

Scalable Bayesian Structure Learning for Gaussian Graphical Models Using Marginal Pseudo-Likelihood

Reza Mohammadi^{*}, Marit Schoonhoven[†], Lucas Vogels[†], and Ş. İlker Birbil[†]

Abstract. Bayesian methods for learning Gaussian graphical models offer a principled framework for quantifying model uncertainty and incorporating prior knowledge. However, their scalability is constrained by the computational cost of jointly exploring graph structures and precision matrices. To address this challenge, we perform inference directly on the graph by integrating out the precision matrix. We adopt a marginal pseudo-likelihood approach, eliminating the need to compute intractable normalizing constants and perform computationally intensive precision matrix sampling. Building on this framework, we develop continuous-time (birth–death) and discrete-time (reversible jump) Markov chain Monte Carlo (MCMC) algorithms that efficiently explore the posterior over graph space. We establish theoretical guarantees for posterior contraction, convergence, and graph selection consistency. The algorithms scale to large graph spaces, enabling parallel exploration for graphs with over 1,000 nodes, while providing uncertainty quantification and supporting flexible prior specification over the graph space. Extensive simulations show substantial computational gains over state-of-the-art Bayesian approaches without sacrificing graph recovery accuracy. Applications to human and mouse gene expression datasets demonstrate the ability of our approach to recover biologically meaningful structures and quantify uncertainty in complex networks. An implementation is available in the R package *BDgraph*.

Keywords: Markov random field, model selection, link prediction, network reconstruction, Bayes factor.

1 Introduction

Undirected graphical models (Lauritzen, 1996; Koller and Friedman, 2009) serve as fundamental tools for analyzing conditional dependencies among variables. A conditional dependency represents the association between variables conditional on the presence of other variables. These dependencies are naturally represented by graphs, where nodes correspond to random variables (Lauritzen, 1996), and the absence of an edge between two nodes signifies conditional independence (Rue and Held, 2005). The process of estimating this underlying graph structure is known as *structure learning*.

In this article, we consider Bayesian structure learning approaches for estimating Gaussian graphical models (GGMs), in contrast to frequentist techniques such as neigh-

arXiv: 2307.00127v3

^{*}Department of Business Analytics, Faculty of Economics and Business, University of Amsterdam, a.mohammadi@uva.nl

[†]Department of Business Analytics, Faculty of Economics and Business, University of Amsterdam

neighborhood selection (Meinshausen and Bühlmann, 2006) and the optimization of the likelihood function (Friedman et al., 2008). Bayesian methods offer key advantages, including the ability to quantify model uncertainty through posterior distributions and incorporate prior knowledge. However, their computational scalability often lags behind frequentist alternatives as the dimensionality of the problem increases.

The primary goal of Bayesian Structure Learning is to infer the underlying graph given the observed data. This is typically achieved by computing the posterior distribution of the graph conditional on the data. For GGMs, this requires the evaluation of complex integrals, which becomes increasingly challenging or even infeasible for large-scale graphs. Consequently, most Bayesian methods compute the joint posterior distribution of the graph and precision matrix. These methods have two primary bottlenecks in each Markov chain Monte Carlo (MCMC) iteration: (i) approximating intractable normalizing constants and (ii) iteratively updating the precision matrix. Consequently, full joint posterior exploration becomes computationally prohibitive for graphs exceeding 100 nodes in reversible jump and birth-death MCMC algorithms (Mohammadi and Wit, 2015), and 250 nodes in the spike-and-slab approach (Wang, 2015). To mitigate these limitations, Mohammadi et al. (2023) introduced an MCMC-based method incorporating normalizing constant approximations, enabling scalability to a few hundred nodes. Similarly, van den Boom et al. (2022) proposed a G -Wishart weighted proposal algorithm that leverages delayed acceptance MCMC and an informed proposal distribution to reduce the computational costs. These methods, despite their advances, remain computationally infeasible for modern applications involving thousands of variables.

Alternatively, several techniques have been proposed that bypass the full exploration of the graph space. They include methods based on multiple testing of marginal and conditional independence relationships (Williams and Mulder, 2020; Leday and Richardson, 2019). These methods are effective for large-scale problems but primarily focus on controlling the type I error rather than optimizing goodness-of-fit (Drton and Perlman, 2007). Another approach avoids sampling over the graph space entirely by sampling solely from the posterior distribution of the precision matrix. Examples include block Gibbs samplers (Wang, 2012; Li et al., 2019; Sagar et al., 2024) and, recently, low-rank matrix decomposition methods (Chandra et al., 2024). Although these approaches improve computational efficiency, they do not fully explore the posterior distribution of the graph space, restricting their ability to quantify model uncertainty. Moreover, approaches that focus on the precision matrix rather than the graph itself lack priors on the graphical structure and require additional steps to infer the underlying graph.

Another strategy to gain scalability is the approximation of the Gaussian likelihood. It has been applied to Bayesian structure learning for GGMs, most successfully by Atchadé (2019) and Jalali et al. (2020). Both methods back their approximations with theoretical guarantees and push the boundaries of computational efficiency. Their algorithms still require sampling a precision matrix at every iteration, which poses challenges for scalability in high-dimensional settings. Instead, one could integrate out the precision matrix and target only the posterior on the graph space. The resulting marginal likelihood is not available in closed-form, but can be approximated using the pseudo-likelihood approximation by Besag (1975), resulting in the marginal pseudo-likelihood

(MPL). Early studies (Pensar et al., 2017; Dobra and Mohammadi, 2018) applied MPL to undirected graphical models with discrete variables. Consonni and Rocca (2012); Carvalho and Scott (2009); Stingo and Marchetti (2015) extended MPL to GGMs but initially restricted it to decomposable graphs. Leppä-aho et al. (2017) later adapted MPL to non-decomposable graphs via a score-based hill-climbing algorithm. However, this method only estimates the maximum a posteriori probability rather than fully characterizing posterior uncertainty.

Our Proposed Method and Key Contributions: This article makes two main contributions to Bayesian structure learning in GGMs. First, we introduce a scalable framework capable of fully exploring the graph space. Second, we provide theoretical guarantees for consistency and convergence, which ensure reliable inference in large-scale graphical models.

We propose a scalable Bayesian framework for Gaussian graphical models by replacing the Gaussian likelihood with a marginal pseudo-likelihood (MPL) formulation. While the MPL approximation has been explored previously in Bayesian structure learning (Pensar et al., 2017), we extend its use to the design of scalable MCMC-based algorithms that operate directly in the graph space. Specifically, we develop two algorithms that combine the MPL approach with birth–death and reversible jump MCMC frameworks, enabling scalable and parallelizable exploration of large graph spaces and making inference feasible for graphs with more than 1,000 nodes. Our framework differs from the likelihood approximation methods of Atchadé (2019) and Jalali et al. (2020) in two important respects. First, their methods require sampling a precision matrix at every iteration, whereas our approach bypasses the precision matrix space entirely and samples solely over the graph space, leading to substantial gains in computational efficiency. Second, their methods approximate the full likelihood, namely the probability of the data given the precision matrix, while we approximate the marginal likelihood, namely the probability of the data given the graph.

Beyond scalability, we establish theoretical guarantees for our approach. We prove that the pseudo-posterior concentrates around the true posterior as the sample size increases. Moreover, Theorem 4.3 establishes the consistency of our algorithms, ensuring recovery of the true graph as both the sample size and the number of MCMC iterations grow. The theoretical guarantees we establish are general, applying not only to our proposed algorithms but to any MCMC procedure that targets the pseudo-posterior. In addition, our framework provides uncertainty quantification through edge inclusion probabilities and other graph characteristics, as illustrated in Section 6. It can also incorporate prior knowledge about the graph structure, making the method adaptable to diverse applications.

Complementing our theoretical results, we present an extensive simulation study in Section 5 to assess the practical performance of our proposed algorithms and compare them with the state-of-the-art Bayesian structure learning methods for GGMs. To highlight improvements in computational efficiency and graph recovery accuracy, Figure 1 shows the convergence of the Area Under the Precision-Recall Curve (AUC-PR) over running time for our proposed algorithms—Birth-Death MCMC with Marginal Pseudo-Likelihood (BD-MPL) and Reversible-Jump MCMC with Marginal Pseudo-Likelihood

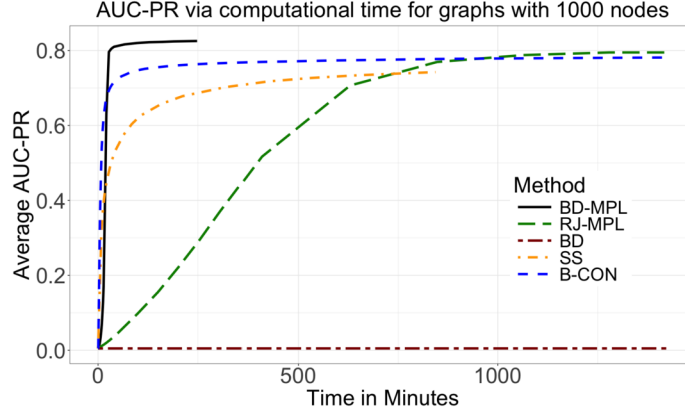


Figure 1: Average AUC-PR over running time for five algorithms applied to a simulated Cluster graph with 1,000 nodes, 0.5% edge density, and 1,050 observations (see Section 5), with 8 replications. BD-MPL and RJ-MPL are our proposed algorithms (Algorithms 1 and 2, respectively). SS is the spike-and-slab method of Wang (2015), BD is the birth-death MCMC algorithm of Mohammadi et al. (2023), and B-CON is the method of Jalali et al. (2020).

(RJ-MPL). Their performance is compared with leading alternatives, including the spike-and-slab (SS) method (Wang, 2015), the standard birth-death (BD) algorithm (Mohammadi and Wit, 2015; Mohammadi et al., 2023), and the Bayesian analogue of the CONvex CORrelation selection method (B-CONCORD; hereafter referred to as B-CON) (Jalali et al., 2020). The simulation is based on a Cluster graph with 1,000 nodes, an edge density of 0.5%, and 1050 observations. The results demonstrate the superior convergence speed of our BD-MPL algorithm, which achieves an AUC-PR above 0.8 in under 10 minutes. In comparison, the B-CON algorithm requires approximately 30 minutes to reach a moderate AUC-PR and fails to match BD-MPL’s performance even after one full day. The SS method also takes nearly a day to achieve a reasonable AUC-PR, but still lags in overall accuracy. The BD algorithm, by contrast, struggles considerably in this large-scale setting, remaining near an AUC-PR of 0.0 even after several days of computation.

The article is organized as follows. Section 2 introduces the fundamental concepts of Bayesian structure learning for GGMs. Section 3 presents the MPL approach and the two proposed MCMC-based algorithms for large-scale graph recovery. Section 4 establishes the theoretical properties of the algorithms, including posterior contraction (Lemma 4.1), convergence (Lemma 4.2), and graph selection consistency (Theorem 4.3). Section 5 provides a comprehensive simulation study assessing the computational efficiency and accuracy of our methods compared to leading Bayesian approaches. In Section 6, we demonstrate the versatility of our methods in uncertainty quantification through two real-world applications, showcasing their strengths on both medium- and large-scale datasets. Finally, we conclude with reflections and future research directions. Our implementation is available in the R package `BDgraph` (Mohammadi et al., 2024).

2 Bayesian Structure Learning for GGMs

We denote an undirected graph as $G = (V, E)$, where V is the set of p nodes representing variables, and $E \subset \{(i, j) \mid 1 \leq i < j \leq p\}$ is the set of edges. An edge $(i, j) \in E$ indicates a connection between nodes i and j , where each node corresponds to a distinct random variable, collectively forming a p -dimensional random vector. The observed data matrix is $\mathbf{X} = \left(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)}\right)^T$ with dimensions $n \times p$, where each independent sample $\mathbf{X}^{(k)}$ ($k \in \{1, \dots, n\}$) is a p -dimensional random vector. In GGMs, each $\mathbf{X}^{(k)}$ follows a multivariate Gaussian distribution $\mathcal{N}_p(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the covariance matrix and $\mathbf{K} = \mathbf{\Sigma}^{-1}$ is the precision matrix with elements K_{ij} . Two nodes i and j are conditionally independent if and only if $K_{ij} = 0$ (Lauritzen, 1996).

In Bayesian structure learning, the goal is to estimate the posterior probability of a graph G given the data $\mathbf{X}$

$$P(G|\mathbf{X}) \propto P(G)P(\mathbf{X}|G), \quad (1)$$

where $P(G)$ is a prior distribution over the graph-space $\mathcal{G}_p$ of undirected graphs with p nodes, and $P(\mathbf{X}|G)$ represents the marginal likelihood of G . A common specification for the prior $P(G)$ assumes independent edge inclusion probabilities $\beta_{ij} \in (0, 1)$ for each edge $e = (i, j)$, enabling the incorporation of prior domain knowledge. This flexible formulation allows structural information to be encoded, for example, by assigning higher values of β_{ij} to edges believed to exist and lower values otherwise. When all β_{ij} are set to a common value $\beta \in (0, 1)$, the prior simplifies to

$$P(G) \propto \beta^{|E|}(1 - \beta)^{|\bar{E}|}, \quad (2)$$

where $|E|$ denotes the number of edges in G , and $|\bar{E}|$ is the number of absent edges. Smaller values of β favor sparser graphs. When $\beta = 0.5$, the prior becomes uniform over the graph space. The hyperparameter β directly controls the expected number of edges and encodes prior beliefs about graph sparsity. It is important to note that our Bayesian approach is not restricted to this prior form and can accommodate any prior distribution on G . For alternative graph prior specifications, we refer the reader to Scutari (2013); van den Boom et al. (2023).

For the marginal likelihood of G , we have

$$P(\mathbf{X}|G) = \int_{\mathbf{K}} P(\mathbf{X}|G, \mathbf{K})P(\mathbf{K}|G)d\mathbf{K}, \quad (3)$$

where $P(\mathbf{K}|G)$ denotes the prior for $\mathbf{K}$ given G and $P(\mathbf{X}|G, \mathbf{K})$ is the likelihood function. A well-defined choice for the prior distribution of the precision matrix $\mathbf{K}$ is the G -Wishart distribution (Roverato, 2002; Letac and Massam, 2007), which serves as the conjugate prior for the multivariate Gaussian likelihood. The G -Wishart density is

$$P(\mathbf{K}|G) = \frac{1}{I_G(b, \mathbf{D})} |\mathbf{K}|^{\frac{b-2}{2}} \exp \left\{ \frac{-1}{2} \text{tr}(\mathbf{K}\mathbf{D}) \right\} \mathbf{1}(\mathbf{K} \in P_G), \quad (4)$$

where $|\mathbf{K}|$ denotes the determinant of $\mathbf{K}$, $\text{tr}(\mathbf{A})$ is the trace of a square matrix $\mathbf{A}$, $I_G(b, \mathbf{D})$ is the normalizing constant, and P_G is the set of positive definite matrices $\mathbf{K}$ with $K_{ij} = 0$ if $(i, j) \notin E$, and $\mathbf{1}(\mathbf{K} \in P_G)$ is an indicator function that equals 1 if $\mathbf{K} \in P_G$ and 0 otherwise. We denote this distribution with $W_G(b, \mathbf{D})$, where the symmetric positive definite matrix $\mathbf{D}$ and the scalar $b > 2$ are the scale and the shape parameters of the G -Wishart distribution, respectively. Using the G -Wishart prior, (3) becomes

$$P(\mathbf{X}|G) = (2\pi)^{-\frac{np}{2}} \frac{I_G(b+n, \mathbf{D} + \mathbf{U})}{I_G(b, \mathbf{D})},$$

where $\mathbf{U} = \mathbf{X}^T \mathbf{X}$. Since this ratio of normalizing constants is intractable (Atay-Kayis and Massam, 2005; Mohammadi et al., 2023; Wong et al., 2024, 2025; Uhler et al., 2018), most Bayesian methods circumvent it with MCMC algorithms that sample over the joint space of graphs and precision matrices.

The joint posterior distribution of the graph G and the precision matrix $\mathbf{K}$ is

$$\begin{aligned} P(G, \mathbf{K}|\mathbf{X}) &\propto P(\mathbf{X}|\mathbf{K}, G)P(\mathbf{K}|G)P(G) \\ &\propto P(G) \frac{1}{I_G(b, \mathbf{D})} |\mathbf{K}|^{\frac{b+n-2}{2}} \exp \left\{ \frac{-1}{2} \text{tr}(\mathbf{K}(\mathbf{D} + \mathbf{U})) \right\}. \end{aligned} \quad (5)$$

Computing this posterior distribution for all graphs $G \in \mathcal{G}_p$ is computationally infeasible for $p > 10$ due to the exponential number of possible graphs. Consequently, most Bayesian methods rely on MCMC-based algorithms, such as the reversible jump MCMC algorithm (Green, 1995), a discrete-time Markov chain approach (Dobra et al., 2011; Lenkoski and Dobra, 2011; Cheng and Lenkoski, 2012; Lenkoski, 2013; Hinne et al., 2014). reversible jump MCMC explores the graph space by proposing to add or remove a single edge at each iteration, accepting the move with a probability that depends on the ratio of posterior probabilities (the conditional Bayes factor). However, reversible jump MCMC often suffers from low acceptance rates, requiring many iterations to converge. To mitigate this, Mohammadi and Wit (2015) proposed a continuous-time MCMC algorithm, in which transitions between neighboring graphs, via edge additions or deletions, are modeled as independent Poisson processes.

A major computational bottleneck in these algorithms is evaluating the ratio of posterior probabilities, which requires computing expensive normalizing constants. Additionally, each new graph necessitates deriving a precision matrix by sampling from the G -Wishart distribution, further increasing computational overhead. Several improvements have been proposed to these challenges – for a comprehensive review, see Vogels et al. (2024) – but efficient exploration of large graph space remains challenging.

3 Bayesian Structure Learning with MPL

Recall that we aim to reduce the computational cost by sampling directly over the graph space instead of the joint space of graphs and precision matrices using MCMC-based search algorithms. To achieve this, we introduce two MCMC-based search algorithms that employ the MPL approach in conjunction with birth-death and reversible jump

MCMC algorithms. In Section 3.1, we illustrate how the MPL approach facilitates the derivation of Bayes factors for MCMC algorithms. The birth-death and reversible jump MCMC-based algorithms are described in Sections 3.2 and 3.3, respectively.

3.1 Marginal Pseudo-Likelihood

Bayesian structure learning in GGMs relies on the development of computationally efficient search algorithms, such as those outlined in Sections 3.2 and 3.3. A key component of these algorithms, often termed neighborhood search algorithms, is the calculation of Bayes factors between pairs of neighboring graphs as

$$\frac{P(G'|\mathbf{X})}{P(G|\mathbf{X})} = \frac{P(\mathbf{X}|G')P(G')}{P(\mathbf{X}|G)P(G)}, \quad (6)$$

where the graphs $G = (V, E)$ and $G' = (V, E')$ differ by a single edge $e = (i, j)$, such that $G' = (V, E \cup e)$ or $G' = (V, E \setminus e)$. To compute $P(G|\mathbf{X})$, we require the marginal likelihood $P(\mathbf{X}|G)$ in (3), which does not have a closed-form expression. Instead, we use the pseudo-likelihood (Besag, 1975), which approximates the marginal likelihood with a product of conditional likelihoods as

$$P(\mathbf{X}|G) \approx \tilde{P}(\mathbf{X}|G) := \prod_{h=1}^p P(\mathbf{X}_h | \mathbf{X}_{nb(h)}, G), \quad (7)$$

where $\mathbf{X}_h$ is a n -dimensional vector corresponding to node h , $nb(h)$ refers to the set of neighbors of node h with respect to $G = (V, E)$ and $\mathbf{X}_{nb(h)}$ is the sub-matrix of $\mathbf{X}$ corresponding to the nodes that are in $nb(h)$. We then have

$$\begin{aligned} \frac{P(\mathbf{X}|G')}{P(\mathbf{X}|G)} &\approx \frac{\prod_{h=1}^p P(\mathbf{X}_h | \mathbf{X}_{nb(h)}, G')}{\prod_{h=1}^p P(\mathbf{X}_h | \mathbf{X}_{nb(h)}, G)} \\ &= \frac{P(\mathbf{X}_i | \mathbf{X}_{nb(i)}, G') P(\mathbf{X}_j | \mathbf{X}_{nb(j)}, G')}{P(\mathbf{X}_i | \mathbf{X}_{nb(i)}, G) P(\mathbf{X}_j | \mathbf{X}_{nb(j)}, G)}, \end{aligned} \quad (8)$$

where the last step is based on the fact that the graphs G and G' differ by a single edge $e = (i, j)$. To finalize the approximation of the Bayes factor (6), one task remains: a closed-form expression of the terms $P(\mathbf{X}_h | \mathbf{X}_{nb(h)}, G)$ in equation (8). However, a closed-form expression is intractable due to the presence of the intractable normalizing constants of the G-Wishart prior (4). Instead, Leppä-aho et al. (2017) provide an approximation

$$P(\mathbf{X}_h | \mathbf{X}_{nb(h)}, G) \approx \tilde{P}(\mathbf{X}_h | \mathbf{X}_{nb(h)}, G) := \pi^{-\frac{n-1}{2}} \frac{\Gamma(\frac{n+p_h}{2})}{\Gamma(\frac{p_h+1}{2})} n^{-\frac{2p_h+1}{2}} \left(\frac{|\mathbf{U}_{nb(h) \cup h}|}{|\mathbf{U}_{nb(h)}|} \right)^{-\frac{n-1}{2}}, \quad (9)$$

where p_h is the size of the neighborhood $nb(h)$, and $\mathbf{U}_A$ denotes the submatrix of $\mathbf{U}$ corresponding to the variables in the set A . For (9) to be well defined, the matrices $\mathbf{U}_{nb(h)}$ and $\mathbf{U}_{nb(h) \cup h}$ must be positive definite, which requires $n \geq p_h + 1$. Since this

must hold for all $h = 1, \dots, p$, the condition becomes $n \geq \max\{p_h + 1 : h = 1, \dots, p\}$. This requirement is automatically satisfied when $n \geq p$, but it often also holds when $n < p$ due to the typical sparsity of precision matrices. The condition fails only if a node has more than $n - 1$ neighbors. For example, with $p = 100$ and $n = 50$, the condition is violated only if a node is connected to 50 or more neighbors, in which case $p_h = 50$.

To clarify the approximation in (9), we turn to directed acyclic graphical models (DAGs). Consonni and Rocca (2012) derived a closed-form expression for the marginal likelihood $P(\mathbf{X} \mid M)$ of a Gaussian DAG, M . Their approach build on the Geiger and Heckerman (2002, Theorem 2) that proved the problem reduces to computing the marginal likelihood $P(\mathbf{X}_A \mid M_c)$, where $\mathbf{X}_A$ is the submatrix of $\mathbf{X}$ corresponding to the variables in the set A , and M_c is a complete DAG. In analogy to (3), this can be written as

$$P(\mathbf{X}_A \mid M_c) = \int_{\mathbf{K}} P(\mathbf{X}_A \mid M_c, \mathbf{K}) P(\mathbf{K} \mid M_c) d\mathbf{K}. \quad (10)$$

To evaluate $P(\mathbf{X}_A \mid M_c)$, we must specify a prior $P(\mathbf{K} \mid M_c)$. This prior should be proper (meaning it integrates to one), objective (containing no subjective information), and ideally computationally convenient for evaluating (10). Consonni and Rocca (2012) show that the Wishart prior $W(p, \mathbf{U}/n)$ satisfies these requirements. This prior is an example of a fractional (or data-dependent) prior, as it incorporates a fraction of the observed data. Crucially, it allows for a closed-form expression of $P(\mathbf{X}_A \mid M_c)$ analogous to (9), as shown in Consonni and Rocca (2012, Equations 24 and 25). These results extend to decomposable undirected graphs (Consonni and Rocca, 2012; Carvalho and Scott, 2009). In particular, Consonni and Rocca (2012, Equation 29) derived a closed-form expression identical to (9), which is exact for decomposable graphs. For non-decomposable graphs, the equality no longer holds, but Leppä-aho et al. (2017) demonstrated that the right-hand side of (9) still provides a useful approximation.

Consequently, our approach to evaluate the Bayes factor (8) involves two layers of approximation: the use of a pseudo-likelihood in (7), and the approximation of its local components via (9). Combining these two approximations gives the pseudo-posterior as

$$P(G \mid \mathbf{X}) \approx \tilde{P}(G \mid \mathbf{X}) \propto P(G) \prod_{h=1}^p \tilde{P}(\mathbf{X}_h \mid \mathbf{X}_{nb(h)}, G). \quad (11)$$

Sampling from this pseudo-posterior requires computing the Bayes factor in (8), which can be done using the closed-form expression in (9). This requires only four evaluations of (9), making the computation highly efficient for MCMC-based search algorithms. In the following sections, we introduce two such algorithms for general undirected GGMs, applicable to both decomposable and non-decomposable graphs. Section 4 provides the theoretical justification for employing the MPL approximation within our structure learning framework.

3.2 Birth-Death MCMC Algorithm

The birth-death MCMC algorithm, based on a continuous-time Markov process (Preston, 1976), was applied to GGMs by Mohammadi and Wit (2015) for sampling from

the joint posterior of graphs and precision matrices (5). We propose a modified version that incorporates the MPL approximation, allowing sampling exclusively over the graph space $\mathcal{G}_p$ from the pseudo-posterior (11). At iteration $s \in \{1, \dots, S\}$, the state of the Markov chain is a graph $G^{(s)}$, which transitions to $G^{(s+1)}$ by either adding (birth) or removing (death) a single edge. These birth and death events are modeled as independent Poisson processes, each occurring with rate $R_e(G)$. If a birth of edge $e = (i, j)$ occurs, the process moves to $G^{+e} = (V, E \cup e)$; if a death of edge e occurs, it moves to $G^{-e} = (V, E \setminus e)$. Since the events are governed by independent Poisson processes, the waiting time between consecutive events follows an exponential distribution with mean

$$W(G) = \frac{1}{\sum_e R_e(G)}, \quad (12)$$

where the summation is over all $e \in \{(i, j) | 1 \leq i < j \leq p\}$. The associated birth/death probabilities are

$$P(\text{birth/death of edge } e) = R_e(G)W(G) \text{ for all } e \in \{(i, j) | 1 \leq i < j \leq p\}. \quad (13)$$

The birth-death MCMC search algorithm converges to the target posterior distribution $P(G|\mathbf{X})$ in (1) by setting the birth/death rates to

$$R_e(G) = \min \left\{ \frac{P(G'|\mathbf{X})}{P(G|\mathbf{X})}, 1 \right\}, \quad \text{for each } e \in \{(i, j) | 1 \leq i < j \leq p\}, \quad (14)$$

where G' is either G^{+e} or G^{-e} . Dobra and Mohammadi (2018, Theorem 5.1) showed that these rates lead to convergence of the algorithm to the target posterior distribution in (1); see also Lemma 4.2. Their proof relies on the detailed balance condition, which is sufficient for convergence, though not strictly necessary (Cappé et al., 2003). This birth-death algorithm, which searches exclusively over the graph space, is referred to as BD-MPL. Algorithm 1 provides the pseudo-code for this algorithm.

Algorithm 1: BD-MPL search algorithm.

Input: Data $\mathbf{X}$ and an initial graph $G_0 = (V, E)$.
Output: Samples from the posterior distribution (1).

- 1 Calculate in parallel the MPL of each node given G_0 by (9);
- 2 Calculate in parallel all the rates for each edge given G_0 by (14);
- 3 **for** S iterations **do**
- 4 **for** the rates that need to be re-evaluated **do**
- 5 | Calculate in parallel the birth and death rates by (14);
- 6 **end**
- 7 Calculate the waiting time by (12);
- 8 Update the graph by the birth/death probabilities in (13);
- 9 Update the MPL of the two nodes associated with the flipped edge.
- 10 **end**

Algorithm 1 offers a significant computational advantage, particularly in determining birth and death rates, which are well-suited for parallel execution. By retaining the

marginal pseudo-likelihood of the current graph nodes, recalculations are required only for the two nodes associated with the flipped edge. This is why all marginal pseudo-likelihoods are initially calculated outside the main loop in line 1. Similarly, the majority of the birth and death rates remain unchanged between iterations, justifying their initial calculation outside the main loop in line 2. By retaining rates across iterations, only a small fraction requires re-evaluation. For a graph with p nodes, only $2p - 3$ of the possible $p(p - 1)/2$ rates need to be updated. For example, for a graph with $p = 100$ nodes, the BD-MPL algorithm recalculates just 197 rates per iteration, compared to the 4950 rates that would otherwise need updating in the traditional birth-death MCMC algorithm. These computational optimizations have been implemented in Algorithm 1, which is coded in C++ and ported to R. This implementation is available in the R package `BDgraph` (Mohammadi et al., 2024) as the `bdgraph.mpl()` function.

The output of Algorithm 1 consists of a set of S sampled graphs, $\{G^{(1)}, \dots, G^{(S)}\}$, and their corresponding waiting times, $\{W^{(1)}, \dots, W^{(S)}\}$. These outputs facilitate various types of inference. Specifically, the expected value of any information function $f : \mathcal{G}_p \rightarrow \mathbb{R}$ can be approximated using the Rao-Blackwellized estimator (Cappé et al., 2003) as

$$E(f(G)|\mathbf{X}) \approx \frac{\sum_{s=1}^S W^{(s)} f(G^{(s)})}{\sum_{s=1}^S W^{(s)}}, \quad (15)$$

which is well-suited for uncertainty quantification, as it accounts for posterior uncertainty. By defining different information functions $f(G)$, various posterior probabilities can be computed. For instance, setting $f(G)$ to detect highly connected nodes (hubs) enables inference on hub probabilities, as demonstrated in Section 5. Similarly, defining $f(G)$ as 1 when an edge e is in G and 0 otherwise allows the estimation of posterior edge inclusion probabilities as

$$\hat{P}_e = \hat{P}_e(e \in G^*|\mathbf{X}) = \frac{\sum_{s=1}^S W^{(s)} \mathbf{1}(e \in G^{(s)})}{\sum_{s=1}^S W^{(s)}}. \quad (16)$$

Here, G^* denotes the underlying graph that encodes the true, but unknown, conditional dependence structure. Within the Bayesian model averaging framework, these probabilities provide a valuable summary of the explored graph space, highlighting the relative importance of all edges. They are commonly used for graph selection based on a threshold $0 < v < 1$, with a typical choice of $v = 0.5$ leading to the estimated graph $\hat{G} = (V, \hat{E})$, where $\hat{E} = \{e = (i, j) \mid \hat{P}_e \geq 0.5\}$. For graph estimation, we recommend this median-probability approach, as suggested by Barbieri and Berger (2004), over selecting the graph with the highest posterior probability.

Note that Algorithm (1) relies on the local approximation in (9), which is exact for decomposable graphs. Therefore, the algorithm can be adapted specifically for use with decomposable graphs through two straightforward modifications. First, the initial graph $G^{(0)}$ must be decomposable (for example, the empty graph satisfies this condition). Second, before computing the birth and death rates (lines 4 and 5 of the algorithm), we must check whether the proposed graph G' remains decomposable, accepting only moves that preserve decomposability.

3.3 Reversible Jump MCMC Algorithm

To sample from the pseudo-posterior (11), we present an alternative to BD-MPL (Algorithm 1). Specifically, we integrate the MPL approach with the reversible jump MCMC algorithm (Green, 1995), a discrete-time method that explores the graph space by proposing edge additions or deletions. In each iteration, the reversible jump MCMC algorithm proposes a new graph G' by adding or deleting an edge from the current graph G . The acceptance probability for the proposed move, which ensures that the Markov chain has the correct stationary distribution, is

$$\alpha(G, G') = \min \left\{ \frac{P(G'|\mathbf{X})q(G'|G)}{P(G|\mathbf{X})q(G|G')}, 1 \right\}, \quad (17)$$

where $q(G'|G)$ is the proposal probability of transitioning from graph G to graph G' . Clearly, $q(G'|G) = 0$ when G' and G are not neighbors. When G' and G are neighbors, we assume a uniform proposal distribution. For neighboring graphs, we adopt a uniform proposal distribution, setting $q(G'|G) = q(G|G') = 1/nb_{\max}$, where $nb_{\max} = p(p-1)/2$ is the total number of graphs differing from G by one edge (for alternative proposals, see Dobra et al. 2011; van den Boom et al. 2022). With this setup, $\alpha(G, G')$ aligns with the birth-death rate as defined in (14), highlighting the connection between the reversible jump MCMC and birth-death MCMC algorithms.

Our proposed reversible jump algorithm is abbreviated to RJ-MPL and the pseudo-code for this algorithm is described in Algorithm 2. Similar to Algorithm 1, we implement this algorithm in C++ and ported it to R. It is available within the R package `BDgraph` (Mohammadi et al., 2024) as `bdgraph.mpl()` function.

Algorithm 2: RJ-MPL search algorithm.

Input: Data $\mathbf{X}$ and an initial graph $G_0 = (V, E)$.
Output: Samples from the posterior distribution (1).
1 Calculate in parallel the MPL of each node given G_0 by (9);
2 **for** S iterations **do**
3 Draw a proposal graph by selecting an edge to flip;
4 Calculate the acceptance probability by (17) and update the graph;
5 Update the MPL for the pair of nodes associated with the flipped edge.
6 **end**

The output of Algorithm 2 consists of a set of S sampled graphs, $\{G^{(1)}, \dots, G^{(S)}\}$, representing the posterior graph space. These samples can be leveraged for various types of inference and model uncertainty quantification by applying (15), where $W^{(s)} = 1$ for $s \in \{1, \dots, S\}$. For example, the sampled graphs can be used to calculate the estimated edge-inclusion probabilities ($\hat{P}_e$) using (16), where $W^{(s)} = 1$ for $s \in \{1, \dots, S\}$. Similar to the BD-MPL approach discussed in Section 3.2, $\hat{P}_e$ provides a means to quantify model uncertainty and can also be employed for model selection.

3.4 Precision Matrix Estimation

The BD-MPL and RJ-MPL algorithms (Algorithms 1 and 2) are designed to handle large-scale problems by recovering the underlying graph structure from the data. In practical applications, it may also be necessary to estimate the precision matrix. Here, we present two approaches for estimating the precision matrix using the graph samples generated by the BD-MPL and RJ-MPL algorithms.

One approach is first to estimate the underlying graph structure using the edge inclusion probabilities (16) derived from the BD-MPL or RJ-MPL algorithms. Specifically, we can obtain the estimated graph $\hat{G} = (V, \hat{E})$ where $\hat{E} = \{e = (i, j) \mid \hat{P}_e \geq 0.5\}$, and $\hat{P}_e$ represents the estimated edge-inclusion probabilities (16). This estimated graph can then be used to sample from the precision matrix using the G -Wishart distribution (4), where $\mathbf{K}|\hat{G} \sim W_{\hat{G}}(b + n, \mathbf{D} + \mathbf{U})$, representing the posterior distribution of the precision matrix. This step can be performed using the sampling algorithm developed by Lenkoski (2013), which is implemented in the R package **BDgraph** (Mohammadi et al., 2024) as `rgwish()` function. The precision matrix can then be estimated as the mean of the sampled matrices. It is important to emphasize that the estimated precision matrix will always be positive definite, as the sampling algorithm by Lenkoski (2013) guarantees positive definiteness.

Another approach is to use the sampled graphs $\{G^{(1)}, \dots, G^{(S)}\}$ generated by the BD-MPL or RJ-MPL algorithms, and then sample the corresponding precision matrices $\{\mathbf{K}^{(1)}, \dots, \mathbf{K}^{(S)}\}$ from $W_G(b + n, \mathbf{D} + \mathbf{U})$, similar to the first approach. Note that, this method may not be feasible for very large-scale graphs, as saving all the sampled graphs can lead to memory issues; see, for example, Mohammadi and Wit (2019, Appendix).

4 Theoretical Properties

Here, we provide theoretical results that validate the use of the pseudo-posterior $\tilde{P}(G \mid \mathbf{X})$ in (11) as an approximation to the true posterior $P(G \mid \mathbf{X})$ in (1). Specifically, Lemma 4.1 establishes that the pseudo-posterior contracts around the true graph. Lemma 4.2 shows that both proposed algorithms converge to the target pseudo-posterior. Most importantly, Theorem 4.3 demonstrates that any MCMC algorithm targeting the pseudo-posterior, including RJ-MPL and BD-MPL, yields a consistent estimator of the graph. These results require only the assumption of Gaussianity and impose no restrictions on dimensionality, minimal signal strength, or hyperparameter choices.

We start with the following lemma, which establishes posterior contraction. It states that the posterior mass assigned to the true sparsity pattern (the true graph G^*) converges to one in probability as n tends to infinity. Equivalently, the posterior probability of graphs G that differ from the true graph G^* approaches zero as $n \rightarrow \infty$.

Lemma 4.1 (Posterior Contraction). *Let $\mathbf{X} = (\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)})^T$ be an $n \times p$ data matrix, where each independent observation $\mathbf{X}^{(k)}$, for $k \in \{1, \dots, n\}$, is distributed as $\mathcal{N}_p(\mathbf{0}, (\mathbf{K}^*)^{-1})$. Let $G^* = (V, E^*)$ denote the true underlying graph that encodes*

the conditional independence structure implied by $\mathbf{K}^*$. Then, as $n \rightarrow \infty$, the pseudo-posterior distribution $\tilde{P}(G|\mathbf{X})$ defined in (11) satisfies

$$\tilde{P}(G^*|\mathbf{X}) \rightarrow 1 \quad \text{in probability.}$$

Proof. Based on Equation (11) we have

$$\tilde{P}(G^*|\mathbf{X}) \propto P(G^*) \prod_{j=1}^p \tilde{P}(\mathbf{X}_j|\mathbf{X}_{nb^*(j)}, G^*),$$

where $nb^*(j)$ denotes the set of neighbors of node j and these neighbors together uniquely define the true graph G^* . Now, considering an arbitrary graph G' (not equal to G^*) with corresponding neighbors $nb'(j)$ of node $j \in \{1, \dots, p\}$, we have

$$\frac{\tilde{P}(G^*|\mathbf{X})}{\tilde{P}(G'|\mathbf{X})} = \frac{P(G^*)}{P(G')} \times \prod_{j=1}^p \frac{\tilde{P}(\mathbf{X}_j|\mathbf{X}_{nb^*(j)}, G^*)}{\tilde{P}(\mathbf{X}_j|\mathbf{X}_{nb'(j)}, G')}.$$

For all $j \in \{1, \dots, p\}$, as $n \rightarrow \infty$, we can derive

$$\log \left(\frac{\tilde{P}(\mathbf{X}_j|\mathbf{X}_{nb^*(j)}, G^*)}{\tilde{P}(\mathbf{X}_j|\mathbf{X}_{nb'(j)}, G')} \right) \rightarrow \infty$$

in probability; this result follows directly from Leppä-aho et al. (2017, Theorem 2, Lemma 1, and 2). Essentially, this means that by using pseudo-posterior (11), the true neighbors $nb^*(j)$ are preferred over any other set $nb'(j)$ as the number of observations n approaches infinity. Considering this, as $n \rightarrow \infty$,

$$\frac{\tilde{P}(G^*|\mathbf{X})}{\tilde{P}(G'|\mathbf{X})} \rightarrow \infty,$$

in probability. The convergence then follows. $\square$

The following lemma establishes that, as the number of MCMC iterations S tends to infinity, the BD-MPL and RJ-MPL sampling algorithms converge to the pseudo-posterior distribution $\tilde{P}(G|\mathbf{X})$.

Lemma 4.2 (Convergence). *Let $\{G^{(1)}, \dots, G^{(S)}\}$ denote the Markov chain generated by either the BD-MPL or RJ-MPL algorithm. As the number of iterations S approaches infinity, the Markov chain converges to the target pseudo-posterior distribution $\tilde{P}(G|\mathbf{X})$ defined in (11). Furthermore, for any function $f : \mathcal{G}_p \rightarrow \mathbb{R}$, we have*

$$E(f(G)|\mathbf{X}) = \lim_{S \rightarrow \infty} \frac{\sum_{s=1}^S W^{(s)} f(G^{(s)})}{\sum_{s=1}^S W^{(s)}}, \quad (18)$$

where $\{W^{(1)}, \dots, W^{(S)}\}$ are the waiting times in the BD-MPL algorithm. For the RJ-MPL algorithm, these waiting times are equal to one.

Proof. Convergence requires three conditions: irreducibility, aperiodicity, and the balance condition, ensuring that the Markov chain has a well-defined stationary distribution (Tierney, 1994). Both BD-MPL and RJ-MPL naturally satisfy irreducibility and aperiodicity. While detailed balance is a sufficient condition but not necessary for general balance, it is commonly imposed in practical sampler design (Green, 1995). For Algorithm 1, detailed balance holds by construction, as ensured by the birth-death rates defined in (14); see Dobra and Mohammadi (2018, Theorem 5.1) for further details. Similarly, Algorithm 2 satisfies these conditions through the acceptance probability defined in (17) (Green, 1995). Since both BD-MPL and RJ-MPL converge to the target posterior distribution, equation (18) follows directly from the Rao-Blackwellized estimator (Cappé et al., 2003, Section 2.5). $\square$

An immediate consequence of Lemma 4.2 is the convergence of the estimated edge-inclusion probability $\hat{P}_e$, as defined in (16), to the true pseudo-posterior edge-inclusion probability P_e . Specifically, for all $e \in \{(i, j) \mid 1 \leq i < j \leq p\}$, we have

$$\lim_{S \rightarrow \infty} \hat{P}_e = P_e = \sum_{G \in \mathcal{G}_p} \mathbf{1}(e \in G) \tilde{P}(G|\mathbf{X}).$$

The following theorem presents the main result. It involves the graph $\hat{G}$ obtained by thresholding the edge inclusion probabilities (16) at some threshold $0 < v < 1$, so that $\hat{G} = (V, \hat{E})$ with $\hat{E} = \{e = (i, j) \mid \hat{P}_e \geq v\}$. We prove that, for any fixed threshold v , the estimated graph $\hat{G}$ converges in probability to the true graph G^* as both the number of observations and the number of MCMC iterations tend to infinity. In other words, the edge inclusion probabilities asymptotically converge to either 0 or 1, accurately reflecting the true edge structure of G^* .

Theorem 4.3 (Selection Consistency). *Let $\hat{P}_e$ denote the estimated edge inclusion probabilities from Equation (16), obtained via an MCMC sampling algorithm (such as BD-MPL or RJ-MPL) with S iterations targeting the pseudo-posterior distribution (11). Define the estimated graph $\hat{G} = (V, \hat{E})$, where an edge $e = (i, j)$ is included in $\hat{E}$ if $\hat{P}_e \geq v$ for some fixed threshold $0 < v < 1$. Then, as the number of observations $n \rightarrow \infty$ and the number of MCMC iterations $S \rightarrow \infty$,*

$$P(\hat{G} = G^*) \rightarrow 1 \quad \text{in probability.}$$

Proof. Due to Lemma 4.2, for all $e \in \{(i, j) \mid 1 \leq i < j \leq p\}$, we have

$$\lim_{S \rightarrow \infty} \hat{P}_e = P_e = \sum_{G \in \mathcal{G}_p} \mathbf{1}(e \in G) \tilde{P}(G|\mathbf{X}).$$

Considering $P_e \geq \tilde{P}(G^*|\mathbf{X})$ for all $e \in G^*$ and $P_e \leq 1 - \tilde{P}(G^*|\mathbf{X})$ for all $e \notin G^*$, we have

$$\begin{aligned} P(\hat{G} = G^*) &= P(P_e \geq v \quad \forall e \in G^* \text{ and } P_e < v \quad \forall e \notin G^*) \\ &\geq P(\tilde{P}(G^*|\mathbf{X}) \geq v \text{ and } 1 - \tilde{P}(G^*|\mathbf{X}) < v) \\ &= P(\tilde{P}(G^*|\mathbf{X}) \geq \max(v, 1 - v)). \end{aligned}$$

The result follows from Lemma 4.1. $\square$

We note that, in connection with the theoretical results presented in this section, Jalali et al. (2020, Theorem 1) also established posterior contraction and selection consistency for their B-CON method, which combines a spike-and-slab prior on the precision matrix with a generalized likelihood approximation to the Gaussian likelihood. Three key differences distinguish their results from ours. First, their framework requires structural conditions on the true precision matrix, including a minimal signal size and bounded eigenvalues (Jalali et al., 2020, Assumptions 3–4), whereas our approach imposes no such constraints. Second, their results are tied to the specific graph prior in (2) with a fixed value of β (Jalali et al., 2020, Assumption 5), while ours hold for any prior on the graph. Third, our results assume Gaussian data, whereas B-CON requires only that the data-generating distribution has sub-Gaussian tails (Jalali et al., 2020, Assumption 2).

Theorem 4.3 shows that an MCMC algorithm targeting the pseudo-posterior recovers the true graph as the sample size and number of iterations grow. We next evaluate its finite-sample performance through simulations against state-of-the-art Bayesian structure learning methods.

5 Simulation Study

We assess the accuracy and computational efficiency of BD-MPL (Algorithm 1) and RJ-MPL (Algorithm 2) with three state-of-the-art Bayesian approaches.¹ The first is the birth-death MCMC algorithm (BD) introduced by Mohammadi and Wit (2015) and Mohammadi et al. (2023). The second method, SS, is established by Wang (2015), and it employs a block Gibbs sampler based on the spike-and-slab prior distribution. The third is the B-CON approach developed by Jalali et al. (2020), which uses a generalized likelihood function together with a spike-and-slab prior distribution.

The simulation study covers small ($p = 10$), medium ($p = 100$), and large-scale ($p = 1000$) graphs, considering three structural types. The first type, **Random**, consists of graphs in which edges are randomly sampled without replacement. The second type, **Cluster**, contains either two clusters (for $p \in 10, 100$) or eight clusters (for $p = 1000$), with each cluster following the structure of a **Random** graph. The third type, **Scale-free**, comprises spanning trees generated using the B-A algorithm of Albert and Barabási (2002). For both **Random** and **Cluster** graphs, we examine ‘*sparse*’ and ‘*dense*’ variants. To accurately represent these graphs, we determine the number of edges n_e using $\max(ap, bp(p-1)/2)$, with $a = 0.5$ and $b = 0.5\%$ for sparse graphs, and $a = 2$ and $b = 5\%$ for dense graphs. The edge densities of all the simulated graph types are reported in the Supplementary Material (Mohammadi et al., 2025). For the number of observations n , we follow a logarithmic relation between n and p as suggested by posterior contraction rates (Sagar et al., 2024, Theorem 4.6). We select $n = 20 \log(p)$ for ‘few’ observations and $n = 350 \log(p)$ for ‘many’ observations. An exception is made for $p = 1000$, where $n = 20 \log(p) = 60$ was too low for all algorithms to provide meaningful results, so we chose $n = 400$.

¹For comparisons with frequentist methods such as the graphical lasso (Friedman et al., 2008), we refer the reader to Vogels et al. (2024).

In each simulated graph G , the precision matrix $\mathbf{K}$ was generated from the G -Wishart distribution $W_G(3, \mathbf{I}_p)$. For $p \in \{10, 100\}$, we generated 16 graphs along with their corresponding precision matrices, while for $p = 1000$, we obtained 8 such pairs. Subsequently, for each pair G and $\mathbf{K}$, we sampled n data points from the p -dimensional Gaussian distribution $\mathcal{N}_p(\mathbf{0}, \mathbf{\Sigma})$ with mean zero and covariance matrix $\mathbf{\Sigma} = \mathbf{K}^{-1}$. The data was generated using the `bdgraph.sim()` function from the R package `BDgraph`.

To ensure fair comparisons in computing time, all five algorithms were coded in C++ and then ported to R, using the same routines wherever feasible. Following the approach of Wang (2015), the hyperparameters for the SS method were chosen as $\epsilon = 0.02$, $v = 2$, and $\lambda = 1$. Each MCMC run started with an empty graph. For the prior on the graph, we use the distribution (2) with prior density β . In real-life cases, one can use domain knowledge to select an appropriate value for β . In a simulation study, however, such prior knowledge is not available. It is therefore common to set β to either the uninformative value $\beta = 0.5$ (Gan et al., 2019) or to a function that decreases with p , for example $\beta = 2/(p - 1)$ (van den Boom et al., 2022). Here, we opt for a middle ground and set $\beta = 0.2$. For the B-CON method, we used $\beta = 0.5$ and initialized with a full graph, consistent with the authors’ original setup, since their provided code (Jalali et al., 2020) does not allow specifying either the initial graph or β . The BD-MPL and RJ-MPL were implemented using the `BDgraph` R package (Mohammadi et al., 2024), and the SS method utilized the `ssgraph` R package (Mohammadi, 2022).

The methods are evaluated based on graph recovery accuracy and computational cost. To assess accuracy, we compute edge inclusion probabilities from the MCMC-sampled graphs $\{G^{(1)}, \dots, G^{(S)}\}$ using (16). Five accuracy metrics are considered, with the Area Under the Precision-Recall Curve (AUC-PR) (Davis and Goadrich, 2006) highlighted here. The remaining metrics are provided in the Supplementary Material. AUC-PR is particularly useful for evaluating performance on imbalanced datasets, such as sparse graphs. The AUC-PR curve plots precision (the proportion of true positive edges among all predicted edges) against recall (the proportion of actual edges correctly identified) at various thresholds. Figure 2 presents AUC-PR scores over time for two scenarios: the left plot shows results for medium-scale graphs ($p = 100$, $n = 700$, *sparse Cluster* graph), while the right plot corresponds to large-scale graphs ($p = 1000$, $n = 1050$, *dense Cluster* graph). Additional instances are provided in the Supplementary Material. For the medium-scale problem, all methods except BD converge rapidly. In the large-scale setting, BD-MPL converges significantly faster in graph recovery precision compared to the other methods.

To compare algorithms, we require a measure of computational cost. Ideally, this would capture the time needed for an MCMC chain to meet a formal convergence criterion. However, defining such a criterion for large-scale, MCMC algorithms is challenging. Standard diagnostics such as the Gelman–Rubin statistic (Gelman and Rubin, 1992), effective sample size, or trace plots (Figure 3) require storing entire chains (or multiple chains), which can be prohibitively memory-intensive at scale. We therefore adopt a more practical approach, stopping the chains when a relevant performance metric — here the AUC-PR — shows no meaningful improvement. Following Vogels et al. (2024), we define computational cost as the time required for the AUC-PR to come within 0.01

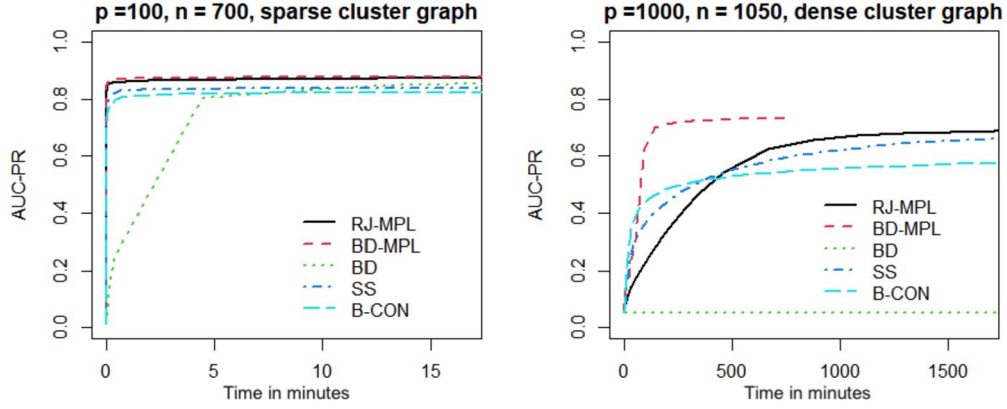


Figure 2: The convergence of AUC-PR scores over time for all algorithms (RJ-MPL, BD-MPL, BD, SS, B-CON). The left plot represents the instance with $p = 100$, $n = 700$ for the sparse Cluster graph, while the right plot corresponds to $p = 1000$, $n = 1050$ for the dense Cluster graph.

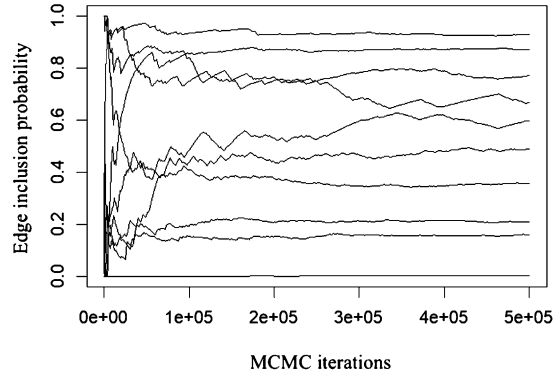


Figure 3: Edge inclusion probabilities across MCMC iterations for 10 randomly selected edges, created by the BD-MPL algorithm on the human gene dataset ($p = 100$).

of its final iteration value. While this measure is inherently subjective, it serves as a useful indicator of the runtime needed for an algorithm’s output quality to stabilize rather than a formal proof of convergence. Table 1 summarizes the results. For $p = 10$, all algorithms converge in under a minute with similar costs. At $p = 100$, RJ-MPL, BD-MPL, SS, and B-CON converge within seconds to minutes, whereas BD requires several hours. For $p = 1000$, BD fails to converge within five days, while BD-MPL, RJ-MPL, SS, and B-CON converge within hours to days, with BD-MPL running up to ten times faster than the others.

Table 2 reports the AUC-PR scores for all methods. For small ($p = 10$) and medium

p	Graph	Density	n	RJ-MPL	BD-MPL	BD	SS	B-CON
1000	Random	Sparse	400	2494	41	-	531	972
		Sparse	1050	1998	96	-	543	418
		Dense	400	2706	350	-	3012	3731
		Dense	1050	2434	1065	-	3297	2099
	Cluster	Sparse	400	2212	33	-	436	975
		Sparse	1050	1794	74	-	489	615
		Dense	400	2509	283	-	2386	3681
		Dense	1050	3003	396	-	2654	1711
	Scale-free	Sparse	400	4104	43	-	339	1808
		Sparse	1050	4053	144	-	1281	1508
100	Random	Sparse	40	6	2	103	0	7
		Sparse	700	0	0	34	2	3
		Dense	40	2	0	101	0	5
		Dense	700	2	0	72	1	3
	Cluster	sparse	40	6	3	86	1	6
		Sparse	700	1	1	45	1	4
		Dense	40	0	0	115	0	4
		Dense	700	1	0	78	1	1
	Scale-free	Sparse	40	2	1	103	0	5
		Sparse	700	2	1	67	1	4

Table 1: Computational cost (T) in minutes until AUC-PR convergence for various instances. T represents the average time until AUC-PR convergence, based on 16 replications for $p \in \{10, 100\}$ and 8 replications for $p = 1000$. The table excludes the $p = 10$ case since the computational time for all algorithms was less than one minute. A “-” indicates that an algorithm did not converge within five days. For each setting, the best-performing algorithm is highlighted in **bold**.

($p = 100$) graphs, performances are comparable, with BD showing slightly higher values. For large graphs ($p = 1000$), differences are pronounced: BD yields near-zero AUC-PR, reflecting its inability to handle large-scale problems within a reasonable runtime (five days in this study). In contrast, RJ-MPL and BD-MPL achieve the highest AUC-PR, followed by SS, B-CON, and BD. For example, in the dense **Cluster** graph with $p = 1000$ and $n = 1050$, BD-MPL reaches the highest AUC-PR (over 0.7) in under two hours, whereas SS and B-CON require more than 27 hours, and BD fails to converge within five days. Together with Table 1, these results indicate that BD-MPL delivers the best overall performance for $p = 1000$, combining both computational cost and accuracy.

In summary, for large-scale problems (graphs with $p = 1000$), the BD-MPL algorithm achieves significantly lower computational costs while maintaining high accuracy compared to state-of-the-art methods (BD, SS, and B-CON). BD-MPL also has several advantages over RJ-MPL, as it converges faster (Figure 2) and its computationally intensive components can be parallelized (Section 3.2). This makes BD-MPL particularly

p	Graph	Density	n	RJ-MPL	BD-MPL	BD	SS	B-CON
1000	Random	Sparse	400	0.67	0.70	0.01	0.66	0.66
		Sparse	1050	0.79	0.81	0.00	0.72	0.79
		Dense	400	0.39	0.43	0.05	0.42	0.31
		Dense	1050	0.56	0.61	0.05	0.56	0.50
	Cluster	Sparse	400	0.70	0.72	0.01	0.67	0.68
		Sparse	1050	0.81	0.83	0.01	0.74	0.78
		Dense	400	0.55	0.59	0.05	0.55	0.48
		Dense	1050	0.71	0.74	0.05	0.7	0.58
	Scale-free	Sparse	400	0.66	0.68	0.00	0.68	0.62
		Sparse	1050	0.78	0.8	0.00	0.76	0.72
100	Random	Sparse	40	0.50	0.50	0.54	0.55	0.50
		Sparse	700	0.89	0.89	0.89	0.84	0.86
		Dense	40	0.37	0.37	0.40	0.43	0.38
		Dense	700	0.86	0.86	0.86	0.79	0.82
	Cluster	Sparse	40	0.49	0.49	0.52	0.53	0.49
		Sparse	700	0.87	0.88	0.88	0.84	0.83
		Dense	40	0.40	0.39	0.43	0.45	0.41
		Dense	700	0.87	0.87	0.87	0.80	0.83
	Scale-free	Sparse	40	0.41	0.41	0.47	0.47	0.41
		Sparse	700	0.87	0.87	0.89	0.82	0.77
10	Random	Sparse	20	0.52	0.52	0.54	0.52	0.52
		Sparse	350	0.91	0.91	0.92	0.89	0.89
		Dense	20	0.68	0.69	0.69	0.68	0.68
		Dense	350	0.94	0.93	0.93	0.93	0.91
	Cluster	Sparse	20	0.50	0.50	0.50	0.50	0.48
		Sparse	350	0.84	0.85	0.85	0.84	0.86
		Dense	20	0.81	0.81	0.83	0.81	0.77
		Dense	350	0.95	0.95	0.95	0.94	0.94
	Scale-free	Sparse	20	0.61	0.61	0.65	0.65	0.60
		Sparse	350	0.90	0.91	0.91	0.89	0.87

Table 2: AUC-PR scores of the algorithms for different instances. The values are averages over 16 replications for $p \in \{10, 100\}$ and over 8 replications for $p = 1000$. For each setting, the best-performing algorithm is highlighted in **bold**.

well suited for large-scale problems with $p = 1000$ or more. For medium-scale problems (around 100 nodes), BD and BD-MPL have similar and higher accuracy than other methods (Table 2), but BD-MPL is more efficient, reducing computational cost from one–two hours to under three minutes (Table 1). For small-scale problems (around 10 nodes), BD achieves slightly higher accuracy in most cases. Overall, we recommend BD-MPL for large- and medium-scale problems and BD or SS for small-scale problems.

6 Applications

We apply our proposed BD-MPL and RJ-MPL algorithms to two real-world data sets: a medium-scale data set (with 100 variables) in Section 6.1 and a large-scale data set (with 623 variables) in Section 6.2. We report results from the BD-MPL algorithm, as both algorithms target the same pseudo-posterior distribution and produce virtually identical outputs when run for a sufficient number of iterations. For both datasets, our approach provides point estimates of the conditional dependence structure. Moreover, we illustrate how the method facilitates uncertainty quantification over graph structures and key graph characteristics, extending beyond traditional point estimation.

6.1 Application to Human Gene Expression

We apply the BD-MPL algorithm to infer a human gene network. The dataset consists of genetic data for $p = 100$ genes from $n = 60$ unrelated individuals and is available in the **BDgraph** R package (Mohammadi et al., 2024). Detailed information on data collection can be found in Stranger et al. (2007) and Bhadra and Mallick (2013). Several other Bayesian structure learning methods have also been applied to this dataset (Mohammadi and Wit, 2015; Li et al., 2019; Mohammadi and Wit, 2019; van den Boom et al., 2022; Vogels et al., 2024).

Genes are specific sequences of DNA that play a critical role in the functioning of organisms. Gene expression is the process through which these genes produce proteins, which then influence various biological functions. Some proteins have direct effects on the organism, such as initiating the breakdown of food. Others serve a regulatory role by *activating* other genes, leading to the production of additional proteins that further activate more genes, creating a cascade of interactions. These activation relationships can be represented in a gene network, where each node corresponds to a gene, and each edge signifies an activation relationship between two genes. Mapping and understanding these gene networks are vital for elucidating disease susceptibility, ultimately contributing to advancements in treatment and public health (Stranger et al., 2007).

Before applying the algorithm, we normalized the dataset using a transformation based on cumulative distribution functions (Liu et al., 2009), implemented via the `bdgraph.npn()` function in the **BDgraph** package. The initial parameter settings matched those used in the simulation study described in Section 5. For the graph prior defined in (2), we set $\beta = 0.2$, building on the results of earlier studies (van den Boom et al., 2022). The MCMC procedure was initialized at the empty graph. To determine the number of MCMC iterations, we use a trace plot (Figure 3). Trace plots are a common tool to determine MCMC convergence, see for example van den Boom et al. (2022). They depict the edge inclusion probabilities for a set of randomly selected edges across MCMC iterations. Based on Figure 3, we observe that the BD-MPL algorithm typically converges after approximately 500,000 iterations. We therefore set the number of iterations to one million, half of which are burn-in iterations. The total computation time remained under 10 minutes on a standard desktop computer.

Figure 4 (left) presents a heatmap of the estimated edge inclusion probabilities (16) obtained from the BD-MPL algorithm. Figure 4 (right) presents a point estimate of

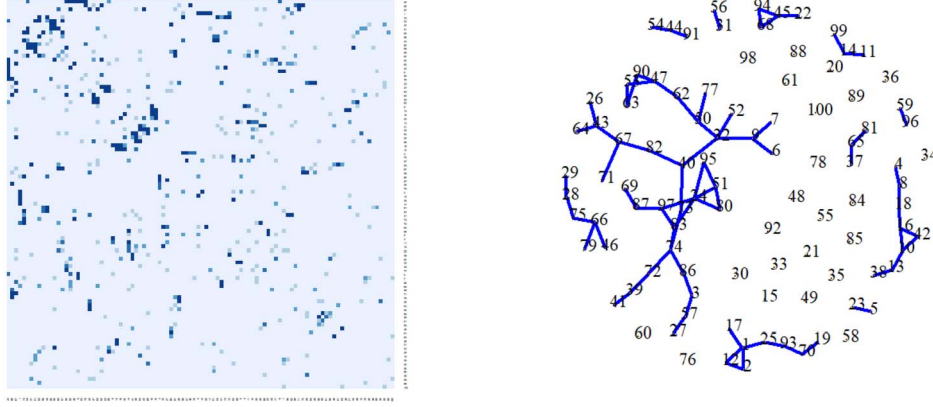


Figure 4: (Left) Heatmap of estimated edge inclusion probabilities from the BD-MPL algorithm for the human gene dataset ($p = 100$), with values ranging from 0 (gray) to 1 (dark blue). (Right) Estimated graph from the BD-MPL algorithm for the same dataset, showing only edges with inclusion probabilities greater than 0.9.

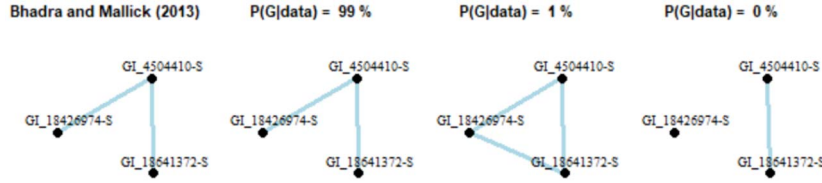


Figure 5: The structure estimated by Bhadra and Mallick (2013, Figure 4) among three genes part of the major histocompatibility complex (left), alongside the three most likely structures estimated by the BD-MPL.

the gene network obtained using a threshold of 0.9 on the edge inclusion probabilities. We use this relatively high threshold to prevent the plot from becoming overly dense and difficult to interpret. However, for optimal point estimation, a threshold of 0.5 is generally recommended, as suggested by Barbieri and Berger (2004). The inferred gene network displays structural patterns consistent with those reported in Bhadra and Mallick (2013, Figure 4).

In addition to estimating edge inclusion probabilities, the BD-MPL algorithm can quantify uncertainty in various graph structures. Specifically, they estimate the expected value $E(f(G)|\mathbf{X})$ for any function $f(G)$ using (15). Bhadra and Mallick (2013) examined the conditional dependence structure among three genes in the major histocompatibility complex, a DNA region involved in immune system function. The BD-MPL algorithm confirms this as the most likely structure, estimating a 99% probability given the data (Figure 5). Additionally, Bhadra and Mallick (2013) identified a conditional dependence structure among six genes, five of which are located on the Y chromosome.

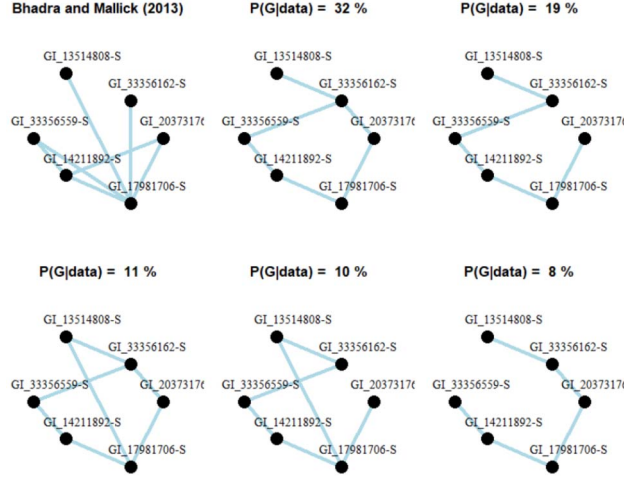


Figure 6: The structure estimated by Bhadra and Mallick (2013, Figure 4) for six selected genes (top left), alongside the five most likely structures estimated by the BD-MPL.

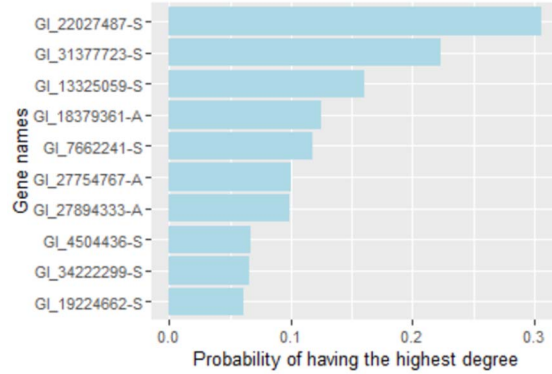


Figure 7: The ten most connected genes in the human gene dataset ($p = 100$), along with their corresponding posterior probabilities of being the gene with the highest degree.

The BD-MPL algorithm estimates a near-zero probability for this structure and instead suggests alternative likely structures for these six genes (Figure 6). Another choice of $f(G)$ allows for identifying highly connected nodes in a network, a measure known as degree centrality, which is particularly useful for analyzing gene networks (Koschützki and Schreiber, 2008). Bhadra and Mallick (2013) found that the gene with the highest degree is GI-22027487-S, which regulates iron and copper levels (Xu et al., 2022). Our results confirm that this gene is the most likely to have the highest degree, with a posterior probability of 30%. Figure 7 presents the highest degree probabilities for the other nodes.

6.2 Application to Gene Expression in Immune Cells

This subsection presents an application of the BD-MPL algorithm on a large dataset with $p = 623$ variables and $n = 653$ observations, demonstrating its ability to estimate uncertainty across a wide range of graph structures. The dataset is the GSE15907 microarray dataset from Painter et al. (2011) and Desch et al. (2011), comprising gene expression data from 24,922 genes in 653 mouse immune cells, obtained from the Immunological Genome Project (Heng et al., 2008). It was previously analyzed by Chandra et al. (2024) using Bayesian methods. These immune cells play a crucial role in defending organisms against diseases, with genes producing proteins that either directly impact disease response or initiate cascades by activating other genes. These activation relationships form gene networks, where nodes represent genes and edges denote activation connections. Understanding these networks is essential for advancing research on immune system function and disease treatment strategies.

For data preparation, following Chandra et al. (2024), we apply a $\log_2$ transformation and retain the top 2.5% of genes with the highest variance. To normalize the dataset, we use a cumulative distribution function-based transformation, as described in Subsection 6.1. For the graph prior defined in (2), we select $\beta = 0.01$, based on the results of Chandra et al. (2024). To determine the number of MCMC iterations, creating a trace-plot as in Figure 3 requires excessive memory for large-scale instances. Instead, we look at results from our simulation study in Section 5 (see Table 7 in the Supplementary Material). For the case $p = 1000$, these results suggest that the number of MCMC iterations depends on the sparsity and the number of observations, with dense and low-observation instances requiring more MCMC iterations. In all cases though, AUC-PR convergence happens within 1.5 million MCMC iterations. To be on the safe side we run 4 million MCMC iterations, discarding the first half as burn-in. The total runtime is approximately 17 hours. All other initial parameters are consistent with those used in the simulation study. We provide a heatmap of edge inclusion probabilities estimated by the BD-MPL algorithm in the supplementary material. While thresholding these probabilities could produce inferred gene networks, we do not present the complete networks here because of their large size and limited interpretability. Instead, following Chandra et al. (2024), we focus on specific gene subsets documented in the literature as conditionally dependent. These include histone genes (e.g., *Hist1h1a*, *Hist1h1b*) (Wolffe, 2001), B-cell leukemia genes (e.g., *Bcl2a1a*, *Bcl2a1b*) (Chandra et al., 2024), leukocyte antigen genes (e.g., *Ly6c1*, *Ly6c2*) (Lee et al., 2013), and membrane-spanning 4A genes (e.g., *Ms4a1*, *Ms4a4c*) (Liang et al., 2001). Figure 8 highlights these subsets, showing edges with inclusion probabilities greater than 0.99. These results demonstrate that the BD-MPL algorithm successfully recovers known conditional dependencies within biologically relevant gene groups.

To illustrate the ability of BD-MPL to quantify uncertainty in specific graph structures, we focus on a group of genes encoding chemokine ligands (e.g., *Ccl3*, *Ccl5*) and chemokine receptors (e.g., *Ccr2*, *Ccr5*). These genes produce CC chemokines, which play a crucial role in immune responses (Raman et al., 2011). Figure 9 presents the four most likely conditional dependence structures among these genes. We also find

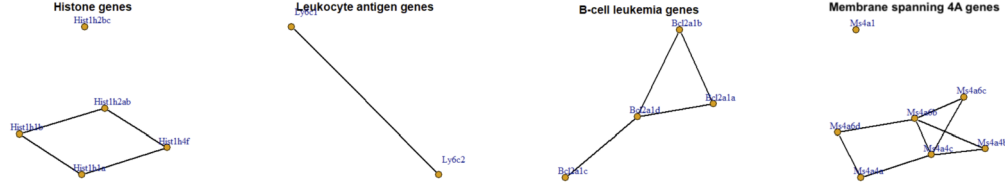


Figure 8: The networks for histone genes, leukocyte antigen genes, B-cell leukemia genes, and membrane-spanning 4A genes, estimated by the BD-MPL for the mice gene dataset ($p = 623$). All displayed edges have inclusion probabilities exceeding 99%.

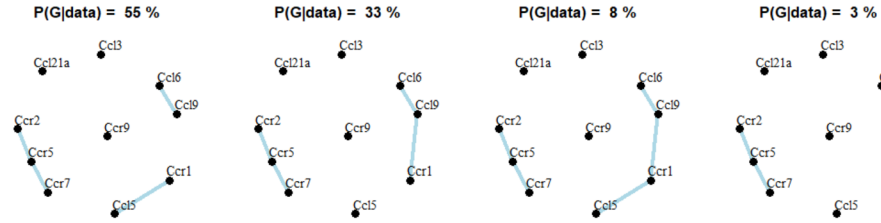


Figure 9: The most likely structures between chemokine ligand (Ccl) and chemokine receptor (Ccr) genes, estimated by the BD-MPL on the mice gene dataset ($p = 623$).

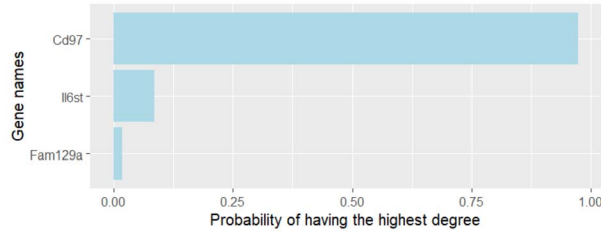


Figure 10: The three most connected genes in the mice gene dataset ($p = 623$), with their corresponding posterior probability of being the gene with the highest degree.

that the CD97 gene has the highest degree with a posterior probability of 97% (Figure 10). This aligns with the conclusion of Safaee et al. (2013) that the CD97 protein, encoded by this gene, is broadly expressed and plays diverse roles in the immune system.

7 Conclusion

We address the computational challenges of posterior sampling for GGMs in large-scale problems by introducing an MCMC-based framework that combines birth-death and reversible jump algorithms with the marginal pseudo-likelihood approach. The pro-

posed algorithms scale to large graph spaces, enabling parallel exploration for graphs with over 1,000 nodes, as demonstrated in our extensive simulation study. We provide theoretical guarantees, including posterior contraction and graph selection consistency. In addition to scalability and theoretical support, the proposed method offers reliable uncertainty quantification and the flexibility to incorporate prior knowledge about the graph structure, as illustrated in our application section.

Future research could enhance these methods along several directions. In Section 3.1, we employ a fractional prior to derive a closed-form analytical expression for the local components, see Equation (9). One potential enhancement is to tune the prior to improve inferential accuracy and improve the approximation of the true posterior. Alternatively, one could adopt the closed-form approximation for the ratio of normalizing constants proposed by Mohammadi et al. (2023). Another promising avenue is the exploration of alternative priors on the precision matrix. For instance, priors such as the spike-and-slab formulation used in Jalali et al. (2020) may offer improved structure learning and posterior inference.

The BD-MPL and RJ-MPL algorithms rely on the addition or removal of a single edge per MCMC iteration to explore the graph space. Their computational efficiency can be further improved by allowing multiple edge updates per iteration. For instance, for the BD-MPL algorithm, after computing all possible edge update probabilities from (13), one can select a fixed number of edges ($k > 1$) to update in each iteration. The choice of k can be guided by the computational time required per iteration. This strategy is implemented in the `BDgraph` package (Mohammadi et al., 2024) through the function `bdgraph.mpl()` using the option `jump = k`; see Dobra and Mohammadi (2018, Section 5.5) for further details.

Another promising direction is to adopt a blocking approach, as proposed by van den Boom et al. (2023) and Colombi et al. (2024). In particular, the block-Bernoulli prior introduced in van den Boom et al. (2023) enables joint updates of groups of edges by operating at the block level. Analogous to (9), local pseudo-likelihoods for blocks can be expressed in closed form using Equation (29) from Consonni and Rocca (2012). Since the proposed algorithm assumes that each block is either fully connected or empty, the corresponding block marginal likelihoods can be computed exactly.

Scalability is a major barrier to applying Bayesian inference in graphical models. In this work, we address this challenge by proposing a scalable Bayesian framework based on the marginal pseudo-likelihood approach for vanilla (single) GGMs. A natural extension of this framework is to the multiple GGMs setting Peterson et al. (2015). A related extension has been developed by Jalali et al. (2023), who proposed a Bayesian method for multiple GGMs using a jointly convex, regression-based pseudo-likelihood with a spike-and-slab prior. More recently, Avalos-Pacheco et al. (2025) applied a similar pseudo-likelihood framework to scale Bayesian inference for multiple Ising graphical models with binary data. For possible extensions of our proposed algorithms to multiple Ising or general discrete graphical models, we refer readers to Dobra and Mohammadi (2018, Section 5), which employs a birth-death MCMC algorithm with an MPL approach for multivariate discrete data.

Acknowledgments

We thank three anonymous referees and the Editor, Prof. Igor Pruenster, for their insightful suggestions and comments, which have significantly improved this article.

Supplementary Material

Supplementary Material for “Scalable Bayesian Structure Learning for Gaussian Graphical Models Using Marginal Pseudo-Likelihood” (DOI: [10.1214/25-BA1561SUPP](https://doi.org/10.1214/25-BA1561SUPP); .pdf). The supplementary material provides further detailed results of the simulation study in Section 5 and the applications in Section 6.

References

- Albert, R. and Barabási, A.-L. (2002). “Statistical mechanics of complex networks.” *Reviews of Modern Physics*, 74(1): 47. MR1895096. doi: <https://doi.org/10.1103/RevModPhys.74.47>. 15
- Atay-Kayis, A. and Massam, H. (2005). “A Monte Carlo method for computing the marginal likelihood in nondecomposable Gaussian graphical models.” *Biometrika*, 92(2): 317–335. MR2201362. doi: <https://doi.org/10.1093/biomet/92.2.317>. 6
- Atchadé, Y. F. (2019). “Quasi-Bayesian estimation of large Gaussian graphical models.” *Journal of Multivariate Analysis*, 173: 656–671. MR3959745. doi: <https://doi.org/10.1016/j.jmva.2019.03.005>. 2, 3
- Avalos-Pacheco, A., Lazzerini, A., Lupporelli, M., and Stingo, F. C. (2025). “Bayesian inference of multiple Ising models for heterogeneous public opinion survey networks.” *Journal of the Royal Statistical Society Series C: Applied Statistics*, 1–32. 25
- Barbieri, M. M. and Berger, J. O. (2004). “Optimal predictive model selection.” *Annals of Statistics*, 870–897. MR2065192. doi: <https://doi.org/10.1214/009053604000000238>. 10, 21
- Besag, J. (1975). “Statistical analysis of non-lattice data.” *Journal of the Royal Statistical Society: Series D (The Statistician)*, 24(3): 179–195. 2, 7
- Bhadra, A. and Mallick, B. (2013). “Joint high-dimensional Bayesian variable and covariance selection with an application to eQTL analysis.” *Biometrics*, 69: 447–457. MR3071063. doi: <https://doi.org/10.1111/biom.12021>. 20, 21, 22
- Cappé, O., Robert, C., and Rydén, T. (2003). “Reversible jump, birth-and-death and more general continuous time Markov chain Monte Carlo samplers.” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(3): 679–700. MR1998628. doi: <https://doi.org/10.1111/1467-9868.00409>. 9, 10, 14
- Carvalho, C. M. and Scott, J. G. (2009). “Objective Bayesian model selection in Gaussian graphical models.” *Biometrika*, 96(3): 497–512. MR2538753. doi: <https://doi.org/10.1093/biomet/asp017>. 3, 8

- Chandra, N. K., Müller, P., and Sarkar, A. (2024). “Bayesian scalable precision factor analysis for Gaussian graphical models.” *Bayesian Analysis*, 1(1): 1–29. 2, 23
- Cheng, Y. and Lenkoski, A. (2012). “Hierarchical Gaussian graphical models: Beyond reversible jump.” *Electronic Journal of Statistics*, 6: 2309–2331. MR3020264. doi: <https://doi.org/10.1214/12-EJS746>. 6
- Colombi, A., Argiento, R., Paci, L., and Pini, A. (2024). “Learning block structured graphs in Gaussian graphical models.” *Journal of Computational and Graphical Statistics*, 33(1): 152–165. MR4713950. doi: <https://doi.org/10.1080/10618600.2023.2210184>. 25
- Consonni, G. and Rocca, L. L. (2012). “Objective Bayes factors for Gaussian directed acyclic graphical models.” *Scandinavian Journal of Statistics*, 39(4): 743–756. MR3000846. doi: <https://doi.org/10.1111/j.1467-9469.2011.00785.x>. 3, 8, 25
- Davis, J. and Goadrich, M. (2006). “The relationship between precision-recall and ROC curves.” In *Proceedings of the 23rd International Conference on Machine Learning*, 233–240. New York. 16
- Desch, A., Randolph, G., Murphy, K., Kedl, R., Lahoud, M., Caminschi, I., Shortman, K., Henson, P., and Jakubzick, C. (2011). “CD103+ pulmonary dendritic cells preferentially acquire and present apoptotic cell-associated antigen.” *The Journal of Experimental Medicine*, 208: 1789–97. 23
- Dobra, A., Lenkoski, A., and Rodriguez, A. (2011). “Bayesian inference for general Gaussian graphical models with application to multivariate lattice data.” *Journal of the American Statistical Association*, 106(496): 1418–1433. MR2896846. doi: <https://doi.org/10.1198/jasa.2011.tm10465>. 6, 11
- Dobra, A. and Mohammadi, R. (2018). “Loglinear model selection and human mobility.” *The Annals of Applied Statistics*, 12(2): 815–845. MR3834287. doi: <https://doi.org/10.1214/18-AOAS1164>. 3, 9, 14, 25
- Drton, M. and Perlman, M. D. (2007). “Multiple testing and error control in Gaussian graphical model selection.” *Statistical Science*, 22(3): 430–449. MR2416818. doi: <https://doi.org/10.1214/088342307000000113>. 2
- Friedman, J., Hastie, T., and Tibshirani, R. (2008). “Sparse inverse covariance estimation with the graphical Lasso.” *Biostatistics*, 9(3): 432–441. 2, 15
- Gan, L., Narisetty, N. N., and Liang, F. (2019). “Bayesian regularization for graphical models with unequal shrinkage.” *Journal of the American Statistical Association*, 114(527): 1218–1231. MR4011774. doi: <https://doi.org/10.1080/01621459.2018.1482755>. 16
- Geiger, D. and Heckerman, D. (2002). “Parameter priors for directed acyclic graphical models and the characterization of several probability distributions.” *The Annals of Statistics*, 30(5): 1412–1440. MR1936324. doi: <https://doi.org/10.1214/aos/1035844981>. 8

- Gelman, A. and Rubin, D. B. (1992). “Inference from iterative simulation using multiple sequences.” *Statistical Science*, 7(4): 457–472. [16](#)
- Green, P. (1995). “Reversible jump Markov chain Monte Carlo computation and Bayesian model determination.” *Biometrika*, 82(4): 711–732. [MR1380810](#). doi: <https://doi.org/10.1093/biomet/82.4.711>. [6](#), [11](#), [14](#)
- Heng, T., Painter, M., Elpek, K., Lukacs-Kornek, V., Mauermann, N., Turley, S., Koller, D., Kim, F., Wagers, A., Asinowski, N., Davis, S., Fassett, M., Feuerer, M., Gray, D., Haxhinasto, S., Hill, J., Hyatt, G., Laplace, C., Leatherbee, K., and Kang, J. (2008). “The immunological genome project: Networks of gene expression in immune cells.” *Nature Immunology*, 9: 1091–1094. [23](#)
- Hinne, M., Lenkoski, A., Heskes, T., and van Gerven, M. (2014). “Efficient sampling of Gaussian graphical models using conditional Bayes factors.” *Stat*, 3(1): 326–336. [MR4027347](#). doi: <https://doi.org/10.1002/sta4.66>. [6](#)
- Jalali, P., Khare, K., and Michailidis, G. (2020). “B-CONCORD—A scalable Bayesian high-dimensional precision matrix estimation procedure.” *arXiv preprint arXiv:2005.09017*. [2](#), [3](#), [4](#), [15](#), [16](#), [25](#)
- Jalali, P., Khare, K., and Michailidis, G. (2023). “A Bayesian subset specific approach to joint selection of multiple graphical models.” *Statistica Sinica*, 33: 1–24. [MR4647050](#). [25](#)
- Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: Principles and techniques*. Cambridge, Massachusetts: MIT Press. [MR2816736](#). [1](#)
- Koschützki, D. and Schreiber, F. (2008). “Centrality analysis methods for biological networks and their application to gene regulatory networks.” *Gene Regulation and Systems Biology*, 2: 193–201. [22](#)
- Lauritzen, S. L. (1996). *Graphical models*, volume 17. U.K.: Clarendon: Oxford: Oxford University Press. [MR1419991](#). [1](#), [5](#)
- Leday, G. G. and Richardson, S. (2019). “Fast Bayesian inference in large Gaussian graphical models.” *Biometrics*, 75(4): 1288–1298. [MR4041830](#). doi: <https://doi.org/10.1111/biom.13064>. [2](#)
- Lee, P. Y., Wang, J.-X., Parisini, E., Dascher, C. C., and Nigrovic, P. A. (2013). “Ly6 family proteins in neutrophil biology.” *Journal of Leukocyte Biology*, 94(4): 585–594. [23](#)
- Lenkoski, A. (2013). “A direct sampler for G-Wishart variates.” *Stat*, 2(1): 119–128. [MR4027305](#). doi: <https://doi.org/10.1002/sta4.23>. [6](#), [12](#)
- Lenkoski, A. and Dobra, A. (2011). “Computational aspects related to inference in Gaussian graphical models with the G-Wishart prior.” *Journal of Computational and Graphical Statistics*, 20: 140–157. [MR2816542](#). doi: <https://doi.org/10.1198/jcgs.2010.08181>. [6](#)
- Leppä-aho, J., Johan, Roos, T., and Corander, J. (2017). “Learning Gaussian graphical models with fractional marginal pseudo-likelihood.” *International Journal of Approx-*

- imate Reasoning*, 83: 21–42. MR3614245. doi: <https://doi.org/10.1016/j.ijar.2017.01.001>. 3, 7, 8, 13
- Letac, G. and Massam, H. (2007). “Wishart distributions for decomposable graphs.” *The Annals of Statistics*, 35(3): 1278–1323. MR2341706. doi: <https://doi.org/10.1214/009053606000001235>. 5
- Li, Y., Craig, B. A., and Bhadra, A. (2019). “The graphical horseshoe estimator for inverse covariance matrices.” *Journal of Computational and Graphical Statistics*, 28(3): 747–757. MR4007755. doi: <https://doi.org/10.1080/10618600.2019.1575744>. 2, 20
- Liang, Y., Buckley, T. R., Tu, L., Langdon, S. D., and Tedder, T. F. (2001). “Structural organization of the human MS4A gene cluster on Chromosome 11q12.” *Immunogenetics*, 53: 357–368. 23
- Liu, H., Lafferty, J., and Wasserman, L. (2009). “The nonparanormal: semiparametric estimation of high dimensional undirected graphs.” *Journal of Machine Learning Research*, 10(80): 2295–2328. MR2563983. 20
- Meinshausen, N. and Bühlmann, P. (2006). “High-dimensional graphs and variable selection with the Lasso.” *The Annals of Statistics*, 34(3): 1436–1462. MR2278363. doi: <https://doi.org/10.1214/009053606000000281>. 2
- Mohammadi, A. and Wit, E. (2015). “Bayesian structure learning in sparse Gaussian graphical models.” *Bayesian Analysis*, 10(1): 109–138. MR3420899. doi: <https://doi.org/10.1214/14-BA889>. 2, 4, 6, 8, 15, 20
- Mohammadi, R. (2022). *ssgraph: Bayesian graph structure learning using spike-and-slab priors*. R package version 1.15. URL <https://CRAN.R-project.org/package=ssgraph> 16
- Mohammadi, R., Massam, H., and Letac, G. (2023). “Accelerating Bayesian structure learning in sparse Gaussian graphical models.” *Journal of the American Statistical Association*, 118(542): 1345–1358. MR4595499. doi: <https://doi.org/10.1080/01621459.2021.1996377>. 2, 4, 6, 15, 25
- Mohammadi, R., Wit, E., and Dobra, A. (2024). *BDgraph: Bayesian structure learning in graphical models using birth-death MCMC*. R package version 2.73. URL <https://CRAN.R-project.org/package=BDgraph> 4, 10, 11, 12, 16, 20, 25
- Mohammadi, R., Schoonhoven, M., Vogels, L., Birbil, Ş. I. (2025). “Supplementary Material for “Scalable Bayesian Structure Learning for Gaussian Graphical Models Using Marginal Pseudo-Likelihood”” *Bayesian Analysis*. doi: <https://doi.org/10.1214/25-BA1561SUPP>. 15
- Mohammadi, R. and Wit, E. C. (2019). “BDgraph: An R package for Bayesian structure learning in graphical models.” *Journal of Statistical Software*, 89(3): 1–30. 12, 20
- Painter, M., Davis, S., Hardy, R., Mathis, D., and Benoist, C. (2011). “Transcriptomes of the B and T lineages compared by multiplatform microarray profiling.” *The Journal of Immunology*, 186(5): 3047–3057. 23

- Pensar, J., Nyman, H., Niiranen, J., and Corander, J. (2017). “Marginal pseudo-likelihood learning of discrete Markov network structures.” *Bayesian Analysis*, 12(4): 1195–1215. MR3724983. doi: <https://doi.org/10.1214/16-BA1032>. 3
- Peterson, C., Stingo, F. C., and Vannucci, M. (2015). “Bayesian inference of multiple Gaussian graphical models.” *Journal of the American Statistical Association*, 110(509): 159–174. MR3338494. doi: <https://doi.org/10.1080/01621459.2014.896806>. 25
- Preston, C. J. (1976). “Special birth-and-death processes.” *Bulletin of the International Statistical Institute*, 46: 371–391. MR0474532. 8
- Raman, D., Sobolik, T., and Richmond, A. (2011). “Chemokines in health and disease.” *Experimental Cell Research*, 317: 575–89. 23
- Roverato, A. (2002). “Hyper inverse Wishart distribution for non-decomposable graphs and its application to Bayesian inference for Gaussian graphical models.” *Scandinavian Journal of Statistics*, 29(3): 391–411. MR1925566. doi: <https://doi.org/10.1111/1467-9469.00297>. 5
- Rue, H. and Held, L. (2005). *Gaussian Markov random fields: Theory and applications*. London: Chapman and Hall-CRC Press. MR2130347. doi: <https://doi.org/10.1201/9780203492024>. 1
- Safaei, M., Clark, A., Ivan, M., Oh, M., Bloch, O., Sun, M., Oh, T., and Parsa, A. (2013). “CD97 is a multifunctional leukocyte receptor with distinct roles in human cancers (Review).” *International Journal of Oncology*, 43. 24
- Sagar, K., Banerjee, S., Datta, J., and Bhadra, A. (2024). “Precision matrix estimation under the horseshoe-like prior–penalty dual.” *Electronic Journal of Statistics*, 18(1): 1–46. MR4693861. doi: <https://doi.org/10.1214/23-ejs2196>. 2, 15
- Scutari, M. (2013). “On the prior and posterior distributions used in graphical modelling.” *Bayesian Analysis*, 8(3): 505–532. MR3102220. doi: <https://doi.org/10.1214/13-BA819>. 5
- Stingo, F. and Marchetti, G. M. (2015). “Efficient local updates for undirected graphical models.” *Statistics and Computing*, 25: 159–171. MR3304918. doi: <https://doi.org/10.1007/s11222-014-9541-6>. 3
- Stranger, B. E., Nica, A. C., Forrest, M. S., Dimas, A., Bird, C. P., Beazley, C., Ingle, C. E., Dunning, M., Flicek, P., Koller, D., Montgomery, S., Tavaré, S., Deloukas, P., and Dermitzakis, E. T. (2007). “Population genomics of human gene expression.” *Nature Genetics*, 39: 1217–1224. 20
- Tierney, L. (1994). “Markov chains for exploring posterior distributions.” *The Annals of Statistics*, 22(4): 1701 – 1728. MR1329166. doi: <https://doi.org/10.1214/aos/1176325750>. 14
- Uhler, C., Lenkoski, A., and Richards, D. (2018). “Exact formulas for the normalizing constants of Wishart distributions for graphical models.” *The Annals of Statistics*, 46(1): 90–118. MR3766947. doi: <https://doi.org/10.1214/17-AOS1543>. 6

- van den Boom, W., Beskos, A., and De Iorio, M. (2022). “The G-Wishart weighted proposal algorithm: Efficient posterior computation for Gaussian graphical models.” *Journal of Computational and Graphical Statistics*, 31(4): 1215–1224. MR4513382. doi: <https://doi.org/10.1080/10618600.2022.2050250>. 2, 11, 16, 20
- van den Boom, W., De Iorio, M., and Beskos, A. (2023). “Bayesian learning of graph substructures.” *Bayesian Analysis*, 18(4): 1311–1339. MR4675040. doi: <https://doi.org/10.1214/22-ba1338>. 5, 25
- Vogels, L., Mohammadi, R., Schoonhoven, M., and Birbil, Ş. İ. (2024). “Bayesian structure learning in undirected Gaussian graphical models: Literature review with empirical comparison.” *Journal of the American Statistical Association*, 119(548): 3164–3182. MR4833945. doi: <https://doi.org/10.1080/01621459.2024.2395504>. 6, 15, 16, 20
- Wang, H. (2012). “The Bayesian graphical Lasso and efficient posterior computation.” *Bayesian Analysis*, 7: 771–790. MR3000017. doi: <https://doi.org/10.1214/12-BA729>. 2
- Wang, H. (2015). “Scaling it up: Stochastic search structure learning in graphical models.” *Bayesian Analysis*, 10(2): 351–377. MR3420886. doi: <https://doi.org/10.1214/14-BA916>. 2, 4, 15, 16
- Williams, D. R. and Mulder, J. (2020). “Bayesian hypothesis testing for Gaussian graphical models: Conditional independence and order constraints.” *Journal of Mathematical Psychology*, 99: 102441. MR4156963. doi: <https://doi.org/10.1016/j.jmp.2020.102441>. 2
- Wolffe, A. (2001). “Histone genes.” In Brenner, S. and Miller, J. H. (eds.), *Encyclopedia of Genetics*, 948–952. New York: Academic Press. 23
- Wong, C., Moffa, G., and Kuipers, J. (2024). “A new way to evaluate G-Wishart normalising constants via Fourier analysis.” *arXiv preprint arXiv:2404.06803*. 6
- Wong, C., Moffa, G., and Kuipers, J. (2025). “On a conjecture of Roverato regarding G-Wishart normalising constants.” *arXiv preprint arXiv:2503.13046*. 6
- Xu, M., Evans, L., Bizzaro, C., Quaglia, F., Verrillo, C., Li, L., Stieglmaier, J., Schiewer, M., Languino, L., and Kelly, W. (2022). “STEAP1–4 (Six-transmembrane epithelial antigen of the Prostate 1–4) and their clinical implications for Prostate cancer.” *Cancers*, 14: 4034. 22