algorithms

MDPI

Article

EPSOM-Hyb: A General Purpose Estimator of Log-Marginal Likelihoods with Applications in Probabilistic Graphical Models

Eric Chuu [1], Yabo Niu [2], Anirban Bhattacharya [1,*] and Debdeep Pati [1]

[1] Department of Statistics, Texas A&M University, 3143 TAMU, College Station, TX 77843, USA; eric.chuu71996@gmail.com (E.C.); debdeep@stat.tamu.edu (D.P.)
[2] Department of Mathematics, University of Houston, 4800 Calhoun Rd., Houston, TX 77004, USA; yniu4@central.uh.edu
* Correspondence: anirbanb@stat.tamu.edu

Abstract: We consider the estimation of the marginal likelihood in Bayesian statistics, with primary emphasis on Gaussian graphical models, where the intractability of the marginal likelihood in high dimensions is a frequently researched problem. We propose a general algorithm that can be widely applied to a variety of problem settings and excels particularly when dealing with near log-concave posteriors. Our method builds upon a previously posited algorithm that uses MCMC samples to partition the parameter space and forms piecewise constant approximations over these partition sets as a means of estimating the normalizing constant. In this paper, we refine the aforementioned local approximations by taking advantage of the shape of the target distribution and leveraging an expectation propagation algorithm to approximate Gaussian integrals over rectangular polytopes. Our numerical experiments show the versatility and accuracy of the proposed estimator, even as the parameter space increases in dimension and becomes more complicated.

Keywords: deterministic approximation; expectation propagation; graphical model; junction tree; marginal likelihood; partition function

Citation: Chuu, E.; Niu, Y.; Bhattacharya, A.; Pati, D. EPSOM-Hyb: A General Purpose Estimator of Log-Marginal Likelihoods with Applications in Probabilistic Graphical Models. *Algorithms* **2024**, *17*, 213. https://doi.org/10.3390/a17050213

Academic Editor: Frank Werner

Received: 23 April 2024
Revised: 10 May 2024
Accepted: 12 May 2024
Published: 15 May 2024

1. Introduction

In Bayesian inference, many tasks, such as model selection, averaging, and comparison, rely on being able to compute the marginal likelihood (or model evidence) in order to form the Bayes factor. The marginal likelihood is the normalizing constant of a posterior distribution, which plays a particularly important role in probabilistic graphical models. Suppose γ is a general Gibbs distribution defined on $\mathcal{U} \subseteq \mathbb{R}^d$, $\gamma(u) = \mathcal{Z}^{-1} e^{-\Psi(u)}$, with

$$\mathcal{Z} = \int_{\mathcal{U}} e^{-\Psi(u)} du < +\infty, \quad \mathcal{U} \subseteq \mathbb{R}^d. \tag{1}$$

In a Bayesian setup, $\Psi : \mathbb{R}^d \to \mathbb{R}$ is the negative log posterior. Since Ψ is typically complicated, and the space over which we are integrating tends to be high-dimensional, the resulting integral is usually intractable. This calculation is exacerbated further for Gaussian graphical models (GGM) [1,2] because the integral is taken over a specialized subset of positive definite matrices.

Generic algorithms for approximating the marginal likelihood include Laplace's method [3], the harmonic mean estimator [4,5], the corrected arithmetic mean estimator [6], annealed importance sampling [7], Chib's method [8], (warp) bridge sampling [9,10], and nested sampling [11]. More recent developments include the integrated nested Laplace approximation [12] and the expectation propagation–approximate Bayesian computation algorithm [13]. While these methods can be applied to many marginal likelihood estimation problems, they do not provide a straightforward way to deal with the specificity of GGMs and the positive definite restriction on the precision matrices. Additionally, many of these

aforementioned methods tend to rely on high-quality MCMC samples and have numerous problem-specific model settings that limit the overall practicality.

While there have been developments that specifically target the marginal likelihood of GGMs [14], the stringent prior restrictions (inverse-Wishart and hyper-inverse Wishart) prevent many of these approaches from being used in broader contexts. Furthermore, there exists substantial literature on inference for both decomposable [15] and non-decomposable [16–18] graphs; however, the need for more general and scalable algorithms is ever present. This has consequently led to dedicated methods for sampling from the G-Wishart distribution and conducting model comparison [19–21]. Also relevant is the availability of the G-Wishart normalizing constant in closed form [22], but the viability of these results is limited when considering most high-dimensional graphs of practical interest. Bhadra et al. [23] propose an application of Chib's method and a telescoping decomposition of the precision matrix that simplifies the ensuing marginal likelihood calculations, but the main advantages of this approach are seen with element-wise priors, and in most cases, the time complexity is on par with other GGM-specific methods.

A recent approach that combines some of the probabilistic ideas from MCMC-based methods while drawing inspiration from quadrature is the hybrid estimator [24]. This method moves away from an over-reliance on MCMC samples, as these can be time-consuming to obtain if the likelihood is expensive to evaluate. Rather, the MCMC samples are utilized during a preliminary step to learn a partition of the parameter space in order to identify regions of posterior concentration. By making local approximations to the negative log posterior over these partition sets, the the hybrid algorithm simplifies the integral and forms a piecewise estimate of the marginal likelihood. The high-probability partition of the parameter space yields multiple benefits. First, the areas of the parameter space that admit finer partitions encourage precise approximations to the log posterior. More importantly, this partitioning routine redirects attention away from regions that have little to no contribution to the posterior distribution, saving both time and computation.

The hybrid estimator establishes the foundation for a promising way to estimate the marginal likelihood, and the experimental results demonstrate its competitiveness with other well-known estimators in various problem settings [24]. Moreover, the methodology's generality makes it convenient to refine the algorithm so that it can be applied to more specific problem setups. With the ultimate goal of accurately computing the marginal likelihood for GGMs, we introduce an alternative parametrization of precision matrices and restrict our scope to target the normalizing constants of a specific class of densities—unimodal densities that are approximately log-concave around the mode. In this paper, we leverage core ideas of the hybrid estimator in conjunction with higher order approximations to the negative log posterior for accuracy, as well as expectation propagation techniques for scalability. This results in a novel estimator that is suitably equipped for high-dimensional problems. We call the estimator that arises from these modifications the EP-guided second-order modified hybrid estimator (EPSOM-Hyb).

After verifying the accuracy of EPSOM-Hyb in initial experiments for decomposable graphs, we further extend the methodology to better handle non-decomposable GGMs. By incorporating junction tree (JT) representations of connected graphs, we enhance the computational efficiency of the marginal likelihood calculation. This leads to a GGM-specific estimator, denoted EPSOM-HybJT. Our contribution is multifaceted; despite the introduction of higher-order terms that inherently bring additional computational challenges to the hybrid estimator framework, we maintain practical utility in the EPSOM-Hyb methodology. This approach leads to an estimator that not only excels in highly specific problem settings, but also outperforms well-established estimators. Our experiments indicate that EPSOM-HybJT is more accurate and more than 100 times faster in higher dimensions than other viable estimators.

The outline of the paper is as follows. In Section 2, we provide the background for the hybrid estimator so that the modifications that we propose in Section 3 to formulate the EPSOM-Hyb estimator have relevant context. In Section 4, we investigate the performance

of the EPSOM-Hyb estimator for GGMs, and we also develop the EPSOM-HybJT estimator. In Section 5, we conclude and briefly discuss future work.

2. The Hybrid Estimator

We first review the salient details of the hybrid estimator as a means of providing background on the existing work and establishing context for our development of the EPSOM-Hyb estimator. As an initial step, the hybrid algorithm learns a dyadic partition of the parameter space that helps identify high-probability regions of the target distribution. In order to accomplish this, we form covariate–response pairs, $\{(u_j, \Psi(u_j))\}_{1 \le j \le J}$, using MCMC samples from the target distribution γ and evaluating them with Ψ. Clearly, we make the assumption of being able to sample from γ and evaluate Ψ, but both of these are basic requirements in many MCMC-based algorithms. We can then use a regression tree algorithm to obtain optimal splits of the parameter space, where each of the partition sets is the hyper-rectangle that defines the corresponding leaf node in the fitted tree. The hybrid estimator uses the classification and regression tree (CART) algorithm [25], which returns a partition of the parameter space that can be further restricted to the compactification A of $\mathcal{U}$ by setting A to be bounding box defined by the range of the posterior samples: $A = \prod_{l=1}^{d} [\min_j \{u_j^{(l)}\}, \max_j \{u_j^{(l)}\}]$, where $u_j^{(l)}$ is the l-th coordinate of the j-th sample. Thus, the resulting partition $\mathcal{A}$ is a dyadic partition of the compact set A. We work with a compactification of the parameter space as a means of eliminating low-probability regions of the domain whose contributions to the integral in Equation (1) are negligible. This is particularly useful in Bayesian contexts where the posterior concentrates with increasing sample size [26,27].

Next, we formulate the piecewise estimator to Ψ defined over the compact set $A \subseteq \mathcal{U}$,

$$\Psi(u) \approx \widehat{\Psi}(u) = \sum_{k=1}^{K} \widehat{\Psi}_k(u) \mathbb{1}_{A_k}(u). \tag{2}$$

where $\mathcal{A} = \{A_1, \ldots, A_K\}$, $A = \bigcup_{k=1}^{K} A_k$, and $A_k \cap A_{k'} = \emptyset$ for all $k \neq k'$. We set $\widehat{\Psi}_k(u) = c_k^\star$, a representative point in A_k that is constant in u. Since each $A_k = \prod_{l=1}^{d} [a_k^{(l)}, b_k^{(l)}]$ is a hyper-rectangle, the hybrid estimator reduces to

$$\int_A e^{-\Psi(u)}\, du \approx \int_A e^{-\widehat{\Psi}(u)}\, du = \sum_{k=1}^{K} e^{-c_k^\star} \cdot \mu(A_k) =: \widehat{\mathcal{Z}}_{\text{Hyb}} \tag{3}$$

Here, $\mu(B) = \int_B 1\, du$ denotes the d-dimensional volume of a set B.

3. The Epsom-Hyb Estimator

In the discourse that follows, we propose some key modifications to the hybrid algorithm to develop a novel estimator better suited for tackling higher-dimensional problems. For the initial development, we do not restrict our analysis to posterior distributions and illustrate the modified algorithm in the general case of log-concave target densities. Our additional assumption about the shape of Ψ is employed only to ensure that the Hessian is positive definite everywhere and can, in fact, be relaxed to unimodal densities that are approximately log-concave in a suitable neighborhood around the mode. This assumption is widely satisfied by Bayesian posteriors in regular parametric models, and due to the Bernstein–von Mises phenomenon, most regular posterior distributions are approximately log-concave with sufficiently large sample size [28].

As before, we work with a compactification of the parameter space, for which we have a dyadic partition $\mathcal{A}$ obtained through the same tree-fitting procedure as before. However,

instead of the constant approximation to Ψ within each partition set A_k in $\mathcal{A}$, we first retain the general estimator given in Equation (2),

$$\mathcal{Z} \approx \int_A e^{-\Psi(u)}\, du \approx \int_A e^{-\widehat{\Psi}(u)}\, du = \sum_{k=1}^{K} \int_{A_k} e^{-\widehat{\Psi}_k(u)} du. \tag{4}$$

Below, we highlight three fundamental features of the EPSOM-Hyb algorithm that distinguish it from the hybrid algorithm, followed by detailed discussions of each step. See Algorithm 1 for a formal statement of the EPSOM-Hyb estimation procedure.

- In Equation (2), we take $\widehat{\Psi}_k$ to be a local approximation to Ψ constructed using a second-order Taylor expansion of Ψ around a representative point $u_k \in A_k$.
- The second-order Taylor approximation complicates the calculation of the integrals in Equation (4). We address the intractability of the resulting integral using an expectation propagation (EP) algorithm that targets high-dimensional Gaussian integrals.
- We exploit the unimodality of γ to identify suitable points within each partition set around which we perform the Taylor expansion required for $\widehat{\Psi}_k$.

Algorithm 1: EPSOM-Hyb

Input : Sampler for γ, functions for evaluating Ψ, $\nabla\Psi$, $\nabla^2\Psi$.
Output: Estimate of the logarithm of the normalizing constant of γ.

Sample $u_1, \ldots, u_J \sim \gamma$.

Fit a CART model, $\mathcal{T}$, to $(u_1, \Psi(u_1)), \ldots, (u_J, \Psi(u_J))$.

Extract the partition $\mathcal{A} = \{A_1, \ldots, A_K\}$ from $\mathcal{T}$ of the bounding box A of $\mathcal{U}$.

Calculate the global mode, u_0, of γ.

for $k \in \{1, \ldots, K\}$ **do**

$u_k \leftarrow \text{argmin}_{u \in A_k} ||u - u_0||_1$.

$\lambda_k \leftarrow \nabla\Psi(u_k)$.

$H_k \leftarrow \nabla^2\Psi(u_k)$.

$C_k \leftarrow (2\pi)^{d/2} |H_k|^{-1/2} \exp\left(\frac{1}{2}\left(u_k' H_k^{-1} u_k - 2\lambda_k' u_k + \lambda_k' H_k^{-1} \lambda_k\right)\right)$.

$b_k \leftarrow H_k u_k - \lambda_k$.

$G_k \leftarrow \int_{A_k} \mathcal{N}\left(u \mid H_k^{-1} b_k, H_k^{-1}\right) du$.

$\widehat{\mathcal{Z}}_k \leftarrow C_k \cdot G_k$.

end

return $\log \widehat{\mathcal{Z}} = \texttt{log-sum-exp}(\log \widehat{\mathcal{Z}}_1, \ldots, \log \widehat{\mathcal{Z}}_K)$.

3.1. Local Approximation Using a Taylor Expansion

We revisit Equation (2) and consider the following piecewise quadratic approximation to Ψ,

$$\Psi(u) \approx \sum_k \left[\Psi(u_k) + (u - u_k)'\nabla\Psi(u_k) + \frac{1}{2}(u - u_k)'\nabla^2\Psi(u_k)(u - u_k)\right] \mathbb{1}_{A_k}(u), \tag{5}$$

where u_k is a representative point of A_k. This improves upon the initial idea of the piecewise constant approximation in the hybrid methodology by introducing a second-order Taylor approximation. By introducing higher-order terms, we trade the convenient simplification of the integral in Equation (3) for increased local accuracy of the approximation. Upon

exponentiating the approximation in Equation (5), we observe that the second exponential in the summation below is proportional to a Gaussian density,

$$e^{-\widehat{\Psi}(u)} = \sum_k \exp\Big\{ -\Psi(u_k) + u_k'\lambda_k - \tfrac{1}{2}u_k'H_ku_k \Big\} \exp\Big\{ -\tfrac{1}{2}u'H_ku + (H_ku_k - \lambda_k)'u \Big\} \mathbb{1}_{A_k}(u),$$

where $\lambda_k := \nabla\Psi(u_k)$ and $H_k := \nabla^2\Psi(u_k)$ represent the gradient vector and the Hessian matrix of Ψ evaluated at u_k, respectively. Since γ is log-concave, the Hessian matrix is positive definite, and hence invertible. Integrating the exponential above, we arrive at the following approximation to $\mathcal{Z}$:

$$\int_A e^{-\Psi(u)}du \approx \int_A e^{-\widehat{\Psi}(u)}du = \sum_k C_k \int_{A_k} \mathcal{N}\Big(u \mid H_k^{-1}b_k, H_k^{-1}\Big)du =: \widehat{\mathcal{Z}}_{\text{EPSOM}-\text{Hyb}}. \quad (6)$$

Here, $C_k = \exp\big(-\Psi(u_k) + \lambda_k'u_k - 0.5u_k'H_ku_k + 0.5m_k'H_km_k\big)$, $b_k = H_ku_k - \lambda_k$, and $m_k = H_k^{-1}b_k$. Before $\widehat{\mathcal{Z}}_{\text{EPSOM}-\text{Hyb}}$ can serve as a viable estimator for $\mathcal{Z}$, we must address the remaining questions of how to determine the value of each $u_k \in A_k$ and how to compute the intractable Gaussian integrals in Equation (6). In the next two sections, we provide scalable methods to accomplish these tasks. It is worth noting that the original hybrid estimator does not face the same intractability issue due to the simplistic nature of the constant approximation used in Equation (3). Our incorporation of higher-order terms complicates the subsequent calculation, which necessitates a scalable algorithm like EP to compute the truncated Gaussian integrals.

3.2. Estimating Truncated Gaussian Probabilities

Despite the prevalence of Gaussian densities in statistical modeling, multivariate Gaussian probabilities remain difficult to compute, as they typically require high-dimensional integration. Numerical integration [29] is a common solution, but it is prohibitively expensive in the number of points required, thus preventing it from being a scalable solution in the high-dimensional space in which we wish to operate. A more recent alternative that specifically targets truncated normal distributions is the minimax tilting method [30], and while we used this method in the initial development of our algorithm, it failed to produce accurate results beyond trivial settings. This ultimately led us to consider more computationally efficient integration techniques, such as expectation propagation [31], which is widely used to compute approximate integrals. To better understand how the expectation propagation (EP) algorithm can be applied to the intractable Gaussian integral in the previous section, we lay out some preliminary groundwork for the EP algorithm. Starting with the Gaussian distribution $p_0(x) = \mathcal{N}(x \mid m, K)$, we define the un-normalized truncated distribution $p(x) = p_0(x)\mathbb{1}_{\mathcal{A}}(x)$. The Gaussian probability of interest can be written as:

$$F(\mathcal{A}) = \int_{\mathcal{A}} p_0(x)dx = \int p(x)dx. \quad (7)$$

Cunningham et al. [32] propose a framework for approximating $F(\mathcal{A})$ that consists of two main approximations. The first approximation is a tractable distribution q that minimizes the Kullback–Leibler (KL) divergence between p and q, denoted $D(p\,||\,q)$. However, the intractability of p complicates the minimization and subsequently requires another approximation, for which we instead work with a simplified representation of p. Namely, we replace $p(x)$ with the product of a prior distribution $p_0(x)$ and factors $t_i(x)$, such that

$$p(x) = p_0(x)\prod_i t_i(x).$$

In order to facilitate tractability, we assume $t_i(x) = \mathbb{1}\{a_i < x_i < b_i\}$, where x_i is the i-th coordinate of x, and a_i, b_i are simply the lower and upper bounds of integration, respectively. Then, the target integral in Equation (7) can be written as

$$F(\mathcal{A}) = \int p(x)dx = \int p_0(x) \prod_{i=1}^{d} t_i(x_i)dx_i.$$

We proceed to approximate each of the intractable factors t_i with a tractable, un-normalized Gaussian $\tilde{t}_i(x)$, which produces the final approximation q of p. More specifically, we take q to mirror the product form of p,

$$q(x) = p_0(x) \prod_i \tilde{t}_i(x) = p_0(x) \prod_i \tilde{Z}_i \mathcal{N}\left(x \mid \tilde{\mu}_i, \tilde{\sigma}_i^2\right) = Z\mathcal{N}(x \mid \mu, \Sigma),$$

where the parameters of these unnormalized Gaussian distributions, $\{\tilde{\mu}_i, \tilde{\sigma}_i^2, \tilde{Z}_i\}$, admit closed form updates that are the result of an iterative moment matching scheme. From this, we observe that by estimating the normalizing constant of q, we also obtain an approximation for the normalizing constant of the target distribution p. See Equations (21)–(23) from Cunningham et al. [32] for the closed-form updates for each of these parameters and more details regarding the relevant notation. Essentially, the EP algorithm iteratively constructs the approximating distribution $q(x)$ to minimize $D\left(t_i q^{\setminus i} \,||\, \tilde{t}_i q^{\setminus i}\right)$, which in turn approximately minimizes $D(p \,||\, q)$. Here, $q^{\setminus i}(x) = q(x)/\,\tilde{t}_i(x)$ is defined as the cavity distribution. By running the EP algorithm to convergence, we can calculate the following mean and covariance parameters of the normal distribution q,

$$\mu = \Sigma\left(K^{-1}m + \sum_{i=1}^{d} \frac{\tilde{\mu}_i}{\tilde{\sigma}_i^2} e_i\right), \quad \Sigma = \left(K^{-1} + \sum_{i=1}^{d} \frac{1}{\tilde{\sigma}_i^2} c_i c_i'\right)^{-1},$$

where e_i is the i-th standard basis vector. With this, we also obtain a closed-form expression for the normalizing constant of q,

$$\begin{aligned}\log Z = &-\frac{1}{2}\left(m'K^{-1}m + \log|K|\right) + \sum_{i=1}^{d}\left(\log \tilde{Z}_i - \frac{1}{2}\left(\frac{\tilde{\mu}_i^2}{\tilde{\sigma}_i^2} + \log \tilde{\sigma}_i^2 + \log(2\pi)\right)\right) \\ &+ \frac{1}{2}\left(\mu'\Sigma^{-1}\mu + \log|\Sigma|\right),\end{aligned}$$

which approximates the Gaussian probability $F(\mathcal{A})$ in Equation (7). It is worth noting that the algorithm is still valid for arbitrary factors $t_i(x)$, albeit with different parameter updates. Our simplistic choice of $t_i(x)$ to be the indicator function defined over the constant lower and upper limits of integration reflects the rectangular nature of the partition sets used in the Hybrid-EP estimator.

With this, recall the Gaussian integral in Equation (6),

$$\int_{A_k} \mathcal{N}\left(u \mid H_k^{-1}b_k, H_k^{-1}\right) du. \tag{8}$$

Taking $p_0(u) \equiv \mathcal{N}\left(u \mid H_k^{-1}b_k, H_k^{-1}\right)$, we observe that the above integral is exactly the one given in Equation (7). Thus, we can directly use the EPMGP algorithm to obtain an estimate for the Gaussian probability in Equation (8). Another benefit of using the EPMGP algorithm is its exceptional accuracy when the constraint set is rectangular, which coincides with our setup, where $A_k = \prod_{l=1}^{d} [a_k^{(l)}, b_k^{(l)}]$ is a d-dimensional hyper-rectangle.

3.3. Selecting the Candidate Point in Each Partition Set

The final piece of EPSOM-Hyb is the point u_k used in the piecewise Taylor approximation $\widehat{\Psi}$ in Equation (5). In our unimodal setup, a natural choice for each partition set's representative point is one that is closest to the global mode of γ. More specifically, if u_0 is the global mode, then $u_k = \text{argmin}_{u \in A_k} \|u - u_0\|_1$. Conveniently, we can obtain the global mode of γ using Newton's method with little additional effort because we already have expressions for the gradient and Hessian of Ψ as part of the approximation in Equation (6). With this, all components of EPSOM-Hyb are accounted for and can be summarized in Algorithm 1.

4. Application to Gaussian Graphical Models

Next, we investigate the performance of EPSOM-Hyb on marginal likelihood estimation for GGMs. Let $G = (V, E)$ be an undirected graph with vertex set $V = \{1, \ldots, p\}$ and edge set E. Define $\mathbb{S}^p$ as the set of symmetric $p \times p$ matrices and $\mathbb{S}^p_{\succ 0}$ as the cone of positive definite $p \times p$ matrices in $\mathbb{S}^p$. Let $X \sim \mathcal{N}(\mu, \Sigma)$, $\Sigma^{-1} \in \mathbb{S}^p_{\succ 0}(G)$, where $\mathbb{S}^p_{\succ 0}(G) = \{M = (M_{ij}) \in \mathbb{S}^p_{\succ 0} \mid M_{ij} = 0, \forall (i,j) \notin E\}$. Then, the likelihood can be written as follows,

$$L(\Sigma) = (2\pi)^{-np/2} \det(\Sigma)^{-n/2} e^{-\operatorname{tr}\left(\Sigma^{-1} B\right)/2}, \; B = \sum_{i=1}^{n} x_i x_i'. \tag{9}$$

Further, X satisfies the GGM with graph G, where G dictates the conditional dependence structure and restricts the sparse concentration matrix $\Omega = (\omega_{ij})_{p \times p} = \Sigma^{-1}$ so that $(i,j) \in E$ if and only if $\omega_{ij} \neq 0$, and $x^{(i)}$ and $x^{(j)}$ are conditionally independent if and only if $\omega_{ij} = 0$. Hence, an undirected graphical model corresponding to G restricts the inverse covariance matrix Ω to a linear subspace of the cone of positive definite matrices.

We first consider decomposable GGMs where the tractability of the marginal likelihood makes it easy to evaluate estimates. In Section 4.2, we broaden our focus to general, non-decomposable graphs. To address the additional computational challenges that accompany non-decomposable graphs and high-dimensional parameter spaces, we propose an extension to EPSOM-Hyb that facilitates dimension reduction and drastically reduces the time complexity of the normalizing constant calculation. In the following GGM experiments, we refer to the estimator from Atay-Kayis and Massam [17] as GNORM, which is well-established for general graphs and can be computed easily using `BDgraph` [33].

4.1. Hyper Inverse-Wishart Induced Cholesky Factor Density

We first consider the case where G is decomposable. The hyper-inverse Wishart (HIW) distribution [34] for $\Sigma = \Omega^{-1}$ and the corresponding induced class of distributions [14] for Ω are attractive choices for priors. Given G, we place a hyper-inverse Wishart prior $\text{HIW}_G(\delta, \Lambda)$ on $\Omega = \Sigma^{-1}$, where $\delta > 2$ is the degrees of freedom and $\Lambda \in \mathbb{S}^p_{\succ 0}$ is fixed. The HIW distribution is defined over the cone of $p \times p$ positive definite matrices, with the corresponding density:

$$f(\Omega \mid G) \propto |\Omega|^{(\delta-2)/2} e^{-\operatorname{tr}(\Omega\Lambda)/2}. \tag{10}$$

Despite being able to sample from the posterior distribution, $\text{HIW}_G(\delta + n, \Lambda + B)$, we cannot directly carry out the EPSOM-Hyb algorithm. Since samples are drawn from a sub-manifold of $\mathbb{R}^{p \times p}$, proceeding to obtain a partition over $\mathbb{R}^{p \times p}$ does not guarantee that a given point in the partition can be reconstructed to form a valid covariance matrix. As such, we circumvent this issue by working with an alternative representation of Σ given by the Cholesky decomposition $\Sigma^{-1} = \Omega = \phi' \phi$. Provided that the vertices of G are enumerated according to a perfect vertex elimination scheme, the upper triangular matrix ϕ observes the same sparsity as Ω. Consequently, we need only compute the induced prior on the nonzero elements of ϕ, which, together with the likelihood in Equation (9), gives us an

explicit expression for the negative log posterior Ψ. The determinant of the Jacobian matrix J of the transformation $\Omega \to \phi$ is given by $|J| = 2^p \prod_{i=1}^{p} \phi_{ii}^{\nu_i+1}$, where the i-th row of ϕ has exactly $\nu_i + 1$ nonzero elements [14]. Further, the distribution of Σ has the strong hyper-Markov property [35], so we can ascertain mutual independence of the rows of ϕ. Then, we can specify the joint density of the free elements of ϕ as follows,

$$\pi(\phi) = \left[\prod_{i=1}^{p} \frac{2^{-(\delta+\nu_i)/2}}{\Gamma((\delta+\nu_i)/2)} \phi_{ii}^{\delta+\nu_i-2} e^{-\frac{1}{2}\phi_{ii}^2} (2\phi_{ii}) \right] \times \left[\prod_{(r,s):s>r,(v_s,v_r)\in E} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\phi_{rs}^2} \right]. \quad (11)$$

Recall that we also require expressions for the gradient and Hessian of Ψ, both of which are provided in Appendix D. Putting these ideas together, we can employ the EPSOM-Hyb estimation framework by drawing samples from the posterior distribution, taking the Cholesky factor of each sample, and computing $\Psi(\phi)$ using the likelihood in Equation (9) and the prior in Equation (11). While this procedure appears to be quite simple, the implications are significant; if we have a different prior on Σ for which we can do the posterior computation, all other aspects of the algorithm would remain the same. All that is required is a way to sample from the posterior of Ω and an expression for the Jacobian of the corresponding transformation.

In our simulations, we emulate a high-dimensional setting by stacking the adjacency matrix of the decomposable graph G_9 in Figure 1 multiple times along a block diagonal to construct larger graphs, e.g., $A_{G_p} = I_r \otimes A_{G_9}$, where A_{G_p} is the adjacency matrix of G_p and r is the number of times we stack the graph. For $r = 8, 10$, we form the graphs G_{72} and G_{90}, which have $d = 200, 250$ free elements, respectively. Drawing data that satisfy their corresponding GGMs and taking $\delta = 3, n = 100$, and $\Lambda = I_p$, we can sample from the posterior distribution and compute the marginal likelihood estimates. Among the generic methods mentioned in Section 1, only the bridge sampling estimator (BSE) and the warp bridge sampling estimator (WBSE) are viable and accurate methods for computing the marginal likelihood. As seen in Table 1, both BSE and WBSE are competitive with EPSOM-Hyb when $d = 200$, but the quality of the former two estimators deteriorates when $d = 250$. The relatively small error of EPSOM-Hyb in this high-dimensional graphical models setting is particularly encouraging. Interestingly, GNORM fails to produce sensible estimates for these high-dimensional graphs.

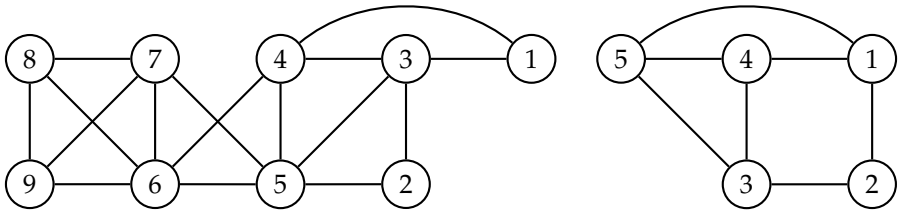

Figure 1. G_9 (**left**) , an undirected, decomposable graph, and G_5 (**right**), an undirected, non-decomposable graph.

Table 1. Mean, average error (AE), and root mean squared error (RMSE) for the approximations to the log normalizing constant of the $\text{HIW}_G(\delta + n, I_p + B)$ distribution. G_{72} has 72 vertices with 200 parameters and G_{90} has 90 vertices and 250 parameters. Estimates are reported over 100 replications, each using 1000 samples from the true posterior.

Dimension	Method	Mean	AE	RMSE
$p = 72, d = 200$	EPSOM-HybJT	−6230.59	0.41	0.45
	BSE	−6230.56	−0.74	1.93
	WBSE	−6230.59	−0.70	2.22
$p = 90, d = 250$	EPSOM-HybJT	−7881.46	0.51	0.56
	BSE	−7875.70	−5.26	6.31
	WBSE	−7875.76	−5.19	6.09

4.2. G-Wishart Prior for General Graphs

In contrast with the previous section, we now broaden the scope of examples and consider general, non-decomposable graphs. For the conditional general graph G, a popular choice for the prior is the G-Wishart prior on Ω, $\mathrm{GW}(\delta, \Lambda)$, which has the following density,

$$f(\Omega \mid G) \propto |\Omega|^{(\delta-2)/2} e^{-\mathrm{tr}(\Omega\Lambda)/2}. \tag{12}$$

Here, Ω, Λ are $p \times p$ positive definite matrices, and $\delta > 2$. While the density expression is similar to that of the HIW density, the tractability of the normalizing constant is no longer assumed. The normalizing constant that we wish to compute is of the form

$$C_G(\delta, \Lambda) = 2^{\frac{1}{2}p\delta + |E|} \cdot I_G((\delta-2)/2, \Lambda), \tag{13}$$

where $I_G(\delta, \Lambda) = \int_{\mathbb{S}^p_{\succ 0}} |\Omega|^{\delta} \exp(-\mathrm{tr}(\Omega\Lambda))\, d\Omega$. Here, $d\Omega = \prod_{i=1}^{p} d\omega_{ii} \cdot \prod_{i<j,(i,j)\in E} d\omega_{ij}$.

G-Wishart-Induced Cholesky Factor Density

We first establish some of the notations relevant to the GW density. For $G = (V, E)$, define $\mathscr{V} = \{(i,j) : i \le j \text{ where } i = j, i \in V \text{ or } (i,j) \in E\}$ and $\mathscr{W} = \{(i,j) : i, j \in V, i \le j\}$. Then, let $\bar{\mathscr{V}} = \mathscr{W} \setminus \mathscr{V}$, and $A_{p\times p} = (a_{ij})$, where $a_{ij} = 0$ if $(i,j) \in \bar{\mathscr{V}}$ or if $i = j$, and $a_{ij} = 1$ otherwise. Set k_i to be the number of 1's in the i-th column of A. Proceeding in a similar fashion as in the HIW example, we take the Cholesky decomposition of $\Omega = \phi'\phi$ and $\Lambda = (T'T)^{-1}$, where $T = (t_{ij})_{1\le i\le j\le p}$. We make an additional change of variable $\zeta = \phi T^{-1}$. The Jacobian of the first transformation $\Omega \to \phi$ is identical to the one given in Section 4.1, and the determinant of the Jacobian of the second transformation $\phi \to \zeta$ is given by $\prod_{i=1}^{p} t_{ii}^{k_i+1}$. Putting this all together, we can rewrite the normalizing constant as an integral over the free variables of $\zeta = (\zeta_{ij})_{1\le i\le j\le p}$,

$$\begin{aligned} C_G(\delta, \Lambda) = 2^p \prod_{i=1}^{p} (t_{ii}^2)^{(\delta+b_i-1)/2} \int \exp\left(-\frac{1}{2}\sum_{(i,j)\in\bar{\mathscr{V}}} \zeta_{ij}^2\right) \prod_{i=1}^{p} (\zeta_{ii}^2)^{(\delta+\nu_i-1)/2} \exp\left(-\frac{1}{2}\sum_{i=1}^{p}\zeta_{ii}^2\right) \\ \times \exp\left(-\frac{1}{2}\sum_{(i,j)\in\mathscr{V}, i\neq j} \zeta_{ij}^2\right) \prod_{i=1}^{p} d\zeta_{ii} \prod_{(i,j)\in\mathscr{V}, i\neq j} d\zeta_{ij}, \end{aligned} \tag{14}$$

where $b_i = \nu_i + k_i + 1$, $\nu_i = |\mathrm{ne}(i) \cap \{i+1, \ldots, p\}|$, and $\mathrm{ne}(i) = \{j \in V : (i,j) \in E\}$. Taking the logarithm of the integrand above, we can write the following expression for $\Psi(\zeta)$,

$$\Psi(\zeta) = C + \sum_{i=1}^{p} (\delta + \nu_i - 1)\log\zeta_{ii} - \frac{1}{2}\sum_{i=1}^{p}\zeta_{ii}^2 - \frac{1}{2}\sum_{(i,j)\in\bar{\mathscr{V}}}\zeta_{ij}^2 - \frac{1}{2}\sum_{(i,j)\in\mathscr{V}, i\neq j}\zeta_{ij}^2, \tag{15}$$

where $C = p\log(2) + \sum_{i=1}^{p}(\delta + b_i - 1)\log t_{ii}$. Another key difference between the GW and HIW setups is how the non-free elements interact with the objective function Ψ. In the case of decomposable G, the Cholesky factor ϕ observes the same sparsity pattern as the adjacency matrix of G. This simplifies the evaluation of Ψ, as Ψ can then be evaluated over the nonzero elements of ϕ. However, in the case of non-decomposable G, the sparsity in G is not necessarily reflected in the Cholesky factors. Therefore, Ψ has nonzero contribution from the non-free elements, denoted as $\zeta_{ij}, (i,j) \in \bar{\mathscr{V}}$. Using Lemma 2 [17], we can express the non-free elements, ζ_{rs} for $(r,s) \in \bar{\mathscr{V}}$ and $r < s$, recursively in terms of the free elements,

$$\zeta_{rs} = \sum_{j=r}^{s-1}\left(-\zeta_{rj}\frac{\lambda_{js}}{\lambda_{ss}}\right) - \sum_{i=1}^{r-1}\left(\frac{\zeta_{ir} + \sum_{j=i}^{r-1}\zeta_{ij}\frac{\lambda_{jr}}{\lambda_{rr}}}{\zeta_{rr}}\right)\left(\zeta_{is} + \sum_{j=i}^{s-1}\zeta_{ij}\frac{\lambda_{js}}{\lambda_{ss}}\right). \tag{16}$$

Subsequent expressions for the gradient and Hessian of $\Psi(\zeta)$ can be found in Appendix E.

4.3. Junction Tree Representation

In order to combat the computational overhead associated with larger graphs, we modify EPSOM-Hyb to first break down the graph into subgraphs according to its topology using a junction tree representation, which exists for all connected graphs. By leveraging the attractive properties of junction tree representations, we not only achieve significant computational speedup in the marginal likelihood calculation, but we can also more adequately handle general graphs. We briefly discuss the properties of decomposable versus non-decomposable graphs and how their respective junction tree representations can simplify the normalizing constant calculation. A junction tree representation of a graph decomposes the graph into a sequence of interconnecting subgraphs separated by complete subgraphs [36], which has the following form:

$$G \mapsto \mathcal{J}_G = \{P_1, S_2, P_2, S_3, \ldots, P_{m-1}, S_m, P_m\}. \tag{17}$$

In $\mathcal{J}_G$, each prime component P_i is a proper subgraph of G, each separator S_i is a complete subgraph of G, and each S_i is the intersection of P_i with all the previous components $\{P_1, P_2, \ldots, P_{i-1}\}$. If any of the prime components found by this decomposition procedure is not complete and cannot be further decomposed, then G is non-decomposable [37]. With these concepts in place, we recall the original goal of marginal likelihood estimation but look to include the intermediate step of obtaining the junction tree representation of the graph. By working with the smaller prime components, we can overcome the complexity associated with the original high-dimensional graph. Moreover, we can take advantage of the distributional properties of the complete prime components.

Concretely, suppose we have data $X = \{x_1, \ldots, x_n\}$ satisfying the GGM with a graph G and corresponding junction tree $\mathcal{J}_G$, as given in Equation (17). Let $\mathscr{P}$ and $\mathscr{S}$ represent the prime component and separator sequences, respectively. If all of the prime components are complete (cliques), we denote the clique sequence as $\mathscr{C}$. Regardless of the decomposability of G, we can use $\mathcal{J}_G$ to express the joint density of X given Σ,

$$p(X \mid \Sigma) = \frac{\prod_{P \in \mathscr{P}} p(X_P \mid \Sigma_P)}{\prod_{S \in \mathscr{S}} p(X_S \mid \Sigma_S)}, \tag{18}$$

where Σ_P and Σ_S represent the submatrices corresponding to the subgraphs P and S, respectively. We first consider the case where G is decomposable, with $\Sigma^{-1} \sim \text{HIW}_G(\delta, \Lambda)$. Then, the prior density factorizes over the cliques and separators,

$$p(\Sigma \mid G) = \frac{\prod_{C \in \mathscr{C}} p(\Sigma_C \mid G)}{\prod_{S \in \mathscr{S}} p(\Sigma_S \mid G)}. \tag{19}$$

The completeness of the prime components admits distributional properties that make the normalizing constants tractable. In particular, the prior density on Σ_C is the inverse-Wishart (IW). Putting this together with the likelihood given in Equation (18), we deduce that the marginal likelihood also factorizes over the cliques and separators as follows,

$$\begin{aligned} p(X \mid G) &= \int_{\Sigma \mid G} p(X \mid G) p(\Sigma \mid G) d\Sigma \\ &= (2\pi)^{-\frac{np}{2}} \frac{h(G, \delta, \Lambda)}{h(G, \delta + n, \Lambda + B)} \qquad (20) \\ &= (2\pi)^{-\frac{np}{2}} \frac{\prod_{C \in \mathscr{C}} w(C)}{\prod_{S \in \mathscr{S}} w(S)}. \qquad (21) \end{aligned}$$

Here, $B = \sum_i x_i x_i'$. For a decomposable graph, the factorization of the likelihood and IW prior over the cliques and separators yields a product of tractable normalizing constants. Exact formulae for $h(G, \delta, \Lambda)$, $w(C)$ and $w(S)$ can be found in Equations (A4) and (A5).

We can easily generalize the calculations above to accommodate non-decomposable graphs with GW priors. Since the GW density in Equation (12) has a similar form to the HIW density, the marginal likelihood calculation also factorizes over prime components and separators. The difference from the previous calculation is that for decomposable G, the product in Equation (21) is taken over the cliques $\mathscr{C}$, whereas in the GW case, the product is taken over prime components $\mathscr{P}$, which may or may not be complete. Since normalizing constants for the non-complete prime components are generally intractable, they typically require MCMC estimation.

Essentially, the factorized representation of the marginal likelihood induced by the junction tree representation of G splits the original calculation involving the entire graph G into sub-problems involving the prime components and separators. When $P \in \mathscr{P}$ is complete, we can rely on the closed form IW normalizing constant. For non-complete $P \in \mathscr{P}$, we take the corresponding parameters, $\Sigma_P^{-1} = \Omega_P, B_P, \Lambda_P$, and write the normalizing constant for the non-complete prime component as

$$h(P, \delta, \Lambda_P) = \int |\Omega_P|^{\frac{\delta-2}{2}} \exp(-\operatorname{tr}(\Omega_P \Lambda_P)/2) d\Omega_P.$$

Although this quantity remains intractable, it poses a lower dimensional problem for which we can use the procedure outlined in Section 4.2 along with the EPSOM-Hyb algorithm. The posterior normalizing constant $h(P, \delta + n, \Lambda_P + B_P)$ can be similarly computed. After computing the normalizing constants for each of the prime components and separators, the individual approximations are summed together to form the log normalizing constant approximation corresponding to the original graph G. These steps make up the EPSOM-HybJT algorithm, which is outlined in Algorithm 2.

4.3.1. Experiment 1: Block Diagonal Graph Structure

Consider a graph G_p formed by stacking the adjacency matrix A_{G_5} of G_5 (given in Figure 1) r times along a block diagonal. Using the notation in Section 4.1, $A_{G_p} = I_r \otimes A_{G_5}$, for $r = 1, 10, 20, 30$. Similarly, for a randomly generated scale matrix $\Lambda \in \mathbb{R}^{5\times 5}$, the corresponding scale matrix for G_p is $\Lambda_p = I_r \otimes \Lambda$, where $p = 5r$ denotes the dimension of the vertex set of G_p. For specific graphs, the corresponding GW normalizing constants can be computed analytically. In particular, for G_5, we can use the formula in Equation (4.1) from Uhler et al. [22] to compute $I_{G_5}(\delta, \Lambda)$, which can then be plugged into Equation (13) to obtain the normalizing constant of $\mathrm{GW}_{G_5}(\delta, \Lambda)$. Since G_p explicitly uses G_5 to create its dependence structure, we can then easily calculate the normalizing constant of $\mathrm{GW}_{G_p}(\delta, \Lambda_r)$.

The results in Table 2 indicate that both GNORM and EPSOM-HybJT produce accurate estimates, with EPSOM-HybJT having a slightly lower relative error. However, the time savings that accompany EPSOM-HybJT are substantial. In the case of $\mathrm{GW}_{G_5}(\delta, \Lambda)$, GNORM is notably faster, but for high-dimensional graphs, there is a dramatic slowdown. Indeed, for $p = 150$, GNORM is more than 100 times slower than EPSOM-HybJT.

Algorithm 2: EPSOM-HybJT

Input : Graph G, prior parameters (δ, Λ) for the GW_G density.
Output: Estimate of the logarithm of the normalizing constant of $\mathrm{GW}_G(\delta, \Lambda)$.
Obtain the JT representation of $G \mapsto \mathcal{J}_G = \{P_1, S_2, P_2, S_3, \ldots, P_{m-1}, S_m, P_m\}$.
for $i \in \{1, \ldots, m\}$ **do**
  if $i > 1$ **then**
    $\log \widehat{\mathcal{Z}}_{S_i} \leftarrow \log h\big(S_i, \delta, \Lambda_{S_i}\big)$.
  end
  if *P_i is complete* **then**
    $\log \widehat{\mathcal{Z}}_{P_i} \leftarrow \log h\big(P_i, \delta, \Lambda_{P_i}\big)$.
  else
    Compute the Cholesky decomposition, $\Lambda_{P_i}^{-1} = T_i' T_i$.
    for $j \in \{1, \ldots, J\}$ **do**
      Sample $\Omega_{(j)} \sim \mathrm{GW}_{P_i}\big(\delta, \Lambda_{P_i}\big)$.
      Compute the Cholesky decomposition, $\Omega_{(j)} = \phi_{(j)}' \phi_{(j)}$.
      $\zeta_{(j)} \leftarrow \phi_{(j)} T_i^{-1}$.
      Extract the free parameters u_j from $\zeta_{(j)}$.
    end
    Fit a CART model, $\mathcal{T}_i$, to $(u_1, \Psi_{P_i}(u_1)), \ldots, (u_J, \Psi_{P_i}(u_J))$.
    Extract the partition $\mathcal{A} = \{A_k\}_{1 \le k \le K}$ from $\mathcal{T}_i$ of the bounding box A of $\mathcal{U}$.
    Calculate the global mode, u_0, of Ψ_{P_i}.
    for $k \in \{1, \ldots, K\}$ **do**
      $u_k \leftarrow \mathrm{argmin}_{u \in A_k} ||u - u_0||_1$.
      $\lambda_k \leftarrow \nabla \Psi_{P_i}(u_k)$.
      $H_k \leftarrow \nabla^2 \Psi_{P_i}(u_k)$.
      $C_k \leftarrow (2\pi)^{d/2} |H_k|^{-1/2} \exp\Big(\frac{1}{2}\Big(u_k' H_k^{-1} u_k - 2\lambda_k' u_k + \lambda_k' H_k^{-1} \lambda_k\Big)\Big)$.
      $b_k \leftarrow H_k u_k - \lambda_k$.
      $G_k \leftarrow \int_{A_k} \mathcal{N}\Big(u \mid H_k^{-1} b_k, H_k^{-1}\Big)\, du$.
      $\widehat{\mathcal{Z}}_k \leftarrow C_k \cdot G_k$.
    end
    $\log \widehat{\mathcal{Z}}_{P_i} \leftarrow \texttt{log-sum-exp}\big(\log \hat{\mathcal{Z}}_1, \ldots, \log \hat{\mathcal{Z}}_K\big)$.
  end
end
return $\log \widehat{\mathcal{Z}} = \sum_i \log \widehat{\mathcal{Z}}_{P_i} - \sum_i \log \widehat{\mathcal{Z}}_{S_i}$.

Table 2. Mean estimate, mean relative error (MRE), and relative runtime of EPSOM-HybJT and GNORM approximations to the log normalizing constant of the $\mathrm{GW}_{G_p}(\delta, \Lambda_p)$ density. G_p has $p = 5r$ vertices and $d = 12r$ free elements. Here, $\delta = 100$ and $\Lambda_p = I_r \otimes \Lambda$. Estimates are reported over 20 replications, each using 1000 samples from the target distribution. The runtime of the GNORM algorithm is calculated relative to the runtime of the EPSOM-HybJT algorithm.

Dimension	Method	Mean	MRE	Runtime
$p = 5, d = 12$	EPSOM-HybJT	920.16	6.46×10^{-4}	1
	GNORM	920.17	6.63×10^{-4}	0.03
$p = 50, d = 120$	EPSOM-HybJT	9201.55	6.44×10^{-4}	9.60
	GNORM	9201.74	6.63×10^{-4}	35.94
$p = 100, d = 240$	EPSOM-HybJT	18,403.14	6.45×10^{-4}	19.26
	GNORM	18,403.46	6.63×10^{-4}	796.07
$p = 150, d = 360$	EPSOM-HybJT	27,604.72	6.46×10^{-4}	28.92
	GNORM	27,605.18	6.63×10^{-4}	4587.46

4.3.2. Experiment 2: Normalizing Constants of General Graphs

Next, we investigate the performance of the two estimators for more general graphs. The results of the previous experiment showed that both GNORM and EPSOM-HybJT are very close to the true normalizing constants. For general graphs that do not satisfy the conditions that previously allowed us to calculate the exact GW normalizing constant, we compare the values of the GNORM and EPSOM-HybJT estimates against each other and keep track of the runtime relative to that of the EPSOM-HybJT algorithm when $p = 10$. In Table 3, for $p \leq 50$, very little separates the log normalizing constant estimates, and GNORM even demonstrates its strength in low dimensions with a runtime that is about 50 times faster than EPSOM-HybJT for $p = 10$. However, EPSOM-HybJT flips the script in all subsequent experiments and scales better as p grows. For $p = 60$, the GNORM estimator fails to give a finite estimate for the log normalizing constant.

Table 3. Mean estimate, standard deviation, and relative runtime of EPSOM-HybJT and GNORM approximations to the log normalizing constant of the $\mathrm{GW}_{G_p}(\delta, \Lambda)$. Here, Λ is not block-diagonal. Estimates are reported over 20 replications, each using 1000 samples from the corresponding G-Wishart distribution. The runtime of the GNORM algorithm is calculated relative to the runtime of the EPSOM-HybJT algorithm.

# Vertices	Method	Mean	SD	Runtime
$p = 10$	EPSOM-HybJT	−2477.40	0.003	1
	GNORM	−2468.19	0	0.02
$p = 30$	EPSOM-HybJT	−7450.69	0.012	2.99
	GNORM	−7428.00	0.773	12.34
$p = 40$	EPSOM-HybJT	−10,030.95	0.015	4.68
	GNORM	−10,003.81	1.596	42.76
$p = 50$	EPSOM-HybJT	−12,563.87	0.014	6.76
	GNORM	−12,528.42	2.385	108.87
$p = 60$	EPSOM-HybJT	−15,170.05	0.017	15.26
	GNORM	`-Inf`	—	231.32

In these experiments, each graph is generated using Bernoulli draws to determine the existence of its edges and checked to ensure that the graph is non-decomposable; otherwise, this would simply reduce the problem to a summation of tractable IW constants and would be unable to fairly assess the approximating ability of EPSOM-HybJT. Contrast these results with the examples from the previous section where the scale matrix was assumed to be block-diagonal and the dependence structure was relatively simple. In those simpler

settings, GNORM retains its accuracy even for high-dimensional graphs, albeit with a much slower runtime. Evidently, the added complexity induced by a nontrivial dependence structure contributes to the computational burden that cannot be easily overcome using standard methods.

4.4. Software Contribution

With the architecture in place for both the EPSOM-Hyb algorithm and its junction tree extension, we are equipped with the necessary tools for computing normalizing constants for general cases, as well as GGM-specific examples. For ease of use, EPSOM-Hyb is implemented in the `hybrid` package. Note that the hyperparameter initialization in line 26 of Figure A1 refers to problem-specific hyperparameters, rather than algorithm-specific hyperparameters or settings. We emphasize that other than supplying a sampler for the target distribution and functions to evaluate Ψ and its gradient and Hessian, the EPSOM-Hyb methodology does not require hyperparameter tuning or convergence monitoring, beyond Newton's method for finding the global mode of the unimodal target distribution, as described in Section 3.3. Recognizing the intricacy of the EPSOM-HybJT algorithm and the difficulty of having to manually combine the EPSOM-Hyb and the junction tree methodologies, we have developed an additional package dedicated specifically to estimating the normalizing constant of G-Wishart densities. The `R` package `graphml` serves as a black box method that seamlessly integrates the junction tree algorithm into the existing EPSOM-Hyb methodology without any user input other than the adjacency matrix representation of the graph and the G-Wishart density parameters. See Appendixes A and B for more details regarding the installation and use of these packages. Additionally, in the package repositories, we provide the `R` programs that replicate the experimental results reported in this paper.

5. Discussion

Our methodological contributions are twofold: firstly, we developed EPSOM-Hyb as a general algorithm for computing the normalizing constant of log-concave densities, and secondly, we extended this methodology to specifically target the marginal likelihood of non-decomposable GGMs. While the GNORM estimator from Atay-Kayis and Massam [17] is a valuable tool that performs well for simpler and lower-dimensional graph structures, it is evident that more scalable and robust solutions are needed in the nontrivial problem settings presented in this paper. By taking advantage of junction tree representations of connected graphs and pairing them with the EPSOM-Hyb methodology, we significantly simplify the normalizing constant calculation for GGMs. The resulting EPSOM-HybJT algorithm is not only able to compete with other widely accepted estimators, but it also presents itself as a scalable solution for high-dimensional scenarios where other methods either fail to produce sensible results or succumb to time complexity constraints.

We recognize that the junction tree representation of a connected graph can be integrated into the GNORM methodology to create a more robust algorithm, leading to substantial time savings in subsequent normalizing constant calculations. However, such an implementation is presently unavailable. In the meantime, our contributions represent a universal method for marginal likelihood estimation that can be easily adapted to be as accurate and significantly faster than specialized methods for graphical models. Finally, while our methodologies empirically demonstrate their competitiveness across a variety of examples, many popular and long-established marginal likelihood estimation methods are accompanied by strong theoretical guarantees. As such, this remains an area of future work and development.

Author Contributions: Conceptualization, A.B. and D.P.; methodology, E.C. and Y.N.; software, E.C.; validation, Y.N.; formal analysis, E.C.; investigation, E.C.; resources, A.B. and D.P.; data curation, E.C. and Y.N.; writing—original draft preparation, E.C.; writing—review and editing, Y.N., A.B. and D.P.; visualization, E.C.; supervision, A.B. and D.P.; project administration, A.B.; funding acquisition, A.B. and D.P. All authors have read and agreed to the published version of the manuscript.

Funding: Pati and Bhattacharya acknowledge support from NSF DMS (2210689).

Informed Consent Statement: Not applicable.

Data Availability Statement: No datasets were used in this article. Simulations and examples can be replicated using the scripts made available in the following repositories: https://github.com/echuu/hybrid (accessed on 12 May 2024) and https://github.com/echuu/graphml (accessed on 12 May 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

GGM	Gaussian Graphical Model
JT	Junction Tree
KL	Kullback–Leibler
IW	Inverse Wishart
HIW	Hyper-Inverse Wishart
GW	G-Wishart
MCMC	Markov Chain Monte Carlo
EP	Expectation Propagation
CART	Classification and Regression Tree
EPSOM-Hyb	EP-guided Second-Order Modified Hybrid
EPSOM-HybJT	EP-guided Second-Order Modified Hybrid Junction Tree
BSE	Bridge Sampling Estimator
WBSE	Warp Bridge Sampling Estimator
AE	Average Error
MRE	Mean Relative Error

Appendix A. General Use of the `hybrid` Package

We developed `hybrid` [38], an `R` package that allows practitioners to easily compute estimates of the log-marginal likelihood. In Figure A1, we provide a snippet of code to demonstrate how both the hybrid approximation and the EPSOM-Hyb approximation can be used in practice. For the hybrid method, users only need to provide a way to evaluate the negative log posterior Ψ and a sampler for the target distribution γ. After drawing the samples using a user-defined `sample_post()` function and evaluating them using the `hybrid::preprocess()` function, we can calculate the log-marginal likelihood estimate with the `hybrid::hybml_const()` function.

For EPSOM-Hyb, users will need to supplement the input of the hybrid method with function definitions for the gradient vector and Hessian matrix of Ψ. These function definitions, along with the posterior samples, are then passed into the `hybrid::hybml()` function. Optionally, a representative point (typically the global mode) can be supplied to the function call to play the role of u_0 as defined in Section 3.3, but in the case where no point is given, the implementation will take u_0 to be the point with the highest posterior mass. We emphasize that beyond specifying the model and supplying a sampler, which is typically required in all other competing methods, there are no hyperparameters to tune and no problem-specific settings that require modification or attention. This makes our solution one of the few black box marginal likelihood estimation methods that has been empirically shown to scale well with dimension and accommodate complex parameter spaces. The repository that contains the source code for these algorithm implementations can be found at https://github.com/echuu/hybrid, accessed on 12 May 2024. The repository also contains installation instructions and a working example.

```
#### ------------------ HYBRID, EPSOM-Hyb DEMO  ------------------ ####

## In the functions below, 'params' is an object that stores
## any miscellaneous values (hyperparameters, sample size, dimensions)
## that may be necessary to compute the corresponding~functions.

library(hybrid)

#### -- The following 4 functions must be supplied by the user -- ####

# sample_post(): returns a (J x d) matrix of samples from the
# target distribution
sample_post = function(J) { ... }

# psi(): returns the (scalar) negative log posterior evaluated at u
psi  = function(u, params) { ... }

# grad(): returns the (d x 1) gradient vector of psi evaluated at u
grad = function(u, params) { ... }

# hess(): returns the (d x d) Hessian matrix of psi evaluated at u
hess = function(u, params) { ... }

#### ------------- Problem-specific initializations ------------- ####

params = init( ... )    # initialize any hyperparameters
J = 5000                # number of posterior samples to draw
samps = sample_post(J)  # (J x d) samples from the target~distribution

# evaluate posterior samples using psi()
psi_df = hybrid::preprocess(samps, d, params)

# compute hybrid estimate for the log Z
hybrid::hybml_const(psi_df)$logz

# compute EPSOM-Hyb estimate for the log Z
hybrid::hyb(psi_df, params, grad, hess)$logz
```

Figure A1. Demonstration of how to use `hybrid` package in R. This package contains the implementation for both the Hybrid and EPSOM-Hyb algorithms for estimating the log normalizing constant of a target distribution.

Appendix B. General Use of the `graphml` Package

While the graphical modeling examples can be adapted to be used with the `hybrid` package, we have also developed a package specific to graphical models that further simplifies the process of weaving together the EPSOM-Hyb methodology with the junction tree representation of general graphs—as discussed in Section 4.3 and presented in Algorithm 2—because of the importance of the normalizing constant calculation in graphical modeling literature. With the `graphml` [39] package, we can easily use the EPSOM-HybJT methodology to compute the normalizing constant of the G-Wishart density given the adjacency matrix for a general graph, the scale matrix, the degrees of freedom, and the number of MCMC samples to be drawn from the corresponding G-Wishart density.

```
#### ------------------ EPSOM-Hyb ALGORITHM DEMO  ------------------ ####

library(graphml)

#### ----- Initialize G-Wishart parameters ----- ####

G = ...                  # initialize adj. matrix representation of graph
delta = ...              # degrees of freedom, delta > 2
V = ...                  # non-negative definite scale matrix
J = ...                  # number of samples to draw from target~density

# Compute the edge matrix for the JT part of the algorithm
EdgeMat = graphml::getEdgeMat(G)

# --- Compute the log normalizing constant of the GW(b, V) density --- #

# Compute the log normalizing constant using Atay's algorithm
BDgraph::gnorm(G, delta, V, J)

# Compute the log normalziing constant using the EPSOM-HybJT algorithm
graphml::hybridJT(G, EdgeMat, delta, V, J)
```

Figure A2. Demonstration of how to use `graphml` package in `R`. This package contains the implementation of the EPSOM-HybJT algorithm for estimating the log normalizing constant of G-Wishart densities. Atay's estimator can similarly be computed using the `gnorm` function from the `BDgraph` package.

We juxtapose the code necessary for computing the GNORM estimate via the `BDgraph` package to demonstrate the comparable ease of use of the `graphml` package. We use the C++ implementation of junction tree representation [40], which can be found at https://www2.stat.duke.edu/~mw/mwsoftware/GGM/index.html (accessed on 10 April 2024). We adapt their implementation into the `graphml` package since the original C++ implementation is not directly importable in `R`. Recall that for the general `hybrid` package, we require user-defined functions for the objective function, gradient, and Hessian, whereas, in this package, all of these functions are already optimally implemented. In the case of the G-Wishart density, these functions are quite cumbersome to write because of the recursive structure, so removing this from the list of user responsibilities is especially convenient. The Github repository that contains the source code for this package can be found at https://github.com/echuu/graphml (accessed on 12 May 2024).

Appendix C. Hyper-Inverse Wishart Density

We introduce some notation to help us obtain a closed form for the marginal likelihood in the case of a decomposable graph G. For an $n \times p$ matrix X, X_C is defined as the sub-matrix of X consisting of columns with indices in the clique C. Let $(X_1, X_2, \ldots, X_p) = (x_1, x_2, \ldots, x_n)'$, where X_i is the ith column of $X_{n\times p}$. If $C = \{i_1, i_2, \ldots, i_{|C|}\}$, where $1 \leq i_1 < i_2 < \ldots < i_{|C|} \leq p$, then $X_C = (X_{i_1}, X_{i_2}, \ldots, X_{i_{|C|}})$. For any square matrix $A = (a_{ij})_{p\times p}$, define $A_C = (a_{ij})_{|C|\times|C|}$ where $i, j \in C$, $|C|$ is the cardinality of the clique C, and the order of entries carries into the new sub-matrix A_C. Therefore, $X_C' X_C = (X'X)_C$.

Decomposable graphs correspond to a special kind of sparsity pattern in Σ, henceforth denoted Σ_G. Suppose we have a $\text{HIW}_G(b, D)$ distribution on the cone of $p \times p$ positive definite matrices with $b > 2$ degrees of freedom and a fixed $p \times p$ positive definite matrix D such that the joint density factorizes on the junction tree of the given decomposable graph G as

$$p(\Sigma_G \mid b, D) = \frac{\prod_{C\in\mathcal{C}} p(\Sigma_C \mid b, D_C)}{\prod_{S\in\mathcal{S}} p(\Sigma_S \mid b, D_S)}, \tag{A1}$$

where for each $C \in \mathcal{C}$, $\Sigma_C \sim \mathrm{IW}_C(b, D_C)$ has the density

$$p(\Sigma_C \mid b, D_C) \propto |\Sigma_C|^{-(b+2|C|)/2} \mathrm{etr}\Big\{ -\frac{1}{2}\Sigma_C^{-1} D_C \Big\}, \tag{A2}$$

where $\mathrm{etr}(\cdot) = \exp\{\mathrm{tr}(\cdot)\}$. Here, $\mathrm{IW}(b, D)$ denotes the inverse-Wishart distribution with degrees of freedom b and a fixed $p \times p$ positive definite matrix D with normalizing constant

$$\left|\frac{1}{2}D\right|^{(b+p-1)/2} \Gamma_d^{-1}\left(\frac{b+p-1}{2}\right).$$

Note that we can establish equivalence to the parametrization used in Section 4.1 by taking $\delta = b + p - 1$. Since the joint density in Equation (9) factorizes over cliques and separators in the same way as in Equations (A1) and (A2),

$$f(X \mid \Sigma_G) = (2\pi)^{-\frac{np}{2}} \frac{\prod_{C\in\mathcal{C}} |\Sigma_C|^{-\frac{n}{2}} \mathrm{etr}\Big(-\frac{1}{2}\Sigma_C^{-1} X_C' X_C \Big)}{\prod_{S\in\mathcal{S}} |\Sigma_S|^{-\frac{n}{2}} \mathrm{etr}\Big(-\frac{1}{2}\Sigma_S^{-1} X_S' X_S \Big)}.$$

Using the above equations, we can obtain the marginal likelihood,

$$f(X \mid G) = (2\pi)^{-\frac{np}{2}} \frac{h(G, b, D)}{h(G, b+n, D+S)} = (2\pi)^{-\frac{np}{2}} \frac{\prod_{C\in\mathcal{C}} w(C)}{\prod_{S\in\mathcal{S}} w(S)}, \tag{A3}$$

where

$$h(G, b, D) = \frac{\prod_{C\in\mathcal{C}} \left|\frac{1}{2}D_C\right|^{\frac{b+|C|-1}{2}} \Gamma_{|C|}^{-1}\left(\frac{b+|C|-1}{2}\right)}{\prod_{S\in\mathcal{S}} \left|\frac{1}{2}D_S\right|^{\frac{b+|S|-1}{2}} \Gamma_{|S|}^{-1}\left(\frac{b+|S|-1}{2}\right)}, \tag{A4}$$

$$w(C) = \frac{|D_C|^{\frac{b+|C|-1}{2}} \left|D_C + X_C' X_C\right|^{-\frac{b+n+|C|-1}{2}}}{2^{-\frac{n|C|}{2}} \Gamma_{|C|}\left(\frac{b+|C|-1}{2}\right) \Gamma_{|C|}^{-1}\left(\frac{b+n+|C|-1}{2}\right)}. \tag{A5}$$

Appendix D. Hyper-Inverse Wishart Objective Function

Recall that we take the Cholesky decomposition of $\Omega = \phi'\phi$, where ϕ is upper triangular. Using Equations (9) and (11), we define $\Psi(\phi) = -\log L(\phi) - \log \pi(\phi)$. Even though Ψ is expressed as a function of the upper Cholesky factor ϕ, it is inherently a function of only the free elements of ϕ. As a result, the gradient of Ψ is defined element-wise,

$$\frac{\partial \Psi}{\partial \phi_{ij}} = \begin{cases} -\frac{1}{\phi_{ii}}(\eta_i + n) + \phi_{ii} + \sum_{m=i}^{p} \phi_{im} b_{mi} & i = j, \\ \phi_{ij} + \sum_{m=i}^{p} \phi_{im} b_{mj} & i \neq j. \end{cases}$$

Similarly, the Hessian can be computed element-wise,

$$\frac{\partial^2 \Psi}{\partial \phi_{ij} \partial \phi_{kl}} = \begin{cases} 0 & i \neq k, \\ \frac{1}{\phi_{ii}^2}(n + \eta_i) + b_{ii} + 1 & i = j = k = l, \\ b_{li} & i = j, i = k, l > j, \\ 1 + b_{lj} & i \neq j, i = k, l = j, \\ b_{lj} & i \neq j, i = k, l > j, \end{cases}$$

where $(i, j), (k, l) \in \mathcal{V}$, $\eta_i = \delta + \nu_i - 1$, and $B = (b_{ij})_{1 \leq i,j \leq p}$.

Appendix E. G-Wishart Objective Function

Next, we provide the calculation details for the derivation of the gradient and Hessian of $\Psi(\zeta)$, as defined in Equation (15). First, we can compute the terms of the gradient element-wise by taking the derivative of Ψ with respect to the free elements of ζ,

$$\frac{\partial \Psi(\zeta)}{\partial \zeta_{ij}} = \begin{cases} \sum\limits_{(r,s)\in\bar{\mathscr{V}}} \zeta_{rs}\dfrac{\partial \zeta_{rs}}{\partial \zeta_{ii}} - \dfrac{(\delta+\nu_i-1)}{\zeta_{ii}} + \zeta_{ii} & i=j, \\ \sum\limits_{(r,s)\in\bar{\mathscr{V}}} \zeta_{rs}\dfrac{\partial \zeta_{rs}}{\partial \zeta_{ij}} + \zeta_{ij} & i\neq j, (i,j)\in\mathscr{V}. \end{cases} \tag{A6}$$

Note that because the non-free elements are functions of the free elements, each gradient term involves additional recursive derivative calculations. As given in Equation (16), for $(r,s)\in\bar{\mathscr{V}}$ and $r<s$,

$$\zeta_{rs} = \sum_{j=r}^{s-1}\left(-\zeta_{rj}\frac{\lambda_{js}}{\lambda_{ss}}\right) - \sum_{i=1}^{r-1}\left(\frac{\zeta_{ir}+\sum_{j=i}^{r-1}\zeta_{ij}\frac{\lambda_{jr}}{\lambda_{rr}}}{\zeta_{rr}}\right)\left(\zeta_{is}+\sum_{j=i}^{s-1}\zeta_{ij}\frac{\lambda_{js}}{\lambda_{ss}}\right). \tag{A7}$$

Finally , using the expression for the gradient in Equation (A6), we can perform a similar calculation for the Hessian. The elements on and above the diagonal are defined as follows,

$$\frac{\partial^2 \Psi(\zeta)}{\partial \zeta_{ij}\partial \zeta_{kl}} = \begin{cases} \sum\limits_{(r,s)\in\bar{\mathscr{V}}}\left[\left(\dfrac{\partial \zeta_{rs}}{\partial \zeta_{ii}}\right)^2 + \zeta_{rs}\dfrac{\partial^2 \zeta_{rs}}{\partial \zeta_{ii}^2}\right] + \dfrac{(\delta+\nu_i-1)}{\zeta_{ii}^2} + 1, & i=j=k=l, \\ \sum\limits_{(r,s)\in\bar{\mathscr{V}}}\left[\dfrac{\partial \zeta_{rs}}{\partial \zeta_{kl}}\dfrac{\partial \zeta_{rs}}{\partial \zeta_{ij}} + \zeta_{rs}\dfrac{\partial^2 \zeta_{rs}}{\partial \zeta_{ij}\partial \zeta_{kl}}\right] + \dfrac{\partial \zeta_{ij}}{\partial \zeta_{kl}}, & i\neq j,(i,j),(k,l)\in\mathscr{V}. \end{cases} \tag{A8}$$

Below, we provide the derivation for the partial derivative terms of the non-free elements taken with respect to the free elements by again starting with the recursive definition of ζ_{rs} in Equation (16). In the following calculations, we define $\xi_{ks} = \lambda_{ks}/\lambda_{ss}$, where $\Lambda = (\lambda_{ij})$, where $1 \le i,j \le p$. We consider two cases.

Case 1: for $(r,s)\in\bar{\mathscr{V}}$, $r<s$, $i\neq j$, and (r,s) coming after (i,j), we have

$$\begin{aligned} \frac{\partial \zeta_{rs}}{\partial \zeta_{ij}} &= -\sum_{k=r}^{s-1}\xi_{ks}\frac{\partial \zeta_{rk}}{\partial \zeta_{ij}} \\ &\quad - \frac{1}{\zeta_{rr}}\frac{\partial}{\partial \zeta_{ij}}\sum_{k=1}^{r-1}\left[\zeta_{ks}\zeta_{kr} + \zeta_{kr}\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ls} + \zeta_{ks}\sum_{l=k}^{r-1}\zeta_{kl}\xi_{lr} + \left(\sum_{l=k}^{k-1}\zeta_{kl}\xi_{kr}\right)\left(\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ks}\right)\right]. \end{aligned} \tag{A9}$$

Case 2: for $r=i=j$, and $s>r$, we obtain the derivative:

$$\begin{aligned} \frac{\partial \zeta_{rs}}{\partial \zeta_{rr}} &= -\sum_{k=r}^{s-1}\xi_{ks}\frac{\partial \zeta_{rk}}{\partial \zeta_{rr}} \\ &\quad + \frac{1}{\zeta_{rr}^2}\sum_{k=1}^{r-1}\left[\zeta_{ks}\zeta_{kr} + \zeta_{kr}\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ls} + \zeta_{ks}\sum_{l=k}^{r-1}\zeta_{kl}\xi_{lr} + \left(\sum_{l=k}^{k-1}\zeta_{kl}\xi_{kr}\right)\left(\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ks}\right)\right]. \end{aligned}$$

In case 1, each term in the outer summation in Equation (A9) can be computed as:

$$\frac{\partial}{\partial \zeta_{ij}}\left[\zeta_{ks}\zeta_{kr}\right] = \frac{\partial \zeta_{kr}}{\partial \zeta_{ij}}\zeta_{ks} + \zeta_{kr}\frac{\partial \zeta_{ks}}{\partial \zeta_{ij}},$$

$$\frac{\partial}{\partial \zeta_{ij}}\left[\zeta_{kr}\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ls}\right] = \frac{\partial \zeta_{kr}}{\partial \zeta_{ij}}\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ls} + \zeta_{kr}\sum_{l=k}^{s-1}\frac{\partial \zeta_{kl}}{\partial \zeta_{ij}}\xi_{ls},$$

$$\frac{\partial}{\partial \zeta_{ij}}\left[\zeta_{ks}\sum_{l=k}^{r-1}\zeta_{kl}\xi_{lr}\right] = \frac{\partial \zeta_{ks}}{\partial \zeta_{ij}}\sum_{l=k}^{r-1}\zeta_{kl}\xi_{lr} + \zeta_{ks}\sum_{l=k}^{r-1}\frac{\partial \zeta_{kl}}{\partial \zeta_{ij}}\xi_{lr},$$

$$\frac{\partial}{\partial \zeta_{ij}}\left[\left(\sum_{l=k}^{r-1}\zeta_{kl}\xi_{kr}\right)\left(\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ks}\right)\right] = \left[\sum_{l=k}^{r-1}\xi_{kr}\frac{\partial \zeta_{kl}}{\zeta_{ij}}\right]\left[\sum_{l=k}^{s-1}\zeta_{kl}\xi_{ks}\right] + \left[\sum_{l=k}^{r-1}\zeta_{kl}\xi_{kr}\right]\left[\sum_{l=k}^{s-1}\xi_{ks}\frac{\partial \zeta_{kl}}{\partial \zeta_{ij}}\right].$$

These scalar, element-wise quantities can be substituted back into the piecewise definitions of the gradient and Hessian matrices, as given in Equations (A6) and (A8), respectively.

References

1. Lauritzen, S.L. *Graphical Models*; Clarendon Press: Oxford, UK, 1996; Volume 17.
2. Maathuis, M.; Drton, M.; Lauritzen, S.; Wainwright, M. *Handbook of Graphical Models*, 1st ed.; CRC Press, Inc.: Boca Raton, FL, USA, 2018.
3. Tierney, L.; Kadane, J.B. Accurate Approximations for Posterior Moments and Marginal Densities. *J. Am. Stat. Assoc.* **1986**, *81*, 82–86. [CrossRef]
4. Newton, M.A.; Raftery, A.E. Approximate Bayesian Inference with the Weighted Likelihood Bootstrap. *J. R. Stat. Soc. Ser. B* **1994**, *56*, 3–26. [CrossRef]
5. Lenk, P. Simulation Pseudo-Bias Correction to the Harmonic Mean Estimator of Integrated Likelihoods. *J. Comput. Graph. Stat.* **2009**, *18*, 941–960. [CrossRef]
6. Pajor, A. Estimating the Marginal Likelihood Using the Arithmetic Mean Identity. *Bayesian Anal.* **2017**, *12*, 261–287. [CrossRef]
7. Neal, R.M. Annealed importance sampling. *Stat. Comput.* **2001**, *11*, 125–139. [CrossRef]
8. Chib, S. Marginal Likelihood from the Gibbs Output. *J. Am. Stat. Assoc.* **1995**, *90*, 1313–1321. [CrossRef]
9. Meng, X.L.; Wong, W.H. Simulating ratios of normalizing constants via a simple identity: A theoretical exploration. *Stat. Sin.* **1996**, *6*, 831–860.
10. Meng, X.L.; Schilling, S. Warp Bridge Sampling. *J. Comput. Graph. Stat.* **2002**, *11*, 552–586. [CrossRef]
11. Skilling, J. Nested sampling for general Bayesian computation. *Bayesian Anal.* **2006**, *1*, 833–859. [CrossRef]
12. Hubin, A.; Storvik, G. Estimating the marginal likelihood with Integrated nested Laplace approximation (INLA). *arXiv* **2016**, arXiv:1611.01450. [CrossRef]
13. Barthelmé, S.; Chopin, N. Expectation-Propagation for Likelihood-Free Inference. *arXiv* **2011**, arXiv:1107.5959. [CrossRef]
14. Roverato, A. Cholesky decomposition of a hyper inverse Wishart matrix. *Biometrika* **2000**, *87*, 99–112. [CrossRef]
15. Giudici, P.; Green, P. Decomposable graphical Gaussian model determination. *Biometrika* **1999**, *86*, 785–801. [CrossRef]
16. Dellaportas, P.; Giudici, P.; Roberts, G. Bayesian inference for nondecomposable graphical Gaussian models. *Sankhyā Indian J. Stat.* **2003**, *65*, 43–55.
17. Atay-Kayis, A.; Massam, H. A Monte Carlo method for computing the marginal likelihood in nondecomposable Gaussian graphical models. *Biometrika* **2005**, *92*, 317–335. [CrossRef]
18. Khare, K.; Rajaratnam, B.; Saha, A. Bayesian inference for Gaussian graphical models beyond decomposable graphs. *arXiv* **2015**, arXiv:1505.00703. [CrossRef]
19. Piccioni, M. Independence Structure of Natural Conjugate Densities to Exponential Families and the Gibbs' Sampler. *Scand. J. Stat.* **2000**, *27*, 111–127. [CrossRef]
20. Rajaratnam, B.; Massam, H.; Carvalho, C.M. Flexible covariance estimation in graphical Gaussian models. *Ann. Stat.* **2008**, *36*, 2818–2849. [CrossRef]
21. Lenkoski, A. A Direct Sampler for G-Wishart Variates. *Stat* **2013**, *2*, 119–128. [CrossRef]
22. Uhler, C.; Lenkoski, A.; Richards, D. Exact Formulas for the Normalizing Constants of Wishart Distributions for Graphical Models. *Ann. Stat.* **2018**, *46*, 90–118. [CrossRef]
23. Bhadra, A.; Sagar, K.; Banerjee, S.; Datta, J. Graphical Evidence. *arXiv* **2022**, arXiv:2205.01016. [CrossRef]
24. Chuu, E.; Pati, D.; Bhattacharya, A. A Hybrid Approximation to the Marginal Likelihood. In Proceedings of the AISTATS, San Diego, CA, USA, 13–15 April 2021.
25. Breiman, L. *Classification and Regression Trees*; Wadsworth International Group: Franklin, TN, USA, 1984.

26. Kleijn, B.J.K.; van der Vaart, A.W. Misspecification in infinite-dimensional Bayesian statistics. *Ann. Statist.* **2006**, *34*, 837–877. [CrossRef]
27. Ghosal, S.; Van Der Vaart, A. Convergence rates of posterior distributions for noniid observations. *Ann. Stat.* **2007**, *35*, 192–223. [CrossRef]
28. Vaart, A.W.v.d. *Asymptotic Statistics*; Cambridge Series in Statistical and Probabilistic Mathematics; Cambridge University Press: Cambridge, UK, 1998. [CrossRef]
29. Genz, A. Numerical Computation of Multivariate Normal Probabilities. *J. Comput. Graph. Stat.* **1992**, *1*, 141. [CrossRef]
30. Botev, Z.I. The normal law under linear restrictions: Simulation and estimation via minimax tilting. *J. R. Stat. Soc. Ser. B* **2016**, *79*, 125–148. [CrossRef]
31. Minka, T.P. Expectation Propagation for approximate Bayesian inference. *arXiv* **2013**, arXiv:1301.2294. [CrossRef]
32. Cunningham, J.P.; Hennig, P.; Lacoste-Julien, S. Gaussian probabilities and expectation propagation. *arXiv* **2011**, arXiv:1111.6832.
33. Mohammadi, R.; Wit, E.C. BDgraph: An R Package for Bayesian Structure Learning in Graphical Models. *J. Stat. Softw.* **2019**, *89*, 1–30. [CrossRef]
34. Diaconis, P.; Ylvisaker, D. Conjugate priors for exponential families. *Ann. Stat.* **1979**, *7*, 269–281. [CrossRef]
35. Dawid, A.P.; Lauritzen, S.L. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Ann. Statist.* **1993**, *21*, 1272–1317. [CrossRef]
36. Robertson, N.; Seymour, P. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms* **1986**, *7*, 309–322. [CrossRef]
37. Fitch, A.M.; Jones, M.B.; Massam, H. The Performance of Covariance Selection Methods That Consider Decomposable Models Only. *Bayesian Anal.* **2014**, *9*, 659–684. [CrossRef]
38. Chuu, E. *hybrid: Hybrid Approximation to the Marginal Likelihood*, R package version 1.0.; 2022. Available online: https://proceedings.mlr.press/v130/chuu21a.html (accessed on 10 May 2024).
39. Chuu, E. *graphml: Hybrid Approximation for Computing Normalizing Constants of G-Wishart Densities*, R package version 1.0.; 2022.
40. Jones, B.; Carvalho, C.; Dobra, A.; Hans, C.; Carter, C.; West, M. Experiments in Stochastic Computation for High-Dimensional Graphical Models. *Stat. Sci.* **2005**, *20*, 388–400. [CrossRef]