. As mentioned in the main body, we construct the auxiliary variables K' in the reversal move where K' consists of the same elements of K but with different order. One property of such a design is that the intermediate DAGs in the reversal move are identical to the those ones in the proposal move but only with opposite order. We then can re-use the posterior model probabilities calculated during the proposal move to compute the reversal probabilities directly. In addition, the Proposition 3 of Liang et al. [2022] implies that the acceptance probability in (7) can be expressed in terms of the product of the normalising constants. Given the normalising constants

$$Z(r) = \sum_{\gamma^* \in \mathcal{N}(\gamma(r-1), K_r)} g\left(\frac{\pi(\gamma^*)p(K_r|\gamma^*)}{\pi(\gamma(r-1))p(K_r|\gamma(r-1))}\right) \tag{S.18}$$

$$Z(R-r+1)' = \sum_{\gamma^* \in \mathcal{N}(\gamma(r), K_r)} g\left(\frac{\pi(\gamma^*)p(K_r|\gamma^*)}{\pi(\gamma(r))p(K_r|\gamma(r))}\right), \tag{S.19}$$

the MH acceptance probability is then simplified as follows

$$\alpha(\gamma, \gamma') = \min\left\{1, \prod_{r=1}^{R} \frac{Z(r)}{Z'(R-r+1)}\right\}. \tag{S.20}$$

To summarise the PARNI-DAG proposal, we now give its full algorithmic pseudo-code is given in **Algorithm** 1.

Algorithm 1 The PARNI-DAG proposal

Compute $\{\tilde{\pi}_{ij}\}$ as described in **Section 3.2** and **Appendix C**
Initialise $\gamma^{(1)}$, $\omega^{(1)}$ and $\hat{\eta}^{(1)} = \{\hat{\eta}_{ij}^{(1)}\}$
for $t = 1$ to $t = T$ **do**
  Sample $k \sim p_{\hat{\eta}^{(t)}}(\ \cdot\ |\ \gamma^{(t)})$ as in (3)
  Construct $K = \{K_r\}$ as described in **Appendix D**
  Set $\gamma(0) = \gamma^{(i)}$ and $\tilde{\mathcal{N}} = 0$
  for $r = 1$ to $r = R$ **do**
    if K_r has one element **then**
      Construct $\mathcal{N}(\gamma(r-1), K_r)$ as in (S.16)
    else
      Construct $\mathcal{N}(\gamma(r-1), K_r)$ as in (S.17)
    end if
    Sample $U_1 \sim \text{Unif}(0, 1)$
    if $U_1 < \omega^{(t)}$ **then**
      Sample $\gamma(r) \sim q_{g,K_r}(\gamma(r-1),\ \cdot\ )$ as in (5)
      Calculate $Z(r)$ as in (S.18)
      Calculate $Z'(R-r+1)$ as in (S.19)
    else
      Set $\gamma(r) = \gamma(r-1)$, $Z(r) = Z(R-r+1)' = 1$.
    end if
  end for
  Set $\gamma' = \gamma(R)$, sample $U_2 \sim \text{Unif}(0, 1)$ and compute $\alpha(\gamma^{(i)}, \gamma')$ as in (S.20);
  if $U_2 < \alpha(\gamma^{(i)}, \gamma')$ **then**
    Set $\gamma^{(t+1)} = \gamma'$
  else
    Set $\gamma^{(t+1)} = \gamma^{(t)}$
  end if
  Update $\hat{\eta}^{(t+1)} = \{\hat{\eta}_{ij}^{(t+1)}\}$ according to (9).
  Update $\omega^{(t+1)}$ according to (13).
end for

E Implementation of other MCMC algorithms

In this section, we will briefly describe the Add-Delete-Reverse (ADR) proposal, the order MCMC and the partition MCMC implemented in the experimental Section 4.

E.1 The Add-Delete-Reverse proposal

The ADR proposal is another example of random neighbourhood proposal. After the random neighbourhood is generated, unlike the PARNI-DAG proposal, ADR uniformly proposes a model within that neighbourhood.

In ADR, there are three possible moves, "Addition", "Deletion" and "Reversal", which defines the space $\mathcal{K}$ of indicator k. The indicator k is then uniformly drawn from one of these three possible moves. We construct the neighbourhood $\mathcal{N}(\gamma, k)$ according the the current DAG γ and indicator k as the following

$$\begin{aligned}
\mathcal{N}(\gamma, \text{"Addition"}) &= \{\gamma^* \in \Gamma \,|\, d_H(\gamma, \gamma^*) = 1,\, d_{\gamma^*} = d_\gamma + 1, \gamma^*_{ji} = 0 \text{ if } \gamma_{ij} = 1\} \\
\mathcal{N}(\gamma, \text{"Deletion"}) &= \{\gamma^* \in \Gamma \,|\, d_H(\gamma, \gamma^*) = 1,\, d_{\gamma^*} = d_\gamma - 1\} \\
\mathcal{N}(\gamma, \text{"Reversal"}) &= \{\gamma^* \in \Gamma \,|\, d_H(\gamma, \gamma^*) = 2,\, d_{\gamma^*} = d_\gamma, \exists\, i, j\colon \gamma_{ij} = \gamma^*_{ji} = 0 \text{ if } \gamma_{ji} = \gamma^*_{ij} = 1\}.
\end{aligned}$$

where $d_H(\cdot, \cdot)$ denotes the Hamming distance and d_γ is the number of edges induced by γ. A new DAG model $\gamma' \in \Gamma$ is proposed according to the uniform density:

$$q_k^{\text{ADR}}(\gamma, \gamma') = \begin{cases} \frac{1}{|\mathcal{N}(\gamma,k)|}, & \text{if } \gamma \in \mathcal{N}(\gamma, k), \\ 0, & \text{otherwise.} \end{cases} \tag{S.21}$$

As ADR is not guaranteed (unlike PARNI-DAG) to return a DAG, we carry out acyclicity diagnostics check when γ' is sampled. If γ does not lead to a valid DAG, we will accept the current state γ as the next state of the Markov chain. If γ' leads to a valid DAG, to preserve π-reversibility, we consider k' in the reverse move where k' is the opposite move to k (*e.g.* the addition is opposite to the deletion move, and vice versa). To complete the ADR proposal, the new DAG γ' is accepted as the next state of the Markov chain with probability

$$\begin{aligned}
\alpha_k(\gamma, \gamma') &= \min\left\{1, \frac{\pi(\gamma') q_{k'}(\gamma', \gamma)}{\pi(\gamma) q_k(\gamma, \gamma')}\right\} \\
&= \min\left\{1, \frac{\pi(\gamma') |\mathcal{N}(\gamma, k)|}{\pi(\gamma) |\mathcal{N}(\gamma', k')|}\right\}.
\end{aligned}$$

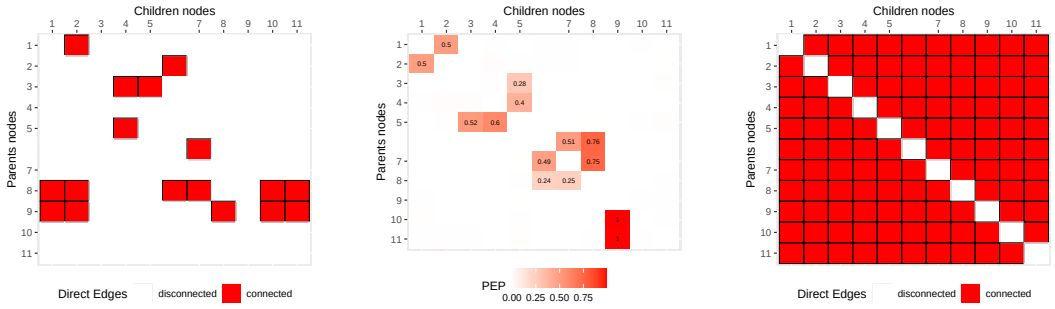
E.2 Order MCMC and Partition MCMC

Order and Partition MCMC samplers rely on DAG scoring measures where $p(\mathcal{G}|\mathcal{D}) = \prod_{j=1}^n S(X_j, \text{Pa}(X_j)|\mathcal{D})$, and on restricting the search space of DAGs to some lower capacity set. Friedman and Koller [2003] first proposed order MCMC, which operates in the space of orders $(\boldsymbol{X}, \prec)$ instead of the DAGs one. This search space is smoother, so that order MCMC generally achieves faster convergence with respect to classic neighborhood MCMC. The method considers permutations $(j_1, ..., j_n)$ of nodes that imply a linear order $j_1 \prec ... \prec j_n$. Each linear order is then given a score $R(\prec|\mathcal{D})$ equal to the sum of scores of all the compatible DAGs with the order $\prec \in (\boldsymbol{X}, \prec)$. To restrict the space capacity even more, sum and product operations can be swapped so that the order's score is:

$$\begin{aligned}
R(\prec|\mathcal{D}) = \sum_{\mathcal{G} \in \prec} p(\mathcal{G}|\mathcal{D}) &\propto \sum_{\mathcal{G} \in \prec} \prod_{j=1}^n S(X_j, \text{Pa}(X_j)|\mathcal{D}) \\
&\propto \prod_{j=1}^n \sum_{\text{Pa}(X_j) \in \prec} S(X_j, \text{Pa}(X_j)|\mathcal{D}),
\end{aligned}$$

where $S(\cdot)$ is a DAG scoring function, such as the Bayesian Gaussian equivalent (BGe) score for continuous variables [Geiger and Heckerman, 2002]. Constructing a MCMC in the space of orders means moving from order $\prec$ to order $\prec'$ with MH acceptance probability $\alpha_\prec(\prec, \prec') = \min\left\{1, \frac{R(\prec'|\mathcal{D})}{R(\prec|\mathcal{D})}\right\}$, according to following moves: a) "Local Swap" swaps adjacent nodes in $\prec$; b) "Global Swap" swaps random nodes in $\prec$; c) "Relocate" picks a node and replaces it in all possible positions in $\prec$, while keeping the rest of the order unchanged. Order MCMC is efficient, but has two main drawbacks compared to neighborhood-like structure MCMC: i) it outputs a sample of orders, not a DAG directly; sample of DAGs can be obtained by sampling, for each node independently, a parent set compatible with the relative order $\prec$, thanks to score decomposability (then get the DAG with maximum score in the sample as MAP estimate); ii) Order MCMC places a non-uniform prior on some DAGs by over-representing them in more than one order $\prec$, which causes the sample to be biased [Ellis and Wong, 2008].

In the attempt to correct sampling bias, Kuipers and Moffa [2017] proposes Partition MCMC, which operates in the space of ordered partitions instead. A labelled partition Λ is a pair of order $\prec$ and



(a) (b) (c)

Figure S.2: Protein dataset: (a) The ground-truth DAG. (b) The true PEPs estimated by a long chain of partition MCMC algorithm. (c) The full skeleton used in Section 5.1.

partitioning vector $\boldsymbol{\delta} = (\delta_1, ..., \delta_p)$ where $p < n$, and $\sum_{j=1}^{p} \delta_j = n$. Vector $\boldsymbol{\delta}$ divides the order into p parts $v_1, ..., v_p$ such that v_1 contains the first δ_1 nodes of the order $\prec$ and so on. A DAG is compatible with partition $\Lambda = (\prec, \boldsymbol{\delta})$ if for every node $X_j \in v_j$: i) X_j has at least one parent in part v_{j+1}; ii) all $\text{Pa}(X_j)$ belong to parts with indices higher than j; iii) $\text{Pa}(X_j) = \emptyset$ only if $j = p$. The possible moves of partition MCMC are: a) swap 2 nodes from different parts; b) swap 2 nodes from adjacent parts; c) split or join two parts; d) move a node in another existing part or create a new part with that node. Partition MCMC is unbiased in terms of DAG sampling; however it is extremely slow as it has very high computational complexity, and therefore can only be used to sample DAGs with very few nodes [Kuipers et al., 2022].

The orderMCMC and partitionMCMC schemes in the experiments are implemented via the `BiDAG`[1] package [Suter et al., 2023] (available from **CRAN**) written in `rcpp` [Eddelbuettel and François, 2011].

F Additional experimental results

F.1 Addition to Section 5.1

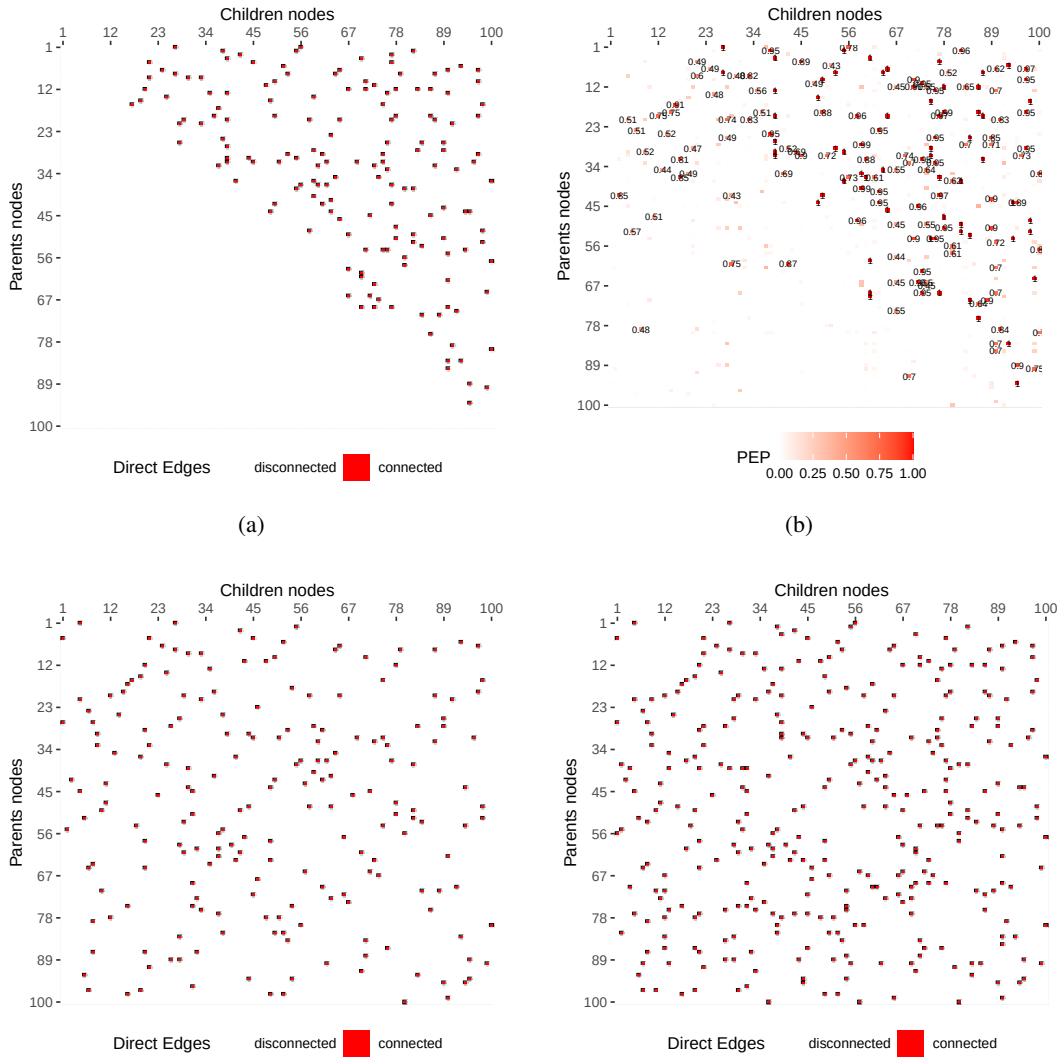
Protein data. We provide additional information on the real-world protein-signalling dataset ($n = 11$, $N = 853$) studied in **Section 4.1**. The ground-truth DAG constructed through expert knowledge is provided in Figure S.2 (a). The ground-truth estimate of PEP estimated by running a long chain of the partition MCMC algorithm is provided in Figure S.2 (b). The full skeleton used to implement different MCMC schemes is provided in S.2 (c).

gsim100 data. We also provide additional information on the simulated `gsim100` dataset ($n = 100$, $N = 100$) studied in **Section 4.1**, and featured in Suter et al. [2023]. The ground-truth DAG, containing 161 true edges, used to simulate data is presented in Figure S.3 (a). The ground-truth PEP estimates obtained from a long chain run of the PARNI-DAG proposal are provided in S.3 (b). The skeleton learned via the PC-algorithm (with a significance rate of 5%) $\mathcal{H}_{\text{PC}}$ and the skeleton learned from the iterative MCMC $\mathcal{H}_{\text{iter}}$ are provided in Figure S.3 (c) and (d) respectively.

F.2 Additional studies on the effects of tuning parameter of the PC-algorithm

In addition to **Section 4.1** results, we perform another batch of numerical studies to study the PARNI-DAG proposal run over different starting skeletons, by comparing their convergence behavior. The skeletons are mainly obtained by running the PC-algorithm with different levels of significance. We consider 4 different values of significance level, 0, 0.01, 0.05 and 0.1. The resulting skeletons are provided in Figure S.4 (The skeleton with significance level of 0.05 is provided in Figure S.3 (c)). As the significance level of the PC-algorithm increases, the resulting skeleton includes more edges

[1]More information on: `https://cran.r-project.org/web/packages/BiDAG/BiDAG.pdf`.



(a) (b)

(c) $\mathcal{H}_{\text{PC}}$ (d) $\mathcal{H}_{\text{iter}}$

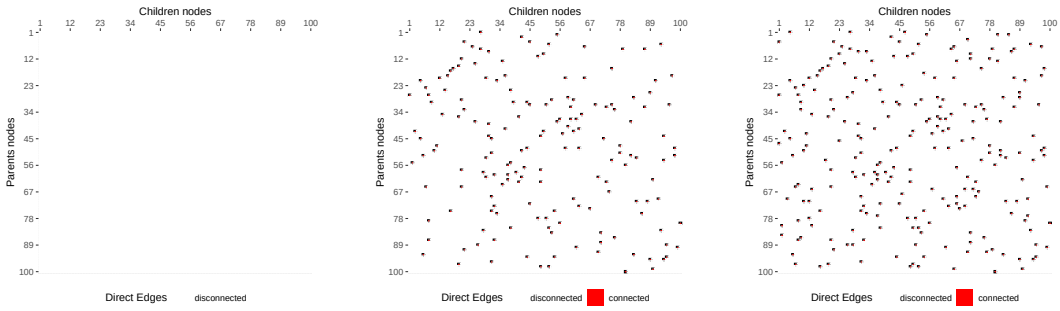
Figure S.3: `gsim100` dataset: (a) The ground-truth DAG. (b) The true PEPs estimated by a long chain of the PARNI-DAG proposal. (c) $\mathcal{H}_{\text{PC}}$ with significance level of 0.05. (d) $\mathcal{H}_{\text{iter}}$.

whilst the skeleton is empty when significance level is 0. In addition, we also compare these results to the skeleton from iteration MCMC as in **Section 4.1**.

Figure S.5 presents the trace plots of the log posterior model probabilities from the PARNI-DAG proposal under different skeletons. Using the skeleton $\mathcal{H}_{\text{PC}}$ with significance level of 0 trivially leads to the slowest convergence rate, while the skeleton $\mathcal{H}_{\text{iter}}$ (which includes more possible parent sets) results in the fastest convergence. The behaviour of the chains is similar. We can draw similar conclusions from the median MSEs presented in Table 1. The skeleton $\mathcal{H}_{\text{iter}}$ leads to the minimum MSE, while $\mathcal{H}_{\text{PC}}$ with significance level of 0 again trivially results in the largest MSE. The MSEs from other three options of significance levels are not significant different between each other, so using any one of them will give similar level of accuracy.

F.3 Additional studies on DAG learning performance

In addition to the results provided in Section 4.2 of the main body, in a similar way we compare PARNI-DAG with ADR and the same other methods in terms of accuracy in recovering the underlying DAG structure on an another example. We consider a randomly generated DAG structure featuring



(a) (b) (c)

Figure S.4: gsim100 dataset: (a) $\mathcal{H}_{\text{PC}}$ with significance level of 0. (b) $\mathcal{H}_{\text{PC}}$ with significance level of 0.01. (c) $\mathcal{H}_{\text{PC}}$ with significance level of 0.1

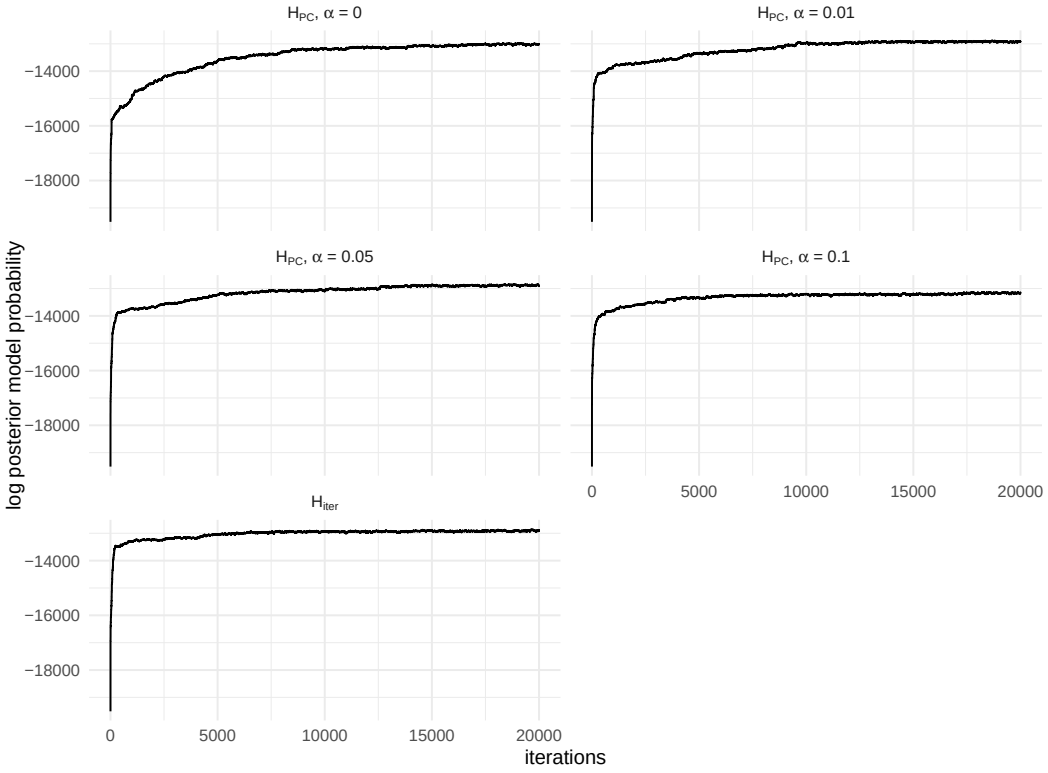


Figure S.5: gsim100 dataset: trace plots of log posterior model probabilities for the PARNI-DAG proposal under different skeletons.

50 nodes and 119 edges. Given this DAG structure, we simulate Gaussian data for three different sample sizes $N \in \{50, 100, 150\}$.

To recall, the models compared are: i) PC algorithm [Spirtes et al., 2000]; ii) GES algorithm [Chickering, 1996, 2002]; iii) Order MCMC [Friedman and Koller, 2003]; iv) Iterative MCMC [Kuipers et al., 2022]; v) Partition MCMC [Kuipers and Moffa, 2017]; vi) ADR; vii) PARNI-DAG.

Models' performance is compared according to same metric, the *Structural Hamming Distance* $d_H(\hat{\gamma}, \gamma)$ between the estimated and the true DAG. The experiments are replicated for 10 times, for each of the sample sizes above. Results on the distributions of SHD are depicted in Figure S.6. We can see how Iterative MCMC and PARNI-DAG are the best competing models out of this example. Note that this is a medium size graph, and while we have argued in the main body

skeleton	significance rate (α)	MSE
$\mathcal{H}_{\text{PC}}$	0	2.40
	0.01	1.92
	0.05	1.95
	0.1	1.95
$\mathcal{H}_{\text{iter}}$	-	1.85

Table 1: `gsim100` dataset: time-normalised median MSE ($\times 10^{-7}$) on estimating posterior edge probabilities. (*Lower is better.*)

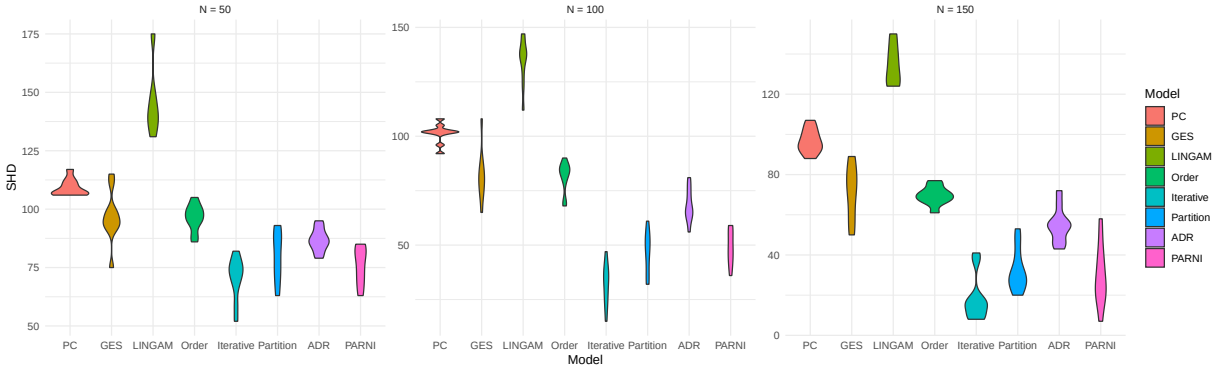


Figure S.6: Distribution of SHD of each compared model, for different sample sizes $N \in \{50, 100, 150\}$, over 10 replications.

that PARNI-DAG particularly excels in settings characterized by larger size graphs ($|\mathcal{V}| \geq N$), it is nonetheless competitive with the best performing method, Iterative MCMC here, also on lower dimensional DAGs.

References

David Maxwell Chickering. Learning Bayesian networks is NP-complete. *Learning from data: Artificial intelligence and statistics V*, pages 121–130, 1996.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

Dirk Eddelbuettel and Romain François. Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*, 40(8):1–18, 2011. doi: 10.18637/jss.v040.i08.

Byron Ellis and Wing Hung Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103(482):778–789, 2008.

Nir Friedman and Daphne Koller. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine learning*, 50:95–125, 2003.

Dan Geiger and David Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. *The Annals of Statistics*, 30(5):1412–1440, 2002.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Jack Kuipers and Giusi Moffa. Partition MCMC for Inference on Acyclic Digraphs. *Journal of the American Statistical Association*, 112(517):282–299, 2017.

Jack Kuipers, Polina Suter, and Giusi Moffa. Efficient Sampling and Structure Learning of Bayesian Networks. *Journal of Computational and Graphical Statistics*, 31(3):639–650, 2022.

Xitong Liang, Samuel Livingstone, and Jim Griffin. Adaptive random neighbourhood informed Markov chain Monte Carlo for high-dimensional Bayesian variable selection. *Statistics and Computing*, 32(5):84, 2022.

Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000.

Polina Suter, Jack Kuipers, Giusi Moffa, and Niko Beerenwinkel. Bayesian Structure Learning and Sampling of Bayesian Networks with the R Package BiDAG. *Journal of Statistical Software*, 105 (9):1–31, 2023.