Information and Inference: A Journal of the IMA (2026) **15**, iaag016
https://doi.org/10.1093/imaiai/iaag016

Property testing in Gaussian graphical models: trees and small separation numbers

Luc Devroye
School of Computer Science, McGill University, 3480 University Street, Montreal, H3A 0E9, Canada

Gábor Lugosi
ICREA Pg. Lluis Companys 23 08010, Spain
Universitat Pompeu Fabra, Department of Economics and Business, Ramon Trias Fargas 25-27, 08005 Barcelona, Spain
Barcelona School of Economics, Ramon Trias Fargas 25-27, 08005, Barcelona, Spain

and

Piotr Zwiernik*
Universitat Pompeu Fabra, Department of Economics and Business, Ramon Trias Fargas 25-27, 08005 Barcelona, Spain
Barcelona School of Economics, Ramon Trias Fargas 25-27, 08005, Barcelona, Spain
*Corresponding author. Email: piotr.zwiernik@upf.edu

[Received on 21 May 2024; revised on 9 August 2025; accepted on 26 March 2026]

We study property testing for graphical models in a setting where an algorithm may adaptively query individual entries of the covariance matrix Σ, and cost is measured by the number of queried entries. Under natural (strong) faithfulness assumptions, we design divide-and-conquer tests guided by *balanced separators* that operate on small submatrices and avoid global matrix inversion. Our first result is a tester for whether the underlying graph is a tree; with high probability it decides correctly using a subquadratic number of correlation queries, and for bounded maximum degree Δ the complexity is near-linear up to logarithmic factors. Our second result concerns graphs with a small separation number (hence small treewidth). We present two complementary procedures: a *conditional descent* test that never breaks when $\mathrm{sn}(G) \leq k$ and, upon termination, returns an $O(k\log(n/k))$ certificate; and a *marginal descent* test that, under a 'good run' condition, either certifies $\mathrm{sn}(G) \leq 2k$ or proves $\mathrm{sn}(G) > k$. The approach extends beyond the Gaussian case whenever reliable conditional-independence queries are available (e.g., nonparanormal models), yielding tests that access only a vanishing fraction of the entries of Σ.

Keywords: covariance matrix estimation; property testing; separation number; graphical models.

1. Introduction

Graphical models are widely used to encode multivariate dependence structures in a parsimonious and interpretable way [22]. In a Gaussian graphical model, a graph $G = ([n], E)$ is associated with a random vector $X = (X_1, \ldots, X_n)$ through zeros of the precision matrix: if $\Sigma = \mathrm{cov}(X)$, then

$$(\Sigma^{-1})_{ij} = 0 \quad \iff \quad X_i \perp\!\!\!\perp X_j \mid X_{[n]\setminus\{i,j\}},$$

so the edge set E records conditional independences directly. Let $\mathbb{S}^n_+$ be the cone of positive definite matrices and define

$$M(G) \ = \ \{\Sigma \in \mathbb{S}^n_+ : (\Sigma^{-1})_{ij} = 0 \text{ whenever } i \neq j \text{ and } ij \notin E\}. \tag{1.1}$$

This representation supports statistical inference and scalable computation in many applied domains [22, 40].

In modern applications, n can be so large that even storing Σ or computing Σ^{-1} becomes prohibitive. This occurs, for example, in reconstructing gene regulatory networks from high-throughput expression data [3, 15, 41] and in constructing functional connectivity networks from fMRI data [14]. In such regimes, full structure learning may be unnecessary or infeasible; instead, it is natural to ask whether *specific* properties of the underlying graph hold. Typical targets include whether the graph is a tree, whether its maximum degree is bounded, or whether it has small treewidth (equivalently, admits small balanced separators). Our work adopts this property-testing viewpoint while accessing only a vanishing fraction of the entries of Σ, in the entry-query spirit of Lugosi *et al.* [25].

Property testing studies algorithms that decide whether a large object has a property using highly limited access to the object [12, 32]. Here, the object of interest is the graph G, but we do not query edges directly. Instead, the algorithm may adaptively query individual entries of the covariance matrix Σ and must draw global conclusions about G from such local access. This indirect access makes the design of efficient testers delicate. Related hypothesis-testing ideas in graphical models have largely focused on structure learning and parametric hypotheses [7, 19], or on testing local substructures [39]. Our emphasis is on *global* structural properties and on computational efficiency.

We focus on two canonical properties. The first is treeness:

$$H_0 : \ G \text{ is a tree} \qquad \text{vs.} \qquad H_1 : \ G \text{ has a cycle.} \tag{1.2}$$

For connected graphs, acyclicity and treeness coincide, so (1.2) is the natural formulation. The second property is a small separation number, a balanced-separator notion closely related (within constant factors) to treewidth (see Section 4): for $k \geq 1$,

$$H_0 : \ \mathrm{sn}(G) \leq k \qquad \text{vs.} \qquad H_1 : \ \mathrm{sn}(G) > k. \tag{1.3}$$

By definition, $\mathrm{sn}(G) \leq k$ means that for every $W \subseteq V$, there exist disjoint $A, A' \subseteq W$ and $S \subseteq V$ with $|S| \leq k$ such that $\max\{|A|, |A'|\} \leq \frac{2}{3}|W|$ and no path in $G \setminus S$ connects a vertex of A to a vertex of A'. Graphs of small treewidth are central in exact and approximate inference [4, 17, 21], and empirical studies suggest that many real networks have modest treewidth [1, 2, 26].

We work in an *entry-query* oracle model: the algorithm may adaptively query selected entries of Σ, and cost is the number of queried entries. This model captures settings where Σ is distributed or too large to invert, but entries can be obtained on demand. We design tests that avoid global linear-algebra operations and achieve subquadratic total query complexity (often near-linear when the maximum degree Δ is bounded). The entry-query paradigm has proved useful beyond testing, for example, in learning sparse graphical structures [25] and in recovering spectral information from few entries [18].

1.1 *Related work and comparison*

There is a complementary, *sample-based* literature on testing graph properties in Gaussian graphical models. Neykov *et al.* [30] observe m i.i.d. samples and build de-biased estimators for entries of the precision matrix; these are aggregated into tests for global properties, such as connectivity, acyclicity/forestness, bounded degree and graph distances [30]. Guarantees are expressed in terms of *sample size* and rely on concentration/normality of the de-biased statistics; see also [5, 29, 36, 37] for related combinatorial inference problems. By contrast, we assume no samples and no global precision estimation: our guarantees control *query complexity* (how many entries of Σ we read). Methodologically, we follow the entry-query approach of Lugosi *et al*. [25] but focus on testing rather than full graph recovery. Because the information models differ (samples vs. entry queries), the guarantees are not directly comparable.

1.2 *Main contributions*

We develop divide-and-conquer tests guided by *balanced separators*. At each recursion step, the algorithm inspects only small submatrices (and their Schur complements), performs localized rank/zero checks to detect separators and then descends into components whose sizes shrink geometrically. This locality drives both correctness and efficiency.

Our first result concerns testing whether G is a tree. Under a natural 1-faithfulness assumption (formalized in Section 2), the procedure repeatedly finds an approximately central cut vertex using small conditional-independence checks on Schur complements $\Sigma^{(v)}$ and recurses on the resulting components. With probability at least $1-\epsilon$, it decides correctly while reading a subquadratic number of entries of Σ; when the maximum degree Δ is bounded, the query complexity is near-linear up to logarithmic factors (Theorem 3.5).

Our second set of results treats small separation numbers (hence, bounded treewidth). Section 4 presents two descent schemes. The *(MD)* procedure mirrors the tree tester but searches for balanced separators inside successively smaller blocks obtained by marginalization. Because marginalization can introduce fill-in, a separator found inside a block need not be a valid separator of G; we therefore state a precise 'good run' condition under which every local separator decision is globally valid, leading to correctness and high-probability subquadratic query bounds (Theorem 4.11). We prove that decomposable (chordal) graphs always yield good runs (Theorem 4.13). The *(CD)* procedure conditions on accumulated separators and works directly with Schur complements; strong faithfulness propagates to induced subgraphs, so the algorithm never breaks when $\mathrm{sn}(G) \le k$ and, upon termination, returns an $O(k\log(n/k))$ certificate with subquadratic query complexity (Theorem 4.18).

1.3 *Notation*

We write $\mathbb{S}^n$ for symmetric $n \times n$ matrices and $\mathbb{S}^n_+$ for the positive definite cone. For $\Sigma \in \mathbb{S}^n$ and $A, B \subseteq [n]$, $\Sigma_{A,B}$ denotes the corresponding submatrix; $\Sigma_{\setminus i,i}$ is the ith column with the ith entry removed. For a graph $G = (V, E)$ and $B \subseteq V$, G_B is the induced subgraph.

The rest of the paper is structured as follows. Section 2 formalizes the faithfulness assumptions and gives an entry-query routine for connected components. Section 3 presents the tree tester and proves Theorem 3.5. Section 4 describes the two descent schemes and proves Theorems 4.11 and 4.18. Section 5 discusses non-Gaussian extensions where reliable conditional-independence queries can replace Gaussian algebra; we note, for instance, that in the Gaussian case the test $X_i \perp\!\!\!\perp X_j \mid X_k$ reduces to verifying $\Sigma_{ij}\Sigma_{kk} = \Sigma_{ik}\Sigma_{jk}$ using four entries.

2. Faithfulness and connected components

2.1 *Separation in graphs*

For a graph $G = (V, E)$ we say that $A, A' \subset V$ are separated by a vertex set S if every path between a vertex in A and a vertex in A' contains a vertex in S. In other words, A and A' are disconnected in the graph $G \setminus S$ obtained from G by removing the vertices in S and all the incident edges.

In graphical models, vertices of the graph G represent random variables and no edge between i and j implies conditional independence $X_i \perp\!\!\!\perp X_j \mid X_{V\setminus\{i,j\}}$. For strictly positive densities, the Hammersley–Clifford theorem also implies that $X_i \perp\!\!\!\perp X_j \mid X_S$ whenever S separates i from j in G; see [22] for more details.

In the Gaussian case the conditional independence $X_i \perp\!\!\!\perp X_j \mid X_S$ is equivalent to $\Sigma_{ij}^{(S)} = 0$, where $\Sigma_{ij}^{(S)}$ is the (i,j)th entry of the conditional covariance matrix

$$\Sigma^{(S)} := \Sigma_{\overline{S},\overline{S}} - \Sigma_{\overline{S},S}\Sigma_{S,S}^{-1}\Sigma_{S,\overline{S}} \qquad \text{with } \overline{S} = V \setminus S. \tag{2.1}$$

Equivalently, by the Guttman rank additivity formula,

$$\operatorname{rank}\big(\Sigma_{A\cup S,\, A'\cup S}\big) = |S| \iff X_A \perp\!\!\!\perp X_{A'} \mid X_S,$$

so whenever S separates A and A' in G one has $\operatorname{rank}(\Sigma_{A\cup S,A'\cup S}) = |S|$. Note also that if $S = \{v\}$ for $v \in [n]$, (2.1) specializes to

$$\Sigma^{(v)} := \Sigma_{V\setminus\{v\},\, V\setminus\{v\}} - \frac{1}{\Sigma_{v,v}}\, \Sigma_{V\setminus\{v\},\, \{v\}}\, \Sigma_{\{v\},\, V\setminus\{v\}}, \tag{2.2}$$

and so $\Sigma_{ij}^{(v)} = 0$ if and only if $\Sigma_{ij}\Sigma_{vv} = \Sigma_{iv}\Sigma_{vj}$.

All our procedures rely on a divide-and-conquer approach, where in each step the graph is divided into balanced components by a small separator S, in which the matrix $\Sigma^{(S)}$ plays a crucial role.

2.2 *Faithfulness and learning connected components*

It follows from the definition of $M(G)$ in (1.1) that, if H is a subgraph of G, then $M(H) \subset M(G)$. In particular, if Σ is diagonal, then $\Sigma \in M(G)$ *for every* G with the given vertex set V. To be able to read from $\Sigma \in M(G)$ structural information about G, we need to require that Σ is in some way generic in $M(G)$. In this paper we consider two such genericity conditions.

Definition 2.1. We say that $\Sigma \in M(G)$ is *faithful* to G if for any $A, A', S \subseteq V$ we have $\operatorname{rank}\big(\Sigma_{A\cup S,\, A'\cup S}\big) = |S|$ if and only if S separates A and A' in G.

The implication 'S separates $A, A' \Rightarrow \operatorname{rank}(\Sigma_{A\cup S,A'\cup S}) = |S|$' is guaranteed by Section 2.1; faithfulness requires the converse to hold as well. This is a generic condition: the set of $\Sigma \in M(G)$ that violate it is a finite union of proper algebraic subsets, hence measure zero in $M(G)$.

Definition 2.2. We say that $\Sigma \in M(G)$ is *strongly faithful* over G if for any $A, A' \subseteq V$, the value $\operatorname{rank}(\Sigma_{A,A'})$ equals the size of a minimal separator of A and A' in G. Denote the set of strongly faithful covariance matrices by $M^\circ(G)$.

By standard trek-separation results (e.g., [35, Theorem 2.15]), $\operatorname{rank}(\Sigma_{A,A'})$ is always upper bounded by the size of a minimal A–A' separator, and equality holds on a Zariski open dense subset of $M(G)$. In particular, $M^\circ(G)$ is dense in $M(G)$ and strong faithfulness implies faithfulness.

In applications one may only have access to a noisy oracle returning approximate entries of Σ. In such cases, one replaces equalities by thresholds, which leads to additional complications; see, e.g., [38]. We note that some of our procedures work under weaker, localized genericity conditions:

DEFINITION 2.3. Fix $\tau \in \mathbb{N}$. The matrix $\Sigma \in M(G)$ is *τ-faithful to G* if the condition in Definition 2.1 holds whenever $|S| \le \tau$. We write $\Sigma \in M^{\tau}(G)$. Moreover, Σ is *τ-strongly faithful* to G if the condition in Definition 2.2 holds whenever $\mathrm{rank}(\Sigma_{A,A'}) \le \tau$. We write $\Sigma \in M^{\tau,\circ}(G)$.

The role of the faithfulness assumptions is that Σ then encodes accurately the underlying graph (or at least the parts we query). The next lemma is the first instance of this principle.

LEMMA 2.4. If Σ is 0-faithful then Σ has a block diagonal structure with blocks corresponding to the connected components of G and each block has all non-zero off-diagonal entries. More generally, if $\Sigma \in M^{\tau}(G)$ and $|S| \le \tau$, then $\Sigma^{(S)} \in M^{0}(G \setminus S)$.

Proof. If i,j are in different connected components of G, then $(\Sigma^{-1})_{ij} = 0$. Consequently, Σ^{-1} (equiv. Σ) is block diagonal with respect to the component partition. Under 0-faithfulness, the converse holds: if $\Sigma_{ij} = 0$ then i,j are in different components, so within each block all off-diagonal entries are non-zero. For the second claim, apply (2.1): if $|S| \le \tau$, then for $i,j \notin S$ we have $\Sigma^{(S)}_{ij} = 0$ if and only if S separates i and j in G, i.e., if and only if i and j lie in distinct components of $G \setminus S$. Hence $\Sigma^{(S)} \in M^{0}(G \setminus S)$. □

By Lemma 2.4, if $\Sigma \in M^{0}(G)$, the connected component containing a vertex i is the support of the vector $\Sigma_{\setminus i,\, i}$ consisting of all off-diagonal entries of the ith column of Σ. This observation underlies the following procedure `Components`(Σ, W) (Algorithm 1) that identifies the connected components of G intersected with an arbitrary index set $W \subseteq [n]$.

Algorithm 1 COMPONENTS(Σ, W)

Require: Index set $W \subseteq [n]$; oracle access to Σ
Ensure: $C_1, \dots, C_\ell$ (the components of G intersected with W)

```
1: C_0 ← ∅, ℓ ← 0, A_0 ← W
2: while A_ℓ ≠ ∅ do
3:     pick a random vertex i ∈ A_ℓ
4:     v ← Σ_{A_ℓ \ {i}, i}
5:     C_{ℓ+1} ← supp(v) ∪ {i}
6:     A_{ℓ+1} ← A_ℓ \ C_{ℓ+1}
7:     ℓ ← ℓ + 1
8: end while
```

LEMMA 2.5. Let $G = (V,E)$ have connected components $C_1, \dots, C_r$ and fix $W \subseteq V$. If $\Sigma \in M^{0}(G)$, then `Components`(Σ, W) outputs the sets $C_1 \cap W, \dots, C_r \cap W$. The algorithm uses $\mathcal{O}\big(|W| r\big)$ covariance queries.

Proof. Correctness follows from Lemma 2.4 and the construction. For the complexity,

$$|C_1 \cap W| + 2\,|C_2 \cap W| + \cdots + r\,|C_r \cap W| \;=\; \mathcal{O}\big(|W| r\big).$$

□

REMARK 2.6. Directly from (2.1) and Lemma 2.4, `Components` also identifies the components of $G \setminus S$ by replacing Σ with $\Sigma^{(S)}$, provided $\Sigma \in M^{|S|}(G)$. In the tree case of Section 3 it suffices to assume 1-faithfulness.

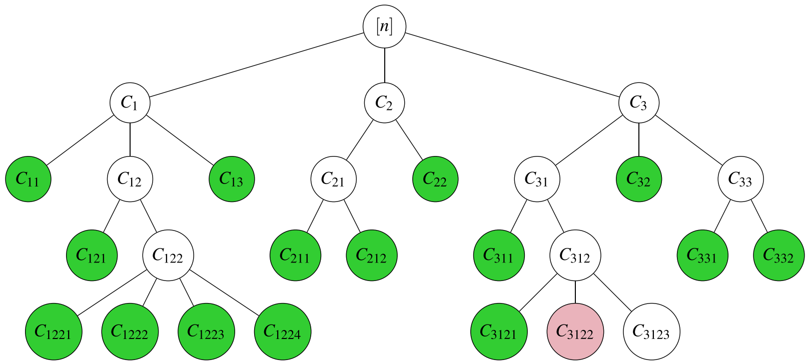

FIG. 1. Schematic picture of our divide-and-conquer algorithm.

2.2.1 *Connectedness assumption.* From now on we assume that G is connected (equivalently, we restrict attention to a single component). If the number of connected components is $o(n)$, they can be identified within our computational budget; moreover, $\mathrm{sn}(G)$ equals the maximum of the separation numbers of its components.

3. Testing if G is a tree

Recall that our general goal is to test properties of the underlying graph G with n nodes, where the graph is encoded in the support of Σ^{-1}. In this section we present a simple randomized procedure, which tests whether G is a tree; cf. (1.2).

Our testing procedure follows a divide-and-conquer approach. First an approximately central cut-vertex v^* is found (see Section 3.1) and the procedure descends to the connected components of $G \setminus \{v^*\}$ (see Section 3.2). In Fig. 1 we provide a simple example, where $G\setminus\{v^*\}$ has three components C_1, C_2, C_3. Here, centrality assures that each of these components has size at most $2n/3$ with high probability. Now, the second step of the procedure starts, where the same step is repeated in each of the subsets $B_i = C_i \cup \{v^*\}$. If a component C is small enough (green nodes in Fig. 1), we query the whole submatrix $\Sigma_{C,C}$ and run a direct check if this component is tree supported. We do not descend in this component any further. It may happen that we get evidence that one of the components is not a tree (the red node in Fig. 1), which may happen either because there is no cut-vertex in the given component or the direct check in a small component fails. In this case the algorithm stops and rejects the null hypothesis.

3.1 *Finding a central vertex in G*

Our procedure starts by finding an approximately central vertex in the graph G. We use a standard definition of node centrality in trees [16]. For $v \in V$, denote by $\mathscr{C}^{(v)}$ the set of connected components of $G \setminus \{v\}$. Define

$$M_{\mathrm{comp}}(v) := \max_{C \in \mathscr{C}^{(v)}} |C|. \tag{3.1}$$

Then v is central if it minimizes $M_{\mathrm{comp}}(v)$ over all $v \in V$. If the minimum is attained more than once, pick one of the optimal vertices arbitrarily. It is well known that, if G is a tree, then $\min_{v\in V} M_{\mathrm{comp}}(v) \le n/2$ and the minimum is attained at most twice (see [13]).

Since n is large, computing $M_{\text{comp}}(v)$ directly exceeds our computational bounds. We need to find a reliable and computationally efficient method to estimate the maximal component size in each $G \setminus \{v\}$. This can be done in a randomized way as follows:

(S1) Sample m nodes uniformly at random $W = \{v_1, \ldots, v_m\}$ without replacement from V.

(S2) For each $v \in V$ find the restricted decomposition

$$\mathscr{C}^{(v)}(W) \;:=\; \{C \cap W : C \in \mathscr{C}^{(v)} \text{ and } C \cap W \neq \emptyset\},$$

by running `Components`$(\Sigma^{(v)}, W \setminus \{v\})$.

(S3) Use the size of the largest element in $\mathscr{C}^{(v)}(W)$ as the estimator of $M_{\text{comp}}(v)$ in (3.1) by defining

$$\widehat{M}_{\text{comp}}(v) \;=\; n \max_{C \in \mathscr{C}^{(v)}} \frac{|C \cap W|}{|W|}, \qquad v^* := \arg\min_{v \in V} \widehat{M}_{\text{comp}}(v). \tag{3.2}$$

The parameter m is a computational budget parameter, which is required to be not too small (see (3.5)). The whole procedure is outlined in `FindBalancedPartitionTree` (Algorithm 2) and we now explain it in more detail.

Algorithm 2 FindBalancedPartitionTree(Σ, m)

Require: oracle for Σ, $m \in \mathbb{N}$
Ensure: v^* and $C_1, \ldots, C_k$

```
Let V be the index set of the rows/columns of Σ
Sample m nodes uniformly at random without replacement to obtain W ⊆ V
for v ∈ V do
    run Components(Σ^(v), W \ {v})
    compute M̂_comp(v) as in (3.2)
end for
if min_{v∈V} M̂_comp(v) > |V|/2 then
    break                                                ▷ G is not a tree
else
    return v* = arg min_{v∈V} M̂_comp(v) and
     Components(Σ^(v*), V \ {v*})
end if
```

Lemma 3.1. Let V be the set of vertices in the current call of the algorithm. The query complexity of `FindBalancedPartitionTree` is

$$\mathscr{O}\big(|V|\, m\, \Delta\big),$$

where Δ is the maximal degree of G.

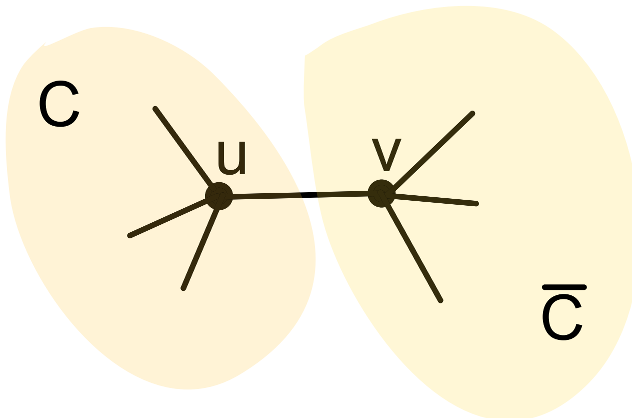

FIG. 2. Illustration of the proof of Lemma 3.2.

Proof. The algorithm queries Σ while running `Components`$(\Sigma^{(v)}, W \setminus \{v\})$. By Lemma 2.5, every call makes $\mathcal{O}(mr)$ queries, where r is the number of elements in $\mathscr{C}^{(v)}(W)$. Since removing a single vertex v can create at most $\deg(v) \le \Delta$ components, $r \le \Delta$. Moreover, by (2.2), each queried entry of $\Sigma^{(v)}$ uses $O(1)$ entries of Σ (namely, Σ_{ij}, Σ_{iv}, Σ_{vj}, Σ_{vv}). Thus this loop costs $\mathcal{O}(m\Delta)$ per v, and $\mathcal{O}(|V|m\Delta)$ overall. Once the minimizer v^* is found, the partition $\mathscr{C}^{(v^*)}(W)$ in step (S2) can be found efficiently using `Components`. This has query complexity $\mathcal{O}(|V|\,\Delta)$, which yields the stated bound. □

We still need to justify that the optimal separator v^* for W, as defined in (3.2), remains a good separator of the whole G. Recall that, if G is a tree, then $\min_{v\in V} M_{\text{comp}}(v) \le n/2$. What is less clear is that the same holds for $\widehat{M}_{\text{comp}}(v)$, which explains the `BREAK` line in `FindBalancedPartitionTree`. This fact relies on the following basic result.

LEMMA 3.2. Consider a tree $T = (V, E)$ with each node $v \in V$ having a weight $w(v) \ge 0$. Let $w(C) := \sum_{v\in C} w(v)$ for any $C \subseteq V$. Then there exists a *central* node v^* such that its removal splits the vertices into disjoint subsets $C_1, \ldots, C_\ell$ with

$$\max_i w(C_i) \;\le\; \frac{w(V)}{2}. \tag{3.3}$$

3.1.1 *Consequence for our estimator.* Taking $w(v) = 1$ if $v \in W$ and $w(v) = 0$ otherwise, we obtain $\min_{v\in V} \widehat{M}_{\text{comp}}(v) \le n/2$.

Proof. Take any vertex v. If v satisfies (3.3), we are done. Otherwise, let C be a component of $T \setminus \{v\}$ with $w(C) > \frac{w(V)}{2}$ and let u be the neighbour of v in C. Denote by $\overline{C}$ the union of all other components of $T \setminus \{v\}$; then $w(\overline{C}) < \frac{w(V)}{2}$. If u does not satisfy (3.3), repeat the move towards the unique neighbour in the heaviest component. The maximal component weight is non-increasing and must strictly decrease at some step; the process terminates at a vertex satisfying (3.3). □

Lemma 3.2 implies that, if $\min_{v\in V} \widehat{M}_{\text{comp}}(v) > \frac{n}{2}$, we get an immediate guarantee that G is not a tree. Moreover, if $M_{\text{comp}}(v)$ is bounded away from $\frac{n}{2}$, say if $M_{\text{comp}}(v) > \frac{2}{3}n$, then $\widehat{M}_{\text{comp}}(v) > \frac{n}{2}$ with high probability if m is sufficiently large.

LEMMA 3.3. Let $W \subseteq V$ be sampled uniformly without replacement with $|W| = m$. For any $v \in V$, if $M_{\text{comp}}(v) > \frac{2}{3}n$ then

$$\mathbb{P}\Big(\widehat{M}_{\text{comp}}(v) \le \tfrac{n}{2}\Big) \;\le\; \exp\Big(-\tfrac{m-1}{18}\,\tfrac{n-1}{n-m}\Big) \;\le\; e^{-(m-1)/18}.$$

Consequently,

$$\mathbb{P}\Big(\exists v \in V \text{ with } M_{\text{comp}}(v) > \tfrac{2n}{3} \text{ and } \widehat{M}_{\text{comp}}(v) \le \tfrac{n}{2}\Big) \;\le\; n\,e^{-(m-1)/18}.$$

Proof. Fix $v \in V$ such that $M_{\text{comp}}(v) > \frac{2}{3}n$. Let $C \subseteq V \setminus \{v\}$ be a component of $G \setminus \{v\}$ with $|C| = M_{\text{comp}}(v)$. Set $N := n-1$ and define the deterministic vector $x = (x_u)_{u\in V\setminus\{v\}} \in \{0,1\}^N$ by

$$x_u = \begin{cases} 0, & u \in C, \\ 1, & u \notin C. \end{cases}$$

Denote $\mu := \frac{1}{N}\sum_u x_u = \frac{N-|C|}{N} < \frac{1}{3}$ where inequality follows from $|C| > \frac{2}{3}n$. Draw $W \subseteq V$ uniformly amongst all m-subsets and let $W' := W \setminus \{v\}$ with size $m' := |W'| \in \{m, m-1\}$. Conditional on m', the set W' is uniform amongst all m'-subsets of $V \setminus \{v\}$. Define the sample mean

$$S_{m'} \;=\; \frac{1}{m'}\sum_{u\in W'} x_u.$$

Then $|W' \cap C| = m'(1 - S_{m'})$. By definition,

$$\widehat{M}_{\text{comp}}(v) \;=\; n \cdot \max_{C'\in\mathscr{C}^{(v)}} \frac{|C' \cap W'|}{|W'|}.$$

The event $\{\widehat{M}_{\text{comp}}(v) \le n/2\}$ implies $|W' \cap C| \le m'/2$, equivalently $S_{m'} \ge 1/2$. By Serfling's inequality for sampling without replacement [34, Cor. 1.1], for any $t > 0$,

$$\mathbb{P}(S_{m'} - \mu \ge t) \;\le\; \exp\Big(-2m't^2\frac{N}{N-m'}\Big).$$

Taking $t = \frac{1}{2} - \mu > \frac{1}{6}$ yields

$$\mathbb{P}\Big(S_{m'} \ge \tfrac{1}{2}\Big) \;\le\; \exp\Big(-\tfrac{m'}{18}\frac{n-1}{n-1-m'}\Big).$$

For $m' \in \{m, m-1\}$, the factor $\frac{n-1}{n-1-m'}$ is minimized when $m' = m-1$, while m' itself is also minimized at $m' = m-1$. Therefore, we can bound the exponent from below by substituting $m' = m-1$ and using the smallest ratio $\frac{n-1}{n-m}$:

$$\mathbb{P}\Big(\widehat{M}_{\text{comp}}(v) \le n/2\Big) \;\le\; \exp\Big(-\tfrac{m-1}{18}\frac{n-1}{n-m}\Big).$$

Finally, since $\frac{n-1}{n-m} \geq 1$, we have

$$\mathbb{P}(\widehat{M}_{\text{comp}}(v) \leq n/2) \ \leq \ e^{-(m-1)/18}.$$

The union bound over all $v \in V$ gives the claimed result. □

3.2 *Descending into sub-components*

After completing steps (S1)–(S3), our procedure finds v^*, which optimizes $\widehat{M}_{\text{comp}}(v)$ over $v \in V$, and the corresponding components $C_1, \ldots, C_\ell$. If $\widehat{M}_{\text{comp}}(v^*) > n/2$, it stops with a guarantee that G cannot be a tree (like the red node in Fig. 1). If $\widehat{M}_{\text{comp}}(v^*) \leq n/2$ it descends into the connected components of $G \setminus \{v^*\}$. Namely, for every $C \in \mathscr{C}^{(v^*)}$, we apply our procedure to the smaller matrix $\Sigma_{B,B}$ with $B = C \cup \{v^*\}$. For each such B we first check if $|B| \leq m$. If yes (green nodes in Fig. 1), we query the whole matrix $\Sigma_{B,B}$, invert it and identify the underlying subgraph directly. Otherwise ($|B| > m$), we run `FindBalancedPartitionTree` on $\Sigma_{B,B}$ and proceed recursively. The whole procedure is outlined in `TestTree` (Algorithm 3).

To prove the correctness of this approach, we argue that the structure of the induced subgraph G_B can be read directly from the submatrix $\Sigma_{B,B}$. This is a matrix-algebraic instance of [11].

LEMMA 3.4. Let G be any graph and let C be a connected component of $G \setminus S$. Denote $B := C \cup S$ and assume $\Sigma \in M(G)$. Then $\Sigma_{B,B} \in M(G_B)$ if and only if for every pair $i,j \in S$ that are not connected by an edge in G_B, we have $(\Sigma_{B,B})^{-1}_{ij} = 0$. In particular, if S is a clique then $\Sigma_{B,B} \in M(G_B)$.

Proof. By definition, $\Sigma_{B,B} \in M(G_B)$ means that for every non-edge $\{i,j\}$ of the induced subgraph G_B, the corresponding entry of the inverse satisfies $(\Sigma_{B,B})^{-1}_{ij} = 0$. Let $A := V \setminus B$ and let $K := \Sigma^{-1}$. By the block matrix inversion formula,

$$(\Sigma_{B,B})^{-1} \ = \ K_{B,B} \ - \ K_{B,A} K_{A,A}^{-1} K_{A,B}. \tag{3.4}$$

Assume first that $\Sigma_{B,B} \in M(G_B)$. Then by the above characterization, for every non-edge $\{i,j\}$ of G_B we have $(\Sigma_{B,B})^{-1}_{ij} = 0$, in particular for all non-edges $\{i,j\} \subseteq S$. Hence the stated condition holds.

Conversely, assume that for every pair $i,j \in S$ that are not connected by an edge in G_B, we have $(\Sigma_{B,B})^{-1}_{ij} = 0$. We need to show that $(\Sigma_{B,B})^{-1}_{ij} = 0$ also for all non-edges where at least one of i,j lies in C. For such a pair $\{i,j\}$ with $i \in C$ (and possibly $j \in C$ or $j \in S$), the definition of G_B implies that i has no neighbours outside B. Since $\Sigma \in M(G)$, in particular, for $i \in C$ we have $K_{iA} = 0$, because i is not adjacent to any vertex in $A = V \setminus B$. Moreover, $K_{ij} = 0$ because $\{i,j\}$ is a non-edge in G. Thus, by (3.4), $(\Sigma_{B,B})^{-1}_{ij} = 0$ for all non-edges involving a vertex from C. □

The following result bounds the query complexity and shows correctness.

THEOREM 3.5. Let G be a connected graph and let $\Sigma \in M^1(G)$. Fix $\epsilon \in (0,1)$ and let

$$m \ = \ \left\lceil 18 \log\left(\frac{5n^2}{\epsilon} \log n\right)\right\rceil. \tag{3.5}$$

Algorithm 3 TestTree(Σ, m)

```
Require: oracle for Σ, m ∈ ℕ
Let V be the index set of the rows/columns of Σ
if |V| ≤ m then
    run a direct test
else
    run FindBalancedPartitionTree(Σ, m) to get v* and components C_1, ..., C_ℓ
    for i = 1, ..., ℓ do
        B ← C_i ∪ {v*}
        run TestTree(Σ_{B,B}, m)
    end for
end if
```

Then `TestTree` correctly identifies whether G is a tree. Moreover, with probability at least $1 - \epsilon$, `TestTree` runs with total query complexity

$$\mathscr{O}\left(n \log(n)\ \log(n/\epsilon)\ \Delta\right),$$

where Δ is the maximum degree of G.

The upper bound shows that, whenever $\Delta = O(n^{1-\gamma})$ for some $\gamma > 0$, the algorithm requires sub-quadratic query complexity. In particular, when the maximum degree is bounded, the query complexity is quasi-linear. It is not difficult to see that, without any bound on Δ, one cannot hope for non-trivial query complexity. For example, in order to test whether G is a star of degree $n-1$ or it is a star with one extra edge added, any algorithm needs $\Omega(n^2)$ query complexity. Instead of giving a formal statement, we refer the reader to [25] for related arguments in the context of structure learning.

Proof. By Lemma 3.1, the initial call has cost $\mathscr{O}(n\, m\, \Delta)$. In the second step, we run the same procedure on each $B = C \cup \{v^*\}$ with cost $\mathscr{O}((|C|+1)\, m\, \Delta)$. Since $\sum_C (|C|+1) \le 2n$, the total still scales as $\mathscr{O}(n\, m\, \Delta)$. The same bound holds at every level, because the sum of block sizes across the components processed at any level is at most n.

Let $\mathscr{E}_\ell$ be the event that all components at level ℓ have size at most $(\frac{2}{3})^\ell n$. Let

$$\ell^* := \left\lceil \frac{\log(n/m)}{\log(3/2)} \right\rceil \le 5 \log\left(\frac{n}{m}\right).$$

On $\mathscr{E}_{\ell^*}$ every component at level ℓ^* has size $\le m$, so one more call stops. Hence the total query complexity is $\mathscr{O}\big((\ell^*+1)\, n\, m\, \Delta\big) = \mathscr{O}\big(n \log n\, m\, \Delta\big)$. The final complexity formula follows then from (3.5).

It remains to bound $\mathbb{P}(\neg\mathscr{E}_{\ell^*})$. By Lemma 3.3, for any component B,

$$\mathbb{P}\Big(\exists v \in B\ \text{ s.t. }\ M_{\text{comp}}(v) > \tfrac{2|B|}{3}\ \text{ and }\ \widehat{M}_{\text{comp}}(v) \le \tfrac{|B|}{2}\Big) \le |B| e^{-m/18}.$$

By the union bound, the probability that in at least one call we do not get a balanced split, can be bounded by

$$n^2\ell^* e^{-m/18} \;\le\; n^2 5\log\left(\frac{n}{m}\right)\frac{\epsilon}{5n^2\log(n)} \;\le\; \frac{\log\left(\frac{n}{m}\right)}{\log(n)}\epsilon \;\le\; \epsilon,$$

which proves that the probability at least $1-\epsilon$ we get the claimed query complexity.

Correctness follows by induction on the recursion: if G is a tree, every block $B = C \cup \{v^*\}$ induces a tree (Lemma 3.4 with $S = \{v^*\}$), and the direct checks on leaves certify 'tree'; if G is not a tree, some leaf B contains a cycle and the direct check rejects. □

4. Testing small separation numbers

In this section, we generalize the procedure for testing trees to a significantly richer class of graphs, characterized by their separation number, as given in (1.3).

DEFINITION 4.1 (Balanced separation number). Let $G = (V,E)$ be a graph and fix $\alpha \in (1/2, 1)$ (we will use $\alpha = 2/3$). The α-separation number $\mathrm{sn}_\alpha(G)$ is the smallest integer k such that for every $W \subseteq V$ there exist a partition $W = A \cup A'$ with

$$\max\{|A|, |A'|\} \le \alpha\,|W|$$

and a set $S \subseteq V$ with $|S| \le k$ such that in $G \setminus S$ there is no path between any vertex of A and any vertex of A'. Such S is called an (α, k)-separator of W. We write $\mathrm{sn}(G) := \mathrm{sn}_{2/3}(G)$.

REMARK 4.2 (Relation to the literature). Our definition uses *global* separators ($S \subseteq V$) to separate the parts A, A' of a chosen subset $W \subseteq V$ *in* G. This matches the Graph Minors formulation of 'w-balanced separations for every $\{0,1\}$ weight w' (take $w = \mathbf{1}_W$); see, e.g., the discussion in the introduction of [8]. Note that [8] also considers a slightly stronger *subgraph* version, which requires that every induced subgraph $G[W]$ admit a balanced separator *inside* W. We work with the global notion throughout.

PROPOSITION 4.3 (Treewidth vs. global separation number). For every graph G,

$$\mathrm{sn}(G) \;\le\; \mathrm{tw}(G)+1 \qquad \text{and} \qquad \mathrm{tw}(G) \;\le\; 4\,\mathrm{sn}(G).$$

Proof. (idea and sources). The implication $\mathrm{sn}(G) \le \mathrm{tw}(G)+1$ is the classical 'balanced separator from bounded treewidth' fact due to Robertson–Seymour: any tree decomposition of width t yields, for *every* vertex weight $w \ge 0$, a w-balanced separator of order at most $t+1$ (see [31,§2, Lemmas (2.5)–(2.6)]; for a textbook presentation, cf. [10,§11.2]). Instantiating $w = \mathbf{1}_W$ gives the stated bound on $\mathrm{sn}(G)$.

Conversely, Robertson & Seymour [31] showed that if for every {0,1}-valued weight function w the graph admits a w-balanced separator of order at most s, then $\mathrm{tw}(G) < 4s$ (cf. the discussion in [8]). We use only this consequence and do not reproduce the argument here. □

Conceptually, our testing algorithm is analogous to the one presented for trees. Thus, the rest of this section is organized in a similar way as Section 3. In Section 4.1 we provide a procedure to efficiently find a small balanced separator. Having such a separator S, the algorithm descends in the components of $G \setminus S$. This is explained in Section 4.2. Then the procedure proceeds recursively and in Sections 4.3 and 4.4 we consider two ways this can happen.

4.1 *Finding a balanced separator*

To identify a small balanced separator in a graph G, we adopt a randomized approach similar to that used for trees but with stronger genericity conditions. Our strategy involves sampling a subset of vertices, denoted as W, and then searching for a compact balanced separator of W. The approach developed in [9] provides a basis for asserting that even for relatively small values of $m = |W|$, a balanced separator of W will, with high probability, serve as a balanced separator for the entire graph. However, it is important to note a critical distinction from the tree case: the number of candidate separators of size at most k in a component C is $\binom{\sum_{s=0}^{k}}{|C|\, s}$, which is exponential in k. We therefore avoid exhaustive enumeration; our query bounds remain subquadratic, while the runtime has an additional factor exponential in k, as discussed at the end of Section 4.3. We now describe in detail how this is performed.

To describe our procedure to find a balanced separator, first note that, under strong faithfulness, if $|S| = r$ and $\mathrm{rank}(\Sigma_{A\cup S, A'\cup S}) = r$ then S is a minimal separator of A, A'. To find a balanced separator of W we proceed as follows: see Algorithm 6. First, search exhaustively through all partitions of W into sets A, A' with $\max\{|A|, |A'|\} \le \frac{2}{3}|W|$. For each such split A/A' compute the rank of $\Sigma_{A,A'}$. If $\mathrm{sn}(G) \le k$, then such a minimal rank needs to be less than or equal to k, which we can use as the early detection for $\mathrm{sn}(G) > k$. Take any split A/A' that minimizes this rank, say $\mathrm{rank}(\Sigma_{A,A'}) = r \le k$.

Now, if Σ is k-strongly faithful, `ABSeparator` (Algorithm 4) finds a minimal separator of A and A'. It first checks for all $v \in V$ whether $\mathrm{rank}(\Sigma_{A\cup\{v\}, A'\cup\{v\}}) = r$, which is equivalent with v being an element in some minimal separator of A, A'; cf. Lemma 3 in [25]. After identifying the set U of all nodes that lie in some minimal separator of A, A', we proceed to find a minimal separator. This is done by picking any element v_0 in U, fixing $S = \{v_0\}$, and then adding nodes from U to S one by one, at each step making sure that S is part of a minimal separator of A, A'; here again we use Lemma 3 in [25] and simply check if $\mathrm{rank}(\Sigma_{A\cup S, A'\cup S}) = r$. Because, Σ is k-strongly faithful, this procedure concludes with $|S| = r$, a $(\frac{2}{3}, k)$-separator S of W; cf. Definition 4.1.

After finding a $(\frac{2}{3}, k)$-separator S of W we would like to argue that S is also an (α, k)-separator of the entire node set V for some $\alpha \in [\frac{2}{3}, 1)$. To determine the size of W that allows us to draw such a conclusion, we follow the discussion in Section 4 in [25]; a key tool is from [9] who bound the vc dimension of the class of sets of vertices forming the connected components of a graph obtained by removing k arbitrary vertices.

THEOREM 4.4. Fix $0 < \delta, \delta' < \frac{1}{3}$. Suppose that $W \subseteq V$ is obtained by sampling m vertices from V uniformly at random, with replacement, where m satisfies

$$m = \left\lceil \max\left(\frac{110k}{\delta^2}\log\left(\frac{88k}{\delta^2}\right), \frac{2}{\delta^2}\log\left(\frac{2}{\delta'}\right)\right)\right\rceil. \tag{4.1}$$

Then, with probability at least $1-\delta'$, every $(\frac{2}{3}, k)$-separator of W is a $(\frac{2}{3}+\delta, k)$-separator of V.

Proof. A set $W \subseteq V$ is a δ-*sample* for (G, k) if for all sets $S \subseteq V$ with $|S| \le k$ and all $C \in \mathscr{C}^{\mathscr{S}}$,

$$\frac{|C|}{|V|} - \delta \le \frac{|W \cap C|}{|W|} \le \frac{|C|}{|V|} + \delta. \tag{4.2}$$

Algorithm 4 ABSeparator(Σ, A, A')

$U \leftarrow \emptyset$
$r \leftarrow \text{rank}(\Sigma_{A,A'})$
Let V be the index set of the rows of Σ
for all $v \in V$ **do**
  if $\text{rank}\big(\Sigma_{A\cup\{v\},A'\cup\{v\}}\big) = r$ **then**
    $U \leftarrow U \cup \{v\}$
  end if
end for
choose some $v_0 \in U$
$S \leftarrow \{v_0\}$
for all $u \in U \setminus \{v_0\}$ **do**
  if $\text{rank}\big(\Sigma_{A\cup S\cup\{u\},A'\cup S\cup\{u\}}\big) = r$ **then**
    $S \leftarrow S \cup \{u\}$
  end if
end for
return S

In other words, W allows the estimation of the relative sizes of all connected components, exactly as in the tree case; cf. Definition 3.2 in [9]. By Lemma 3.3 in [9] every $(\frac{2}{3}, k)$-separator of a δ-sample W is a $(\frac{2}{3}+\delta, k)$ separator of the whole graph. Thus, to conclude the result, we need to show that with probability $\geq 1 - \delta'$, W is a δ-sample if $m = |W|$ satisfies (4.1). This can be done by combining Theorem 22 and Lemma 23 in [25]. $\square$

4.2 *Descending into sub-components*

Our procedure begins by testing equation (1.3) for a given $k \geq 1$. The input is provided by an oracle on Σ. If the cardinality of V is less than or equal to k, there is nothing to verify, and the procedure ends. Otherwise, the procedure attempts to find a small balanced separator S of G using Algorithm 6, as outlined in Section 4.1. If no such small balanced separator can be found, the procedure halts. If a balanced separator is found, we run `Components`($\Sigma^{(S)}, V \setminus S$) to identify the connected components of $G \setminus S$.

The algorithm then descends into the components and applies the same procedure to each subset of nodes. We outline two methods of descent. The first is both conceptually and computationally simpler. It guarantees a correct answer when the graph G is decomposable, but may sometimes halt inconclusively for certain non-decomposable graphs. In such cases, we execute our second algorithm, which always provides a bound on the separation number. These two methods of descent are referred to as the MD and the CD:

(MD) For each $C \in \mathscr{C}^{(S)}$, we apply our procedure to the smaller matrix $\Sigma_{B,B}$, where $B = C \cup S$.

(CD) For each $C \in \mathscr{C}^{(S)}$, we apply our procedure to the smaller matrix $\Sigma^{(S)}_{C,C}$.

The procedure for testing a small separation number is executed by running either `test.marginal` (Algorithm 5), or `test.conditional` (Algorithm 7). As we will demonstrate, both procedures yield significant insights.

4.3 *Marginal descent*

In the MD, we regress into each subset $B = C \cup S$, where C is a connected component of $G \setminus S$. We then repeat the procedure on each of these smaller sets. However, justifying this step is more complex than in the case of trees. The complexity arises from how we access information about the induced subgraph G_B. By applying Lemma 3.4, we deduce that if S is a clique in G, then $\Sigma_{B,B}$ belongs to $M(G_B)$ but this is generally not guaranteed otherwise. Furthermore, we need to monitor how our genericity conditions change when transitioning to $\Sigma_{B,B}$. If S is a clique, we also demonstrate that $\Sigma_{B,B}$ remains generic in the sense that if Σ is strongly faithful, then $\Sigma_{B,B}$ is also strongly faithful. However, again, this statement do not hold universally (refer to Proposition 4.7 for more details).

We start with a graph-theoretic definition that is standard in the study of graph treewidth.

Definition 4.5 (Torso on a vertex set). Let G be a graph and $C \subseteq V(G)$. The *torso* torso(G, C) is the graph with vertex set C in which $u, v \in C$ are adjacent if either $\{u, v\} \in E(G)$ or there exists a path in G connecting u and v whose internal vertices lie outside C.

The following result will be useful in our analysis; see Proposition 2.5 in [27].

Proposition 4.6 (Monotonicity of separations in torsos). If $C_1 \subseteq C_2 \subseteq V(G)$ and $u, v \in C_1$, then $S \subseteq C_1$ separates u and v in torso(G, C_1) if and only if S separates u and v in torso(G, C_2). In particular, with $C_2 = V(G)$, $S \subseteq C_1$ separates u and v in torso(G, C_1) if and only if it separates them in G.

Proposition 4.7 (Marginals are Markov/faithful to the torso). Let $B = C \cup S$ where C is a connected component of $G \setminus S$. If $\Sigma \in M(G)$ then $\Sigma_{B,B} \in M(\text{torso}(G, B))$. Moreover, if Σ is k-faithful to G then $\Sigma_{B,B}$ is k-faithful to torso(G, B).

Proof. Let $A := V \setminus B$ and $K := \Sigma^{-1}$. For Gaussian concentration graphs, marginalizing onto B gives the Schur complement

$$(\Sigma_{B,B})^{-1} \;=\; K_{B,B} \;-\; K_{B,A}\, K_{A,A}^{-1}\, K_{A,B}. \tag{4.3}$$

We claim that for distinct $i, j \in B$, if there is no i–j path in G whose internal vertices lie in A then $\big((\Sigma_{B,B})^{-1}\big)_{ij} = 0$. By definition, this is exactly the adjacency rule of the torso(G, B), so this claim implies $\Sigma_{B,B} \in M(\text{torso}(G, B))$.

Because K is sparse with respect to G, the vectors $K_{i,A}$ and $K_{A,j}$ are supported on neighbours of i and j in A. The term $K_{B,A}K_{A,A}^{-1}K_{A,B}$ aggregates contributions of walks with internal vertices in A. Thus, if there is no path between i, j that lies entirely in A, $K_{i,A}K_{A,A}^{-1}K_{A,j} = 0$, which implies the first claim.

For the second claim. Assume Σ is k-faithful to G. Let $S' \subseteq B$ with $|S'| \le k$ and suppose $\Sigma_{ij}^{(S')} = 0$ for $i, j \in B \setminus S'$. By k-faithfulness in G, the set S' separates i and j in G. By Proposition 4.6, S' separates i, j in torso(G, B). □

We rely on the following important result that shows that local separators found by our procedure in some component B become global separators of G.

Lemma 4.8. Suppose Σ is k-faithful to G. Let S be a separator satisfying $|S| \le k$ found at some point in the algorithm by running `Separator`$(\Sigma_{B,B}, k)$. Let $C_1, \ldots, C_\ell$ be the corresponding components obtained by running `Components`$(\Sigma_{B,B}^{(S)} B \setminus S)$. Then S separates $C_1, \ldots, C_\ell$ in G.

Algorithm 5 TEST.MARGINAL(Σ, k)

Require: an oracle for Σ, $k \geq 1$
Let V be the index set of the rows of Σ
if $|V| \leq k$ **then**
  stop
else
  run SEPARATOR(Σ, k) to get a balanced separator S with $|S| \leq k$
  run COMPONENTS$(\Sigma^{(S)}, V \setminus S)$ to get the corresponding components $C_1, \ldots, C_\ell$
  for $i = 1, \ldots, \ell$ **do**
    $B \leftarrow C_i \cup S$
    run TEST.MARGINAL$(\Sigma_{B,B}, k)$
  end for
end if

Proof. By Proposition 4.7, $\Sigma_{B,B} \in M(\text{torso}(G, B))$, so the computed $C_1, \ldots, C_\ell$ are the components of $\text{torso}(G, B) \setminus S$. Hence S separates them in $\text{torso}(G, B)$. Applying Proposition 4.6 with $C_1 = B$ and $C_2 = V(G)$ shows that S separates the same vertex sets in G.

□

Algorithm 6 SEPARATOR(Σ, k)

Require: an oracle for Σ, $k \geq 1$
Pick a set W by taking m vertices uniformly at random with replacement, where m satisfies (4.1) with $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log(n)}$ for a fixed $\epsilon \in (0, 1)$
Search exhaustively through all partitions of W into sets A, A' with $|A|, |A'| \leq \frac{2}{3}|W|$, minimizing $\text{rank}(\Sigma_{A,A'})$
Let A/A' be any partition minimizing the rank
if $\text{rank}(\Sigma_{A,A'}) > k$ **then**
  break
end if
$S \leftarrow$ ABSEPARATOR(Σ, A, A')
return S

In our algorithm, the crucial step involves finding a balanced separator. Suppose that Σ is k-strongly faithful to graph G. In the initial step, we identify S as a minimal separator in the random sample $W \subseteq V(G)$. However, in subsequent steps, we apply our algorithm to $\Sigma_{B,B}$. By Proposition 4.7, $\Sigma_{B,B}$ is Markov (and k-faithful) with respect to $\text{torso}(G, B)$. Strong faithfulness, however, may *fail* to transfer to $\Sigma_{B,B}$ for $\text{torso}(G, B)$. Consequently, even if $\text{rank}(\Sigma_{A,A'}) = r$ (meaning that A and A' admit a separator of size r in G), `Separator` (Algorithm 6) run on $\Sigma_{B,B}$ may output a proper subset of a minimal separator when that separator is not fully contained in B. This motivates the 'good run' notion below.

DEFINITION 4.9. We say that `test.marginal` has a *good run* if, in each call of `ABSeparator`, the output S of the algorithm is a separator of A and A', equivalently, $|S| = \text{rank}(\Sigma_{A,A'})$.

If Σ is k-strongly faithful to G this definition is just saying that in each call of `ABSeparator` the sets A, A' are minimally separated within the current component B. A sufficient condition for this to happen is that, for each component B, $\Sigma_{B,B}$ is k strongly faithful to torso(G, B).

LEMMA 4.10. If G is a decomposable graph with $\mathrm{sn}(G) \leq k$ and Σ is k-strongly faithful to G then, for every subsequent component B, in the run of `test.marginal`, Σ_{BB} is k-strongly faithful to G_B.

Proof. Let S be a minimal separator found in the first step of the procedure, and suppose we descend into $B = C \cup S$, where C is one of the components in $\mathscr{C}^{(S)}$. Dirac [6] characterized decomposable graphs as those for which every minimal separator is a clique. Thus, in our case, S is a clique, and torso$(G, B) = G_B$ (no new edges are added inside B). To show that $\Sigma_{B,B}$ is k-strongly faithful, suppose that $\mathrm{rank}(\Sigma_{A,A'}) = r \leq k$ for some $A, A' \subseteq B$. Since $\Sigma \in M^{k,\circ}(G)$, A and A' are minimally separated in G by some S' with $|S'| = r$. We show that $S' \subseteq B$. Suppose that S' contains a vertex $v \notin B$. By minimality of S', there is a path P between A and A' that crosses v but no other element of S'. Let P_1 be the part of P that leads from A to v, and P_2 be the part from v to A'. Both P_1 and P_2 contain vertices in S. Let u_1 be the first such vertex on P_1, and u_2 be the last such vertex on P_2 (where $u_1 = u_2$ is possible). Since S forms a clique, u_1 and u_2 are connected. Thus, walking along P_1, jumping from u_1 to u_2, and then going to A' along P_2 gives a path from A to A' with no vertices in S'. But this is a contradiction. □

Decomposability is a sufficient but definitely not a necessary condition for our algorithm to have a good run. Practically, a good run can be detected by checking at each call of `ABSeparator` whether $|S| = \mathrm{rank}(\Sigma_{A,A'})$.

THEOREM 4.11. Let G be a connected graph and let Σ be k-strongly faithful to G. Fix $\epsilon \in (0, 1)$ and let m be given by (4.1) with $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log(n)}$. Suppose `test.marginal` has a good run. If `test.marginal` terminates, then $\mathrm{sn}(G) \leq 2k$. If it breaks, then $\mathrm{sn}(G) > k$. Moreover, with probability at least $1 - \epsilon$, it runs with total query complexity

$$\mathscr{O}\Big(n\log(n)\max\{m, k\Delta\}\Big) = \mathscr{O}\left(n\log(n)\max\left\{\log(\tfrac{n}{\epsilon}), k\Delta\right\}\right).$$

LEMMA 4.12. Running `Separator`$(\Sigma_{V,V}, k)$ in Algorithm 6 takes query complexity $\mathscr{O}(|V|\, m)$.

Proof. To find a balanced partition A, A' of $W \subseteq V$, `Separator`$(\Sigma_{V,V}, k)$ exhaustively scans the $< 2^m$ balanced partitions of W and computes the rank of $\Sigma_{A,A'}$, which requires querying $\Sigma_{W,W}$ and thus $\mathscr{O}(m^2)$ entries. Then `ABSeparator`$(\Sigma_{V,V}, A, A')$ evaluates ranks for m-sized augmentations across all $v \in V$, costing $\mathscr{O}(m|V|)$. The possible additional $\mathscr{O}(k^2)$ entries from the second loop do not change the order, since both $|V|$ and m exceed k. Altogether, the cost is $\mathscr{O}(|V|\, m)$. □

Proof. (of Theorem 4.11). With $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log n}$, the bound (4.1) gives

$$m = \left\lceil \max\left\{\frac{110k}{\delta^2}\log\Big(\frac{88k}{\delta^2}\Big),\ \frac{2}{\delta^2}\log\Big(\frac{2}{\delta'}\Big)\right\}\right\rceil = \left\lceil \max\left\{m_0(k),\ \frac{1800}{49}\log\Big(\frac{20\,n\log n}{\epsilon}\Big)\right\}\right\rceil,$$

where $m_0(k) := \frac{110k}{\delta^2}\log\big(\frac{88k}{\delta^2}\big)$ depends only on k, which is fixed. Hence, we get $m = \mathscr{O}(\log(\frac{n}{\epsilon}))$.

Query complexity. At the root ($|V| = n > k$), Lemma 4.12 gives $\mathscr{O}(nm)$ for `Separator`(Σ, k). If it does not break, we obtain a $(\frac{2}{3}, k)$-separator S_1 of W and then run `Components`$(\Sigma^{(S_1)}, V\backslash S_1)$. Querying $\Sigma^{(S_1)}$ has initial cost $|S_1|^2$ (for Σ_{S_1,S_1}), and each entry $\Sigma^{(S_1)}_{ij}$ costs $\mathscr{O}(|S_1|)$ (for Σ_{ij}, Σ_{i,S_1}, Σ_{j,S_1}). Using

Lemma 2.5, computing components costs $\mathscr{O}(|S_1|\, n\, \Delta) = \mathscr{O}(kn\Delta)$; cf. Remark 4.16. Hence the level-1 cost is $\mathscr{O}(n \max\{m, k\Delta\})$.

At level 2, for each $B = C \cup S_1$, Lemma 4.12 gives $\mathscr{O}(|C|m)$ for $\texttt{Separator}(\Sigma_{B,B}, k)$, thus $\mathscr{O}(nm)$ summed across components. If S is the separator found in B, then $\texttt{Components}(\Sigma^{(S)}_{B,B}, B \setminus S)$ costs $\mathscr{O}(k|C|\Delta)$, additive across components. Therefore, level-2 also costs $\mathscr{O}(n \max\{m, k\Delta\})$. The same bound holds for every subsequent level.

Let $\mathscr{E}_\ell$ be the event that all components in level ℓ have size at most $(\frac{2}{3} + \delta)^\ell n = (\frac{9}{10})^\ell n$. Define

$$\ell^* := \left\lceil \frac{\log\left(\frac{n}{k}\right)}{\log\left(\frac{10}{9}\right)} \right\rceil \leq 10 \log\left(\frac{n}{k}\right). \tag{4.4}$$

Under $\mathscr{E}_{\ell^*}$, every component at level ℓ^* has size $\leq k$, so one more call stops. Summing over $\ell^* + 1$ levels yields total complexity $\mathscr{O}(\ell^*\, n \max\{m, k\Delta\})$, which with (4.4) gives the bound stated in the theorem.

High probability. Failure of $\mathscr{E}_{\ell^*}$ means that in some call the returned S is not a $(\frac{9}{10}, k)$-separator of the current component. By Theorem 4.4 (with $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log(n)}$), the failure probability per call is at most δ'. There are at most $n\ell^*$ calls, so by the union bound

$$n\ell^*\delta' \leq n \cdot 10 \log\left(\tfrac{n}{k}\right) \cdot \frac{\epsilon}{10n\log(n)} \leq \epsilon.$$

Correctness. Assume first that the algorithm terminates without breaking. We show $\mathrm{sn}(G) \leq 2k$. Fix an arbitrary $W \subseteq V$. Let $\mathscr{T}$ be the recursion tree: internal nodes correspond to separators S_t; the root is S_1; leaves are blocks $L_1, \ldots, L_u$ with $|L_i| \leq k$.

Apply the root separator S_1 and let $\mathscr{C}^{(S_1)}$ be the connected components of $G \setminus S_1$. If every $C \in \mathscr{C}^{(S_1)}$ satisfies $|C \cap W| \leq \frac{2}{3}|W|$, then, since these components are pairwise separated by S_1, Lemma 2.6 in [25] ensures that they can be grouped into two disjoint unions A, A' with

$$\max\{|W \cap A|, |W \cap A'|\} \leq \tfrac{2}{3}|W|.$$

Hence S_1 is a $(\frac{2}{3}, k)$-separator of W, and we are done.

Otherwise, there is a unique heavy component $C_1 \in \mathscr{C}^{(S_1)}$ with $|C_1 \cap W| > \frac{2}{3}|W|$. Set $B_1 := S_1 \cup C_1$ and recurse inside B_1. Proceeding in this way we obtain a chain

$$S_1,\ B_1 = S_1 \cup C_1,\ S_2,\ B_2 = S_2 \cup C_2,\ \ldots,\ S_t,\ B_t = S_t \cup C_t,$$

where at each level $j < t$ there is a unique heavy component C_j with $|C_j \cap W| > \frac{2}{3}|W|$, and $B_j = S_j \cup C_j$. Let $A_t := V \setminus B_t$; then $|A_t \cap W| < \frac{1}{3}|W|$.

If at level t we compute an additional separator $S_{t+1} \subseteq B_t$ (so that the components $D_1, \ldots, D_r$ of $\mathrm{torso}(G, B_t) \setminus S_{t+1}$ satisfy $|W \cap D_j| \leq \frac{2}{3}|W|$), then by Proposition 4.6 these sets are also separated by S_{t+1} in G. Moreover, by Lemma 4.8, S_t separates A_t from B_t in G. Therefore $S := S_t \cup S_{t+1}$ separates the family $\{A_t, D_1, \ldots, D_r\}$ pairwise in G. Applying Lemma 2.6 in [25] to this family, we can group its members into two disjoint unions A, A' with

$$\max\{|W \cap A|, |W \cap A'|\} \leq \tfrac{2}{3}|W|.$$

Thus S (of size at most $2k$) is a $(\frac{2}{3}, 2k)$-separator of W.

If instead B_t is a leaf (so $|B_t| \leq k$ and no S_{t+1} is found), then by Lemma 4.8, S_t separates A_t from B_t in G. Set $S := S_t \cup B_t$. Since $|B_t| \leq k$, we have $|S| \leq 2k$. In $G \setminus S$, the set B_t is entirely removed, so every remaining component is contained in A_t and therefore meets W in at most $|A_t \cap W| < \frac{1}{3}|W|$ vertices. Hence S is a $(\frac{2}{3}, 2k)$-separator of W.

As $W \subseteq V$ was arbitrary, we conclude $\mathrm{sn}(G) \leq 2k$. If the algorithm breaks at some recursive call on $B = S \cup C$, then by construction of `Separator` and k-strong faithfulness the sampled $W \subseteq B$ admits no $(\frac{2}{3}, k)$-separator in G, so $\mathrm{sn}(G) > k$.

Break ⇒ lower bound. Assume the algorithm breaks at some recursive call on a component $B = S \cup C$. By construction of `Separator`, at this call we sampled $W \subseteq B$ and exhaustively checked all $(\frac{2}{3})$-balanced splits $W = A \cup A'$; the break means that

$$\mathrm{rank}(\Sigma_{A,A'}) > k \quad \text{for every such balanced split } A/A' \text{ of } W.$$

Under k-strong faithfulness, the minimal size of an A–A' separator in G equals $\mathrm{rank}(\Sigma_{A,A'})$ (cf. Lemma 3 in [25]); hence for this particular W there is no $(\frac{2}{3}, k)$-separator in G. Since the definition of $\mathrm{sn}(G) = \mathrm{sn}_{2/3}(G)$ requires that *every* $W \subseteq V$ admit a $(\frac{2}{3}, k)$-separator in G, this single counterexample W implies $\mathrm{sn}(G) > k$. □

Under a good run, one may use `test.marginal` and Theorem 4.11 to estimate the separation number as follows. Run the algorithm for $k = 1, 2, \ldots$ with per-run failure budgets

$$\epsilon_k := \frac{6}{\pi^2}\frac{\epsilon}{k^2} \qquad \text{and set} \qquad \delta'_k := \frac{\epsilon_k}{10\, n \log n}$$

in (4.1). Stop at the first k where the algorithm terminates and denote this value by k_0. By Theorem 4.11, the break at $k_0 - 1$ implies $\mathrm{sn}(G) > k_0 - 1$, while the termination at k_0 implies $\mathrm{sn}(G) \leq 2k_0$. Hence

$$\mathrm{sn}(G) \in (\,k_0 - 1,\; 2k_0].$$

Moreover, with probability at least $1 - \epsilon$, the total query complexity up to k_0 satisfies

$$\mathcal{O}\Big(n \log n \sum_{k=1}^{k_0} \max\{\, m(k),\; k\Delta \,\}\Big) \;=\; \mathcal{O}\Big(n \log n \cdot \mathrm{sn}(G) \cdot \max\{\, m(\mathrm{sn}(G)),\; \mathrm{sn}(G)\Delta \,\}\Big),$$

where $m(k)$ is the sample size chosen via (4.1) with $\delta = \frac{7}{30}$ and δ'_k as above (note $m(k)$ is non-decreasing in k).

THEOREM 4.13. With the same assumptions as in Theorem 4.11 and assuming in addition that G is decomposable, `test.marginal` terminates if and only if $\mathrm{sn}(G) \leq k$.

Proof. By Lemma 4.10, at each step $B = C \cup S$ we have that $\Sigma_{B,B}$ is k-strongly faithful to G_B. If $\mathrm{sn}(G) \leq k$ then each step finds a small balanced separator, so the algorithm never breaks. Conversely, if $\mathrm{sn}(G) > k$ then G contains a clique of size $> k$, so some step must break. □

Finally, the computational running time (apart from query complexity) is

$$\mathcal{O}\big(n \log n 2^m \max\{m, k\Delta\}\big),$$

because the algorithm enumerates all balanced partitions of W of size m. With our current sampling rule (4.1) we have $m = \Theta(\log n)$, so the factor 2^m can grow as large as $n^{\Theta(1)}$. Consequently, the overall running time may reach n^c for a constant c well above 25 when k or Δ are on the order of $\log n$, and is therefore impractical in the worst case. A recursion-pruning argument at the end of the proof of Theorem 32 in [25] shows how to drive the sample size down to $m = \mathscr{O}(k \log k)$, removing the heavy n-dependence at the cost of a more intricate analysis. We do *not* implement that trick here, because the present work focuses on query complexity; reducing the running time is left for follow-up work.

4.4 *Conditional descent*

Although the MD is theoretically appealing, G may be such that `test.marginal` has a good run with small probability. In this case, we propose an alternative.

Recall that $M(G)$, defined in (1.1), denotes the set of all covariance matrices Σ such that $(\Sigma^{-1})_{ij} = 0$ if i and j are not connected by an edge in G; in this case, we say that Σ is Markov to G. The next lemma shows that $\Sigma^{(S)}_{C,C}$ is Markov (and, under a mild strengthening, faithful) with respect to G_C.

Lemma 4.14. If S is a separator of G and $C \in \mathscr{C}^{(S)}$, then

$$\Sigma^{(S)}_{C,C} = \big((\Sigma^{-1})_{C,C}\big)^{-1} \in M(G_C).$$

Moreover, if Σ is τ-strongly faithful to G with $\tau = |S| + k$, then $\Sigma^{(S)}_{C,C}$ is k-strongly faithful to G_C.

Proof. The identity $\Sigma^{(S)}_{C,C} = \big((\Sigma^{-1})_{C,C}\big)^{-1}$ is standard for Gaussian graphical models when S separates C from $V \setminus (C \cup S)$; see, e.g., Lemma 29 in [25]. Thus $\Sigma^{(S)}_{C,C} \in M(G_C)$. For strong faithfulness, let $A, A' \subseteq C$ and set $r := \operatorname{rank}(\Sigma^{(S)}_{A,A'})$. By the Guttman rank additivity formula,

$$\operatorname{rank}\big(\Sigma_{A\cup S,\, A'\cup S}\big) = r + |S|. \tag{4.5}$$

If Σ is τ-strongly faithful with $\tau = |S| + k$, then the smallest separator of $A \cup S$ and $A' \cup S$ in G has size $r + |S|$. This separator must contain S, hence is of the form $S \cup S'$ with $|S'| = r$ and $S' \subseteq C$. Any A–A' path in G_C is an A–A' path in G avoiding S, so it must meet S'. Minimality in G implies minimality in G_C. Therefore $\Sigma^{(S)}_{C,C}$ is k-strongly faithful to G_C. □

Remark 4.15. A similar argument shows that one could access G_B by computing $\Sigma^{(V\setminus B)}_{B,B}$. However, computing this matrix exceeds our query budget.

An important difference from the MD is that, in each recursive call, the CD keeps track of all previously found separators. Suppose $S_1, \ldots, S_t$ are the separating sets encountered so far and let

$$\overline{S} = \bigcup_{i=1}^{t} S_i.$$

To test whether S_t separates i, j at the current step, we need to check whether $\Sigma^{(\overline{S})}_{ij} = 0$. Despite this additional bookkeeping, we retain tight control over the query complexity, as shown below.

Algorithm 7 TEST.CONDITIONAL(Σ, k)

Require: an oracle for Σ, target $k \geq 1$

Let V be the index set of the rows/columns of Σ
if $|V| \leq k$ **then**
  stop
else
  run SEPARATOR(Σ, k) to get a balanced separator S with $|S| \leq k$
  run COMPONENTS$(\Sigma^{(S)}, V \setminus S)$ to obtain the components $C_1, \dots, C_\ell$
  for $i = 1, \dots, \ell$ **do**
    run TEST.CONDITIONAL$(\Sigma^{(S)}_{C_i,C_i}, k)$
  end for
end if

To analyze `test.conditional`, note that querying

$$\Sigma^{(S)}_{C,C} = \Sigma_{C,C} - \Sigma_{C,S}\, \Sigma^{-1}_{S,S}\, \Sigma_{S,C}$$

for $S \neq \emptyset$ has an initial cost of $|S|^2$ queries to obtain $\Sigma_{S,S}$ and, for each entry $\Sigma^{(S)}_{ij}$, a cost of $1 + 2|S| = \mathcal{O}(|S|)$ to query Σ_{ij}, $\Sigma_{i,S}$ and $\Sigma_{j,S}$.

REMARK 4.16. The query complexity of `Components`$(\Sigma^{(\bar{S})}_{V,V}, V \setminus \bar{S})$ is $\mathcal{O}(|V|\,\Delta)$ if $\bar{S} = \emptyset$, and $\mathcal{O}\big(|\bar{S}|\,|V|\,\Delta\big)$ otherwise.

LEMMA 4.17. Let $W \subseteq V$ with $|W| = m$, and let $\bar{S} \subseteq V$ be the current conditioning set. Running `Separator`$(\Sigma^{(\bar{S})}_{V,V}, k)$ uses $\mathcal{O}(|V|\,m)$ queries if $\bar{S} = \emptyset$ and $\mathcal{O}\big(|\bar{S}|\,|V|\,m\big)$ queries otherwise.

Proof. To form the $m \times m$ submatrix $\Sigma^{(\bar{S})}_{W,W}$ we first query $\Sigma_{\bar{S},\bar{S}}$ once at cost $\mathcal{O}(|\bar{S}|^2)$, and then each of the $\mathcal{O}(m^2)$ entries of $\Sigma^{(\bar{S})}$ at cost $\mathcal{O}(1 + 2|\bar{S}|) = \mathcal{O}(|\bar{S}|)$ per entry, for a total of $\mathcal{O}(|\bar{S}|\,m^2)$. When $\bar{S} = \emptyset$, this reduces to $\mathcal{O}(m^2)$.

For the first `ABSeparator` loop, for each $v \in V$ we need the ranks of matrices obtained from $\Sigma^{(\bar{S})}_{W,W}$ by adding the row/column of v against A and A' (balanced splits of W). This requires $\mathcal{O}(m)$ additional entries per v, each at cost $\mathcal{O}(|\bar{S}|)$ for conditioning, for a total of $\mathcal{O}(|V|\,|\bar{S}|\,m)$ queries. When $\bar{S} = \emptyset$, this becomes $\mathcal{O}(|V|\,m)$.

For the second `ABSeparator` loop, at most $\mathcal{O}(k^2)$ extra entries are queried; since $k \leq m \leq |V|$ in our regime, this is dominated by the terms above. Summing this yields $\mathcal{O}\big(|\bar{S}|^2 + |\bar{S}|\,m^2 + |V|\,|\bar{S}|\,m\big)$, and we include the $\mathcal{O}(|V|\,m)$ term to cover the unconditioned case $\bar{S} = \emptyset$. If $m \leq |V|$ (which holds for the m chosen via (4.1) in our applications), then $|\bar{S}|\,m^2 \leq |\bar{S}|\,|V|\,m$, so the bound simplifies to the stated cases. □

Note. The sampling parameters δ, δ' appear only in (4.1) to ensure that a separator found on W lifts to a $(\frac{2}{3}+\delta, k)$-separator on V with probability $\geq 1-\delta'$. They do not affect the query counting in Lemma 4.17 except through the choice of the sample size m.

THEOREM 4.18. Let G be a connected graph and let $\Sigma \in M(G)$ be strongly faithful. Fix $\epsilon \in (0, 1)$ and let m be minimal satisfying (4.1) with $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log(n)}$. Then, if $\mathrm{sn}(G) \leq k$, `test.conditional` never breaks. Moreover, with probability at least $1 - \epsilon$, it runs with total query complexity

$$\mathcal{O}\big(n \log^2 n\, \Delta\big),$$

and if it terminates without breaking, then $\mathrm{sn}(G) \le 10k \log\big(\frac{n}{k}\big)$.

Proof. Query complexity. At level 1 (with $|V| = n > k$ and $\overline{S} = \emptyset$), Lemma 4.17 gives a cost $\mathcal{O}(nm)$ for `Separator`(Σ, k). If this call does not break, we obtain a $(\frac{2}{3}, k)$-separator S_1 of W and then run `Components`$(\Sigma^{(S_1)}, V \setminus S_1)$, which by Remark 4.16 costs $\mathcal{O}(kn\Delta)$. Hence level 1 costs

$$\mathcal{O}\big(n(m + k\Delta)\big).$$

For level $\ell \ge 2$, write $\overline{S} = \overline{S}_{\mathrm{old}} \cup S_\ell$ with $\overline{S}_{\mathrm{old}} = S_1 \cup \cdots \cup S_{\ell-1}$ and $|S_\ell| \le k$, so $|\overline{S}| = \mathcal{O}(k\ell)$. Consider all current components C of $G \setminus \overline{S}$. For each such C, the call `Separator`$(\Sigma^{(\overline{S})}_{C,C}, k)$ has the following query costs:

- *Incremental conditioning block (one-time per level).* Passing from $\overline{S}_{\mathrm{old}}$ to $\overline{S}$ requires querying $\Sigma_{S_\ell, \overline{S}_{\mathrm{old}}}$ and Σ_{S_ℓ, S_ℓ}, for a total of $\mathcal{O}\big(k\, |\overline{S}_{\mathrm{old}}| + k^2\big) = \mathcal{O}\big(k\, |\overline{S}|\big)$ additional entries at level ℓ. This is reused for all components at the level.
- *Sampled submatrix within a component.* For $W \subseteq C$ with $|W| = m$,

  $$\Sigma^{(\overline{S})}_{W,W} \;=\; \Sigma_{W,W} \;-\; \Sigma_{W,\overline{S}}\, \Sigma^{-1}_{\overline{S},\overline{S}}\, \Sigma_{\overline{S},W},$$

  which costs $\mathcal{O}(m^2)$ for $\Sigma_{W,W}$ plus $\mathcal{O}(m|\overline{S}|)$ for $\Sigma_{W,\overline{S}}$, hence $\mathcal{O}(m^2 + m|\overline{S}|)$.
- *Screening vertices in `ABSeparator`.* For each $v \in C$, we need

  $$\Sigma^{(\overline{S})}_{v,W} \;=\; \Sigma_{v,W} - \Sigma_{v,\overline{S}}\, \Sigma^{-1}_{\overline{S},\overline{S}}\, \Sigma_{\overline{S},W},$$

  at cost $\mathcal{O}(m + |\overline{S}|)$ queries; summed over $v \in C$ this is $\mathcal{O}\big(|C|\,(m + |\overline{S}|)\big)$.

  Therefore, the *per-component* cost is

  $$\mathcal{O}\big(m^2 + m|\overline{S}| + |C|\,(m + |\overline{S}|)\big).$$

Using $m \le |C|$, we have $m^2 \le |C|\,m$ and $m|\overline{S}| \le |C|\,|\overline{S}|$, so the per-component cost simplifies to $\mathcal{O}\big(|C|\,(m + |\overline{S}|)\big)$. Summing over all components at level ℓ and using $\sum_C |C| = n - |\overline{S}| = \mathcal{O}(n)$, the `Separator` cost across the level is

$$\sum_C \mathcal{O}\big(|C|\,(m + |\overline{S}|)\big) = \mathcal{O}\big(n\,(m + |\overline{S}|)\big).$$

Adding the one-time incremental conditioning cost $\mathcal{O}(k\,|\overline{S}|)$ (shared across all components) and noting that $n \ge k$ and $\Delta \ge 1$, we may absorb $\mathcal{O}(k|\overline{S}|)$ into $\mathcal{O}(n|\overline{S}|\Delta)$ below. By Remark 4.16, the aggregate

`Components` cost across level ℓ is

$$\sum_C \mathscr{O}\big(|\overline{S}|\,|C|\,\Delta\big) = \mathscr{O}\big(|\overline{S}|\,n\,\Delta\big).$$

Thus the total cost of level ℓ is

$$\mathscr{O}\big(n\,m + n\,|\overline{S}|\,\Delta\big).$$

Let $\mathscr{E}_\ell$ be the event that every component at level ℓ has size at most $(\frac{2}{3}+\delta)^\ell n = (\frac{9}{10})^\ell n$. Define

$$\ell^* \;:=\; \left\lceil \frac{\log\left(\frac{n}{k}\right)}{\log\left(\frac{10}{9}\right)} \right\rceil \;\le\; 10\log\left(\frac{n}{k}\right). \tag{4.6}$$

On $\mathscr{E}_{\ell^*}$, all components at level ℓ^* have size $\le k$, and the recursion stops after at most one more level. Summing the level costs yields

$$\mathscr{O}\big(n\,m\,\ell^* + n\,\Delta\,\textstyle\sum_{\ell=1}^{\ell^*}|\overline{S}_\ell|\big) \;=\; \mathscr{O}\big(n\,m\,\ell^* + n\,\Delta\,k\,(\ell^*)^2\big),$$

since $|\overline{S}_\ell| = \mathscr{O}(k\ell)$ (each separator has size $\le k$). Because m is the minimal value satisfying (4.1) with $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log n}$, we have $m = \Theta\big(\log n\big)$ for fixed k, ϵ; also $\ell^* = \Theta\big(\log n\big)$ for fixed k. Hence

$$\mathscr{O}\big(n\,m\,\ell^* + n\,\Delta\,k\,(\ell^*)^2\big) = \mathscr{O}\big(n\,\log^2 n + n\,\Delta\,\log^2 n\big) = \mathscr{O}\big(n\,\Delta\,\log^2 n\big).$$

Probability of $\mathscr{E}_{\ell^*}$. By Theorem 4.4 with $\delta = \frac{7}{30}$ and $\delta' = \frac{\epsilon}{10n\log(n)}$, each call to `Separator` returns, with probability at least $1-\delta'$, a $(\frac{9}{10}, k)$-separator for the entire current component (not just for the sample W). There are at most $n\ell^*$ such calls in total, so by the union bound

$$\Pr\big(\neg\mathscr{E}_{\ell^*}\big) \;\le\; n\ell^*\,\delta' \;\le\; \epsilon.$$

Correctness. Assume $\mathrm{sn}(G) \le k$. Fix any recursion level and current component $C \subseteq V \setminus \overline{S}$, and let $W \subseteq C$ be the sample used by `Separator`. By $\mathrm{sn}(G) \le k$, there exist a balanced split $W = A \cup A'$ and a set $S' \subseteq V$ with $|S'| \le k$ that separates A and A' in G. Any A–A' path contained in G_C is also a path in G, hence meets S'; since the path lies in C, it meets $S' \cap C$. Thus $S' \cap C$ separates A and A' in G_C. Because $\Sigma \in M(G)$, Lemma 4.14 yields $\Sigma^{(\overline{S})}_{C,C} \in M(G_C)$, so

$$X_A \perp\!\!\!\perp X_{A'} \mid X_{S'\cap C} \quad \text{under } \Sigma^{(\overline{S})}_{C,C} \quad \Rightarrow \quad \mathrm{rank}\big(\Sigma^{(\overline{S})}_{A,A'}\big) \le |S' \cap C| \le k.$$

Hence the break condition ('rank $> k$ for all balanced splits') cannot occur at this call, and since the call was arbitrary, `test.conditional` never breaks when $\mathrm{sn}(G) \le k$.

Suppose now that the algorithm terminates without breaking. Fix an arbitrary $W \subseteq V$ and define $C_0 := V$. For $\ell \geq 1$, the algorithm selects $S_\ell \subseteq C_{\ell-1}$ with $|S_\ell| \leq k$ and decomposes $C_{\ell-1} \setminus S_\ell$ into connected components $\{C_1^{(\ell)}, \ldots, C_{q_\ell}^{(\ell)}\}$. Let

$$H_\ell(W) \;=\; \text{the unique } C_j^{(\ell)} \text{ with } |W \cap C_j^{(\ell)}| > \tfrac{2}{3}|W| \;\; \text{(if it exists; otherwise } \emptyset\text{)},$$

and set $C_\ell := H_\ell(W)$ whenever $H_\ell(W) \neq \emptyset$; otherwise, stop. Let $t = t(W)$ be the first level at which $H_t(W) = \emptyset$, or the chosen component satisfies $|C_t| \leq k$ (leaf). Write $\overline{S}_\ell := \bigcup_{i=1}^{\ell} S_i$ and $A_{\ell-1} := V \setminus C_{\ell-1}$.

Lifting observation (CD). For every $\ell \geq 1$, the set $\overline{S}_\ell$ separates in G the family

$$\{A_{\ell-1}\} \;\cup\; \big\{D : \; D \text{ is a connected component of } C_{\ell-1} \setminus S_\ell\big\}$$

pairwise. Indeed, any path that leaves $C_{\ell-1}$ must hit $\overline{S}_{\ell-1}$ (by definition of current components), while any path that stays inside $C_{\ell-1}$ must hit S_ℓ.

We analyze the terminal level t.

Case 1: $H_t(W) = \emptyset$. Then every component D of $C_{t-1} \setminus S_t$ satisfies $|W \cap D| \leq \frac{2}{3}|W|$. Moreover, since C_{t-1} was heavy at level $t-1$, we have $|W \cap A_{t-1}| < \frac{1}{3}|W|$. By the lifting observation with $\ell = t$, the family $\{A_{t-1}\} \cup \{D\}$ is pairwise separated by $\overline{S}_t$, and each member has W-mass at most $\frac{2}{3}|W|$. Applying Lemma 2.6 in [25], these pieces can be grouped into two unions A, A' with $\max\{|W \cap A|, |W \cap A'|\} \leq \frac{2}{3}|W|$. Thus $\overline{S}_t$ (of size $\leq kt$) is a $(\frac{2}{3}, kt)$-separator of W.

Case 2: $|C_t| \leq k$ (leaf). If $|W \cap C_t| \leq \frac{2}{3}|W|$, the argument of Case 1 with the family $\{A_{t-1}, C_t\}$ again shows that $\overline{S}_t$ is a $(\frac{2}{3}, kt)$-separator of W. Otherwise, $|W \cap C_t| > \frac{2}{3}|W|$, which forces $|W| \leq \frac{3}{2}k$ (since $|W \cap C_t| \leq |C_t| \leq k$). In this regime, choose any subset $S \subseteq W$ of size $\lceil \frac{2}{3}|W| \rceil \leq k$. Then in $G \setminus S$ every component contains at most $|W \setminus S| \leq \frac{1}{3}|W| \leq \frac{2}{3}|W|$ vertices of W, so S is a $(\frac{2}{3}, k)$-separator of W.

Since $W \subseteq V$ was arbitrary, we obtain that for every W there exists a $(\frac{2}{3}, K)$-separator with $K \leq kt$ in Case 1 and $K \leq k$ in Case 2. Using the depth bound

$$t \;\leq\; \left\lceil \frac{\log(n/k)}{\log(10/9)} \right\rceil \;\leq\; 10 \log\left(\frac{n}{k}\right),$$

we conclude $\mathrm{sn}(G) \leq 10k \log\left(\frac{n}{k}\right)$. □

5. Non-Gaussian case

Although our paper focuses on the Gaussian setting, several parts extend beyond it. First, zeros in the inverse covariance retain a clear statistical meaning even when X is not Gaussian. Let ρ^{ij} denotes the *partial correlation* between X_i and X_j given $X_{V \setminus \{i,j\}}$. Then (see Section 5.1.3 in [22])

$$\rho^{ij} \;=\; -\frac{(\Sigma^{-1})_{ij}}{\sqrt{(\Sigma^{-1})_{ii}(\Sigma^{-1})_{jj}}} \qquad \Longleftrightarrow \qquad (\Sigma^{-1})_{ij} = 0 \iff \rho^{ij} = 0.$$

Thus, sparsity of the precision matrix is *exactly* equivalent to vanishing partial correlations, independent of any distributional assumptions.

Elliptical and mean-independence settings. In several important non-Gaussian families, linear conditional means hold; see, e.g., [33]. In such cases $(\Sigma^{-1})_{ij} = 0$ implies that X_j has zero coefficient in the linear conditional mean of X_i given $X_{V\setminus\{i\}}$, hence

$$\mathbb{E}(X_i | X_{V\setminus\{i\}}) \;=\; \mathbb{E}(X_i | X_{V\setminus\{i,j\}}).$$

This 'mean independence' is weaker than $X_i \perp\!\!\!\perp X_j \,|\, X_{V\setminus\{i,j\}}$ in general, but it is sufficient for our procedures that rely only on *linear* separations. For some discussion of mean independence and its relevance see, for example, [28].

Gaussian copula/non-paranormal models. Consider the non-paranormal (Gaussian–copula) family $X = (f_1(Z_1), \ldots, f_n(Z_n))$ with strictly monotone f_i and $Z \sim N(0, R)$ [23, 24]. The conditional-independence graph of X coincides with that of Z, i.e., $(R^{-1})_{ij} = 0$ if and only if $X_i \perp\!\!\!\perp X_j \,|\, X_{V\setminus\{i,j\}}$. Moreover, rank correlations recover the latent Gaussian correlation R without estimating the f_i:

$$\tau(X_i, X_j) \;=\; \mathbb{E}\Big[\mathrm{sign}(X_i - X_i')\ \mathrm{sign}(X_j - X_j')\Big], \qquad R_{ij} \;=\; \sin\Big(\tfrac{\pi}{2}\,\tau(X_i, X_j)\Big).$$

(see, e.g., [20, 24]). Thus, any τ-oracle provides an R-oracle, which can be used in our presented algorithms.

Arbitrary distributions in the tree case. For testing whether G is a tree, our approach extends to *any* distribution admitting a simple conditional-independence oracle, provided a 1-faithfulness assumption holds:

(i) $X_i \perp\!\!\!\perp X_j$ if and only if i and j lie in different connected components of G.

(ii) $X_i \perp\!\!\!\perp X_j \,|\, X_k$ if and only if i and j lie in different components of $G \setminus \{k\}$.

Assuming G is connected, we then use the same recursive scheme as in the Gaussian case, replacing covariance queries by conditional-independence tests. All query-complexity bounds translate verbatim, with 'one covariance entry' replaced by 'one conditional-independence query'.

Models with Gaussian-algebraic structure. Finally, in some non-Gaussian tree models (e.g., binary Ising trees and, more generally, *linear tree models*) the covariance/precision algebra mirrors the Gaussian case; our tree-testing algorithm then applies *as is*; see Section 2.3 in [42] for details.

Acknowledgements

We would like to thank Vida Dujmović for helpful remarks.

Funding

NSERC (RGPIN-2024-04164 to L.D.) Ayudas Fundación BBVA a Proyectos de Investigación Científica 2021 and the Spanish Ministry of Economy and Competitiveness (PID2022-138268NB-I00 to G.L. and P.Z.), MCIN/AEI/10.13039/501100011033, FSE+MTM2015-67304-P and FEDER, EU. NSERC (RGPIN-2023-03481 to P.Z.). P.Z. is a Serra Húnter Senior Professor financed by the Government of Catalonia.

Data availability statement

No new data were generated or analysed in support of this review.

References

1. Abu-Ata, M. & Dragan, F. F. (2016) Metric tree-like structures in real-world networks: an empirical study. *Networks*, **67**, 49–68.
2. Adcock, A. B., Sullivan, B. D. & Mahoney, M. W. (2013) Tree-like structure in large social and information networks. *2013 IEEE 13th International Conference on Data Mining*. Dallas, TX: IEEE, pp. 1–10.
3. Chan, T. E., Stumpf, M. P. H. & Babtie, A. C. (2017) Gene regulatory network inference from single-cell data using multivariate information measures. *Cell Systems*, **5**, 251–267. https://doi.org/10.1016/j.cels.2017.08.014.
4. Chandrasekaran, V., Srebro, N. & Harsha, P. (2008) Complexity of inference in graphical models. *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI 2008)*. Helsinki, Finland: AUAI Press, pp. 70–78.
5. Chung, M. K, Villalta-Gil, V., Lee, H., Rathouz, P. J., Lahey, B. B. & Zald, D. H. (2017) Exact topological inference for paired brain networks via persistent homology. *Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings 25*, pages 299–310. Springer.
6. Dirac, G. A. (1961) On rigid circuit graphs. *Abh. Math. Sem. Univ. Hamburg*, **25**, 71–76.
7. Drton, M. & Perlman, M. D. (2007) Multiple testing and error control in Gaussian graphical model selection. *Statist. Sci.*, **22**, 430–449.
8. Dvořák, Z. & Norin, S. (2019) Treewidth of graphs with balanced separations. *J. Comb. Theory B*, **137**, 137–144.
9. Feige, U. & Mahdian, M. (2006) Finding small balanced separators. In Jon M. Kleinberg, editors, *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*. Seattle, WA, USA: ACM, pp. 375–384.
10. Flum, J. & Grohe, M. (2006) Parameterized Complexity Theory. *Texts in Theoretical Computer Science. An EATCS Series*. Berlin, Heidelberg: Springer. Editorial Board of the EATCS Series: G. Rozenberg, A. Salomaa & J. D. Ullman. https://doi.org/10.1007/3-540-29953-X.
11. Frydenberg, M. (1990) Marginalization and collapsibility in graphical interaction models. *Ann. Statist.*, **18**, 790–805.
12. Goldreich, O. (2017) *Introduction to Property Testing*. Cambridge, United Kingdom: Cambridge University Press.
13. Harary, F. (1969) *Graph Theory*. Reading, MA: Addison-Wesley.
14. Huang, S., Li, J., Sun, L., Ye, J., Fleisher, A., Wu, T., Chen, K., Reiman, E., Initiative, A. D. N., et al. (2010) Learning brain connectivity of Alzheimer's disease by sparse inverse covariance estimation. *NeuroImage*, **50**, 935–949.
15. Hwang, B., Lee, J. H. & Bang, D. (2018) Single-cell RNA sequencing technologies and bioinformatics pipelines. *Exp. Mol. Med.*, **50**, 1–14. https://doi.org/10.1038/s12276-018-0071-8.
16. Jordan, C. (1869) Sur les assemblages de lignes. *J. Reine Angew. Math.*, **70**, 185–190.
17. Karger, D. & Srebro, N. (2001) Learning Markov networks: maximum bounded tree-width graphs. *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*. Washington, DC, USA: Society for Industrial and Applied Mathematics, pp. 392–401.
18. Khetan, A. & Oh, S. (2019) Spectrum estimation from a few entries. *J. Mach. Learn. Res.*, **20**, 718–772.
19. Klaassen, S., Kueck, J., Spindler, M. & Chernozhukov, V. (2023) Uniform inference in high-dimensional Gaussian graphical models. *Biometrika*, **110**, 51–68.
20. Kruskal, W. H. (1958) Ordinal measures of association. *J. Amer. Statist. Assoc.*, **53**, 814–861.
21. Kwisthout, J., Bodlaender, H. L. & van der Gaag, L. C. (2010) The necessity of bounded treewidth for efficient inference in Bayesian networks. *ECAI*, **215**, 237–242.
22. Lauritzen, S. L. (1996) *Graphical Models*. Oxford, UK: Clarendon Press.
23. Liu, H., Lafferty, J. & Wasserman, L. (2009) The nonparanormal: semiparametric estimation of high-dimensional undirected graphs. *J. Mach. Learn. Res.*, **10**, 2295–2328.

24. Liu, H., Han, F., Yuan, M., Lafferty, J. & Wasserman, L. (2012) High-dimensional semiparametric Gaussian copula graphical models: the nonparanormal SKEPTIC. *Ann. Statist.*, **40**, 2293–2326. https://doi.org/10.1214/12-AOS1037.
25. Lugosi, G., Truszkowski, J., Velona, V. & Zwiernik, P. (2021) Learning partial correlation graphs and graphical models by covariance queries. *J. Mach. Learn. Res.*, **22**, 1–41.
26. Maniu, S., Senellart, P. & Jog, S. (2019) An experimental study of the treewidth of real-world graph data. In Barceló P and Calautti M, editors, *22nd International Conference on Database Theory (ICDT 2019), Lisbon, Portugal, volume 127 of Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:18. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPIcs.ICDT.2019.12.
27. Marx, D., O'Sullivan, B., & Razgon, I. (2010) Treewidth reduction for constrained separation and bipartization problems. *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010*. Nancy, France: Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, pp. 561–572.
28. Mesters, G. & Zwiernik, P. (2024) Non-independent component analysis. *Ann. Statist.*, **52**, 2506–2528.
29. Neykov, M. & Liu, H. (2019) Property testing in high-dimensional ising models. *Ann. Statist.*, **47**, 2472–2503.
30. Neykov, M., Lu, J. & Liu, H. (2019) Combinatorial inference for graphical models. *Ann. Statist.*, **47**, 795–827.
31. Robertson, N. & Seymour, P. D. (1986) Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, **7**, 309–322.
32. Ron, D. (2001) Property testing. *J. Comb. Optim.*, **9**, 597–649.
33. Rossell, D. & Zwiernik, P. (2021) Dependence in elliptical partial correlation graphs. *Electron. J. Stat.*, **15**, 4236–4263.
34. Serfling, R. J. (1974) Probability inequalities for the sum in sampling without replacement. *Ann. Statist.*, **2**, 39–48.
35. Sullivant, S., Talaska, K. & Draisma, J. (2010) Trek separation for Gaussian graphical models. *Ann. Statist.*, **38**, 1665–1685.
36. Sun, W., Lu, J. & Liu, H. (2018) Sketching method for large-scale combinatorial inference. *Adv. Neural Inf. Process. Syst.*, **31**, 1115–1124.
37. Tan, K. M., Lu, J., Zhang, T. & Liu, H. (2020) Inferring the connectivity and maximum degree of time-varying Gaussian graphical models. *Biometrics*, **76**, 1186–1197.
38. Uhler, C., Raskutti, G., Bühlmann, P. & Yu, B. (2013) Geometry of the faithfulness assumption in causal inference. *Ann. Statist.*, **41**, 436–463.
39. Verzelen, N. & Villers, F. (2009) Tests for Gaussian graphical models. *Comput. Statist. Data Anal.*, **53**, 1894–1905.
40. Wainwright, M. J., Jordan, M. I. (2008) Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, **1**, 1–305.
41. Zhang, X., Zhao, X.-M., He, K., Lu, L., Cao, Y., Liu, J., Hao, J.-K., Liu, Z.-P. & Chen, L. (2011) Inferring gene regulatory networks from gene expression data by path consistency algorithm based on conditional mutual information. *Bioinformatics*, **28**, 98–104.
42. Zwiernik, P. (2018) Latent tree models. In Drton, M., Maathuis, M., Lauritzen, S. & Wainwright, M., editors, *Handbook of Graphical Models*. Boca Raton, FL: CRC Press, pp. 283–306.