Accelerating Bayesian Structure Learning in Sparse Gaussian Graphical Models

Reza Mohammadi, Hélène Massam, and Gérard Letac

Department of Business Analytics, University of Amsterdam, Amsterdam, Netherlands; Department of Mathematics and Statistics, York University, Ontario, Canada; Laboratoire de Statistique et Probabilités, Université Paul Sabatier, Toulouse, France

ABSTRACT
Bayesian structure learning in Gaussian graphical models is often done by search algorithms over the graph space. The conjugate prior for the precision matrix satisfying graphical constraints is the well-known $G$-Wishart. With this prior, the transition probabilities in the search algorithms necessitate evaluating the ratio of the prior normalizing constants of $G$-Wishart. In moderate to high-dimensions, this ratio is often approximated by using sampling-based methods as computationally expensive updates in the search algorithm. Calculating this ratio so far has been a major computational bottleneck. We overcome this issue by representing a search algorithm in which the ratio of normalizing constants is carried out by an explicit closed-form approximation. Using this approximation within our search algorithm yields significant improvement in the scalability of structure learning without sacrificing structure learning accuracy. We study the conditions under which the approximation is valid. We also evaluate the efficacy of our method with simulation studies. We show that the new search algorithm with our approximation outperforms state-of-the-art methods in both computational efficiency and accuracy. The implementation of our work is available in the R package BDgraph.

1. Introduction

Gaussian graphical models (GGM) have been widely used in many application areas for learning conditional independence structure among a (possibly large) collection of variables. Bayesian structure learning, for these models, while providing a natural and principled way for uncertainty quantification, often lag behind frequentist approaches (Friedman, Hastie, and Tibshirani 2008) in terms of computational efficiency and scalability. Despite significant developments of Bayesian structure learning methods in recent years, the scalability of these methods has continued to pose challenges regarding the growing demand for higher dimensions.

An essential element of Bayesian structure learning in GGMs is the prior distribution on the precision matrix $K$ given the graph $G$ constraints. Most Bayesian methods use the so-called $G$-Wishart distribution, which is the conjugate prior (Roverato 2002). For structure learning, more recent Bayesian methods, use versions of search algorithms over the graph space with the capability of jointly estimate graph structure and precision matrix, see Hinne et al. (2014), Cheng and Lenkoski (2012), Lenkoski (2013), Dobra and Lenkoski (2011), Dobra, Lenkoski, and Rodriguez (2011), Wang and Li (2012), Wang (2012), Mohammadi and Wit (2015), Mohammadi et al. (2017), and van den Boom, Beskos, and De Iorio (2021). A computationally challenging step in these search algorithms is to estimate the ratio of prior normalizing constants for the $G$-Wishart distribution. This ratio, in general, is not available in closed form, except for specific cases, and typically needs to be evaluated using Monte Carlo-based approaches. Until recently, Uhler, Lenkoski, and Richards (2018) gave the exact analytic expression of the normalizing constants of $G$-Wishart, which gave hope of direct evaluation of this ratio. The capability of applying this expression in the search algorithms need yet to investigate since the expression is mathematically rather complex.

To approximate the ratio of normalizing constants, Wang and Li (2012) introduced the double Metropolis–Hastings algorithm (Liang 2010), by using the block Gibbs sampler from $G$-Wishart. By using direct sampling from $G$-Wishart distribution (Lenkoski 2013; Hinne et al. 2014; Lenkoski 2013) they presented more efficient versions of the search algorithms that combine the concept behind the exchange algorithm (Murray, Ghahramani, and MacKay 2006) with trans-dimensional MCMC algorithm (Green 2003). Likewise, Mohammadi and Wit (2015) proposed a search algorithm over the graph space based on continuous-time birth-death processes and combine it with the exchange algorithm. These algorithms avoid computing the ratio of normalizing constants by using the exchange algorithm. Essentially, the ratio of normalizing constants is canceling out in the probabilities of jumping to the proposal graphs, by using exact samples from the $G$-Wishart distribution. While these algorithms have clear computational benefits compared to earlier approaches, they require exact samples from the $G$-Wishart distribution. These algorithms are computationally expensive updates within the search algorithm. We are going to illustrate it in more detail in Section 2.2.
We aim to offer a search algorithm in which the ratio of normalizing constants is evaluated, by an explicit closed-form approximation. For Bayesian structure learning, we first represent the birth–death Markov Chain Monte Carlo (BDMCMC) search algorithm proposed by Mohammadi and Wit (2015). Then we provide an explicit closed-form approximation to the ratio of the prior normalizing constant of $G$-Wishart, the use of which leads to significant improvement in the scalability of the search algorithms. To immediately illustrate the accuracy, in terms of structure learning and the computational efficiency of our proposed approximation within the search algorithm, we represent here Figure 1 where $G$ has a random graph structure with 150 nodes ($p = 150$) and a sample size of 150. The left-hand side represents the receiver operating characteristic (ROC) plot for comparing the structure learning accuracy of the BDMCMC search algorithm done with our approximation and with the exchange algorithm. We see that our method (BDMCMC-Gamm, in blue) performs at least as well as the state-of-the-art (BDMCMC-DMH, in red). The right-hand side represents the execution time of both search algorithms. We see that for $p = 150$ the execution time when using BDMCMC with our approximation is three times faster than when BDMCMC is done with the exchange algorithm. More details are in Section 6.

The outline of our article is as follows. In Section 2, we introduce background materials for Bayesian structure learning in GGMs. After presenting the birth–death MCMC search algorithm in Section 2.1, we review the existing methods for approximating the ratio of normalizing constants in Section 2.2. Then we introduce our proposed approximation for the ratio of normalizing constants in Section 2.3. In Section 3, we provide the technical detail for proving the accuracy of the proposed approximation. In Sections 4 and 5, we represent our two main results, Theorems 1 and 2.

In Theorem 1, we establish the approximation with explicit bounds in the particular case when all paths between the two nodes corresponding to the removed edge are disjoint (Figure 3 left-side). In Section 4.1, we verify the accuracy of the approximation by various collections of disjoint paths. We compute the theoretical boundary of our approximation as well as the relative error values. We find that, while the theoretical boundary can be as high as 0.30, the actual value of the relative error hardly goes above 0.10 (see Figure 4).

In Theorem 2, we consider the general case where paths between the two nodes corresponding to the removed edge are not necessarily disjoint (Figure 3, right side). In that case, we prove under a technical assumption, our approximation is accurate. The question is then to know whether this assumption is realistic. In Section 5.1, for different types of graphs, we verify numerically how well this assumption holds. To do so, we compute the ratio of normalizing constants in two ways: first following the Monte Carlo approximation of Atay-Kayis and Massam (2005) and, second, using our approximation. We see that in all cases, both approximations take the same range of values. They are both reasonably accurate. When the number of nodes is greater than 30, due to the limitations of the Monte Carlo approximation in Atay-Kayis and Massam (2005), one cannot numerically verify the accuracy of the approximation directly. So, in Section 6, we verify it indirectly: we use both our approximation and the exchange algorithm to compute the ratio in the BDMCMC search algorithm of Mohammadi and Wit (2015) for graphs containing 50, 100, or 150 nodes.

2. Bayesian Structure Learning in GGMs

Graphical models (Lauritzen 1996) are powerful tools to express the conditional dependence structure among random variables by a graph in which each node corresponds to a random variable. For the case of undirected graphs, also known as Markov random fields (Rue and Held 2005), an edge between two nodes determines the conditional dependence of the respective variables. Let $G = (V, E)$ be an undirected graph where $V$ contains $p$ nodes corresponding to the $p$ coordinates and the
edges \( E \) describe the conditional independence relationships among variables. We use the convention that if \((i, j) \in E\) then \( i < j \). Let \( \bar{E} \) be the complement of \( E \) that indexes the missing edges of \( G \).

A GGM for the Gaussian random vector \( X = (X_1, \ldots, X_p) \sim N_p(\mu, \mathbf{K}^{-1}) \) is represented by an undirected graph \( G = (V, E) \). Variables \( X_i \) and \( X_j \) are independent given all the other variables if and only if there is no edge \((i, j) \in E\). It is well-known (Lauritzen 1996) that in that case, the precision matrix \( \mathbf{K} = \mathbf{\Sigma}^{-1} \) belongs to the cone \( \mathcal{P}_G \) of positive definite matrices with \( K_{ij} = 0 \) whenever \((i, j) \in \bar{E}\). In other words, the zero entries in the off-diagonal of the precision matrix correspond to conditional independencies in the graph; it is an essential property of the precision matrix for model selection (Dempster 1972). One can then define the GGM for a given graph \( G \) as the family of distributions

\[
\mathcal{N}_G = \{N(0, \Sigma) : K = \Sigma^{-1} \in \mathcal{P}_G\}.
\]

The likelihood based on a random sample \( X = (X^{(1)}, \ldots, X^{(n)})^\top \) from \( \mathcal{N}_G \) is

\[
P(X|K, G) \propto |K|^{n/2} \exp \left\{ -\frac{1}{2} \text{tr}(KS) \right\},
\]

where \( S = X^\top X \).

The G-Wishart distribution (Roverato 2002; Letac and Massam 2007) is the standard conjugate prior for the precision matrix \( K \) of the above Gaussian distribution. The G-Wishart density \( W_G(b, \Omega) \) is

\[
P(K|G) = \frac{1}{I_G(\delta, \Omega)} |K|^{\frac{\delta}{2}} \exp \left\{ -\frac{1}{2} \text{tr}(K\Omega) \right\} 1_{\mathcal{P}_G}(K),
\]

where \(|K|\) denotes the determinant of \( K \) and the symmetric positive-definite matrix \( \Omega \) and the scalar \( \delta > 2 \) are called, respectively, the scale and shape parameters. The normalizing constant

\[
I_G(\delta, \Omega) = \int_{K \in \mathcal{P}_G} |K|^{\frac{\delta}{2}} \exp \left\{ -\frac{1}{2} \text{tr}(K\Omega) \right\} dK
\]

is of central interest to us. The explicit formula for this normalizing constant is given in Proposition 1. We return to the computation of this quantity in Section 3.

The joint posterior distribution of the graph \( G \) and the precision matrix \( K \) is given as follows:

\[
P(K, G|X) \propto P(X|K, G) P(K|G) P(G)
\]

\[
\alpha P(G) \left( \frac{1}{I_G(\delta, \Omega)} |K|^{\frac{\delta}{2}} \exp \left\{ -\frac{1}{2} \text{tr}(K(\Omega + S)) \right\} \right),
\]

where \( P(G) \) is the prior distribution of the graph \( G \), which here we consider a uniform distribution over all graphs with fixed \( p \) nodes, as a non-informative prior. For other options, see Dobr, Lenkoski, and Rodriguez (2011), Hinoveanu, Leisen, and Villa (2020), and Mohammadi and Wit (2015).

### 2.1. Structure Learning Via Birth–Death MCMC Algorithm

Bayesian structure learning in GGMs which revolves around the joint posterior distribution of the precision matrix and graph (2) requires carefully designed MCMC search algorithms over the graph space. A common way to explore the graph space is by using a search algorithm known as reversible jump MCMC (RJ MCMC) (Green 1995) which is based on a discrete-time Markov chain. These kinds of algorithms often suffer from low acceptance rates since the graph space is enormous and proposals with low probabilities are frequent. Mohammadi and Wit (2015) addressed this issue by developing a continuous-time Markov chain process—or a BDMCMC search algorithm—as an alternative to RJ MCMC. The BDMCMC search algorithm explores the graph space by either jumping to a larger dimension (birth) or lower dimension (death). The birth/death events are modeled as independent Poisson processes, thus the time between two successes events is exponentially distributed. The stationary distribution of the process is determined by the rates of the birth and death events that occur in continuous time; see Figure 2 for a graphical representation of birth and death events from a given graph.

In the birth and death process, given the current state \((G, K)\), each edge is added/deleted independently of the rest as a Poisson process with birth/death rate \( \lambda_e(G, K) \) for each \( e \in \{E \cap \bar{E}\} \). Since birth and death events are independent Poisson processes, the time between two consecutive events has an exponential distribution.
distribution with mean
\[ W(G, K) = \frac{1}{\sum R_e(G, K)} \]
which is the waiting time. The waiting times capture all the possible moves of each step of the BDMMCMC search algorithm. Essentially, the birth–death process tends to stay shorter in the current state for a small waiting time, while the process tends to stay longer for a large waiting time. The birth and death probabilities involved are
\[ P(\text{birth/death of edge } e \in \{E \cup \overline{E}\}) \propto R_e(G, K). \]

The BDMMCMC search algorithm converges to the joint posterior distribution (2) given the birth and death rates as
\[ R_e(G, K) = \min \left\{ \frac{P(G^e, K^e|X)}{P(G, K|X)}, 1 \right\}, \text{ for each } e \in \{E \cup \overline{E}\}. \]

For the birth of edge \( e \in \overline{E} \) we take \( G^e = (V, E \cup e) \) and for the death of edge \( e \in E \) we take \( G^e = (V, E \setminus e) \) and with the corresponding preposition matrix is \( K^e \). Algorithm 1 represents the pseudo-code for the BDMMCMC search algorithm.

**Algorithm 1: BDMMCMC search algorithm**

**Input:** A graph \( G = (V, E) \) with a precision matrix \( K \) and data \( X \).

**for** \( N \) iteration **do**

**for all possible moves in parallel** **do**

- Calculate the birth and death rates by Equation 5;
- Calculate the waiting time by Equation 3;
- Update the graph based on the birth/death probabilities in Equation 4;
- Update the precision matrix;

**Output:** Samples from the joint posterior distribution (2).

The essential element of the BDMMCMC search algorithm is that a continuous-time jump process is associated with the birth and death rates. Whenever a jump occurs, the corresponding move is always accepted, which can be considered more efficient navigation of the graph space. The acceptance probabilities of commonly used RJMCMC algorithms are replaced by the waiting times in the BDMMCMC algorithm. Correspondingly, graphs with high posterior probabilities have larger waiting times while graphs with low posterior probabilities have small waiting times and as a result, die quickly. Another computational advantage of the BDMMCMC algorithm is that the nested for loop, as a computationally expensive part of the algorithm, for computing the birth/death rates can be implemented in parallel since the rates associated with each edge can be calculated independently of each other. We have implemented this part in parallel in the current version of the \( R \) package BDgraph (Mohammadi, Wit, and Dobra 2021). These properties make the BDMMCMC algorithm an efficient search approach to explore the graph space to identify the high posterior probability regimes, particularly for high-dimensional graphical models.

The main computational bottleneck of Algorithm 1 is to evaluate the birth/death rates, which are based on the ratio of the posterior probabilities. These birth/death rates can be considered as the conditional Bayes factor of the comparison between graph \( G \) and \( G^+ / G^- \), similar to Hinne et al. (2014). The ratios in Equation (5) can be derived as
\[ \frac{P(G^e, K^e|X)}{P(G, K|X)} \propto H(K, \Omega + S, \delta + n, e), \]
where
\[ H(K, \Omega + S, \delta + n, e) = \exp \left( \frac{-1}{2} \text{tr}(K^e(\Omega + S)) \right). \]

For details regarding how to compute the above function, see Cheng and Lenkoski (2012), Mohammadi and Wit (2015), and Hinne et al. (2014). We see that computing the ratio of posteriors requires evaluating the ratio of prior normalizing constants. That is the main computational bottleneck of these types of search algorithms.

### 2.2. Existing Methods to Compute the Normalizing Constant

**Exact formula:** Uhler, Lenkoski, and Richards (2018) certified that it is possible to derive an explicit expression for the intractable normalizing constant for general graphs. Since the expression is (by its nature) mathematically complex, the capability of applying this intricate expression for Bayesian structure learning has yet to be investigated. One possibility, as they point it out, would be to find more computationally efficient procedures than (Uhler, Lenkoski, and Richards 2018, Theorem 3.3) for computing the normalizing constant for particular classes of graphs.

**Monte Carlo approximation:** Atay-Kayis and Massam (2005) developed a Monte Carlo (MC) approach to approximate the normalizing constant based on the decomposition described in Section 3. Although the MC approximation is accurate, it can be computationally expensive. In our simulation of Sections 4.1 and 5.1, we faced numerical and computational issues of MC approximation for \( p \) higher than 30.

**Laplace approximation:** Lenkoski and Dobra (2011) developed a Laplace approximation to compute \( I_G(\delta, \Omega) \). Their approximation is based on the iterative proportional scaling algorithm for computing the mode of the integral in Equation 1. This approximation is computationally faster than the MC approach, though it tends to be accurate only for the case of computing the posterior normalizing constant.

**Exchange algorithm:** Murray, Gharahmani, and MacKay (2006) proposed the exchange algorithm for simulating from distributions, where prior distributions—like \( G \)-Wishart—have intractable normalizing constants that vary according to the model. These types of algorithms are also known as auxiliary variable approaches since they require exact sampling from the auxiliary variable to cancel out the ratio of normalizing constants in the Metropolis-Hastings acceptance probabilities (Park and Haran 2018). Hinne et al. (2014), Lenkoski (2013), and Mohammadi and Wit (2015) had implemented this algorithm in GGMs to avoid normalizing constant calculation by using the exact sampler algorithm from \( G \)-Wishart distribution, proposed by Lenkoski (2013). As the state-of-the-art, this development
has proven to yield computational improvement as it avoids the need for expensive approximations within the search algorithm. We briefly review the implementation of the exchange algorithm within the search algorithm. For more details, see (Wang and Li 2012, sec. 5.2).

Suppose we want to compute the birth/death rate (5) for graph $G = (V, E)$ with the precision matrix $K$ as the current state of the search algorithm. In the exchange algorithm, we can approximate the intractable normalizing constant ratio with a single sample at each parameter setting as

$$
\frac{IG(\delta, \Omega)}{IG^*(\delta, \Omega)} \approx \frac{\widetilde{K}^{\frac{1}{2}} e^{\frac{1}{2} \text{tr}(\widetilde{K} \Omega)}}{\widetilde{K}^{\frac{1}{2}} e^{\frac{1}{2} \text{tr}(\widetilde{K}^* \Omega)}}
$$

where $\widetilde{K}$ has to be an exact sample from the prior distribution, $W_G(\delta, \Omega)$. The exchange algorithm replaces the ratio of the intractable normalizing constants with an estimate from a single sample at each parameter setting. By using the above approximation, the birth/death rates will be

$$
R_{e}(G, K) \approx \min \left\{ \frac{H(K, \Omega + S, \delta + n, e)}{H(K, \Omega, \delta, e)}, 1 \right\},
$$

for each $e \in \{E \cup \bar{E}\}$, (7)

where function $H$ is given in Equation (6). Essentially, the intractable prior normalizing constants have been replaced by an evaluation of function $H$ at $\widetilde{K}$ as an exact sample from the prior distribution $W_G(\delta, \Omega)$.

Algorithm 2 represents the pseudo-code for the BD-MCMC search algorithm combined with the exchange algorithm to compute the ratio of normalizing constants. We call it a double BD-MCMC algorithm and consider it here as the state-of-the-art. For more details, see Mohammadi and Wit (2015) and Hinne et al. (2014).

**Algorithm 2: Double BD-MCMC algorithm**

**Input:** A graph $G = (V, E)$ with a precision matrix $K$ and data $X$.

**for** $N$ iteration **do**

- Draw $\widetilde{K} \sim W_G(\delta, \Omega)$;
- **for all the possible moves in parallel**
  - Calculate the birth and death rates by Equation 7;
  - Calculate the waiting time by Equation 3;
  - Update the graph based on birth/death probabilities in Equation 4;
  - Update the precision matrix;
- **Output:** Samples from the posterior distribution (2).

**Remark 1.** Algorithm 2 requires exact sampling from the prior distribution of $G$-Wishart as a computationally expensive update within the BD-MCMC search algorithm. Exact sampling from $G$-Wishart distribution, following Lenkoski (2013), can be done by first sampling a standard Wishart variable from a full model and then using the iterative proportional scaling algorithm to place the variable in the correct space. It requires the solution of systems involving large matrices, in particular, the inverse calculation of matrix $K$.

### 2.3. Proposed Method to Compute the Normalizing Constant

To bypass the computational bottleneck from the intractable normalizing constant in Algorithm 1, we represent a simple explicit analytic formula to approximate the normalizing constant as

$$
\frac{IG(\delta, \Omega_p)}{IG(\delta, \Omega_p)} \approx \frac{1}{2\pi} \frac{\Gamma\left(\frac{\delta + d}{2}\right)}{\Gamma\left(\frac{\delta + d + 1}{2}\right)}
$$

where $d$ is the number of paths of length two, linking the endpoints of $e$. As is the case most of the time, in the absence of prior information, the parameter $\Omega$ is taken to be the $p$-dimensional identity matrix $I_p$. Throughout, we set $\Omega = I_p$. This approximation is exact in some cases, see Remark 4. The following sections are therefore devoted to motivating this approximation and analyzing its accuracy.

### 3. The Ratio of Normalizing Constants

We first recall a result by Atay-Kayis and Massam (2005) which expresses $IG(\delta, \Omega)$ as the product of a constant and an expectation. Let $K$ be the precision matrix and $K = \Psi^T\Psi$ its Cholesky decomposition where $\Psi$ is upper triangular with positive diagonal elements. Given the fact $K_{ij} = 0$ for $(i,j) \in \bar{E}$, through simple matrix multiplication, we can verify

$$
\psi_E = \{\psi_{ij} : (i,j) \in E \text{ and } \psi_{ii} : i \in V\}
$$

is in 1-1 correspondence with $K_E = \{K_{ij} : (i,j) \in E \text{ and } K_{ii} : i \in V\}$. Also, the entries of $\psi_E$ can be expressed in terms of $\psi_E$, a fact used in the following proposition. Thus, the entries of $\psi_E$ are called free variables while the entries of $\psi_{\bar{E}}$ are non-free variables. Using the change of variables from $K_E$ to $\psi_E$, Atay-Kayis and Massam (2005) proved the normalizing constant $IG(\delta, \Omega)$ can be expressed as a known constant multiplied by the expected value of a function of $\psi_E$. In particular, the case where $\Omega = I_p$, which is of concern to us, the result is as follows.

**Proposition 1.** For the undirected graph $G$, we have

$$
IG(\delta, I_p) = \left[ \prod_{i=1}^{p} \psi_{ii}^{2\frac{n_i}{2} + \nu_i} \exp\left(\frac{\delta + \nu_i}{2}\right) \right] \mathbb{E}\left(\psi_e^{\Omega_p}\right),
$$

where $v_i = |nb(i) \cap \{i + 1, \ldots, p\}|$ in which $nb(i)$ denote the set of neighbors of node $i = \{1, \ldots, p\}$ and

$$
D = \sum_{(i,j) \in \bar{E}} \psi_{ij}^2
$$

and the expected value is taken with respect to a product of independent random variables $\psi_{ij} \sim N(0,1)$ where $(i,j) \in E$ and random variables $\psi_{ii} \sim \chi^2_{\delta + \nu_i}$.

The value of $IG(\delta, I_p)$ is independent of the ordering of the nodes, so without loss of generality, in the remainder of this paper, we assume the nodes defining the edge $e$ are $q = p - 1$ and $p$, that is the endpoints of $e$ are numbered last. For convenience, we write $\psi_e = \psi_{qp}$, which is a non-free variable in the graph $G^{-e}$. 
Corollary 1. Let $G^{-e}$ be the graph obtained from $G$ by removing the edge $e = (q, p).$ The ratio of the prior normalizing constants for $G^{-e}$ and $G$ is

$$\frac{I_{G^{-e}}(\delta, \|p\|)}{I_G(\delta, \|p\|)} = \frac{1}{2\sqrt{\pi}} \frac{\Gamma\left(\frac{\delta}{2}\right)}{\Gamma\left(\frac{\delta + d}{2}\right)} \mathbb{E}\left(e^{-\frac{D}{2}}\right).$$

(9)

The proof of Corollary 1 is immediate if we observe that, since $v_i = |nb(i) \cap \{i+1, \ldots, p\}|$, the only $v_i$ that changes between $G^{-e}$ and $G$ is the node $v_q$ and, clearly, $v_q^{G^{-e}} = 0$ while $v_q^G = 1$.

### 3.1. Reformulation of the Ratio of Normalizing Constants

We can derive the non-free entries of $\psi$ as

$$\psi_{ij} = 0 \quad \text{and} \quad \psi_{ji} = -\frac{1}{\psi_{ii}} \sum_{l=1}^{i-1} \psi_{il} \psi_{lj}, \quad \text{for } i \neq 1. \quad (10)$$

The variables $\psi_{ii}$ or $\psi_{ji}$ in the expression of $\psi_{ij}$ above may be free or nonfree variables; see also (Atay-Kayis and Massam 2005, prop. 2).

Remark 2. If $\psi_{ij}$ is nonfree, it follows from Equation (10) that $\psi_{ij}$ can only be a function of free variables $\psi_{ii}, i \neq k$ such that $l \leq i$ and $k < j$ and $\psi_{ii}, l \leq i$.

Since the value of $I_G(\delta, \|p\|)$ does not depend upon the order of the nodes, from now on, we assume the nodes which are neighbors to both $q$ and $p$, are numbered $q-d, q-(d-1), \ldots, q-1$ where $d$ is the number of paths of length 2 between nodes $q$ and $p$; see for example the node orders in Figure 3. With this convention, we have $\psi_{e} = A + b$ where

$$A = -\frac{1}{\psi_{qq}} A_1 \quad \text{where} \quad A_1 = \sum_{l=q-d}^{q-1} \psi_{iq} \psi_{lp},$$

$$b = -\frac{1}{\psi_{qq}} b_1 \quad \text{where} \quad b_1 = \sum_{l=1}^{q-d} \psi_{iq} \psi_{lp}. \quad (11)$$

Remark 3. The numbering we have adopted for nodes that are neighbors both to $q$ and $p$ ensures that $A$ is independent of $b$ and $D$.

With the notations above, Equation (9) can be written

$$\frac{I_{G^{-e}}(\delta, \|p\|)}{I_G(\delta, \|p\|)} = \frac{1}{2\sqrt{\pi}} \frac{\Gamma\left(\frac{\delta}{2}\right)}{\Gamma\left(\frac{\delta + d}{2}\right)} \mathbb{E}\left(e^{-\frac{D}{2}} e^{-\frac{(\Delta+b)^2}{2}}\right).$$

Our aim is to approximate this ratio and, toward this goal, we have

$$\mathbb{E}\left(e^{-\frac{D}{2}} e^{-\frac{(\Delta+b)^2}{2}}\right) \approx \mathbb{E}\left(e^{-\frac{D}{2}}\right) \mathbb{E}\left(e^{-\frac{D^2}{2}}\right). \quad (13)$$

If we prove the above approximation holds, then

$$\frac{I_{G^{-e}}(\delta, \|p\|)}{I_G(\delta, \|p\|)} \approx \frac{1}{2\sqrt{\pi}} \frac{\Gamma\left(\frac{\delta}{2}\right)}{\Gamma\left(\frac{\delta + d}{2}\right)} \mathbb{E}\left(e^{-\frac{D^2}{2}}\right),$$

where

$$\mathbb{E}\left(e^{-\frac{D^2}{2}}\right) = \frac{\Gamma\left(\frac{\delta+1}{2}\right)}{\Gamma\left(\frac{\delta}{2}\right)} \frac{\Gamma\left(\frac{\delta + d + 1}{2}\right)}{\Gamma\left(\frac{\delta + d}{2}\right)},$$

the proof is in Section A of the supplementary file. Thus, we have

$$\frac{I_{G^{-e}}(\delta, \|p\|)}{I_G(\delta, \|p\|)} \approx \frac{1}{2\sqrt{\pi}} \frac{\Gamma\left(\frac{\delta + d}{2}\right)}{\Gamma\left(\frac{\delta + d + 1}{2}\right)}$$

which is the approximation (8) that we want to prove.

**Remark 4.** In Equation (13) if $b = 0$, then our approximation (8) is exact. This means when the paths between nodes $q$ and $p$ are of length 2 or no path, the approximation is exact. It happens also in other cases as (Uhler, Lenkoski, and Richards 2018, theor. 2.5) show that if $G^{-e}$ is such that $G$ is decomposable, then our approximation (8) is exact.

**Lemma 1.** Using the quantities, $D, A, b,$ and $b_1$ defined above, we have

$$\mathbb{E}\left(e^{-\frac{D}{2}} e^{-\frac{(\Delta+b)^2}{2}}\right) = \mathbb{E}\left(e^{-\frac{D}{2}}\right) \mathbb{E}\left(e^{-\frac{D^2}{2}} h(b_1, \delta^*)\right),$$

where $\delta^* = \frac{\delta + d}{2}$ and

$$h(b_1, \delta^*) = \frac{2^{-\delta^*}}{\Gamma(\delta^*)} \int_0^{+\infty} y^{\delta^*-1} e^{-\frac{y}{2}} e^{-\frac{\Delta^2}{2}} dy.$$

(14)

and

$$\Psi_{\bar{U}} = \{\psi_{ij} : (i,j) \in E \setminus (E_q \cup E_p)\}.$$

(15)

where $E_q = \{(i,j) : (i,q) \in E\}$ and $E_p = \{(i,j) : (i,p) \in E\}$. Set $\Psi_{\bar{U}}$ includes all the free elements of the matrix $\Psi$ except those are the neighbors of nodes $p$ and $q$.

The proof is given in Section B of the Supplementary file. Regarding the above lemma, proving

$$\mathbb{E}\left(e^{-\frac{D}{2}} h(b_1, \delta^*)\right) \approx \mathbb{E}\left(e^{-\frac{D}{2}}\right) \mathbb{E}\left(e^{-\frac{D^2}{2}} h(b_1, \delta^*)\right)$$

leads to the approximation in Equation 13. For convenience, we adopt the notation

$$I_1 = \mathbb{E}\left(e^{-\frac{D}{2}} h(b_1, \delta^*)\right) \quad \text{and} \quad I_2 = \mathbb{E}\left(e^{-\frac{D}{2}}\right)$$

and therefore

$$\frac{I_1}{I_2} = \mathbb{E}\left(e^{-\frac{D}{2}} h(b_1, \delta^*)\right) \mathbb{E}\left(e^{-\frac{D^2}{2}}\right).$$

(16)

The accuracy of our approximation in Equation (8) is represented by how close is the above ratio $I_1/I_2$ to 1. Thus, proving our approximation is accurate is equivalent to prove that $I_1/I_2$ can accurately be approximated by 1. For example, for the cases that $I_1/I_2$ is equal to 1, our approximation is exact.

**Remark 5.** It is important to mention that $I_1/I_2$ is always equal to or less than 1 ($I_1/I_2 \leq 1$). It follows immediately from Equation 14 since $b_1^2/2Y$ is always positive and $e^{-b_1^2}/Y \leq 1$. 


Remark 6. If we could show the expectation $\mathbb{E} \left( h(b_1, \delta^*) \mid \Psi_J \right)$ can uniformly be approximated by 1, it would follow that $I_1/I_2$ can also be approximated by 1. We are not able to quite achieve this goal but, first, in the next section (Theorem 1), we establish the approximation with explicit bounds in the special case when all paths between $q$ and $p$ are disjoint. The key to proving this result is the fact that $b_1$ can be expressed as a linear product of independent normal variables, for the cases of disjoint paths. Then, in Section 5, we show, conditional on $\Psi_J$ defined in Equation (15), the distribution of $b_1$ is a scale mixture of normal distributions. We then use this scale mixture of distributions to admit a unique $N(0, \nu_D)$ approximation. Finally, we show that a sufficient condition for $\mathbb{E} \left( h(b_1, \delta^*) \mid \Psi_J \right)$ to be close to 1 is that $\nu_D$ is close to 0.

4. The Ratio in the Disjoint Paths Case

A path is a sequence of nodes in which each node is connected by an edge to the next and the path length is the number of edges between them. Two paths between $q$ and $p$ are disjoint if they have no node other than $p$ and $q$ in common. For example, in the left-hand side graph of Figure 3, the paths between $q = 7$ and $p = 8$ are

$$\lambda_1 = \{q, 1, 2, 3, p\}, \lambda_2 = \{q, 4, 5, p\}, \lambda_3 = \{q, 6, p\},$$

and they are disjoint paths.

A path $\lambda \in \Lambda$ of length $\ell_\lambda + 1$ is a sequence of distinct nodes as $\lambda = \{q, 1_\lambda, 2_\lambda, \ldots, \ell_\lambda, p\}$ where $(q, 1_\lambda), \ldots, ((i + 1)_\lambda), \ldots, (\ell_\lambda, p)$ are edges of $G$. The set of all such paths between $q$ and $p$ is denoted $\Lambda$. We let $E_\lambda$ and $V_\lambda$ be, respectively, the set of edges and the set of interior nodes of path $\lambda$ as

$$E_\lambda = \{(q, 1_\lambda), (1_\lambda, 2_\lambda), \ldots, (\ell_\lambda, p)\}, \text{ and}$$

$$V_\lambda = \{1_\lambda, 2_\lambda, \ldots, \ell_\lambda\}.$$ 

If $L = |\Lambda|$ is the total number of paths, we set an arbitrary order $\lambda_1, \ldots, \lambda_L$ of the paths where, for convenience, we list the paths of length 2, i.e., $\ell_\lambda = 1$ last. The nodes $q$ and $p$ are ranked last so that the order of the nodes in $V$ is

$$1_{\lambda_1}, \ldots, \ell_{\lambda_1}, 1_{\lambda_2}, \ldots, \ell_{\lambda_2}, \ldots, 1_{\lambda_L}, \ldots, \ell_{\lambda_L}, q, p.$$ 

Using these notations, the following lemma gives the expression for $\psi_e$ in terms of the free variables in set $\psi_E$.

**Lemma 2.** In the model with underlying graph $G^{\delta^*}$, the variables $\psi_{qp} = \psi_e$ of the Cholesky decomposition of the precision matrix $K$ is expressed in terms of $\psi_E$ as

$$\psi_e = \frac{1}{\psi_{qq}} \sum_{i \in \Lambda} (-1)^{\ell_i} \prod_{v \in V_\lambda \setminus \{1_i\}} \psi_{vv}.$$  

(17)

The proof relies on a repeated application of Equation 10 and is given in Section C of the supplementary file. We illustrate these calculations with the following example.

**Example 1.** For the graph of Figure 3 (left), the upper triangular matrix $\Psi$ is

$$\Psi = \begin{bmatrix}
\psi_{11} & \psi_{12} & 0 & 0 & 0 & 0 & \psi_{17} & 0 \\
\psi_{12} & \psi_{22} & 0 & 0 & 0 & \ast & \psi_{33} & 0 \\
0 & \psi_{23} & 0 & 0 & 0 & \ast & \psi_{34} & 0 \\
0 & 0 & \psi_{33} & 0 & 0 & \ast & \psi_{45} & 0 \\
0 & 0 & 0 & \psi_{34} & 0 & \ast & \psi_{47} & 0 \\
0 & 0 & 0 & 0 & \psi_{35} & \ast & \psi_{55} & 0 \\
0 & 0 & 0 & 0 & 0 & \psi_{55} & \ast & \psi_{66} \\
0 & 0 & 0 & 0 & 0 & 0 & \psi_{67} & \ast \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \psi_{77} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \psi_{78} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \psi_{88}
\end{bmatrix}$$

where the entries marked with a “∗” are the non-free entries and are given as follows:

$$\psi_{27} = -\frac{\psi_{12}\psi_{17}}{\psi_{22}}, \ \psi_{37} = \frac{\psi_{17}\psi_{12}}{\psi_{22}}\psi_{23}, \ \psi_{57} = -\frac{\psi_{45}\psi_{47}}{\psi_{55}},$$

and

$$\psi_{78} = -\frac{1}{\psi_{77}} \left( \psi_{67}\psi_{68} + \psi_{57}\psi_{58} + \psi_{47}\psi_{48} \right)$$

$$= \frac{1}{\psi_{77}} \left( -\psi_{67}\psi_{68} + \psi_{47}\psi_{45}\psi_{58} - \psi_{47}\psi_{12}\psi_{23}\psi_{38} \right).$$

Equation (17) is verified. We see that the different terms in $\psi_{qp}$ and $\psi_{78}$ above concern, successively, the paths of length 2, 3, and 4.

We are now in a position to state our first theoretical result regarding the error made of our approximation in Equation (8) or equivalently the approximation in Equation (13).
Theorem 1. For the case where in the graph \( G \), the paths between the endpoints of the edge \( e = (q, p) \) are disjoint, the ratio \( I_1/I_2 \) (16) is such that
\[
B(\delta, d, \ell_λ) \leq \frac{I_1}{I_2} \leq 1,
\] (18)
where
\[
B(\delta, d, \ell_λ) = 1 - \frac{\delta^2}{\pi(\delta + 2)} \left( \frac{\Gamma\left(\frac{\delta}{2}\right)}{\Gamma\left(\frac{\delta + 1}{2}\right)} \right)^2 r(\delta + d - 1) \sum_{\lambda \in \Lambda} r(\delta)\ell_λ,
\]
with \( \Lambda \) being the set of paths between \( q \) and \( p \), \( d \) the number of paths of length 2, and
\[
r(\delta) = \frac{\Gamma\left(\frac{\delta}{2}\right)}{\sqrt{\pi} \Gamma\left(\frac{\delta + 1}{2}\right)}.
\]

With an accuracy given by Equation (18), we have the approximation
\[
\frac{I_{G^-}(\delta, \ell_λ)}{I_G(\delta, \ell_λ)} \approx \frac{1}{2\sqrt{\pi}} \frac{\Gamma\left(\frac{\delta + d}{2}\right)}{\Gamma\left(\frac{\delta + d + 1}{2}\right)}.
\]

Proof. The proof is given in Section D of the supplementary file. The proof is based on the fact that the expression of \( b_1 \) (12) can be expressed as a linear product of independent normal variables in the case the paths between \( q \) and \( p \) are disjoint. \( \square \)

4.1. Simulated Experiments for the Disjoint Paths Case
To illustrate the results in Theorem 1, we report the ratio \( I_1/I_2 \) (16) following the MC approach of Atay-Kayis and Massam (2005) as well as the lower bound \( B \) in Equation (19). We note that, if \( I_1/I_2 \approx 1 \) our approximation is good, without any additional conditions. Note that, \( 1 - I_1/I_2 \) reflects the relative error of our approximation (8). Since \( I_1/I_2 \) and \( B \) are functions of \( \delta \) and the type of disjoint paths (\( d \) and \( \ell_λ \)), our simulation is based on graphs with different types of disjoint paths as well as different values of \( \delta \). We consider 15 different types of graphs with five different paths between \( q \) and \( p \). These graphs are indicated on the horizontal axis in Figure 4. Each sequence of four digits denotes the number of paths of length 2, 3, 4, and 5 in the graphs. For example, “3110” indicates a graph configuration with 3 disjoint paths of length 2, 1 of length 3, 1 of length 4, and 0 of length 5.

Figure 4. The ratio \( I_1/I_2 \) and its bound \( B \) in Equation (19) for \( \delta = 3 \) (top) and \( \delta = 10 \) (bottom). The red dotted line is the bound \( B \) and the boxplots are \( I_1/I_2 \) (over 100 replications) computed by the MC algorithm of Atay-Kayis and Massam (2005), with 500 samples. The 15 different graphs are indicated on the horizontal axis. Each sequence of four digits indicates the number of paths of lengths 2, 3, 4, and 5 in the graph. For example, “3110” represents a graph with 3 disjoint paths of length 2, 1 of length 3, 1 of length 4, and 0 of length 5.
Figure 5. Plot visualization for the lower bound in Equation 19 for \( \delta = [3, 4, \ldots, 40] \) and for the 15 different graphs which are indicated on the horizontal axis in Figure 4. Each dotted line represents the \( b \) values for a graph with specific types of paths. For example, the black bottom line is for the case "0500" which means a graph with 0 disjoint paths of length 2, 5 of length 3, 0 of length 4, and 0 of length 5.

Figure 4 represents the values of \( I_1/I_2 \) (over 100 replications) as well as the lower bound \( B \) (19) for two values of \( \delta = [3, 10] \). The ratio \( I_1/I_2 \) is computed by the MC algorithm of Atay-Kayis and Massam (2005) with 500 samples. The worst-case scenarios are for the case \( \delta = 3 \) and no paths of length two \((d = 0)\), like the graph "0500" which has 5 paths of length 3 and no other type of paths; these types of graphs are highly unlikely cases. Even for this case, the relative error is around 0.12. For the case \( \delta = 10 \), we see that our approximation has pretty good performance with the maximum relative error \( 1 - I_1/I_2 \) around 0.025.

Figure 5 reports the values of the lower bound \( B \) for different values of \( \delta \) \((\delta = [3, 4, \ldots, 40])\) and for the 15 different graphs which are indicated on the horizontal axis in Figure 4. Each dotted line represents the \( b \) values for a specific graph with different types of paths. For instance, the black bottom line is for the configuration "0500." In general, this plot indicates that the accuracy of our approximation is increased by increasing the value of \( \delta \). As we can see the worst-case scenario is for the minimum value of \( \delta = 3 \), while for the cases \( \delta > 10 \) the lower bound \( B \) for our approximation is close to 1.

5. The Ratio in the General Case

When the paths between \( q \) and \( p \) are not disjoint, the expression of \( b_1 \) (12) becomes more complicated. It can be expressed in terms of variables \( \psi_{ij} \sim N(0, 1), j < p \) and variables of the type

\[
X_{ij} = \frac{\psi_{ij}}{\psi_{ji}}, \quad i < j,
\]

where \( \psi_{ij} \sim N(0, 1) \) and \( \psi_{ji}^2 \sim X_{ij}^2 + \nu_j \). As a toy example, for the graph of Figure 3 (right) with tedious computations yield

\[
b_1 = \psi_{26}X_{23}^2X_{24}\psi_{47} + \psi_{26}X_{24}\psi_{47} + \psi_{16}X_{14}\psi_{47} + \psi_{26}X_{23}\psi_{37}.
\]

For the details, see Example E.1 of the supplementary file. We see that \( b_1 \) is the sum of polynomials in \( X_{ij}, (i, j) \in E \) multiplied by the product of two independent \( N(0, 1) \). But, unlike in the case of disjoint paths between \( q \) and \( p \), the polynomials are not linear in each \( X_{ij} \). We see in our simple example that one of them has degree 2, and larger graphs would lead to polynomials of higher linear degree. So, we could not find a lower bound, similar to Theorem 1. We therefore should find another argument to prove that \( I_1/I_2 \) is close to 1. This result is given in the following theorem.

Theorem 2. Under the approximation \( b_1 \sim N(0, \nu_D) \), the ratio \( I_1/I_2 \) (16) can be written

\[
I_1 \over I_2 = \frac{\mathbb{E} \left( e^{-\frac{\nu_D}{2}} g(\delta^*, \nu_D) \right)}{\mathbb{E}(e^{-\frac{\nu_D}{2}})}, \tag{20}
\]

where

\[
g(\delta^*, \nu_D) = \left( \frac{\nu_D}{2} \right)^{\frac{\nu_D}{2}} \frac{1}{\Gamma(\delta^*)} \int_0^{\infty} t^{\delta^* - 1} e^{-t} \left( \frac{\nu_D}{2} \right) e^{-\frac{\nu_D}{2}} \frac{1}{\Gamma(\delta^*)} \int_0^{\infty} t^{\delta^* - 1} e^{-t} \left( \frac{\nu_D}{2} \right) e^{-\frac{\nu_D}{2}} dt,
\]

in which \( \delta^* = \frac{\delta + d}{2} \). Moreover, when \( \nu_D \) is small, we have

\[
g(\delta^*, \nu_D) = 1 - \frac{\Gamma(\delta^* + \frac{1}{2})}{\Gamma(\delta^*)} \left( \frac{\nu_D}{2} \right)^{\delta^*} \mathcal{O}\left( \frac{\nu_D}{2} \right)^{\delta^* - 1}.
\]

And when, for all \( D, \nu_D \) is uniformly bounded by a small quantity, we have

\[
I_1 \over I_2 = 1 - \frac{\Gamma(\delta^* + \frac{1}{2})}{\Gamma(\delta^*)} \mathbb{E} \left( e^{-\frac{\nu_D}{2}} \left( \frac{\nu_D}{2} \right)^{\delta^*} \mathcal{O}(\frac{\nu_D}{2}^{\delta^* - 1}) \right) \approx 1
\]

and it leads that our approximation (8) holds.

Proof. The proof is in three steps. First, we show \( b_1 \) can be expressed as a bilinear form. Then, using the bilinear expression, we prove \( b_1 \) is distributed as the continuous scale mixture of centered Gaussian variables. Finally, this allows us to deduce that there exists a unique \( \nu_D \) so that the normal \( N(0, \nu_D) \) distribution best approximates the \( b_1 \) distribution. For detailed proof see Section E of the supplementary file.

In Theorem 2, we prove that \( I_1/I_2 \) can accurately be approximated by 1, under the assumption that \( \nu_D \) is small, or equivalently our approximation in Equation 8 is accurate. The validity of the assumption that \( \nu_D \) is small and the accuracy of the approximation is demonstrated numerically in the following subsection.
5.1. Simulated Experiments for the General Case

We compute the ratio $I_1/I_2$ in two different ways, first following the MC approach of Atay-Kayis and Massam (2005) and second using our approximation in Theorem 2; we call these values $I_{1/2,MC}$ and $I_{1/2,Gamm}$, respectively. We note that, if our approximation ($I_1/I_2 \approx 1$) is good, without any additional conditions, $I_{1/2,MC}$ should reflect that by being close to 1. However, if our approximation ($I_1/I_2 \approx 1$) is good, according to Theorem 2, $I_{1/2,Gamm}$ will be close to 1 if the assumption of $v_D$ small is satisfied.

While it is straightforward to evaluate $I_{1/2,MC}$, it is less obvious how to compute $I_{1/2,Gamm}$ using Equation 20. The pseudo-code for evaluating $I_{1/2,Gamm}$ is given in Section F of the Supplementary file. We represent the boxplot of the numerical values of $I_{1/2,MC}$ and $I_{1/2,Gamm}$ obtained over 100 replications for nine different types of graphs (Figure 6) along with three different numbers of nodes $p = \{10, 20, 30\}$ and two different values for $\delta = \{3, 10\}$. Besides, we report the corresponding values of $v_D$ so that one can see the variation of the accuracy of $I_{1/2,Gamm}$ as $v_D$ varies, as predicted by Theorem 2, but also that of $I_{1/2,MC}$.

For the case, $\delta = 3$, the values of $I_{1/2,MC}$ and $I_{1/2,Gamm}$ are represented in Figure 7 for $p = 20$, and Figures G.1 and G.2 of the supplementary file for $p = \{10, 30\}$. We see that the values of $I_1/I_2$ slightly move away from 1 as $v_D$ moves away from 0. But in all cases, we see that $I_{1/2,MC}$ and $I_{1/2,Gamm}$ cover the same range of values and their medians are between 0.9 and 1, giving relative errors less than 0.10. While, from these facts, we cannot immediately conclude that the assumption of $v_D$ small is always satisfied, it is a strong indication that it is satisfying enough to ensure that our approximation is acceptable.

Figure 6. The 9 different types of undirected graphs for $p = 20$, as a number of nodes. For the case of the Lattice graph, $p = 16$. The graphs Random_1, Random_2, Random_5 are random graphs with edge probabilities equal to 0.1, 0.2, and 0.5, respectively ranging from sparse to dense graphs.
Figure 7. (Top) The boxplot for the ratio $I_1/I_2$ computed by the MC approach of Atay-Kayis and Massam (2005) (in red), with 500 samples, and our approximation (20) (in green). (Bottom) The boxplot of the variance $v_D$ of variable $b_1$ for the corresponding graphs. These computations are done over 100 replications for nine different graphs (Figure 6) with $p = 20$ nodes and $\delta = 3$.

For the case, $\delta = 10$, the values of $I_{1/2,MC}$ and $I_{1/2,Gamm}$ are represented in Figure 8 for $p = 20$, and Figures G.3 and G.4 for $p = \{10, 30\}$ of the Supplementary File. In all cases, we see that $I_{1/2,MC}$ and $I_{1/2,Gamm}$ cover the same range of values and their medians are between 0.995 and 1, giving pretty low relative errors of less than 0.005.

Our simulations indicate that our approximation is more accurate for the sparser graphs. For example, in Figure 7 (top) consider the graphs Random_1, Random_2, and Random_5 which are respectively ranging from sparse to dense graphs. This figure as well as the other figures in this section indicate that our approximation is more accurate for the sparser graphs.

Our numerical results show our approximation $I_1/I_2$ (given by $I_{1/2,Gamm}$) is accurate (close to 1) even for the cases that $v_D$’s are not close to 0. In fact, both sets of values for $I_{1/2,MC}$ and $I_{1/2,Gamm}$ seem to be affected by the size of $v_D$ but are reasonably close to 1, whatever the value of $v_D$. For $p > 30$, we cannot verify the accuracy of our approximation directly by computing $I_{1/2,MC}$ and $I_{1/2,Gamm}$ because of the limitations of the Monte Carlo method of Atay-Kayis and Massam (2005). So, in the next section, for graphs with up to 150 nodes, we will verify the performance of our approximation in the search algorithm represents in Section 2.1.

6. Simulation Study for High-Dimensional Graphs

We perform Bayesian structure learning on simulated data from high-dimensional graphs using the BDMCMC search algorithm, represented in Algorithm 1. We use our approximation (8) within Algorithm 1 and we call it BDMCMC-Gamm. For the sake of comparison, we also evaluate the ratio of normalizing constants, within the BDMCMC search algorithm, using the exchange algorithm which is represented in Algorithm 2, we call it BDMCMC-DMH; this algorithm can be considered as the state-of-the-art. Both approaches are implemented in the BDgraph R package (Mohammadi and Wit 2019; Mohammadi, Wit, and Dobra 2021) in the function bdgraph().

We consider four following graph structures:

1. **Scale-free**: A graph that has a power-law degree distribution generated by the Barabási-Albert algorithm (Albert and Barabási 2002).
2. **Random_p**: A graph in which edges are randomly generated from independent Bernoulli distributions with a mean equal to $p$.
3. **Random_2p**: The same as the Random_p graph with a mean equal to $2p$. 

For the case, $\delta = 10$, the values of $I_{1/2,MC}$ and $I_{1/2,Gamm}$ are represented in Figure 8 for $p = 20$, and Figures G.3 and G.4 for $p = \{10, 30\}$ of the Supplementary File. In all cases, we see that $I_{1/2,MC}$ and $I_{1/2,Gamm}$ cover the same range of values and their medians are between 0.995 and 1, giving pretty low relative errors of less than 0.005.

Our simulations indicate that our approximation is more accurate for the sparser graphs. For example, in Figure 7 (top) consider the graphs Random_1, Random_2, and Random_5 which are respectively ranging from sparse to dense graphs. This figure as well as the other figures in this section indicate that our approximation is more accurate for the sparser graphs.

Our numerical results show our approximation $I_1/I_2$ (given by $I_{1/2,Gamm}$) is accurate (close to 1) even for the cases that $v_D$’s are not close to 0. In fact, both sets of values for $I_{1/2,MC}$ and $I_{1/2,Gamm}$ seem to be affected by the size of $v_D$ but are reasonably close to 1, whatever the value of $v_D$. For $p > 30$, we cannot verify the accuracy of our approximation directly by computing $I_{1/2,MC}$ and $I_{1/2,Gamm}$ because of the limitations of the Monte Carlo method of Atay-Kayis and Massam (2005). So, in the next section, for graphs with up to 150 nodes, we will verify the performance of our approximation in the search algorithm represents in Section 2.1.
4. Cluster: A graph in which the number of clusters is \( \lfloor p/20 \rfloor \).
Each cluster has the same structure as the Random_p graph.

For each graph, we consider various scenarios based on the number of nodes \( p \in \{50, 100, 150\} \) and the sample size \( n \in \{p, 2p\} \). We draw \( n \) independent samples from the normal distribution \( \mathcal{N}_p(0, K) \). We consider \( \delta = 3 \) as the worst-case value for our approximation (see Sections 4.1 and 5.1).

For each scenario, we run Algorithm 1 using our approximation (8) as well as Algorithm 2 which is based on an exchange algorithm. The number of iterations is 100,000 with 60,000 iterations as burn-in. To evaluate the performance of both algorithms we use ROC curves, based on model averaging, by computing true and false-positive rates for each of 50 replicated datasets and then by averaging over the 50 replicates.

Figure 9 represents the ROC curves for the cases \( p = 150 \) with \( n \in \{150, 300\} \). For \( p = \{50, 100\} \), the ROC curves are, respectively, in Figures G.5 and G.6 of the supplementary file. As we can see, in almost all cases, the performance of the BDM-CMC algorithm based on both approximations is the same. In a few cases, the BDMCMC algorithm using our approximation (8) performs slightly better than the BDMCMC algorithm using the exchange algorithm: this happens especially when \( p \) is large, for example, when \( p = 150 \) and \( n = 150 \). This discrepancy can be due to the convergence issue of the exchange algorithm in high-dimensional graphs.

The execution times for both algorithms are represented on the right-hand side of Figure 1. It indicates the computational gain of using our approximation within the search algorithm. For example, in the case \( p = 150 \), the BDMCMC algorithm using our approximation is more than 3 times faster than the BDMCMC algorithm using the exchange algorithm.

In summary, our simulation study shows that, from an accuracy point of view, the BDMCMC algorithm using our approximation (8), performs well especially for high-dimensional sparse graphs, which is the case for many real-world applications. From a computational point of view, using our approximation speeds up the BDMCMC search algorithm for the models with high-dimensional graphs.

7. Conclusion
In this article, we represent a search algorithm in which the ratio of the prior normalizing constants of G-Wishart is carried out by our approximation in Equation (8). Using our approximation allows for Bayesian structure learning to avoid the sampling-based methods as computationally expensive updates within
Figure 9. ROC curves for the BDMCMC algorithm with our approximation (8) (BDMCMC-Gamm) and BDMCMC algorithm with exchange algorithm (BDMCMC-DMH), over 50 replications. Here, $p = 150$, $n \in \{150, 300\}$, and 4 different graph structures.

the search algorithm. We give theoretical results to justify this approximation when certain assumptions are satisfied. Then, as importantly, we show, through numerical experiments that the assumptions are reasonably satisfied and yield a good accuracy of the approximation. In Theorem 1, we consider the specific case where the paths between the endpoints are disjoint. Though this case is unrealistic in practice, it is interesting because we can obtain an analytic lower bound to the ratio $I_1/I_2$, which is a function of $\delta$ and the number of paths and their length. We see that the actual accuracy is much better than that given by the lower bound.

In the realistic and general cases where the paths are not necessarily disjoint, we give an alternative expression in Theorem 2 for the ratio $I_1/I_2$, then an approximation to this expression. We show that when the variance $v_D$ is small, then the accuracy is good. When performing structure learning in practice, one will not verify this assumption any more than one would verify that the paths are disjoint. But we do examine a large array of standard graphs and verify numerically that the assumption of $v_D$ small is satisfied in most cases. Whatever the value of $v_D$, the accuracy of the approximation $I_1/I_2 \approx 1$, or equivalently of the approximation in Equation (8), is very good. We do so by direct computation for graphs of size $p \leq 30$. Due to the limitations of the Monte Carlo method to compute $I_1/I_2$, we cannot perform these direct computations for $p > 30$. In that case, we perform structure learning on graphical models with up to 150 variables and obtain the good results of Section 6. We should emphasize here that we stopped at $p = 150$ because, beyond this size, the state-of-the-art algorithms become computationally expensive but the BDMCMC search algorithm with our approximation (8) can scale up to higher dimensions still.

The accuracy of our approximation (8) depends on (i) the value of $\delta$ (scale parameter of G-Wishart) (ii) the structure of the graphs, more specifically its sparsity. We illustrate it in the simulations of Sections 4.1 and 5.1. It also can be interpreted from Theorems 1 and 2. The accuracy of our approximation is increased by increasing the values of $\delta$. Thus, our recommendation in practice is to choose, preferably, the value of $\delta$ higher than 10. For the case of graph structure, the accuracy of our approximation depends on the sparsity of the graphs. Our results indicated that our approximation is more accurate for the sparser graphs, as is indicated in the simulations of Section 5.1. Since in real-life applications the underlying graphs are not dense (mainly sparse) is safe to use our approximation in practice.

In conclusion, we think that our approximation can be safely adopted in the search algorithm to replace the sampling-based methods such as the exchange algorithm. Finally, we also proved that $I_1/I_2 \leq 1$. It shows that our approximation (8) yields a Bayes factor which favours $G-e$ compared to $G$ so that a model search using our approximation might lead to errors on the side of sparser graphs.

Acknowledgments
We are grateful to the editor, Dr. Wang, and two referees for their insightful comments that have resulted in significant improvements to the article. This work is dedicated to the memory of Hélène Massam who passed away on August 22, 2020.

Software
The BDMCMC search algorithm with the proposed approximation is implemented in the R package BDgraph which is freely available from the Comprehensive R Archive Network (CRAN) at http://cran.r-project.org/packages=BDgraph.

Supplementary Materials
Supplementary material for this article is available online. The supplementary material contains technical proofs and additional simulation results.

ORCID
Reza Mohammadi https://orcid.org/0000-0001-9538-0648

References
Albert, R., and Barabási, A. (2002), “Statistical Mechanics of Complex Networks,” Reviews of Modern Physics, 74, 47. [1355]
Atay-Kayis, A., and Massam, H. (2005), "A Monte Carlo Method for Computing the Marginal Likelihood in Nondecomposable Gaussian Graphical Models," *Biometrika*, 92, 317–335. [1346,1348,1349,1350,1352,1353,1354,1355,1356]

Cheng, Y., and Lenkoski, A. (2012), "Hierarchical Gaussian Graphical Models: Beyond Reversible Jump," *Electronic Journal of Statistics*, 6, 2309–2331. [1345,1348]

Dempster, A. P. (1972), "Covariance Selection," *Biometrics*, 157–175. [1347]

Dobra, A., and Lenkoski, A. (2011), "Copula Gaussian Graphical Models and Their Application to Modeling Functional Disability Data," *The Annals of Applied Statistics*, 5, 969–993. [1345]

Green, P. (1995), "Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination," *Biometrika*, 82, 711–732. [1347]

Green, P. J. (2003), "Trans-Dimensional Markov Chain Monte Carlo," In Highly Structured Stochastic Systems (eds P. J. Green, N. L. Hjort and S. Richardson), pp. 179–196. Oxford: Oxford University Press. [1345]

Hinne, M., Lenkoski, A., Heskes, T., and van Gerven, M. (2014), "Efficient Sampling of Gaussian Graphical Models Using Conditional Bayes Factors," *Statistics*, 3, 326–336. [1345,1348,1349]

Uhler, C., A. Lenkoski, and D. Richards (2018). Exact formulas for the normalizing constants of Wishart distributions for graphical models. *The Annals of Statistics* 46, 90–118. [1345,1348,1350]

van den Boom, W., Beskos, A., and De Iorio, M. (2021), "The g-Wishart Weighted Proposal Algorithm: Efficient Posterior Computation for Gaussian Graphical Models," arXiv:2108.01308. [1345]