université
PARIS-SACLAY

Network inference from incomplete abundance data

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 574, mathématiques Hadamard (EDMH)
Spécialité de doctorat : Mathématiques appliquées
Unité de recherche : Université Paris-Saclay, AgroParisTech, INRAE, UMR MIA-Paris, 75005, Paris, France.
Référent : Faculté des sciences d'Orsay

Thèse présentée et soutenue à Paris, le 12/11/2020, par

Raphaëlle MOMAL

Composition du jury :

Camille Coron Maître de Conférences, Université Paris-Saclay	Examinatrice
Stéphane Dray Directeur de Recherches, Université Claude Bernard Lyon 1	Examinateur
Florence Forbes Directrice de Recherches, Inria Grenoble Rhône-Alpes	Rapportrice
Otso Ovaskainen Professor, University of Helsinky	Rapporteur
Viet-Chi Tran Professeur des Universités, Université Gustave-Eiffel	Examinateur
Stéphane Robin Directeur de recherches, Université Paris-Saclay	Directeur
Christophe Ambroise Professeur des Universités, Université Paris-Saclay	Codirecteur

Thèse de doctorat

NNT : 2020UPASM017

Remerciements

Merci Stéphane et Christophe pour ces trois dernières années (et un peu plus), durant lesquelles vous avez su accueillir mes doutes et mes erreurs avec patience et grand renforts de sourires. Ça a été un grand honneur de travailler avec vous deux, j'ai énormément appris et j'espère qu'on aura l'occasion de faire durer le plaisir encore un peu.

Je remercie Florence Forbes et Otso Ovaskainen d'avoir rapporté cette thèse. Je remercie également Camille Coron, Chi Tran et Stéphane Dray d'avoir accepté de faire partie du jury. Merci aussi à Matthieu Authier et Mahendra Mariadassou pour des discussions intéressantes et enrichissantes.

Je n'aurais jamais cru faire une thèse en mathématiques un jour. Enfin si, quand François Coquet m'a amené doucement à l'idée. Merci François de m'avoir poussée à ne pas me contenter de ce que j'avais sous les yeux. J'ai vraiment pris ma décision lors de mon premier contact avec la recherche. Merci à Andrea Rau d'avoir fait de cette expérience une révélation.

On m'avait prévenu que j'arrivais dans un labo un peu spécial, un labo pas comme les autres. L'Agro c'est comme une grande bourrasque d'été. Ça réchauffe, ça fait rire, ça décoiffe. Un lieu chaleureux et dynamique, peuplé de gens bourrés de talents et d'humour. Je remercie du fond du coeur toute cette grande famille, en commençant par les parents : Liliane, Éric, Céline, Julien, Marie-Laure, Sophie, Isabelle, Laure, Julie, Sarah, Pierre B, Tristan, Maud, Jade, Joon, Erica, Christelle, Christophe D, Marie-Pierre. Un énorme merci aux générations précédentes ~~d'enfants~~ de doctorants que j'ai eu la chance de croiser : Anna, Marie C, Paul, Yann, Mathieu, Thimothée et Rana.

Annarosa, Martina, merci les filles !! Merci d'avoir rendu ce dernier bureau drôle et studieux juste ce qu'il faut, je n'aurais voulu partager les (quelques) craquages de dernière année avec personne d'autre. Ne changez rien vous êtes parfaites. Merci Claire pour tous tes passages (trop courts), pour les orangettes, et ces magnifiques

(c'est le mot !) duos entre flûtistes. Je te souhaite le meilleur pour tes prochaines aventures. Antoine, merci de diffuser la bonne parole de R tidy, grâce à toi nous sommes de plus en plus nombreux dans la paroisse. Merci Félix pour ces moments de calme autour d'un bon thé, et ces week-ends en Bourgogne dont on se souviendra longtemps (qui a dit qu'une chaudière était nécessaire ? Une cheminée suffit !). Merci Saint-Clair de nous rappeler qu'on peut avoir vécu mille vies avant de se lancer dans une thèse ; amuse-toi bien dans la suite de cette aventure. Je remercie aussi les nouveaux que je n'ai pas eu le temps de mieux connaître : Wencan, Gaspard, Tâm et Marina. À vous la suite !
Gabriel (j'ai failli te mettre dans les doctorants, je sais que ça te fait plaisir), merci pour toutes ces histoires, pour tes imitations d'hamsters et de poires astringentes, pour ces moments d'éclats sur isc, et pour ton expertise dans tous les domaines de la vie qu'en tant que petit doctorant on ne comprends pas toujours très bien. Pierre G merci pour tes éclats de voix (notamment de Julien Clerc), pour ta ~~patience~~ tolérance à puyo puyo et pour nous rappeler qu'il est urgent de rire.

Marie est une personne extraordinaire qui a le pouvoir d'augmenter le niveau de bonheur autour d'elle. Elle a l'effet secondaire d'induire une accoutumance assez sévère. Ayant la chance d'avoir partagé son bureau pendant 2 ans, puis d'avoir récidivé aux cours de théâtre, aux soirées et aux week-ends en Normandie, je suis complètement addict, je l'avoue sans rougir. Mais ce traitement de fond fait de moi un être humain plus heureux. Alors merci Marie, je serais toujours là pour toi.

Un grand merci à Charles (et les chichis), Luc, Nico et Joachim pour vos folies respectives. Marine, après toutes ces années tu continues à me suivre. Merci pour toutes ces aventures, hâte de la prochaine ! Merci aussi aux copains sans lesquelles je ne serais pas là aujourd'hui : Jihane, Syrielle, Florian, Rocket, Clément et Laure.

Je remercie enfin ma famille sur laquelle je sais pouvoir compter. Merci beaucoup à mes frères et soeurs pour vos encouragements, Alain, Hélène, Ludo, les belles-soeurs de choc au top Sylvie et Amélie. Je remercie également toute ma belle-famille pour son grand coeur. Je remercie infiniment mes parents d'être des esprits libres, indépendants et originaux, constantes sources d'inspiration pour moi. Merci maman pour ta spontanétité, ta joie et ton air marin. Merci papa pour ta sérénité, ta poésie et tes couleurs.

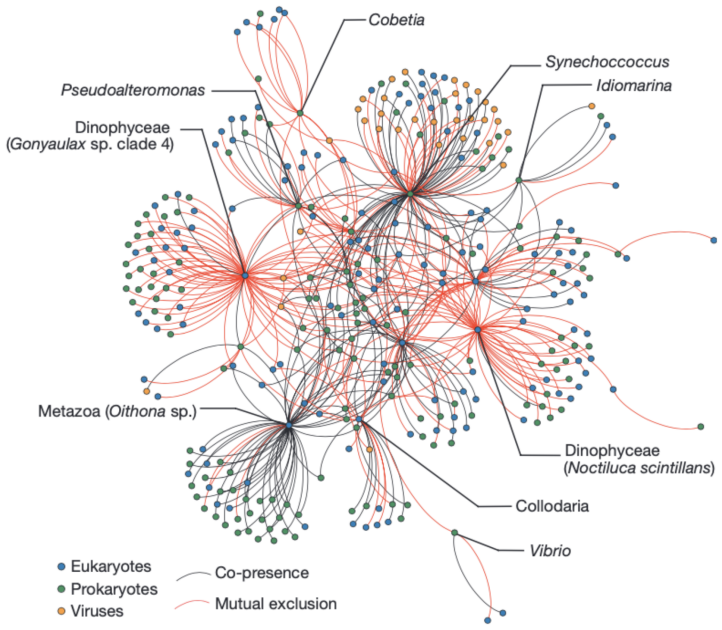
Mes derniers mots vont à Sylvain. Merci d'avoir eu la brillante idée de participer à un certain séminaire, là-haut sur une certaine montagne. Merci de m'honorer chaque jour de ta présence, de ton soutien infaillible et de ton petit air moqueur sans lequel le monde n'aurait plus aucun sens.

Contents

Introduction

Context

Networks are objects representing relationships between entities. They are useful to comprehend systems joint organization and behaviors, leading to discoveries that would not have been possible by analyzing the entities separately. Applications are numerous in life sciences, among which genes regulatory networks (genomics), community assembly mechanisms (community ecology), or pathobiome organization (microbiology). Networks can be built from observed interactions, as is the case of host-parasite, plant-pollinator or trophic networks in ecology, or protein-protein interaction networks in genomics. However this strategy, which we call network reconstruction, limits the identified interactions to observable ones only, where others would be interesting too (e.g. competition for food or space, shelter sharing, etc.).



Integrated plankton community network related to carbon export at 150m (Guidi et al., 2016).

This work is interested in species network inference, which is the art of identifying interactions from observed measures on a set of species. Network inference necessarily relies on a mathematical definition of species interactions, allowing to detect a broad range of interactions. Their biological meaning is unknown and would have

to be identified by experts later on.
A first idea of a mathematical species interaction is the correlation between the species abundances. However using correlation results in dense networks proving hard to analyze, as spurious edges appear between two variables correlated to the same third one [1]. Instead, using conditional dependence relationships between species provides with a clear separation between direct and indirect effects, and therefore yields sparse and easy to interpret networks.
Graphical models are the dedicated mathematical framework for the modeling and inference of such networks, for they graphically represent a multivariate random variable conditional dependencies. Gaussian Graphical Models (GGM) in particular present with specific theoretical and algebraic properties which facilitate the inference. GGM have been widely used on continuous data.

However measures on species are often counts, as is usually the case in ecology and in experiments using high-throughput sequencing technologies in genomics and microbiology. A way to go for network inference from count data with a distinction between direct and indirect effects is then to adopt a modeling for the count allowing the use of the GGM framework. To obtain interpretable results, the model should also account for measured experimental offsets and covariates. Furthermore, the observed measures on species are also very likely to be incomplete as it is difficult to know in advance all the factors governing a phenomenon. This causes the species interaction network to present spurious edges between the species which should be linked to the unobserved actor (species or covariate). A partial observation of the data thus provides with a marginal network instead of the complete one, leading to biased further interpretations and analyzes.

Objectives and outline

The aim of the present work is to develop a methodology for the network inference from incomplete abundance data. This task was divided into two sub-objectives. First, develop a method for network inference from abundance data. To this aim we model counts using the Poisson log-normal distribution, taking advantage of the estimation procedure developed by Chiquet et al. (2018). This specific distribution includes a latent layer of Gaussian parameters, within which the inference of the species interaction network is performed. Following Meilă and Jaakkola (2006) and Schwaller and Robin (2017), the inference is carried out using tree averaging, allowing for a complete and efficient exploration of the space of spanning tree graphs,

1. e.g. the number of covid 19 cases detected correlates with both the real number of cases and the number of tests done on the population, which induces a spurious correlation between the two latter where obviously there is no direct effect of one on the other.

and yielding edges probabilities.

Then, this work includes missing actors in the model to account for incomplete data. There we model the missing actors as additional latent variables of the model Gaussian latent layer. The Gaussian graphical model parameters maximum likelihood estimators detailed in Lauritzen (1996) are adapted to the specific case of spanning tree structures, and applied within a variational Expectation Maximization algorithm.

Chapter 1

The first chapter covers in details the mathematical and technical background used in Chapters 2 and 3. It defines the general framework of graphical models and its link with conditional independence relationships. The particular properties of the Gaussian graphical models are then presented, along with its maximum likelihood estimators. Two network inference methods are detailed: the penalized regularization which estimates the precision matrix in a sparse manner, and tree averaging which efficiently explores the super-exponential space of spanning-tree structures. This chapter then draws a state of the art of the strategies for the modelization of multivariate counts.

Chapter 2

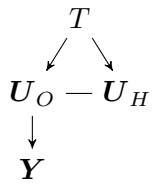
Chapter 2 details the proposed methodology for the inference of species interaction networks from measures of joint abundances. Counts are modeled in a hierarchical manner: a spanning-tree graph T is first drawn, then parameters $\boldsymbol{Z}$ are modeled conditionally on T as a multivariate Gaussian faithful to T, and finally counts $\boldsymbol{Y}$ follow a Poisson log-normal distribution with parameters $\boldsymbol{Z}$. This model thus involves two latent layers of parameters: $\boldsymbol{Z}$ and T, and can be described by the following graph:

$$T \rightarrow \boldsymbol{Z} \rightarrow \boldsymbol{Y}$$

The model inference estimates the tree distribution using an EM algorithm, which had not been done before. The proposed methodology is implemented in the R package EMtree (`github.com/Rmomal/EMtree`). It is compared to state-of-the-art approaches and applied to two empirical datasets from ecology and microbiology. This chapter has been published in the journal *Methods in Ecology and Evolution* (Momal et al., 2020). The presented appendices are extended with a vignette showing usage examples of the EMtree package.

Chapter 3

This chapter presents an extended version of the previous model developed in Chapter 2 to include possible missing actors. The Gaussian latent layer is assumed to involve additional unobserved variables. The Gaussian layer is considered in its normalized form and denoted $\boldsymbol{U}$. This model thus involves three latent layers: T, $\boldsymbol{U}_O$ where "O" stands for "observed", and $\boldsymbol{U}_H$ where "H" stands for "hidden", and can be described by the following graph:



The model is estimated with a variational EM algorithm, which takes advantage of the average on trees to use the adaptation to the context of spanning trees of the maximum likelihood estimators of GGM parameters detailed in Lauritzen (1996). The developed procedure is implemented in the R package nestor (`github.com/Rmomal/nestor`) and illustrated on two empirical datasets from ecology.

This chapter has been submitted for publication in a statistical journal. The submitted supplementary material is enriched with a vignette showing usage examples of nestor, a section presenting different initialization methods and finally a comparison of nestor, EMtree and PLN-network which is another method building on the PLN distribution but uses a penalized approach for the network inference.

Chapter 4

This final chapter introduces some perspectives of this work. After summarizing the specifics of the developed methodology and discussing unresolved issues, natural extensions of the model are presented. They first include extensions about measures on the inferred network, with a method for estimating the latent layer precision matrix, and a strategy to use tree distributions for comparing networks. Then perspectives about the data at hand are discussed, namely how to handle other data types or datasets presenting with spatial dependencies. Finally a model for the network inference not resorting to a latent layer is presented.

Notations

Operations:

$|\cdot|$: matrix determinant

$\odot$: Hadamard product

Matrices:

$\boldsymbol{Y}$: observed counts

$\boldsymbol{Z}$: latent Gaussian parameters

$\boldsymbol{U}$: latent normalized Gaussian parameters

$\boldsymbol{X}$: covariates

O : measured offsets

Dimensions:

n : samples

p : observed species

r : unobserved actors

d : covariates

1

MATHEMATICAL FRAMEWORK

Contents

This first chapter details the technical background extracted from the literature which is used in the next chapters. It is designed to be read independently of the rest of this work, and therefore repetitions with elements from the next chapters are unavoidable. The chapter begins by covering all the required notions from graph theory, with a focus on tree-shaped graphs. A second section presents Gaussian graphical models properties and explains why it is a golden framework for network inference. After a reminder on the Expectation-Maximization algorithm as well as its variational interpretation, the last part is a state of the art of network inference from abundance data and presents methods stemming from both genomics and ecology.

1.1 Graphical Models

A graphical model is classically described as a probabilistic model which conditional dependence structure is given by a graph. This first section gives the general framework of graphical models, and adapts definitions and properties from Lauritzen (1996) to undirected graphs involving only quantitative variables. Then, the specific algebraic properties of spanning tree graphs are presented.

1.1.1 General framework

A graph is defined as a pair $\mathcal{G} = (V, E)$ such that V is a finite set of vertices, and the set of edges E is a subset of $V \times V$. Here we consider E such that vertices are not linked to themselves and there are no multiple edges between two vertices. For any given pair of nodes (k, ℓ), we denote by $k \sim \ell$ an edge and by $k \nsim \ell$ an absence of edge. In the literature, V can be composed of both quantitative and qualitative variables. However here we only use variables of one kind (quantitative) and so V is called pure. The following definitions apply to the particular case of a pure vertex set V and an undirected graph $\mathcal{G}$.

Let's first consider a subset A of the vertex set V. A is said to be *complete* if all the nodes it contains are linked with each other. If A is additionally of maximal size, it is then called a *clique*. This definition makes the expression "maximal clique" a pleonasm. The *subgraph* $\mathcal{G}_A$ defined by A is obtained from $\mathcal{G}$ by keeping edges with both endpoints in A. Furthermore if A, B and S are disjoint subsets of V, S is said to *separate* A from B if any path from $\mathcal{G}_A$ to $\mathcal{G}_B$ intersects with $\mathcal{G}_S$. The following notions of decomposable graphs and perfect sequences are central in the definition of a graphical model.

Definition 1.1 (Proper decomposition)**.** *A triple (A, B, C) of disjoint subsets of the pure vertex set V of an undirected graph $\mathcal{G}$ forms a decomposition of $\mathcal{G}$ if $V = A \cup B \cup C$, and if C satisfies:*

(i) C is a complete subset of V,

(ii) C separates A from B.

If A and B are both non-empty, the decomposition is proper.

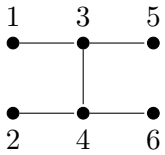


Figure 1.1 – An undirected graph with 6 nodes.

The graph in Figure 1.1 gives an example. Let's define the sets of nodes $A = \{1, 2\}$, $B = \{5, 6\}$ and $C = \{3, 4\}$. As nodes 3 and 4 are linked, C is a complete set. Moreover C separates A from B, and A and B are both non-empty, hence (A, B, C) forms a proper decomposition. A decomposition is thus simply a separation of the set of nodes with complete subsets. A decomposable graph is then defined using Definition 1.1 in a recursive manner:

Definition 1.2 (Decomposable graph)**.** *An undirected graph is decomposable if it is complete, or if there exists a proper decomposition* (A, B, C) *into decomposable subgraphs* $\mathcal{G}_{A\cup B}$ *and* $\mathcal{G}_{B\cup C}$*.*

This definition proves to be very useful in demonstrations of decomposable graphs properties, but not to identify decomposable graphs in practice. The following notion of triangulation will help.

Definition 1.3 (Triangulated graph)**.** *An undirected graph* $\mathcal{G}$ *is said to be triangulated if loops in* $\mathcal{G}$ *connect three nodes at most.*

In other words in a triangulated graph, the loop of maximal size will form a triangle. Triangulated graphs are also called chordal graphs, as it suffices to draw chords in all loops to triangulate a graph. Then, a *sequence* is a numbered list of subsets of the node set. A sequence is called *perfect* under the following conditions on the subsets and the numbering order:

Definition 1.4 (Perfect sequence)**.** *Let* $B_1, ..., B_k$ *be a sequence of subsets of the pure set of vertex* V *of the undirected graph* $\mathcal{G}$*. Let* $H_j = B_1 \cup .. \cup B_j$*, and* $S_j = H_{j-1} \cap B_j$*. The sequence is perfect if it satisfies the following conditions:*

(i) for all $i > 1$ *there is a* $j < i$ *such that* $S_i \subseteq B_j$ *(running intersection property),*

(ii) each S_i *is a complete subset.*

H_j *are called the histories and* S_j *the separators.*

An example where separators are complete but the sequence is not perfect is given in Figure 1.2. A sequence is $B = \big\{B_1 = \{1, 2\}, B_2 = \{3, 4\}, B_3 = \{2, 3, 5\}\big\}$, with the separators $S_2 = \varnothing, S_3 = \{2, 3\}$. As $\{2, 3\}$ is not a subset of any of the B_i with i less than 3, the running intersection property is violated and B is not perfect. Inverting B_2 and B_3 however yields a perfect sequence, which also shows that a simple way of designing a perfect sequence is to ensure that each new set B_j includes its separator with the previous set B_{j-1}.

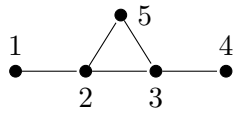


Figure 1.2 – An undirected graph with 5 nodes.

The notion of perfect sequence then allows to define the multiplicity of a separator:

Definition 1.5 (multiplicity of a separator)**.** *The multiplicity $\nu(S)$ of the separator S is an index counting the number of times S occurs in a perfect sequence.*

The graph of Figure 1.3 gives an example. There a perfect sequence of its cliques is $C_1 = \{1, 2, 4\}$, $C_2 = \{2, 3, 4\}$, $C_3 = \{4, 5, 6\}$, $C_4 = \{4, 6, 7\}$, $C_5 = \{4, 8\}$. The corresponding separators are $S_2 = \{2, 4\}$, $S_3 = S_5 = \{4\}$, $S_4 = \{4, 6\}$, which gives the following multiplicities: $\nu(S_2) = 1$, $\nu(S_4) = 1$, $\nu(S_3) = \nu(S_5) = 2$.

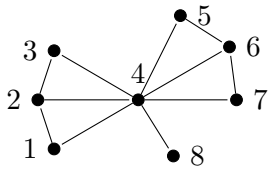


Figure 1.3 – An undirected decomposable graph with 8 nodes.

Proposition 1.1. *The following conditions are equivalent for an undirected graph $\mathcal{G}$:*

(i) $\mathcal{G}$ is decomposable.

(ii) The cliques of $\mathcal{G}$ can form a perfect sequence.

(iii) $\mathcal{G}$ is triangulated.

This property is the most useful in practice, as it states that as soon as a graph does not present with cycles of size 4 or more, it is decomposable on its cliques. This decomposition is essential to the graphical model properties, as we will see in Gaussian graphical models.

1.1.2 Characterization of conditional independence

The notion of conditional independence is central in the theory of graphical models. All that follows focuses on the case of variables associated with positive and continuous densities. Let's consider three random variables X, Y and Z with joint distribution f. The conditional independence of X and Y conditional on Z is linked to the factorization of f. More precisely:

$$X \perp\!\!\!\perp Y \mid Z \iff f(x, y, z) = f(x, z)\, f(y, z)/f(z).$$

Conditional independence can be understood in terms of information. That is, "X is independent of Y, given Z" means that knowing Z, Y will not bring any new information on X, and conversely. The Markov property below is the link between conditional independence of variables and its representation in an undirected graph.

Let's consider a collection of random variables $(X_v)_{v\in V}$ taking values in probability spaces $(\mathcal{X}_v)_{v\in V}$, and $\mathcal{X} = \otimes_{v\in V}\mathcal{X}_v$. We further denote for any subset A of V, $X_A = (X_v)_{v\in A}$, and $\mathcal{G}$ a graph with vertex set V.

Definition 1.6 (Global Markov property)**.** *A probability measure P on $\mathcal{X}$ is global Markov relative to the undirected graph $\mathcal{G}$ if for any triple (A, B, S) of disjoint subsets of V, it holds that:*

$$S \text{ separates } A \text{ from } B \Rightarrow X_A \perp\!\!\!\perp X_B \mid X_S.$$

If additionally $X_A \perp\!\!\!\perp X_B \mid X_S \Rightarrow S$ separates A from B, then P is faithful Markov and $\mathcal{G}$ perfectly describes the conditional dependence structure of P.

Therefore if a distribution is only global Markov, its related graph is possibly too dense: it can contain edges between variables which are actually conditionally independent from one another.

Definition 1.7 (Factorization)**.** *A probability measure P on $\mathcal{X}$ factorizes according to $\mathcal{G}$ if it has density f with respect to measure $\mu = \otimes_{v\in V}\mu_v$, where f can be written as*

$$f(x) = \prod_{c\in\mathcal{C}} \psi_c(x),$$

where $\mathcal{C}$ denotes the set of cliques of $\mathcal{G}$, and for any subset C of the vertex set V, ψ_C is a positive function of X_C only.

The following theorem links definitions 1.6 and 1.7 for positive and continuous distributions:

Theorem 1.1 (Hammersley and Clifford)**.** *For any undirected graph $\mathcal{G}$ and probability distribution P with strictly positive and continuous density f with respect to a product measure μ, it holds that:*

$$P \text{ factorizes according to } \mathcal{G} \iff P \text{ is global Markov relative to } \mathcal{G}.$$

This equivalence means in practice that writing a density in a product form on some combination of its variables helps to identify conditional independence relations.

1.1.3 Spanning trees

Trees are graphs with no cycles, meaning that all cliques are edges only. When a tree connects all the nodes, it is called a spanning tree. This particular type of graph is both the sparsest connected graph, and the most connected graph without loops. This feature makes it a decomposable graph by definition, following condition (*iii*) of Proposition 1.1. The following result is obtained directly.

Proposition 1.2. *If T is a spanning tree, all of its separators are nodes. For any node k of T, $d(k)$ denotes its degree. Then its multiplicity in any perfect sequence is $\nu(k) = d(k) - 1$.*

The study of the spanning tree structure began in the latter part of the 19^{th} century. Then, Arthur Cayley established that the total number of spanning trees of a complete graph with p nodes is p^{p-2}. Kirchhoff generalized this result in a theorem known as the "all-minor" theorem, using the notion of graph Laplacian matrix.

Definition 1.8 (Laplacian matrix)**.** *The Laplacian matrix $\mathbf{Q}$ of a symmetric matrix $\mathbf{W} = [w_{jk}]_{1 \leq j,k \leq p}$ is as follows :*

$$[\mathbf{Q}]_{jk} = \begin{cases} -w_{jk} & 1 \leq j < k \leq p \\ \sum_{u=1}^{p} w_{ju} & 1 \leq j = k \leq p. \end{cases}$$

When $\mathbf{W}$ is a graph adjacency matrix, $\mathbf{Q}$ is called the graph Laplacian matrix and its diagonal is composed of the degrees of each node.

Hereafter $\mathbf{A}^{uv}$ denotes the squared matrix $\mathbf{A}$ deprived from its uth row and vth column, and remind that the (u, v)-minor of $\mathbf{A}$ is the determinant of this deprived matrix, namely $|\mathbf{A}^{uv}|$.

Theorem 1.2 (Kirchhoff's theorem)**.** *For any graph $\mathcal{G}$, let $\mathbf{Q}$ denote its graph Laplacian matrix. Then for any pair of nodes $\{u, v\}$, the total number of spanning trees in $\mathcal{G}$ is equal to $|\mathbf{Q}^{uv}|$.*

The total number of spanning trees in any graph can thus be computed in polynomial time ($\mathcal{O}(p^3)$). For the sake of clarity, we introduce the notation $jk \in T$ which means that nodes j and k are linked in tree T. Theorem 1.2 was extended for weighted graphs in the late 20th century as follows, where $\mathcal{T}$ denotes the spanning tree space on $V = \{1, ..., p\}$:

Theorem 1.3 (Matrix Tree Theorem Chaiken and Kleitman (1978); Meilă and Jaakkola (2006))**.** *For any symmetric weight matrix $\mathbf{W}$ with all entries in $\mathbb{R}^+$, the sum over all spanning trees of the product of the weights of their edges is equal to any minor of its Laplacian $\mathbf{Q}$. That is, for any $1 \leq u, v \leq p$,*

$$W := \sum_{T \in \mathcal{T}} \prod_{jk \in T} w_{jk} = |\mathbf{Q}^{uv}|.$$

Consequently, the operation of summing over all spanning trees can be carried out in a computationally efficient way. Meilă and Jaakkola (2006) built on this result to provide a close form expression for the derivative of the sum-product W with respect to each entry of the input weight matrix $\mathbf{W}$. Without loss of generality, we choose $\mathbf{Q}^{11}$.

Lemma 1.1 (Meilă and Jaakkola (2006))**.** *Define the entries of the symmetric matrix* $\mathbf{M}$ *as*

$$[\mathbf{M}]_{jk} = \begin{cases} \left[(\mathbf{Q}^{11})^{-1}\right]_{jj} + \left[(\mathbf{Q}^{11})^{-1}\right]_{kk} - 2\left[(\mathbf{Q}^{11})^{-1}\right]_{jk} & 1 < j < k \leq p \\ \left[(\mathbf{Q}^{11})^{-1}\right]_{jj} & k = 1, 1 < j \leq p \\ 0 & j = k. \end{cases}$$

it then holds that

$$\partial_{w_{jk}} W = [\mathbf{M}]_{jk} \times W.$$

Theorem 1.3 actually gives the solution to the computation of the sum on trees of a product function on the edges of a spanning tree, which can be used combined with Lemma 1.1 to perform network inference using average on trees, as we will see later on.

1.2 Gaussian Graphical Models

Gaussian graphical models (GGM) describe the conditional dependency structure of a multivariate Gaussian distribution. The different properties of the multivariate Gaussian distribution allow some specific interpretations and estimation results of its related graphical model, making the GGM a sound and very convenient framework to work with.

1.2.1 Specific properties

The multivariate Gaussian distribution possesses various properties which facilitate computations as well as interpretations. Two in particular are of interest in the graphical models setting. The first one concerns the natural writing of its density.

Proposition 1.3. *Let* $X \sim \mathcal{N}(\mu, \mathbf{\Sigma})$ *with precision matrix* $\mathbf{\Omega} = \mathbf{\Sigma}^{-1}$ *and* X_k *denotes the* k^{th} *column of* X*. Then the associated density* f *factorizes as:*

$$f(X) \propto \prod_{k,\ell,\omega_{kl}\neq 0} \exp(-X_k \omega_{k\ell} X_\ell / 2).$$

Proposition 1.4. *The density* f *of a multivariate Gaussian distribution* $\mathcal{N}(\mu, \mathbf{\Omega}^{-1})$ *is global Markov relative to an undirected graph* $\mathcal{G}$*, the edges of which are determined by the non-null entries of the precision matrix* $\mathbf{\Omega}$*.*

Proof. This is a direct application of Theorem 1.1 to the factorized multivariate Gaussian density reminded in Proposition 1.3. □

As f is global Markov, any two non-neighbors in $\mathcal{G}$ are conditionally independent in f. The multivariate Gaussian distribution also presents with a facilitating property

about conditional independence, reminded below.

Proposition 1.5. *If $X \sim \mathcal{N}_V(\mu, \mathbf{\Omega}^{-1})$, then it holds for any $k, \ell \in V$ with $k \neq \ell$ that*

$$X_k \perp\!\!\!\perp X_\ell \mid X_{V\setminus\{k,\ell\}} \iff \omega_{k\ell} = 0.$$

Proof. This comes directly from the fact that the 2×2 covariance matrix of variables k and l conditional on all others expresses in terms of the precision matrix :

$$\mathbf{\Sigma}_{k,l|V\setminus\{k,l\}} = |\mathbf{\Omega}_{\{k,l\}}|^{-1} \begin{pmatrix} \omega_{kk} & -\omega_{kl} \\ -\omega_{kl} & \omega_{ll} \end{pmatrix}.$$

As conditional independence between Gaussian variables is equivalent to 0 terms in the conditional covariance matrix, the above expression shows that it is also equivalent with null precision terms. □

Note that Proposition 1.5 is not a consequence of proposition 1.3.

Proposition 1.6 (Faithfulness property)**.** *Let $X \sim \mathcal{N}(\mu, \mathbf{\Omega}^{-1})$ with associated density f, $\mathcal{G}$ the graph which edges represent the non-null entries of Ω. Then it holds that:*

$$k \nsim \ell \iff X_k \perp\!\!\!\perp X_\ell \mid X_{V\setminus\{k,\ell\}}.$$

and f is faithful Markov relative to $\mathcal{G}$.

Proof. The right implication is the global Markov property of the multivariate Gaussian distribution reminded in Proposition 1.4.
The left implication comes for the combination of Propositions 1.3 and 1.5. From Proposition 1.5:

$$X_k \perp\!\!\!\perp X_\ell \mid X_{V\setminus\{k,\ell\}} \Rightarrow \omega_{k\ell} = 0,$$

and a null precision term implies no edge between the corresponding nodes in $\mathcal{G}$ according to Proposition 1.3. □

Proposition 1.6 is key for estimation in the GGM framework, as it states that finding the null precision entries gives all the graph of conditional dependencies. The Gaussian framework also offers specific conditioning results, and in particular the regression interpretation of $\mathbf{\Omega}$ entries.

Proposition 1.7 (Regression interpretation)**.** *Considering $X \sim \mathcal{N}(\mu, \mathbf{\Omega}^{-1})$, the linear regression of X_j on $X_{\setminus j}$ writes :*

$$X_j = \sum_{k\neq j} \beta_{jk} X_k + \varepsilon_j, \qquad \textit{where} \qquad \varepsilon_j \sim \mathcal{N}(0, \omega_{jj}^{-1}), \qquad \beta_{jk} = -\frac{\omega_{jk}}{\omega_{jj}}.$$

Hence ω_{jk} is, up to a scalar, the coefficient of X_k in the multiple regression of X_j on all other variables. This result is at the basis of GGM inference methods relying

on regression (Meinshausen and Bühlmann, 2006), which we will see later on. But first, let's turn to Lauritzen's parameters estimation for GGM, made possible by the faithfulness property.

1.2.2 Maximum likelihood estimation

In this section we adopt the following notations: for any squared matrix $\mathbf{A}$ of dimension p and B a subset of $V = \{1, ..., p\}$, we let $\mathbf{A}_B$ refer to the block B of $\mathbf{A}$: $\mathbf{A}_B = (a_{ij})_{\{i,j\} \in B}$. $[\mathbf{A}_B]^p$ then denotes the matrix obtained by filling up with zero entries to obtain full dimension $p \times p$, so that:

$$([\mathbf{A}_B]^p)_{ij} = \begin{cases} a_{ij} & \text{if } \{i,j\} \in B, \\ 0 & \text{if } \{i,j\} \in V_{\setminus B}. \end{cases}$$

Hereafter we consider $X \sim \mathcal{N}(\mu, \mathbf{\Omega})$ with dimension p. If X is associated with a decomposable graph $\mathcal{G}$, the Markov property 1.6 applies across the decomposition and reflects in a multiplicative property for the density, and additive property for the concentration matrix. More precisely, considering a perfect sequence of the cliques of $\mathcal{G}$ as in Definition 1.4, we have that

$$f(X) = \frac{\prod_{C \in \mathcal{C}} f(X_C)}{\prod_{S \in \mathcal{S}} f(X_S)^{\nu(S)}}.$$

This general expression of the density across decomposition of the graph is very helpful in likelihood computations, especially when some hypotheses are made on the structure of $\mathcal{G}$. The following lemma gives the expressions for $\mathbf{\Omega}$ and its determinant:

Lemma 1.2. *In a decomposable Gaussian graphical model with precision matrix $\mathbf{\Omega} = \mathbf{\Sigma}^{-1}$ of dimension p:*

$$\begin{cases} \mathbf{\Omega} &= \sum_{C \in \mathcal{C}} \left[(\mathbf{\Sigma}_C)^{-1} \right]^p - \sum_{S \in \mathcal{S}} \nu(S) \left[(\mathbf{\Sigma}_S)^{-1} \right]^p \\ |\mathbf{\Omega}| &= \dfrac{\prod_{C \in \mathcal{C}} |\mathbf{\Sigma}_C|^{-1}}{\prod_{S \in \mathcal{S}} |\mathbf{\Sigma}_S|^{-\nu(S)}} \end{cases}$$

where $\mathcal{C}$ is the set of cliques of $\mathcal{G}$ and $\mathcal{S}$ the set of separators with multiplicities ν in any perfect sequence.

The GGM framework provides exact results about maximum likelihood estimators (MLE) of the mean vector, the precision matrix, its determinant and the terms of the covariance matrix corresponding to edges in the graph. They all depend on the sufficient statistic that is the sum of squared deviations, defined as follows where X^i denotes the i^{th} sample of X:

Definition 1.9. *Denoting $\overline{X}$ the empirical mean of the multivariate Gaussian X,*

the sum of squared deviations matrix, also known as total sum of squares, is given by:

$$SSD = \sum_{i=1}^{n} (X^i - \overline{X})(X^i - \overline{X})^\intercal = X^\intercal X - n\overline{X}\overline{X}^\intercal.$$

The following theorems 1.4 and 1.5 give the MLE estimators in the GGM framework using $SSD = (ssd_{ij})_{ij}$.

Theorem 1.4. *In the Gaussian graphical model, the MLE of the unknown mean and covariance matrix exist with probability one if $n > p$. When they exist, the estimate of the mean is $\widehat{\mu} = \overline{X}$, and the estimate of the unknown covariance matrix $\mathbf{\Sigma}$ is determined as the unique solution of the system of equations*

$$\begin{cases} n\widehat{\sigma}_{jj} = ssd_{jj} & , j \in V \\ n\widehat{\sigma}_{k\ell} = ssd_{k\ell} & , k \sim \ell, (k,\ell) \in V \end{cases}$$

which also satisfies the model restriction $\omega_{k\ell} = 0 \iff k \nsim \ell$.

Theorem 1.5. *In a Gaussian graphical model with graph $\mathcal{G}$, the MLE of the precision matrix exists with probability one if $n > max_{C\in\mathcal{C}}|C|$. It is then given as*

$$\widehat{\mathbf{\Omega}} = n\Big\{ \sum_{C\in\mathcal{C}} \big[(SSD_C)^{-1}\big]^p - \sum_{S\in\mathcal{S}} \nu(S)\big[(SSD_S)^{-1}\big]^p \Big\},$$

where $\mathcal{C}$ is the set of cliques of $\mathcal{G}$ and $\mathcal{S}$ the set of separators with multiplicities ν in any perfect sequence. The determinant of the estimate can be calculated as

$$|\widehat{\mathbf{\Omega}}| = n^p \frac{\prod_{C\in\mathcal{C}} |(SSD_C)^{-1}|}{\prod_{S\in\mathcal{S}} (|(SSD_S)^{-1}|)^{\nu(S)}}.$$

The stronger condition $n > p$ of Theorem 1.4 is only sufficient, whereas the condition $n > max_{C\in\mathcal{C}}|C|$ of Theorem 1.5 is necessary for the existence of all estimators, as it ensures the positive definiteness of all SSD_C (otherwise some might not have full rank).
All of the above maximum likelihood estimators require the precise knowledge of the graph structure and is therefore rarely available. However relying on simple structures (e.g. spanning trees) greatly simplifies the expressions and make their use possible.

1.2.3 Network inference from Gaussian data

Network inference here refers to the inference of the conditional dependence structure of multiple variables jointly observed in a series of sites. The interactions between the species are unknown, therefore this is not to be confused with the art

of inferring a network parameters (e.g. sign and strength of the interactions) from observations of repeated interactions.
In the special framework of GGM, conditional dependence relations are perfectly represented by the non-nul entries of the precision matrix $\mathbf{\Omega}$, as specified in Proposition 1.6. Therefore the general approach to network inference in GGM is then to perform a sparse estimation of $\mathbf{\Omega}$.

Penalized estimation

In a standard linear regression model with data matrix $\mathbf{Y}$ and covariates matrix $\mathbf{X}$, penalized estimation can be used to select predictors. The most popular penalized method is the Lasso (Tibshirani, 1996) which applies the ℓ_1 penalty and estimates the β coefficients as the solution of:

$$\underset{\beta \in \mathbb{R}^p}{\arg\min} \left\{ ||\mathbf{Y} - \mathbf{X}\beta||_2^2 + \lambda ||\beta||_1 \right\}, \qquad ||\beta||_1 = \sum_{i=1}^{p} |\beta_i|,$$

where λ is a penalty called the regularization parameter. The form of the ℓ_1 penalty forces some coefficients to exactly equal zero. Taking advantage of the regression interpretation in GGM reminded in Proposition 1.7, a method for inferring non-nul elements of the precision matrix is to run separate penalized regressions of each node against all others and possible covariates, as in Meinshausen and Bühlmann (2006).

A joint estimation of these regression models is performed by the widely used Graphical Lasso (glasso) (Friedman et al., 2008). The glasso introduces sparsity in the precision matrix by imposing Lasso penalties on its entries. It is an iterative procedure which aims a sparse MLE of the penalized precision matrix $\mathbf{\Omega}$. The log-likelihood of a GGM writes:

$$L(\mathbf{\Omega}) = \log |\mathbf{\Omega}| + \operatorname{tr}(\mathbf{Y}^\intercal \mathbf{Y} \mathbf{\Omega}) + \text{cst}.$$

The precision matrix is then estimated as the matrix which maximizes the penalized log-likelihood:

$$\underset{\mathbf{\Omega} \succeq 0}{\arg\max} \left\{ L(\mathbf{\Omega}) - \lambda ||\mathbf{\Omega}||_1 \right\}, \qquad ||\mathbf{\Omega}||_1 = \sum_{j \neq k} |\omega_{jk}|.$$

The glasso selects the network edges by forcing some of them to zero in the precision matrix. The greater the penalty λ, the less edges are included in the network and therefore choosing λ is critical for the glasso. The selection of λ can be performed using classical tools such as cross-validation, Akaike information criterion (AIC), Bayesian information criterion (BIC) or its extended version. Another popular approach is to perform a stability selection with StARS (Liu et al., 2010a), which

fosters the network robustness under random sampling of the input data.

Averaging on trees

Another inference strategy is to assume a sparse structure. The sparsest connected graphs are spanning trees. If a multivariate variable $\boldsymbol{Y}_i$ of dimension p is faithful to a spanning tree T, its likelihood writes as a factorization on the nodes and the edges of T:

$$p_T(\boldsymbol{Y}_i) = \prod_{j=1}^{p} p(Y_{ij}) \prod_{jk \in T} \frac{p(Y_{ij}, Y_{ik})}{p(Y_{ij})p(Y_{ik})}.$$

The quotient in the product on the edges is close to the mutual information between two variables, which is the Küllback-Leibler divergence between the bivariate density and the product of their marginal densities. It can be seen as a measure of the independence between a pair of variables. The Chow-Liu algorithm (Chow and Liu, 1968) finds the tree structure which maximizes the above likelihood from observations. The maximal spanning tree is thus the tree with the maximal bivariate dependencies on its edges. Note that the Chow-Liu algorithm is not restricted to the Gaussian case, even if it is the most general one.

However the data conditional dependency structure is unlikely to be a tree in general. Assuming the dependency tree T to be random provides with a more flexible approach. Considering any distribution on the space of spanning trees, we now define the tree averaging model (also called tree aggregation or mixture of trees) which is a sum on a set of trees weighted by their respective probability to be the random tree T. This is classically formulated as follows:

Definition 1.10 (Tree averaging (Meilă and Jordan, 2000))**.** *Considering a collection of m spanning trees T_k, $k \in \{1, ..., m\}$ and a choice variable $\boldsymbol{z}$ taking values in $\{1, ..., m\}$, the distribution of an observed variable $\boldsymbol{Y}$ following a mixture of trees is defined as:*

$$p(\boldsymbol{Y}) = \sum_{k=1}^{m} p(\boldsymbol{z} = k) p(T_k).$$

Conditionally on $\boldsymbol{z}$, the distribution of $\boldsymbol{Y}$ is a tree.

Averaging on trees then means to treat the underlying tree T as a latent parameter of the model. In the context of GGM the use of tree averages for network inference relies on a Gaussian tree mixture, which is a mixture model of GGM that are faithful to trees.

Definition 1.11 (Gaussian tree mixture)**.** *The distribution of a variable $\boldsymbol{Y}$ is a*

Gaussian tree mixture on the space of m spanning trees if it writes as follows:

$$p(\boldsymbol{Y}) = \sum_{k=1}^{m} p(T_k) p(\boldsymbol{Y} \mid T_k), \qquad \boldsymbol{Y} \mid T_k \sim \mathcal{N}(0, \boldsymbol{\Omega}_{T_k}^{-1}), \ \ \forall k \in \{1, ..., m\}.$$

Conditional on T_k, $\boldsymbol{Y}$ is faithful relative to this tree and follows a multivariate Gaussian with the corresponding precision matrix $\boldsymbol{\Omega}_{T_k}$.

Instead of a subset m, considering an average on all possible spanning trees presents with the advantage of resorting to edges probabilities. Indeed the probability for an edge to be in the tree T can be defined as the sum of the probabilities of all trees containing this edge:

$$\mathbb{P}\{kl \in T\} = \sum_{\substack{T \in \mathcal{T} \\ T \ni kl}} p_{\mathbf{W}}(T).$$

The final output of a strategy using an average on trees for network inference is not the latent tree presenting with the highest probability, but the matrix filled with the edges probabilities obtained by summing on all trees. Therefore the output is a weighted network which has no reasons to be shaped as a tree itself.

We now detail a useful distribution on the space of trees: the decomposable distribution, which assigns a strictly positive weight to each edge (Meilă and Jaakkola, 2006; Meilă and Jordan, 2000).

Definition 1.12 (Decomposable tree distribution)**.** *For any symmetric weight matrix $\mathbf{W}$ with all positive entries, a decomposable distribution for the tree T is defined as follows:*

$$p_{\mathbf{W}}(T) = \prod_{kl \in T} w_{kl} / W,$$

where W is a normalizing constant with sum-product form as in Theorem 1.3.

Therefore under the decomposable distribution, the probability of a tree is proportional to the product of its edges weights. Such distribution is stable under a multiplicative transformation applied to its weight matrix, which is particularly useful during the implementation. The Chow-Liu algorithm actually maximizes the decomposable distribution with the mutual information as the edges weights.

Defining a tree averaging with the decomposable tree distributions $p_{\mathbf{W}}(T)$ presents several advantages. First, it introduces the quantity $\sum_{T \in \mathcal{T}} \prod_{k\ell \in T} w_{kl}$, which is a sum-product form easily handled thanks to Theorem 1.3 and Lemma 1.1. Then, the following algebraic result from Kirshner (2008) states that under a decomposable distribution it is possible to compute all edges probabilities at the same time, and this at the cost of the inversion of the Laplacian matrix of the edges weights.

Lemma 1.3 (Kirshner (2008)). *Let $p_{\mathbf{W}}$ be a decomposable tree distribution with symmetric and positive weight matrix $\mathbf{W}$. Taking the symmetric matrix $\mathbf{M}$ as defined in Lemma 1.1, the probability for an edge kl to be in the tree T writes:*

$$\mathbb{P}\{kl \in T\} = w_{kl}\, \mathbf{M}_{kl}.$$

Unlike the previous penalized approach which assumes a fixed graph, tree-averaging methods consider the graph as a latent and random tree. Coupled with decomposable tree distributions, they offer an efficient and exhaustive exploration of the space of spanning tree graphs. Tree averaging has recently been used in the context of GGM inference in the Bayesian setting (Schwaller et al., 2019) and for the inference of unobserved data (Robin et al., 2019).

1.3 Inference of incomplete data models

Incomplete data can refer either to unobserved variables due to experimental constraints, or latent variables in the model. The Expectation-Maximization (EM) algorithm (Dempster et al., 1977) is the classic approach to compute the maximum likelihood estimate in presence of hidden variables. In this section we present the general principles of the EM algorithm, as well as its variational version. We let $\boldsymbol{Y}$ denote the observed incomplete data, $\boldsymbol{Z}$ the hidden variables, p their joint distribution with parameter vector $\boldsymbol{\theta}$.

1.3.1 Expectation-Maximization algorithm

The EM algorithm is an iterative procedure which aims at maximizing the log-likelihood $\log p_\theta(\boldsymbol{Y})$ of the observed data Y. It is based on a decomposition of the incomplete data likelihood:

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}) = \mathbb{E}_{\boldsymbol{\theta}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}, \boldsymbol{Z}) \mid \boldsymbol{Y}] - \mathbb{E}_{\boldsymbol{\theta}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{Z} \mid \boldsymbol{Y}) \mid \boldsymbol{Y}].$$

This relation links the observed likelihood and the complete likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}, \boldsymbol{Z})$. The second term is actually the entropy of the latent variables $\boldsymbol{Z}$ given the observed data. The iteration $t+1$ of the EM algorithm then consists of the two following steps:

E step: Given $\boldsymbol{\theta}^t$, compute $\mathbb{E}_{\boldsymbol{\theta}^t}[\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}, \boldsymbol{Z}) \mid \boldsymbol{Y}]$ as a function of θ,

M step: Update $\boldsymbol{\theta}$ as $\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}}\{\mathbb{E}_{\boldsymbol{\theta}^t}[\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}, \boldsymbol{Z}) \mid \boldsymbol{Y}]\}$.

There is no guarantee as for the convergence of the estimate of $\boldsymbol{\theta}$ towards a global maximum of $\log p_\theta(\boldsymbol{Y})$. However an important property of the EM, obtained as a consequence of the Jensen's inequality, is that the log-likelihood of the observed

data $\log p_{\boldsymbol{\theta}}(\boldsymbol{Y})$ increases with the iterations of the EM algorithm (Dempster et al., 1977): $\log p_{\boldsymbol{\theta}^{t+1}}(\boldsymbol{Y}) > \log p_{\boldsymbol{\theta}^t}(\boldsymbol{Y})$.

Another formulation of the EM assumes a distribution q for the hidden variables (Neal and Hinton, 1998) and defines a lower bound $\mathcal{J}$ as:

$$\begin{aligned}\mathcal{J}(\boldsymbol{\theta}, q) &= \log p_{\boldsymbol{\theta}}(\boldsymbol{Y}) - KL(q(\boldsymbol{Z})||p_{\boldsymbol{\theta}}(\boldsymbol{Z} \mid \boldsymbol{Y})) \\ &= \mathbb{E}_q[\log p_\theta(\boldsymbol{Y}, \boldsymbol{Z})] - \mathbb{E}_q[\log q(\boldsymbol{Z})].\end{aligned}$$

where $KL(q(\boldsymbol{Z})||p(\boldsymbol{Z})) = \mathbb{E}_q[\log q(\boldsymbol{Z}) - \log p(\boldsymbol{Z})]$ is the Küllback-Leibler divergence. In this formulation, the link between the maximization of $\mathcal{J}$ and that of the likelihood is made clear. Iteration $t+1$ of the EM can then be written as a double maximization:

E step: $q^{t+1} = \arg\max_q\{\mathcal{J}(\boldsymbol{\theta}^t, q^t)\} = \arg\min_q\{KL(q(\boldsymbol{Z})||p_{\boldsymbol{\theta}}(\boldsymbol{Z} \mid \boldsymbol{Y}))\}$,

M step: $\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}}\{\mathcal{J}(\boldsymbol{\theta}^t, q^{t+1})\} = \arg\max_\theta\{\mathbb{E}_q[\log p_\theta(\boldsymbol{Y}, \boldsymbol{Z})]\}$.

The EM algorithm has become a classic approach to perform inference of models involving hidden or latent variables. In particular it is widely used to infer mixture models, for example mixture of trees Meilă and Jordan (2000). It corresponds to the case where no constraint is imposed to the q distribution: the solution of the E step is thus $q(\boldsymbol{Z}) = p(\boldsymbol{Z} \mid \boldsymbol{Y})$, and the Küllback-Leibler divergence is canceled. Unfortunately the conditional density $p_{\boldsymbol{\theta}}(\boldsymbol{Z} \mid \boldsymbol{Y})$ is not always available, in which case one might resort to a variational computation.

1.3.2 Variational estimation

Variational inference is a method for approximating conditional distributions (Blei et al., 2017; Jordan et al., 1999; Wainwright and Jordan, 2008). It is widely used in Bayesian settings to approximate posterior densities, that is looking for $q(\theta) \approx p(\theta \mid \boldsymbol{Y})$, as an alternative for Monte-Carlo Markov Chain (MCMC) sampling. Another use of the variational inference is to approximate the conditional distribution of a latent variable, thus looking for $q(\boldsymbol{Z}) \approx p(\boldsymbol{Z} \mid \boldsymbol{Y})$. These are two examples of the same general approximation problem.

When the conditional density of latent variables given the observed data cannot be computed, the variational version of the EM reduces the search space of the approximate distribution q to a set Q in the E step, and chooses a divergence D to define the lower bound. Doing so results in a variational approximation of $p_{\boldsymbol{\theta}}(\boldsymbol{Z} \mid \boldsymbol{Y})$. Therefore, in a Variation EM (VEM) algorithm, the E step is replaced by a variational-E (VE) step, which computes an approximate conditional distribution

as the solution of the following optimization problem:

$$q^{t+1} = \arg\min_{q \in Q} \left\{ D\big(q^t(\cdot) \,||\, p_{\theta^t}(\cdot \mid \boldsymbol{Y})\big) \right\}.$$

In other words the idea behind variational estimation is to choose a divergence D and a family of densities Q, and then find the member of this family which is the closest to the target distribution $p_{\boldsymbol{\theta}}(\boldsymbol{Z} \mid \boldsymbol{Y})$ as measured by the divergence. Their exists a variety of divergence measures (see Minka (2005) for an overview), and the Küllback-Leibler divergence as presented in the previous section is generally preferred for maximum-likelihood estimations. The complexity of the optimization is determined by the choice of the variational family. The set Q can be chosen as a parametric family, for example the set of Gaussian distributions: $Q = \{q(\boldsymbol{Z}) = \mathcal{N}(\boldsymbol{Z}; m, S)\}$. Another widely used choice for Q is the set of product-form or factorized distributions, where the latent variables are mutually independent: for K hidden variables $Q = \{q(\boldsymbol{Z}) = \prod_{k=1}^{K} q_k(Z_k)\}$. These two sets can be combined into the set of factorized Gaussian distributions.

The optimization problem of the VE step is most commonly solved using coordinate ascent. With a factorized approximate distribution $q(\boldsymbol{Z})$, coordinate ascent consists in iteratively optimize each hidden factor $q_k(Z_k)$ of the product while keeping the other fixed. A result from Beal and Ghahramani (2003) directly gives the optimal solution for each variable of a factorized distribution and is given in Proposition 1.8 below. It was originally formulated in the Bayesian setting, and as such this result applies to hidden variables as well as Bayesian random parameters. Hidden variables in proposition 1.8 can refer to either latent factors, unobserved variables, or Bayesian parameters.

Proposition 1.8 (Beal and Ghahramani (2003))**.** *In a variational EM using a factorized approximate distribution and the Küllback-Leibler divergence with observed data $\boldsymbol{Y}$ and K hidden variables $\boldsymbol{Z} = \{z_1, ..., z_K\}$, the solution of the VE step optimization problem is the following. At step $t+1$ and for any k in $\{1, ..., p\}$ the optimal variational marginal distribution q_k is proportional to the exponential of the expected log of the complete joint density:*

$$q_k^{(t+1)}(z_k) \propto \exp\left\{ \mathbb{E}_{q^t_{\setminus k}} \left[\log p_{\boldsymbol{\theta}^{t+1}}(\boldsymbol{Y}, \boldsymbol{Z}) \right] \right\}.$$

Proposition 1.8 shows that choosing Q as the set of factorized distributions leads to a so-called "mean-field" approximation, allowing a clearer presentation and easier use of variational inference algorithms with factorized approximations. Variational inference has been widely used in very diverse fields, among which computational biology and genetics (Carbonetto et al., 2012; Raj et al., 2014). Its study in network stochastic block model analysis yielded some specific theoretical results (Bickel et al.,

2013; Celisse et al., 2012), however establishment of general theoretical guarantees of variational inference is an active research area, as underlined in Blei et al. (2017).

1.4 Network inference from count data

Inferring a network from multivariate counts requires to model the joint distribution of discrete variables. Discrete distributions are not particularly practical to work with and their exist few options of multivariate formulations. A solution is then to use Gaussian random latent parameters and rely on a mixed generalized multivariate model, known as Joint Species distribution Model in ecology (Warton et al., 2015). This model then allows to resort to the GGM framework for network inference. The last part of this section summarizes several modeling strategies for network inference, as well as the one we adopted.

1.4.1 Modeling multivariate count data

A convenient distribution for modeling discrete data is the Poisson, which possesses some interesting features as it is part of the exponential family. As explained in Inouye et al. (2017), the first idea for extending the Poisson distribution to the multidimensional framework is it to build a multidimensional Poisson from univariate Poisson distributions. The multivariate Poisson of dimension p is formulated as a collection of variables that are the sums of p univariate Poisson distributions. This construction takes advantage of the subsequent summation of the Poisson distribution parameters (Teicher, 1954). It writes easily and presents the advantage of preserving the marginal Poisson distributions, but the set of possible correlations between variables is restricted. The extension of this formulation allows for a full covariance structure modeling, and can be inferred using an EM algorithm (see Karlis (2003)). However a significant modeling restriction is that only positive dependencies can be modeled, which is generally too strong an assumption. Moreover in higher dimensions the computational cost dramatically increases, and the writing becomes intractable.

Fortunately, there exist other more general ways of jointly modeling discrete data, the majority of which transposes the problem in the Gaussian framework, where they can take advantage of the easy handling of the multivariate Gaussian distribution and its practical properties. These methodologies can be gathered in the framework of multivariate mixed models, which are a general class of statistical parametric models for counts of multiple variables. More specifically, they are an extension of the generalized linear model and thus readily handle covariates and offsets. Additionally multivariate mixed models capture the correlation between the variables, and resort to Gaussian random effects and a link function to do so.

Hereafter we consider the input abundance data matrix $\mathbf{Y}$, and denote by the index i the rows (samples) and the index j the columns (species) of $\mathbf{Y}$. A multivariate random effect $\boldsymbol{Z}_i$ is associated with each data sample $\mathbf{Y}_i$. Denoting $\boldsymbol{x}_i$ the vector of covariates with regression coefficients $\boldsymbol{\theta}_j$, o_{ij} a possible offset and $g(\cdot)$ a link function, the mean abundance m_{ij} can be specified in a general manner as follows:

$$g(m_{ij}) = o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + Z_{ij}, \qquad \mathbf{Z}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}).$$

The art of multivariate mixed modeling then resides in the specification of g and the random effects. We here focus on counts, but binary data (e.g. presence/absence) could also be handle through mixed modeling. This model is known as a Joint Species Distribution Model in ecology, where the data at hand represents the sampled measures on a set of species at different sites. The following section presents some of the most used settings for multivariate mixed modeling of counts in ecology and microbiology.

1.4.2 Shift to Gaussian universe

Transformations

The first idea when it comes to transposing counts into the Gaussian framework is to transform them. Doing so allows to analyze the data without resorting to discrete modeling. A common transformation of counts is to apply the log function, with a small constant c added to counts to avoid zeros (pseudo-counts). In this case the mean of $\log(Y_{ij} + c)$ is then assumed to follow a multivariate Gaussian distribution. In genomics, high throughput sequencing yields samples $\mathbf{Y}_i$ of observed counts of p species or taxa that are constrained by the library size κ_i such that $\sum_j Y_{ij} = \kappa_i$ (Gloor et al., 2017). Thus the available observations are not absolute counts but relative counts, and each sample can be viewed as a compositional vector (Aitchison, 1982) sampled from the simplex

$$\mathcal{S}_i^p = \left\{ \boldsymbol{u} = (u_1, ..., u_p) \in \mathbb{R}^p \middle| \sum_{i=1}^{p} u_i = \kappa_i \right\}.$$

There exist several transformations for compositional data to change the κ_i-sum constraint, the most popular being the centered log-ratio transformation which writes:

$$clr(\mathbf{Y}) = \left(\log \frac{Y_1}{m(\mathbf{Y})}, ..., \log \frac{Y_p}{m(\mathbf{Y})} \right),$$

where $m(\boldsymbol{Y}) = \left(\prod_{j=1}^{p} Y_j \right)^{1/p}$ is the geometric mean. A clr-transformed vector is thus constrained to a zero sum. In the framework of joint modeling, the clr function can be used as a link function to account for covariates and offsets. Recent works

on network inference use the clr transformation (Kurtz et al., 2015), or the logistic normal distribution (Aitchison and Shen, 1980) to model relative counts (Fang et al., 2017).

Copulas

Copulas are joint distributions which map the marginal cumulative distribution functions of a multivariate random vector. Their use with continuous distributions originates from the Sklar's theorem which states that a joint continuous distribution can be decomposed into a copula and marginal distributions. Conversely the pairing of a copula and some marginal distributions yields a valid joint distribution. As copulas fully describe the dependence structure, these models allow to fully separate the modeling of marginal distributions from the modeling of dependencies. In their original form, they only apply to continuous marginals, and their extension to discrete data has been controversial in particular about identifiability, and computationally challenging (Faugeras, 2017). However recent developments made the use of Gaussian copula coupled with discrete marginal distributions possible (Panagiotelis et al., 2012; Popovic et al., 2018), opening the way to the application of copulas in the analysis of multivariate count data (Anderson et al., 2019).
In this context, a first step estimates each p marginal univariate discrete distributions parameters while accounting for covariates and possible offsets. Then the distribution of the response Gaussian copula is computed as a $p-$dimensional integrand (see Popovic et al. (2018)). Popovic et al. (2019) showed that Gaussian copulas are a relevant and promising approach to the problem of network inference from abundance data, even if the computation cost remains substantial as it requires importance sampling as well as Monte Carlo Expectation Maximization algorithms. One way of taking advantage of the copula theory without having to actually estimate the joint distribution is to use copulas as a data transformation. The nonparanormal transformation of counts is a semiparametric Gaussian copula (Liu et al., 2009). It is estimated in a computational efficient manner by marginally transforming the variables using ranks and univariate Gaussian quantiles. This transformation however is sensitive to ex-aequo and 0 counts. Clark et al. (2018) resorts to the nonparanormal approach in an attempt to estimate network parameters varying across a gradient of covariates.

Latent variables

Latent variables enable the modeling of multivariate discrete data using a hierarchical setting where random discrete variables are modeled conditionally to random latent variables. Two specifications of latent variables stand out in community ecology (Warton et al., 2015): the Multivariate Generalized Linear Mixed Model (GLMM) (Ovaskainen et al., 2010; Pollock et al., 2014), and the Latent Variable

Model (LVM) (Ovaskainen et al., 2016, 2017). The difference between these models lies in the dimension of their respective random effects: there are as many latent variables as there are species in the GLMM, whereas in the LVM their number is a parameter of the model.

In more details, a GLMM models the latent variables and abundances as follows:

$$\begin{aligned} Y_{ij} \mid \mathbf{Z}_i &\sim F(m_{ij}, \phi_j) \\ \mathbf{Z}_i &\sim \mathcal{N}(0, \mathbf{\Sigma}), \end{aligned}$$

where F is a distribution with mean m_{ij} and dispersion parameter ϕ_j. There are as many latent variables as observed ones. In the case of LVM the random effect associated with the mean abundance m_{ij} is a linear combination of a set of $K < p$ latent variables, and is generally specified as:

$$g(m_{ij}) = o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + \sum_{k=1}^{K} Z_{ik} \lambda_{kj},$$

where the matrix $\mathbf{\Lambda}$ gathers the factors loadings $\boldsymbol{\lambda}_j$ in columns. Abundances are then assumed to follow

$$\begin{aligned} Y_{ij} \mid \mathbf{Z}_i &\sim F(m_{ij}, \phi_j) \\ \mathbf{Z}_i &\sim \mathcal{N}(0, \boldsymbol{I}). \end{aligned}$$

The covariance matrix of the latent layer of random effects is then of rank K and can be computed from the factor loadings as $\mathbf{\Sigma} = \mathbf{\Lambda}^\intercal \mathbf{\Lambda}$. The LVM is appreciated for its small number of parameters to estimate (Ovaskainen et al., 2017).

The Poisson log-normal distribution (PLN, Aitchison and Ho (1989)) is a GLMM where F is the Poisson distribution and g the log function. This model can also be seen as an infinite Poisson mixture as presented in Inouye et al. (2017), with a log-normal distribution for the parameters. The PLN model presents the advantage of having closed form moments:

$$\begin{aligned} \mathbb{E}[\mathbf{Y}_i] &= e^{\mu_i + \frac{1}{2}\sigma_{ii}} = \alpha_i \\ \mathbb{V}(\mathbf{Y}_i) &= \alpha_i + \alpha_i^2 (e^{\sigma_{ii}} - 1) \\ \mathbb{C}\mathrm{ov}(\mathbf{Y}_i, \boldsymbol{y}_j) &= \alpha_i \, \alpha_j (e^{\sigma_{ij}} - 1). \end{aligned}$$

An interesting property of the multivariate Poisson log-normal law is the conservation of the correlation signs between the observed and latent layers: $\mathrm{sign}(\mathrm{Cor}(Y_{ij}, Y_{ik})) = \mathrm{sign}(\mathrm{Cor}(Z_{ij}, Z_{ik}))$, including the null correlation. Moreover if σ_{jk} is null, then it can be shown that the bivariate distribution $p(Y_{ij}, Y_{ik})$ factorizes

under the product of its marginals, yielding the marginal independence of the two variables (Aitchison and Ho, 1989). This model can be estimated using variational inference in Chiquet et al. (2018). Biswas et al. (2016) and Chiquet et al. (2019a) use the PLN distribution in the context of network inference.

1.4.3 Network inference

As stated in the previous section, most methodologies to infer networks from count data first model count data in a way to transpose the problem in the Gaussian setting. There they take advantage of the GGM framework detailed in section 1.2.1 to perform network inference. The majority of them then use the penalized estimation using either the LASSO or the glasso as detailed in section 1.2.3. Table 1.1 below summarizes the modeling of counts and inference strategy adopted by the methods of interest in this work, which have all been previously mentioned. Note that we focused on methods using the framework of Graphical Models, and included LITree and saturnin even if they do not model count data for they have been a source of inspiration for this work.

		Gaussian shift			Network inference	
	Ref.	Trans.	Copulas	LV	Penalized	Trees
SpiecEasi	Kurtz et al. (2015)	x			x	
MInt	Biswas et al. (2016)			x	x	
gCoda	Fang et al. (2017)	x			x	
MRFcov	Clark et al. (2018)		x		x	
ecoCopula	Popovic et al. (2019)		x		x	
PLNnetwork	Chiquet et al. (2019a)			x	x	
saturnin	Schwaller et al. (2019)					x
LITree	Robin et al. (2019)					x

Table 1.1 – Network inference methods of interest: their strategy to model counts (transformation, copulas or latent variables), and choice for network inference (penalized estimation or tree average).

Inference in the observed layer $\boldsymbol{Y}$

We presented methods relying on a Gaussian layer for the network inference. Another mathematical framework for graphs with discrete data are the Poisson Graphical Models, which are graphical models of the observed layer $\boldsymbol{Y}$ directly, and not the $\boldsymbol{Z}$. These models use properties of the exponential family to derive graphical models with node-conditional Poisson distributions. However this particular class of graphical models entails a major drawback: only negative dependencies relationships can be modeled. As this is a too unrealistic assumption when it comes to

species interactions we did not discuss this solution here, although see Yang et al. (2013) and Inouye et al. (2016) for valuable extensions of this model. In Chapter 4 we propose a model for network inference in the layer of $\boldsymbol{Y}$, using tree averaging.

1.4.4 Proposed methodology

The network inference method which is developed in the following chapters of this work models count data $\boldsymbol{Y}$ using the aforementioned PLN distribution, including offsets and covariates:

$$Y_{ij} \mid \boldsymbol{Z}_i \sim \mathcal{P}(\exp(o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + Z_{ij})), \quad (Y_{ij} \perp\!\!\!\perp) \mid \boldsymbol{Z}_i.$$

The graph underlying the latent layer of Gaussian parameters $\boldsymbol{Z}$ is assumed to be a random tree, so that the conditional distribution is faithful to T:

$$\boldsymbol{Z}_i \mid T \sim \mathcal{N}(0, \boldsymbol{\Omega}_T), \quad \{\boldsymbol{Z}_i\}_i \text{ iid.}$$

The tree T is assumed to be distributed following a decomposable distribution as defined in Definition 1.12:

$$T \sim \prod_{kl \in T} \beta_{kl}/B, \quad B = \sum_{T \in \mathcal{T}} \prod_{kl \in T} \beta_{kl}.$$

The approach then relies on tree averaging for the network inference as detailed in section 1.2.3. Namely, the latent Gaussian parameters $\boldsymbol{Z}$ follow a Gaussian tree mixture on the whole space of spanning trees $\mathcal{T}$, where each Gaussian component is faithful to a spanning tree as is specified in Definition 1.11. This writes

$$\mathbf{Z}_i \sim \sum_{T \in \mathcal{T}} p(T) \mathcal{N}(\boldsymbol{Z}_i \mid T; 0, \boldsymbol{\Omega}_T).$$

Chapter 2 details the inference of this model. Chapter 3 considers the case of additional dimensions in the latent layer $\boldsymbol{Z}$, that is when $\boldsymbol{Z}_i$ is actually of dimension $p + r$, where $r > 0$ is a number of unobserved actors.

2

Network Inference from Abundance Data

Contents

This chapter is based on the article *Tree-based Inference of Species Interaction Networks from Abundance data* published in Methods in Ecology and Evolution (Momal et al., 2020). It details an original network inference method from species observed abundance data using the Poisson log-normal model and tree averaging. The developed methodology is compared to existing alternative methods for network inference,

from ecology and genomics. After an enriching discussion with Nicolas Clark (MRFcov R package), covariates adjustment has been corrected, as well as simulation parameters of the Scale-free structures. Consequently Fig. 2.4 has been updated and Fig. 2.5 added. The appendix has been updated accordingly. This chapter augments the published material, and adds a vignette giving usage examples of the R package implementing the developed method.

2.1 Introduction

There is a growing awareness of biotic interactions being crucial components of biodiversity and relevant descriptors of ecosystems (Jordano, 2016; Valiente-Banuet et al., 2015). Such interactions can be conveniently represented by networks, which have been increasingly studied and used in recent years for describing and understanding living systems in ecology (Poisot et al., 2016), microbiology (Faust and Raes, 2012) or genomics (Evans et al., 2016). Observing species interactions is a laborious task which restricts them to certain categories (e.g. pollination, predation, parasitism), while many other types of interactions may be hard to observe and key in the system organization (e.g. communication, shelter sharing). Many efforts have been devoted in the last decade to get a more complete picture of the biotic interactions existing between species living in the same niche: all these interactions can be gathered in a so-called *species interaction network*.

Network reconstruction. A first attempt consists in using observed interactions to predict other possible links based on species traits matching (see e.g. Bartomeus et al., 2016; Graham and Weinstein, 2018; Olito and Fox, 2015; Weinstein and Graham, 2017). The interaction strength can also be predicted (Wells and O'Hara, 2013). This can be viewed as a prediction task, and modern approaches arising from signal processing and machine learning have been also proposed (Dallas et al., 2017; Desjardins-Proulx et al., 2017; Stock et al., 2017). We name these approaches *network reconstruction* to distinguish them from *network inference*, which is the problem we consider in this article.

Network inference. Network inference approaches also aim at retrieving the interactions among species, but do not rely on observed interactions and therefore, remain agnostic as for their type. Such approaches have been developed in many domains ranging from cell biology (Friedman, 2004, to infer gene regulatory networks) to neurosciences (Zhu and Cribben, 2018, to deciphere brain connectivity structures). In ecology, it will typically aim at inferring the set of biotic interactions linking species from the same guild. As summarized in Fig. 2.1, network inference takes as input measures on species (here abundances) at similar sites, and returns a network of species direct interactions. The importance of distinguishing between

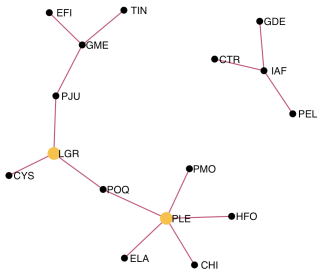
direct interaction and indirect association between species is explained in Popovic et al. (2019).

Species not engaged in biotic interactions can appear linked if they respond similarly or oppositely to an abiotic effect (spurious interaction). Therefore network inference must account for environmental covariates. Fig. 2.2 illustrates this phenomenon: in (c) species (1 and 4) are not in direct interaction, but are affected by the variations of the same environmental covariate x. (d) displays the network when x is not accounted for: a spurious edge appears between species.

date	site	EFI	ELA	GDE	GME	HFA
apr93	km03	71	1	5	6	0
apr93	km03	118	2	3	0	0
apr93	km03	69	0	6	2	0
apr93	km03	56	0	0	0	0
apr93	km17	0	1	1	0	0
apr93	km17	0	0	2	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮

(a) covariates

(b) species abundances

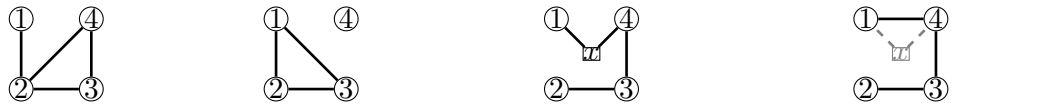


(c) inferred network

Figure 2.1 – Aim of species interaction network inference from abundance data. Data sample from the Fatala river dataset (see Section 2.2.3).

Joint species distribution models. The rationale behind network inference is that interactions between species must affect their joint distribution in a series of similar sites. Such approaches necessarily rely on a *joint* species distribution model (JSDM), as opposed to species distribution models (Elith and Leathwick, 2009) where species are traditionally considered as disconnected entities. A JSDM is a probabilistic model describing the species simultaneous presence/absence (Harris, 2015; Ovaskainen et al., 2017) or joint abundances (Popovic et al., 2018, 2019). An important feature of JSDMs is to include environmental covariates to account for abiotic interactions.

Recently, latent variable models have received attention in community ecology as they provide a convenient way to model the dependence structure between species (Warton et al., 2015). The JSDM proposed by Popovic et al. (2018, 2019) involves a latent layer. So does the Poisson log-Normal model (PLN, Aitchison and Ho, 1989), which combines generalized linear models to account for covariates and offsets, and a Gaussian latent structure to describe the species interactions. It can be seen as a multivariate mixed model, in which correlated random effects encode the dependency between the species abundances.



(*a*) connected (*b*) disconnected (*c*) with covariate (*d*) missing covariate

Figure 2.2 – Examples of graphical models. (*a*) All species are dependent, (*b*) 4 is independent from all others, (*c*) 1 and 4 are independent conditional on x, (*d*) not accounting for x induces a spurious dependence between 1 and 4.

Graphical models: a generic framework for network inference. Although they describe the dependence structure between the distributions of all the species from a same niche, JSDM are not sufficient to perform network inference as they do not distinguish indirect associations from direct interactions (Dormann et al., 2018). Graphical models (Lauritzen, 1996) provide a probabilistic framework to do so and, in the same time, a formal definition of the network to be inferred. This formalism is therefore especially appealing for the inference of species interaction networks (Popovic et al., 2019). In an undirected graphical model (which is the same as a Markov random field: Clark et al., 2018), two species are connected if they are *dependent* conditional on all other species, that is if the variations of their respective abundances would still be correlated if ever both the environmental conditions and the abundances of all other species were kept fixed. Two species are unconnected if they are *independent* conditional on all other species: the observed correlation between them only results from a series of links with other species (Morueta-Holme et al., 2016) or environmental effects. Fig. 2.2 illustrates the concept of conditional dependence/independence with toy graphical models. In (*a*), the network is connected so all species are interdependent: an association exists between any two of them. However, 1 is only directly interacting with 2 which mediates its association with 3 and 4: 1 is independent from them conditional on 2.
In (*b*), the network is disconnected: species 4 is independent from all others. This illustrates that graphical models enjoy all the desirable properties to represent interactions between species in an interpretable manner, so that they can be used as the mathematical counterpart of species interaction networks.

Network inference: the general problem. Network inference methods attempts to retrieve the graphical model underlying the distribution of abundance data. In every domains, network inference is impeded by the huge number of possible graphs for a given set of nodes, which increases super-exponentially with the latter (more than 10^{13} undirected graphs can be drawn between 10 nodes, and more than 10^{57} between 20). The exploration of the graph space is therefore intractable from a combinatorial point of view. To reduce the search space, a common and reasonable assumption is that a relatively small fraction of species pairs are in direct

interaction: the network is sparse. In the case of continuous observations, one of the most popular approaches is the graphical lasso (glasso: Friedman et al. (2008)) which takes advantage of the properties of Gaussian graphical models (GGM) to efficiently infer a sparse network. Alternatively, tree-based approaches have been proposed: Chow and Liu (1968) first made the too stringent assumption that the network is made of a single spanning tree (that is connecting all nodes without any loop, as in Fig. 2.3). More recent approaches introduced by Meilă and Jaakkola (2006) and Kirshner (2008) rely on efficient algebraic tools to average over all possible tree-structured graphical models. The inferred network resulting from such an averaging procedure is not restricted to be a tree: species or groups of species can be isolated (e.g. Fig. 2.1), and loops can appear (e.g. Fig. 2.3).

Network inference from species abundance data. This work focuses on network inference based on abundance data, and not only their presence/absence (as considered in Clark et al., 2018; Ovaskainen et al., 2010). Network inference from species abundance measures is a notoriously difficult problem (Ulrich and Gotelli, 2010), not only because network inference is complex, but also because it has to account for the data specificities. Abundance data may spread over a wide range of values and often result from sampling efforts (sample and/or species-specific), making them difficult to compare. Obviously, count data do not directly fit the Gaussian framework but many network inference methods dedicated to abundance data actually rely on a latent Gaussian structure (see Section 2.2.3).

Contribution. In the present work, we adopt a model-based approach to perform network inference from abundance data. To accommodate the data specificities we use a PLN model, which includes the over-dispersion of the observed counts as well as the sampling effort. Importantly, the PLN model allows us to account for abiotic effects and avoid the detection of spurious interactions between species.
As for the network inference, we adopt a tree-based approach (as opposed to Biswas et al., 2016, which also use a PLN model but resort to glasso), which provides a probability for each edge to be actually part of the underlying graphical model.

Outline. We introduce the method EMtree, which combines two (variational) Expectation-Maximization (EM) algorithms to estimate the model parameters. Importantly, our approach provides the probability for each possible edge to be part of the interaction network. We evaluate our approach on both synthetic and ecological datasets. An R package implementing EMtree is available on GitHub `https://github.com/Rmomal/EMtree`.

2.2 Material and methods

2.2.1 Model

Let us first describe the typical type of data we consider. We assume that p species have been observed in n sites. The abundances are gathered in the $n \times p$ matrix $\mathbf{Y}$. Y_{ij} is the abundance of species j in site i, and $\mathbf{Y}_i$ the abundance vector collected in site i (ith row of $\mathbf{Y}$). We further assume that a vector of covariates $\boldsymbol{x}_i$ of size d has been measured in each site i and that all covariates are gathered in the $n \times d$ matrix $\mathbf{X}$. The sites are supposed to be independent.
Our aim is to decipher the dependency structure between the p species, accounting for the effect of the environmental covariates encoded in $\mathbf{X}$. As explained above, ignoring environmental covariates is more than likely to result in spurious edges.

Mixed model. To distinguish between covariates effects and species interactions, we consider a mixed model which states that each abundance Y_{ij} has a (conditional) Poisson distribution

$$Y_{ij} \sim \mathcal{P}\left(\exp(\boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + o_{ij} + Z_{ij})\right). \tag{2.1}$$

In model (2.1), o_{ij} is the sample- and species-specific offset which accounts for the sampling effort. $\boldsymbol{\theta}_j$ is the vector of fixed regression coefficients measuring the effect of each covariate on species j abundance. The regression part is similar to a general linear model as used in niche modeling (see e.g. Austin, 2007). Z_{ij} is the random effect associated with species j in site i. Importantly, the coordinates of the site-specific random vector $\mathbf{Z}_i = (Z_{i1}, \dots Z_{ip})$ are not independant: the multivariate random term $\mathbf{Z}_i$ precisely accounts for the interactions that are not due to environmental fluctuations. For each site i, a vector $\mathbf{Z}_i$ is associated with the corresponding abundance vector $\mathbf{Y}_i$. The distribution given in Eq. (2.1) is over-dispersed as the Poisson parameter is itself random, which suits ecological modeling of abundance data (Richards, 2008).

We now describe the distribution of the latent vector $\mathbf{Z}_i$. To this aim, we adopt a version of Kirshner's model (Kirshner, 2008), which states that a spanning tree T is first drawn with probability

$$p(T) = \prod_{jk \in T} \beta_{jk}/B, \tag{2.2}$$

where $jk \in T$ means that the edge connecting species j and k is part of the tree T and where B is a normalizing constant. Each edge weight β_{jk} controls the probability for the edge (j, k) to be in the interaction network.
Then for each site i, a vector $\mathbf{Z}_i$ is drawn independently with conditional Gaussian

distribution $(\mathbf{Z}_i \mid T) \sim \mathcal{N}(0, \Sigma_T)$, where the subscript T means that the distribution of $\mathbf{Z}_i$ is faithful to T. When T is a spanning tree, this faithfulness simply means this distribution can be factorized on the nodes and edges of T as follows (see Kirshner, 2008):

$$p(\mathbf{Z}_i \mid T) = \prod_{j=1}^{p} p(Z_{ij}|T) \prod_{jk \in T} \psi_{jk}(\mathbf{Z}_i), \tag{2.3}$$

where $\psi_{jk}(\mathbf{Z}_i)$ does not depend on T. This factorization means that each edge of T corresponds to a species pair in direct interaction; all other pairs are conditionally independent. Experiments are independent, and in the sequel we consider the product of all $p(\mathbf{Z}_i)$ and use the simpler notation $\psi_{jk} = \prod_i \psi_{jk}(\mathbf{Z}_i)$ instead.

According to Eq. (2.2), each $\mathbf{Z}_i$ has a Gaussian distribution conditional on the tree T, so its marginal distribution is a mixture of Gaussians: $\mathbf{Z}_i \sim \sum_{T \in \mathcal{T}} p(T)\mathcal{N}(0, \Sigma_T)$, where $\mathcal{T}$ is the set of all spanning trees. As a consequence, the joint distribution of the $\mathbf{Z}_i$ is modeled by a mixture of distributions with tree-shaped dependency structure.
Besides, for all trees including the edge (j, k), the estimate of the covariance term between the coordinates j and k is the same (see Lauritzen, 1996; Schwaller et al., 2019). Hence, we may define a global covariance matrix $\mathbf{\Sigma}$, filled with covariances that are each common to spanning trees containing a same edge. Each $\mathbf{\Sigma}_T$ is then built by extracting from $\mathbf{\Sigma}$ the covariances corresponding to the edges of T.

2.2.2 Inference with EMtree

We now describe how to infer the model parameters. We gather the edges weights into the $p \times p$ matrix $\boldsymbol{\beta}$ and the vectors of regression coefficients into a $d \times p$ matrix $\boldsymbol{\theta}$. The $p \times p$ matrix $\mathbf{\Sigma}$ contains the variances and covariances between the coordinates of each latent vector $\mathbf{Z}_i$. Hence, the set of parameters to be inferred is $(\boldsymbol{\beta}, \mathbf{\Sigma}, \boldsymbol{\theta})$.

Likelihood. The model described above is an incomplete data model, as it involves two hidden layers: the random tree T and the latent Gaussian vectors $\mathbf{Z}_i$. The most classical approach to achieve maximum likelihood inference in this context is to use the Expectation-Maximization algorithm (EM: Dempster et al., 1977). Rather than the likelihood of the observed data $p(\mathbf{Y})$, the EM algorithm deals with the often more tractable likelihood $p(T, \mathbf{Z}, \mathbf{Y})$ of the complete data (which consists of both the observed and the latent variables). It can be decomposed as

$$p_{\boldsymbol{\beta},\mathbf{\Sigma},\boldsymbol{\theta}}(T, \mathbf{Z}, \mathbf{Y}) = p_{\boldsymbol{\beta}}(T) \times p_{\mathbf{\Sigma}}(\mathbf{Z} \mid T) \times p_{\boldsymbol{\theta}}(\mathbf{Y} \mid \mathbf{Z}), \tag{2.4}$$

where the subscripts indicate on which parameter each distribution depends.

Observe that the dependency structure between the species is only involved in the first two terms, whereas the third term only depends on the regression coefficients $\boldsymbol{\theta}$. We take advantage of this decomposition to propose a two-stage estimation algorithm. The first stage deals with the observed layer $p_{\boldsymbol{\theta}}(\mathbf{Y} \mid \mathbf{Z})$, the second with the two hidden layers $p_{\boldsymbol{\beta}}(T)$ and $p_{\boldsymbol{\Sigma}}(\mathbf{Z} \mid T)$. The network inference itself takes place in the second step.

Inference in the observed layer. The variational EM (VEM) algorithm that provides an estimate of the regression coefficients matrix $\boldsymbol{\theta}$ is described in Appendix 2.A.1 (along with a reminder on EM and VEM). It also provides the (approximate) conditional means $\mathbb{E}(Z_{ij}|\mathbf{Y}_i)$, variances $\mathbb{V}(Z_{ij}|\mathbf{Y}_i)$ and covariances $\mathbb{C}\mathrm{ov}(Z_{ij}, Z_{ik}|\mathbf{Y}_i)$ required for the inference in the hidden layer. As a consequence, this first step provides the estimates $\widehat{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\Sigma}}$.

Inference in the hidden layer. The second step is dedicated to the estimation of $\boldsymbol{\beta}$. The EM algorithm actually deals with the conditional expectation of the complete log-likelihood, namely $\mathbb{E}\left(\log p_{\boldsymbol{\beta},\boldsymbol{\Sigma},\boldsymbol{\theta}}(T, \mathbf{Z}, \mathbf{Y}) \mid \mathbf{Y}\right)$. As shown in Appendix 2.A.2, this reduces to

$$\mathbb{E}\left(\log p_{\boldsymbol{\beta},\boldsymbol{\Sigma},\boldsymbol{\theta}}(T, \mathbf{Z}, \mathbf{Y}) \mid \mathbf{Y}\right) \simeq \sum_{1 \leq j < k \leq p} P_{jk} \log\left(\beta_{jk} \widehat{\psi}_{jk}\right) - \log B + \text{cst} \tag{2.5}$$

where $\widehat{\psi}_{jk}$ is the estimate of ψ_{jk} defined in Eq. (2.3), and the 'cst' term depends on $\boldsymbol{\theta}$ and $\boldsymbol{\Sigma}$ but not on $\boldsymbol{\beta}$. P_{jk} is the approximate conditional probability (given the data) for the edge (j, k) to be part of the network: $P_{jk} \simeq \mathbb{P}\{jk \in T \mid Y\}$. It is also shown in Appendix 2.A.2 that $\widehat{\psi}_{jk} = (1 - \widehat{\rho}_{jk}^2)^{-n/2}$, where the estimated correlation $\widehat{\rho}_{jk}$ depends on the conditional mean, variance and covariances of the Z_{ij}'s provided by the first step. Eq. (2.5) is maximized via an EM algorithm iterating the calculation of the P_{jk} and the maximization with respect to the β_{jk}:

Expectation step: Computing the P_{jk} with tree averaging. The conditional probability of an edge is simply the sum of the conditional probabilities of the trees that contain this edge. Hence, computing P_{jk} amounts to averaging over all spanning trees. Fig. 2.3 illustrates the principle of tree averaging for a toy network with $p = 4$ nodes. Here, five arbitrary spanning trees t_1 to t_5 (among the $p^{p-2} = 16$ spanning trees) are displayed, with their respective conditional probability $p(T \mid Y)$. The edge $(1, 3)$ has a high conditional probability P_{13} because it is part of likely trees such as t_3 and t_4, whereas P_{23} is small because the edge $(2, 3)$ is only part of unlikely trees (e.g. t_1, t_2).
Averaging over all spanning trees at the cost of a determinant calculus (i.e. with complexity $O(p^3)$) is possible using the Matrix Tree theorem (Chaiken and Kleitman, 1978, recalled as Theorem 2.1 in Appendix 2.A.3). Kirshner

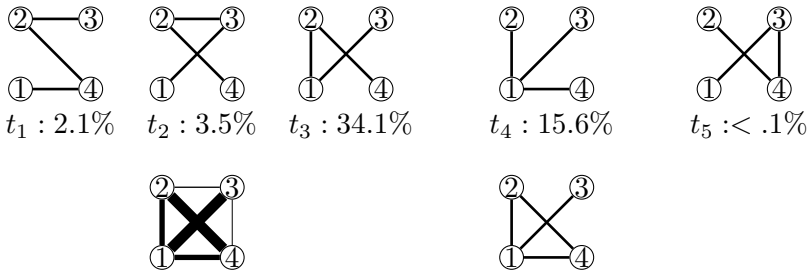


Figure 2.3 – Tree averaging principle. *Top:* 5 spanning trees with 4 nodes $(t_1, \dots t_5)$, with their respective conditional probability given the data $P(T = t \mid Y)$. *Bottom left:* Weighted graph resulting from tree averaging. Each edge has width proportional to its conditional probability. *Bottom right:* Estimated graph (obtained by thresholding edge probabilities) is not a tree.

(2008) further shows that all the P_{jk}'s can be computed at once with the same complexity $O(p^3)$, although the calculation may lead to numerical instabilities for large n and p.

Maximization step: Estimating the β_{jk}. This step is not straightforward, as the normalizing constant $B = \sum_T \prod_{jk \in T} \beta_{jk}$ involves all β_{jk}'s. We propose an exact maximization built upon the Matrix Tree theorem (see Appendix 2.A.2).

Algorithm output: edge scoring and network inference EMtree provides the (approximate) conditional probability P_{jk} for each edge (j, k) to be part of the network. Whenever an actual inferred network $\widehat{G}$ is needed (e.g. for a graphical purpose), it can be obtained by thresholding the P_{jk} (see Fig. 2.3, bottom right). Because we are dealing with trees, a natural threshold is the density of a spanning tree, which is $2/p$. More robust results can be obtained using a resampling procedure similar to the stability selection proposed by Liu et al. (2010b). It simply consists in sampling a series of subsamples $s = 1 \dots S$, to get an estimate $\widehat{G}^s$ from each of them and to collect the selection frequency for each edge. Again, these edge selection frequencies can be thresholded if needed.

2.2.3 Simulation and illustrations

Because network inference is an unsupervised problem (as opposed to network reconstruction), we compare the accuracy of the methods described above on synthetic abundance datasets, for which the true underlying network is known.

Alternative inference methods

We consider network inference methods dedicated to both metagenomics (SPIEC-EASI, gCoda and MInt) and ecology (MRFcov, ecoCopula). All methods can handle count data and rely on some (implicit) Gaussian setting. SPIEC-EASI (Kurtz et al., 2015) and gCoda (Fang et al., 2017) resort to data transformation to fit a Gaussian framework. MInt (Biswas et al., 2016) considers a Poisson mixed model similar to the one we consider. MRFcov (Clark et al., 2018) uses a transformation which is equivalent to a Gaussian copula, and ecoCopula (Popovic et al., 2019) defines a multivariate count distribution, the dependency structure of which is encoded in a Gaussian copula. These methods all rely on a Gaussian graphical model (GGM), so that the network inference problem amounts to estimating a sparse version of the inverse covariance matrix (also named *precision* matrix).

Edge scoring. These methods build upon glasso penalization (Friedman et al., 2008). For each edge, there exists a minimal penalty value above which it is eliminated from the network. The higher this minimal penalty, the more reliable the edge in the network, so it can be used as a score reflecting the importance of an edge. Only SpiecEasi and gCoda provide unthresholded quantities (namely the glasso regularization path) that can be used for edge scoring; the other methods only return their optimal graph.

Covariates. Only MInt and ecoCopula may take covariates into account in their model. MRFcov includes covariates, but in the aim of predicting changes in interactions across covariate gradients, which is different from adjusting for covariates. Thus in order to draw a fair comparison, we give SPIEC-EASI and gCoda access to the covariate information by feeding them with residuals of the linear regression of the transformed data onto the covariates. A generalized linear model of the raw data is adjusted for MRFcov (the same as in ecoCopula), which then infers the network from the Gaussian residuals.

Comparison criteria

False Discovery Rate (FDR) and density ratio criteria. Inferred networks are mostly useful to detect potential interactions between species, which then need to be studied by experts to determine their exact nature. Falsely including an edge lead to meaningless interpretation or useless validation experiments.
A network with a few reliable edges will be preferred to one having more edges with a larger risk of possible false discoveries. Therefore we choose the FDR as an evaluation criterion, which should be close to 0. Comparing FDR's only makes sense for networks with similar densities. We then compute the ratio between the densities of the inferred and the true network (*density ratio*).

Area Under the Curve (AUC) criterion. The AUC criterion allows to evaluate the inferences quality without resorting to any threshold. It evaluates the probability for a method to score the presence of a present edge higher than that of an absent one; it should be close to 1. Note that this criterion cannot be computed for MRFcov, ecoCopula and MInt as they provide a unique network.

Simulation design

Simulated graphs. We consider three typical graph structures: Scale-free, Erdös (short for Erdös-Reyni) and Cluster. Scale-free structure bears the closest similarity to the tree one, with almost the same density and no loops; it is popular in social networks and in genomics as it corresponds to a preferential-attachment behavior. It is simulated following the Barbási-Albert model as implemented in the *huge* R package (Zhao et al., 2012). The degree distribution of Scale-free structure follows a power law, which constrains the edges probabilities such that the network density cannot be controlled. Erdös structure is the most even as the edges all have the same existence probability. It is a step away from the tree as it may contain loops and its density can be increased arbitrarily. Cluster structure spreads edges into highly connected clusters, with few connections between the clusters; the *ratio* parameter controls the intra/inter connection probability ratio.

Simulated counts. The datasets are simulated under the Poisson mixed model described in Eq. (2.1). We first build the covariance matrix Σ_G associated with a graph G following Zhao et al. (2012) and randomly choosing the sign of the link, so that in our simulations we consider both positive and negative interactions. For each site i, we simulate $\mathbf{Z}_i \sim \mathcal{N}(0, \Sigma_G)$, then use these parameters together with a set of covariates to generate count data $\mathbf{Y}$. We use three covariates (one continuous, one ordinal and one categorical), with their regression coefficients θ drawn from a standard uniform distribution to create heterogeneity in environmental response across species.

Experiments. For each set of parameters and type of structure we generate 100 graphs, simulate a dataset under a heterogeneous environment and infer the dependency structure using EMtree, gCoda, SpiecEasi MInt, ecoCopula and MRFcov (the three latter only for Exp. 1). The settings of all methods are set to default, except for ecoCopula for which we use the "AIC" selection criterion ("BIC" gives too many empty results). All computation times are obtained with a 2.5 GH Intel Core 17 processor and 8G of RAM.

Exp. 1: effect of the data dimensions on the inferred network. We compare performances in terms of FDR and density ratios on two scenarios: *easy* ($n = 100$,

$p = 20$), and *hard* ($n = 50$, $p = 30$). The network density for Erdös and Cluster structures is set to $\log(p)/p$.

Exp. 2: effect of the network structure on edge rankings. AUC measures are collected for alternate variations of n and p to get a general idea of each performance. For comparison's sake, the same density is fixed for all structures in this case, so that only n and p vary in turn; the scale-free structure imposes a common density of $2/p$. The default values are $n = 100$, $p = 20$.

Exp. 3: effect of the graph density on edge rankings. AUC measures are collected for variations of n and p with a density of $5/p$ (5 neighbors per node on average), and for variations of density parameters. The default values are $n = 100$, $p = 20$.

Illustrations

The first application deals with fish population measurements in the estuary of the Fatala River, Guinea, (Baran, 1995, available in the R package *ade4*). The data consists of 95 count samples of 33 fish species, and two covariates *date* and *site*. We infer the network using four models including no covariates, either one or both covariates (i.e. respectively the *null*, *site*, *date* and *site+date* models)

The second example is a metabarcoding experiment designed to study oak powdery mildew (Jakuschkin et al., 2016), caused by the fungal pathogen *Erysiphe alphitoides* (Ea). To study the pathobiome of oak leaves, measurements were done on three trees with different infection status. The resulting dataset is composed of 116 count samples of 114 fungal and bacterial operational taxonomic units (OTUs) of oak leaves, including the Ea agent. The original raw data are available at `https://www.ebi.ac.uk/ena/data/view/PRJEB7319`. Several covariates are available, among which the tree status, the orientation of the branch, and three covariates measuring the distances of oak leaves to the ground (D1), to the base of the branch (D2), and to the tree trunk (D3). The experiment used different depths of coverage for bacteria and fungi, which we account for via the offset term. We fitted three Poisson mixed models including either none, the tree status or all of the covariates (i.e. respectively *null*, *tree*, and *tree+D1+D2+D3* models).

To further analyze the inferred networks, we use the betweenness centrality (Freeman, 1978), a centrality measure popular in social network analysis. It measures a node's ability to act as a bridge in the network. High betweenness scores identify sensitive nodes that can efficiently describe a network structure. We compute these using the R package *igraph*.

2.3 Results

2.3.1 On simulated data

Effect of dataset dimensions

Behaviors are compared on an easy setting ($n = 100$, $p = 20$) and a hard setting ($n = 50$, $p = 30$). Fig. 2.4 displays FDR and density ratio measures for all methods on the different cases. Detailed values of medians and standard-deviations are given in Tables 2.3 and 2.4 in appendix. The behaviour of methods remains virtually the same across Erdös and Cluster structures. Scale-free structure appears to entail a greater difficulty for all methods with median FDR above 75%, except for EMtree which stays at 30% in the hard setting. These poor performance are due to the inferred networks being too dense compared to the original Scale-free graphs. Another experiment with the Scale-free structure is detailed in Fig. 2.5, where the number of species is fixed to 50, and the number of samples is 100 or 50. Under such setting the behavior of ecoCopula and MRFcov greatly improves with FDR at about 30%, where EMtree shows about 40% in median (detailed values available in Table 2.6 in appendix).

The greater difficulty affects all methods. Density ratios either increase (MInt, SpiecEasi) or decrease (gCoda, ecoCopula, MRFcov). In the first case, FDRs tend to increase as well (e.g. 40% increase for MInt in Erdös and Cluster structures), where a decrease in density ratio yields more empty results (e.g. in Table 2.5 25% of empty graphs for ecoCopula with Erdös and Cluster structures, 15% with Scale-free structures for p fixed to 50). EMtree seems to remain stable as for the density ratio, however it shows an increase in FDR measures of about 20% for all structures.

Considering FDRs and density ratios combined, EMtree appears to be the method with the lower FDR which maintains a density ratio reasonably close to 1. As a consequence, the proposed methodology compares well to existing tools on problems with varying difficulties. EMtree is also comparable on running times. Table 2.1 shows that for Erdös and Cluster it is the third quicker method in easy cases and the second in hard ones. Table 2.7 (in appendix) shows that on hard scale-free problems ($p = 30$ and $p = 50$ with $n = 50$) EMtree is the quicker method, and third otherwise.

Interestingly, in easy cases when the network density is well estimated, methods yield FDR of $10\% - 30\%$ in median. This reminds that network inference from abundance data is a difficult task, and that perfect inference of the network remains an out-of-reach goal.

Effect of network structure

As expected for a fixed p, the higher the number of observations n, the better the performance for all methods and structures. Interestingly, the same happens when p

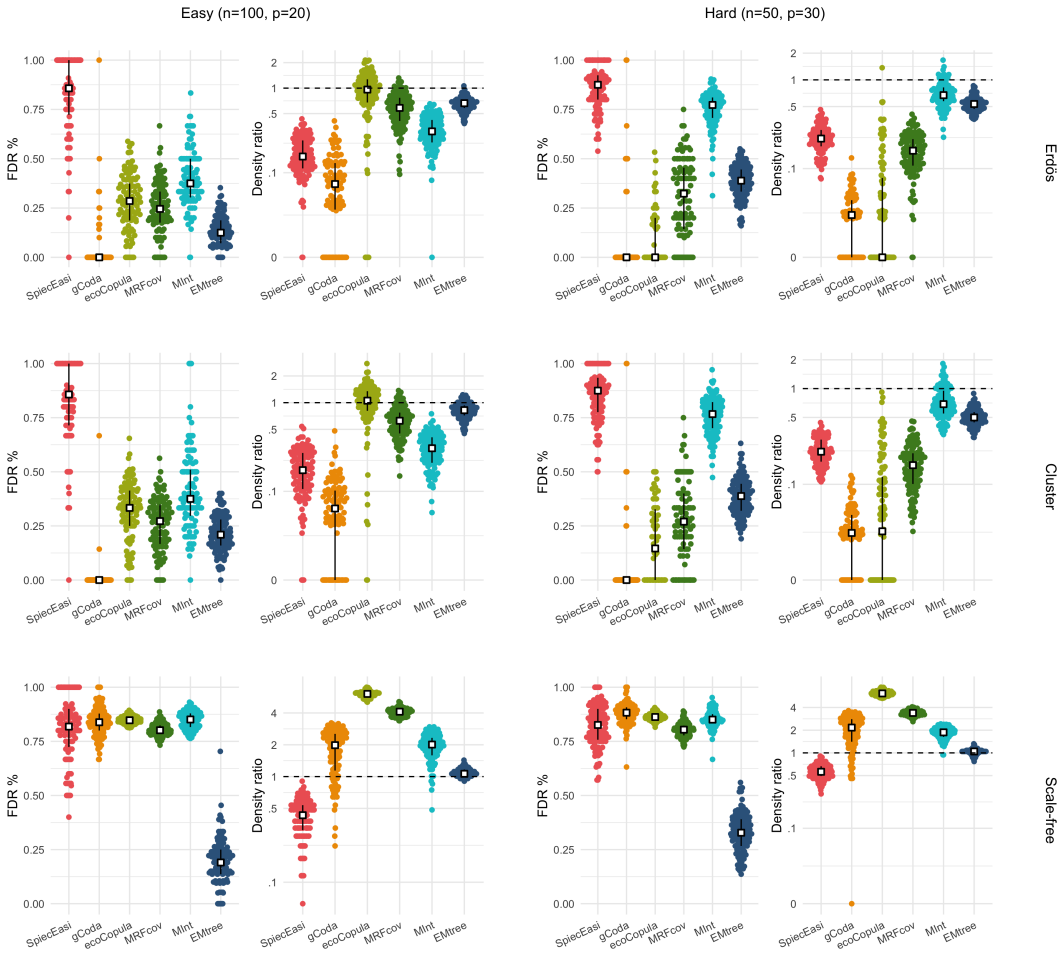


Figure 2.4 – FDR and density ratio measures for all methods at two different difficulty levels and 100 networks of each type. White squares and black plain lines represent medians and quartiles respectively. *ecoCopula selection method: AIC. Number of subsamples for SpiecEasi and EMtree:* $S = 20$. *SpiecEasi and gCoda: lambda.min.ratio* $= 0.001$, *nlambda* $= 100$.

	SpiecEasi	gCoda	ecoCopula	MRFcov	MInt	EMtree
Easy	19.99(4.19)	0.1(0.05)	4.2(0.24)	5.76(0.35)	54(26.9)	4.44(0.64)
Hard	24.29(5.07)	0.5(0.24)	8.19(0.16)	5.52(2.98)	33.87(18.37)	3.29(0.32)

Table 2.1 – Median and standard-deviation running-time values (in seconds) for Cluster and Erdös structures, including resampling with $S = 20$ for SpiecEasi and EMtree.

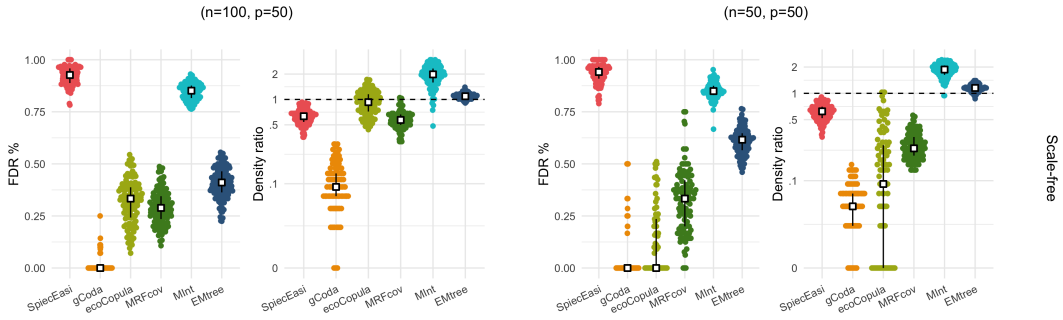


Figure 2.5 – FDR and density ratio measures for all methods on scale-free graphs with $p = 50$. White squares and black plain lines represent medians and quartiles respectively. *ecoCopula selection method: AIC. Number of subsamples for SpiecEasi and EMtree:* $S = 20$. *SpiecEasi and gCoda:* $lambda.min.ratio = 0.001$, $nlambda = 100$.

increases for a fixed $n = 100$ (except for SpiecEasi). EMtree performs well on Scale-free structures (Fig. 2.6) which was also expected; the other methods performance worsen compared to other structures. When lowering n to 30, EMtree performance deteriorates along with p, yet remaining above 70% in median in the extreme case where $p = n$ (Fig. 2.6, right). The structure being Erdös or Cluster, each method is affected in the same way by an increase of n or p (Fig. 2.7). Besides, increasing the difference between the two structures by tuning up the *ratio* parameter has no effect. Overall EMtree performs better than gCoda and SpiecEasi on all the studied configurations. Running times are summarized in Table 2.2. EMtree is about 10 times slower than gCoda (4 for small n), and 4 times faster than SpiecEasi. The high standard deviation for small n seems to be due to gCoda struggling with Scale-free structures.

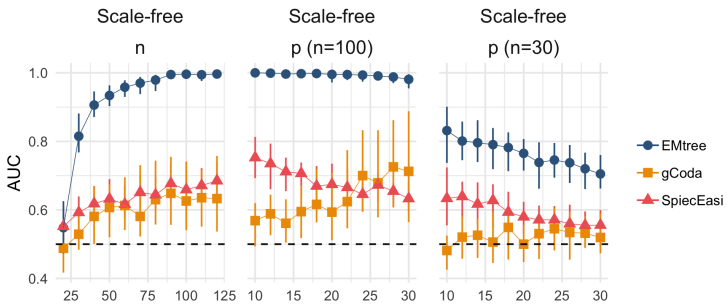


Figure 2.6 – Effect of Scale-free structure on AUC medians and inter-quartile intervals for parameters n and p.

	$n < 50$	$n \geq 50$	$p < 20$	$p \geq 20$
EMtree	0.44 (0.14)	0.60 (0.17)	0.41 (0.13)	0.76 (0.21)
gCoda	0.11 (26.8)	0.05 (0.05)	0.05 (0.04)	0.09 (0.54)
SpiecEasi	2.09 (0.26)	2.37 (0.28)	2.42 (0.27)	2.42 (0.26)

Table 2.2 – Median and standard-deviation of running times for each method in seconds, for n and p parameters.

Effect of network density

The comparison of top and bottom panels of Fig. 2.7 shows that network inference gets harder as the network gets denser, whatever the method and the structure of the true graph. Running times are not affected (Table 2.9). Fig. 2.8 shows that EMtree performance does not deteriorate faster than that of other methods, demonstrating that the tree hypothesis is not a limitation.

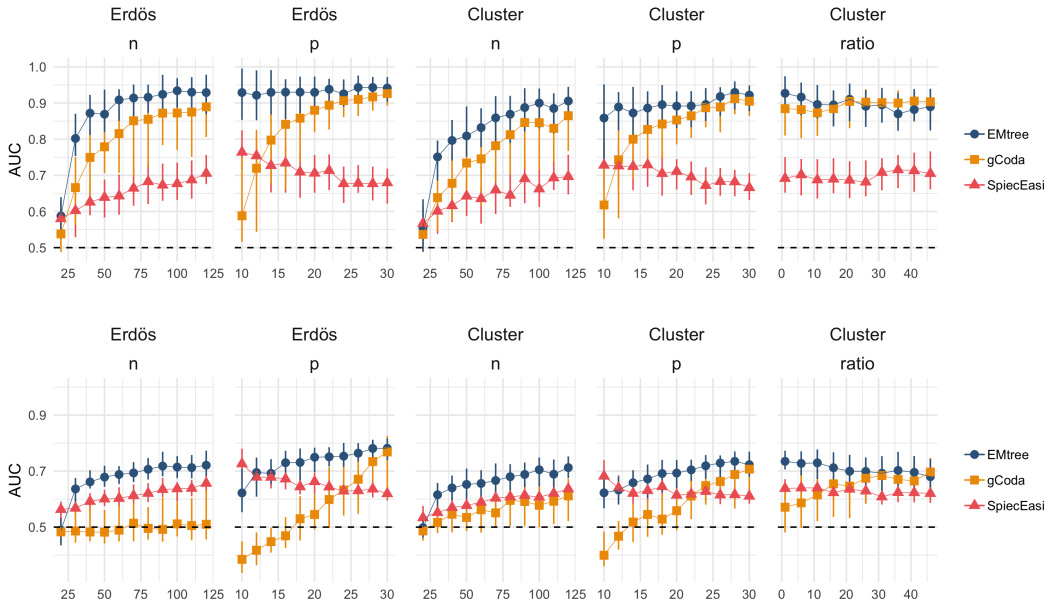


Figure 2.7 – Effect of Erdös and Cluster structures on AUC medians and inter-quartile intervals for parameters n, p and *ratio*. *Top*: densities set to $2/p$, *bottom*: densities set to $5/p$.

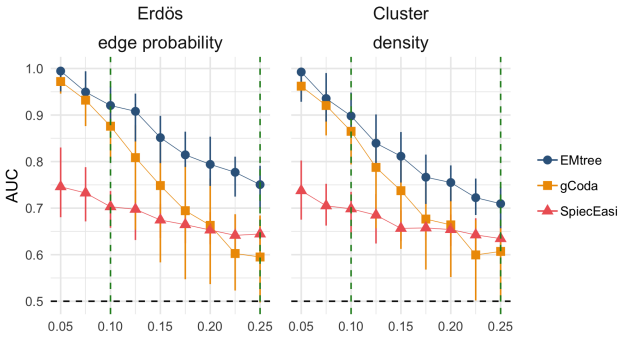


Figure 2.8 – AUC median and inter-quartile intervals for parameters controlling the number of edges in Erdös (*edge probability*) and in Cluster (*density*) structures, $p = 20$, $n = 100$. The two vertical dotted lines are the $3/p$ and $5/p$ values.

2.3.2 Illustrations

In this section we emphasize the importance of covariates for network inference. Accounting for environmental effects changes the structure of all inferred networks we present; nodes with the highest betweenness scores are highlighted to spot these changes. Most frequently, it results in reducing the number of edges (i.e. making the network sparser). However new edges can appear as well, as adjusting for a covariate also reduces the variability, which improves the detection power. In all examples, we used the resampling method described in Section 2.2.2, which provides edge selection frequencies. Eventually, we have to threshold these frequencies to draw actual networks; the value of the threshold obviously affects the density of the plotted networks (see Fig. 2.13).

Fish populations in the Fatala River estuary

Networks on Fig. 2.9 suggest a predominant role of the *site* covariate compared to the *date*. Indeed, adjusting for the *site* results in much sparser networks (Fig. 2.13 in appendix). It deeply modifies the network structure: the *site* network has 12 new edges and only 6 in common with the *null* network. Besides, the highlighted nodes only change when introducing the *site* covariate. This suggests that the environmental heterogeneity between the sites has a major effect on the variations of species abundances, while the effect of the date of sampling is moderate.

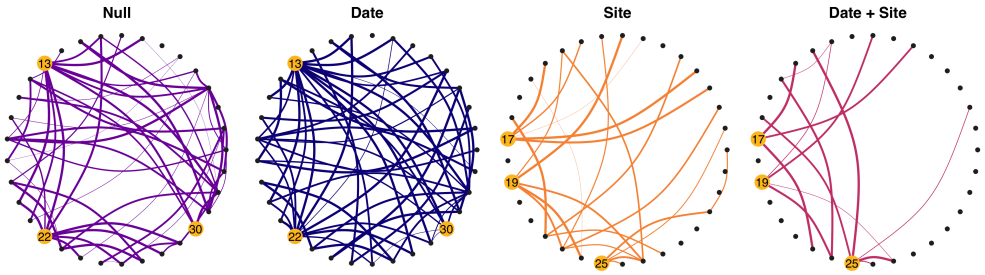


Figure 2.9 – Interaction networks of Fatala River fishes inferred when adjusting for none, both or either one of the covariates among *site* and *date*. Highlighted nodes spot the highest betweenness centrality scores. Widths are proportional to selection frequencies. $S = 100$, $f' = 90\%$.

Oak powdery mildew

When providing the inference with more information (tree status, distances), the structure of the resulting network is significantly modified. Nodes with high betweenness scores differ from one model to another. There is an important gap in density between the *null* model and the others, starting from a 25% selection threshold (Fig. 2.13 in appendix). From a more biological point of view, the features of the pathogen node are greatly modified too: its betweenness score is among the smallest in the *null* network (quantile 16%), and among the highest in the two other networks (quantiles 93% and 96%). Its connections to the other nodes vary as well. Accounting for covariates results in less interactions with the pathogen but a greater role of the latter in the pathobiome organization.

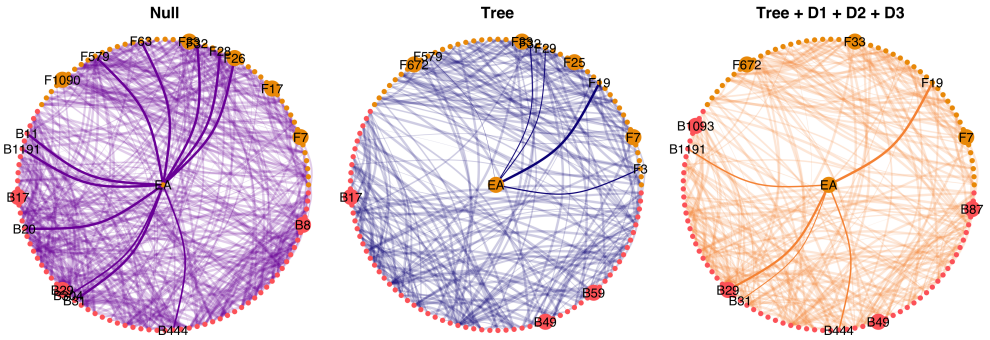


Figure 2.10 – Pathogen interaction networks on oak leaves inferred with EMtree when adjusting for none, the *tree* covariate or *tree* and distances. Bigger nodes represent OTUs with highest betweenness values, colors differentiate fungal and bacterial OTUs. Widths are proportional to selection frequencies. $S = 100$, $f' = 90\%$.

Using the dataset restricted to infected samples (39 observations for 114 OTUs) and correcting for the leaves position in the tree (proxy for their abiotic environment), Jakuschkin et al. (2016) identifies a list of 26 OTUs likely to be directly interacting with the pathogen. Running EMtree on the same restricted dataset with the same correction yields a good concordance with edge selection frequencies, as shown in Fig. 2.11.

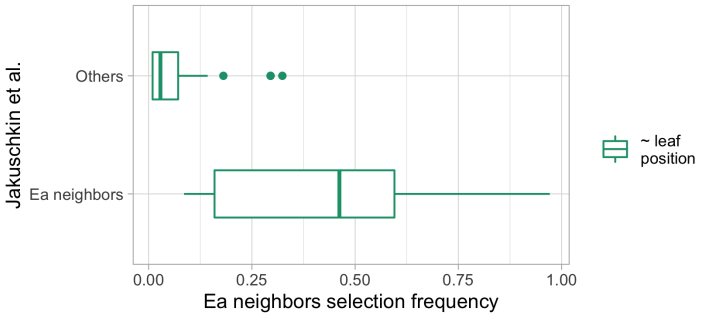


Figure 2.11 – EMtree selection frequencies of pathogen neighbors compared to Jakuschkin et al. (2016) results, computed on infected samples and adjusting for the leaf position (100 subs-samples).

2.4 Discussion

The inference of species interaction network is a challenging task, for which a series of methods have been proposed in the past years. Abundance data seems to be a promising source of information for this purpose. Here we adopt the formalism of graphical models to define a probabilistic model-based framework for the inference of such networks from abundance data. Using a model-based approach offers several important advantages. First, it enables easy and explicit integration of environmental and experimental effects. These could be modeled in a more flexible way using generalized additive models, which include non-linear effects (Hastie, 2017). Then, as it also relies on a formal statistical definition of a *species interaction network* in the context of graphical models, accounting for abiotic effects and modeling species interactions are two clearly defined and distinguished goals. Finally, all the underlying assumptions are explicitly stated in the model definition itself, and can therefore be discussed and criticized.

We developed an efficient method to infer sparse networks, which combines a multivariate Poisson mixed model for the joint distribution of abundances, with an averaging over all spanning trees to efficiently infer direct species interactions. As we do consider a mixture over all spanning trees, our approach remains flexible and can infer most types of statistical dependencies. An EM algorithm (EMtree) max-

imizes the likelihood of the result and returns each edge probability to be part of the network. An optional resampling step increases network robustness.
A simulation study in a heterogeneous environment demonstrates that EMtree compares very well to alternative approaches. The proposed model can take all kind of covariates into account, which when ignored can have dramatic effects on the inferred network structure, as showed here on empirical datasets. Experiments on simulated data and illustrations also demonstrate that EMtree computational cost remains very reasonable.
Alternative methods used in this work all rely on an optimized threshold to tell an edge presence. This particular threshold is obtained after testing a grid of possible values which all yield a different network, and altogether build a path. Making this path available to the user is useful, as the final threshold might need modification and it gives the possibility to build edges scores and get more than a binary result. We found few recent approaches doing this, which prevented us to study their performance in a way that did not impose a threshold.

The proposed methodology could be extended in several ways. Species abundances and interactions indeed vary across space, and depend on local conditions (Poisot et al., 2012, 2015). This can either be considered as nuisance parameter or as feature of interest. In the first case, the method could be extended to account for the spatial autocorrelation of sampling sites, to obtain a "regional" interaction network corrected for this effect, i.e. assuming the network is the same in all sites. If of interest, variation across space and local conditions could be studied by comparing networks inferred from the different sampling locations. Networks comparison is a wide and interesting question and tools lack to check which edges are shared by a set of networks. The approach introduced by Schwaller and Robin (2017) could be adapted to EMtree framework. Lastly, It is also very likely that not all covariates nor even all species have been measured or observed. Another extension may therefore be to detect ignored covariates or missing species. To this purpose EMtree could probably be combined with the approach developed by Robin et al. (2019) to identify missing actors.

Appendices

2.A Supplementary material (augmented)

2.A.1 Variational EM in the observed layer

A reminder on EM and VEM. Expectation-Maximisation (EM: Dempster et al., 1977) has become the standard algorithm for the maximum likelihood inference of latent variable models. Denoting $\boldsymbol{\gamma}$ the unknown parameter, $\mathbf{Y}$ the observed variables and $\mathbf{H}$ the latent variables, the aim of EM is to maximise the *observed* (log-)likelihood $\log p_{\boldsymbol{\gamma}}(\mathbf{Y})$. In the model defined in Section 2.2.1, the set of parameter to estimate is $\boldsymbol{\gamma} = (\boldsymbol{\beta}, \boldsymbol{\Sigma}, \boldsymbol{\theta})$ and the latent variables are $\mathbf{H} = (\mathbf{Z}, T)$. Because the *complete* (log-)likelihood $\log p_{\boldsymbol{\gamma}}(\mathbf{Y}, \mathbf{H})$ is often much easier to handle, EM alternatively evaluates the conditional distribution of the latent variables $p_{\boldsymbol{\gamma}}(\mathbf{H} \mid \mathbf{Y})$ (E step) and updates the parameter estimates by maximizing the conditional expectation of the complete log-likelihood (M step).
Unfortunately, for many models, the conditional distribution $p_{\boldsymbol{\gamma}}(\mathbf{H}|\mathbf{Y})$ is intractable. The variational EM (VEM) algorithm has been designed to deal with such cases. Briefly speaking, the E step (during which the intractable conditional distribution should be evaluated) is replaced with a VE step, during which an approximate distribution $\widetilde{p}(\mathbf{H}) \simeq p_{\boldsymbol{\gamma}}(\mathbf{H} \mid \mathbf{Y})$ is determined. Actually, the VEM algorithm maximizes a lower bound of the genuine log-likelihood, similar to this given in Eq. (2.6) (see Blei et al., 2017; Ormerod and Wand, 2010, for an introduction).

Application to the Poisson log-normal model. To estimate the fixed regression parameters gathered in $\boldsymbol{\theta}$, we resort to a surrogate model where the entries of the abundance matrix $\mathbf{Y}$ still have the conditional distribution given in Eq. (2.1), but where the distribution of the $\mathbf{Z}_i$ is not constrained to be faithful to a specific graphical model. Namely, the latent vectors $\mathbf{Z}_i$ are only supposed to be independent and identically distributed (iid) Gaussian with distribution $\mathcal{N}(0, \boldsymbol{\Sigma})$, without any restriction on $\boldsymbol{\Sigma}$.
This surrogate model is actually a Poisson log-normal model as introduced by Aitchison and Ho (1989), the parameters of which can be estimated using a variational approximation similar to this introduced in Chiquet et al. (2018). More specifically, we maximize with respect to the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\Sigma}$ the following lower bound of the log-likelihood $\log p(\mathbf{Y})$:

$$\mathcal{J}(\mathbf{Y}; \boldsymbol{\theta}, \boldsymbol{\Sigma}, \widetilde{p}) := \log p_{\boldsymbol{\theta},\boldsymbol{\Sigma}}(\mathbf{Y}) - KL\left(\widetilde{p}(\mathbf{Z}) || p_{\boldsymbol{\theta},\boldsymbol{\Sigma}}(\mathbf{Z} \mid \mathbf{Y})\right), \tag{2.6}$$

where $KL(q||p)$ stands for Küllback-Leibler divergence between distributions q and

p and where the approximate distribution $\widetilde{p}(\mathbf{Z})$ is chosen to be Gaussian. This means that each conditional distribution $p(\mathbf{Z}_i \mid \mathbf{Y}_i)$ is approximated with a normal distribution $\mathcal{N}(\widetilde{\mathbf{m}}_i, \widetilde{\mathbf{S}}_i)$. As shown in Chiquet et al. (2018), $\mathcal{J}(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\Sigma}, \widetilde{p})$ is bi-concave in $(\boldsymbol{\theta}, \boldsymbol{\Sigma})$ and $\{(\widetilde{\mathbf{m}}_i, \widetilde{\mathbf{S}}_i)_i\}$, so that gradient ascent can be used. The `PLNmodels` R-package –available on CRAN– provides an efficient implementation of it.
The entries of the $\widetilde{\mathbf{m}}_i$ and $\widetilde{\mathbf{S}}_i$ provide us with approximations of the conditional expectation, variance and covariance of the Z_{ij} conditionally on the $\mathbf{Y}$, which we use to get the estimates $\widehat{\sigma}_j^2$ and $\widehat{\rho}_{jk}$ given in Eq. (2.8). More specifically, we use $\mathbb{E}(Z_{ij} \mid \mathbf{Y}_i) \simeq \widetilde{m}_{ij}$, $\mathbb{E}(Z_{ij}^2 \mid \mathbf{Y}_i) \simeq \widetilde{m}_{ij}^2 + \widetilde{S}_{i,jj}$ and $\mathbb{E}(Z_{ij} Z_{ik} \mid \mathbf{Y}_i) \simeq \widetilde{m}_{ij}\widetilde{m}_{ik} + \widetilde{S}_{i,jk}$.

2.A.2 EM in the latent layer

Complete log-likelihood conditional expectation

Because of the specific form given in Eq. (2.3), and because the $\mathbf{Z}_i \mid T$ are Gaussian, we have that

$$\begin{aligned}
\log p_{\boldsymbol{\Sigma}}(\mathbf{Z} \mid T) &= \sum_{j=1}^{p}\sum_{i=1}^{n} \log P(Z_{ij} \mid T) + \sum_{jk \in T}\sum_{i=1}^{n} \log\left(\frac{P(Z_{ij}, Z_{ik})}{P(Z_{ij})P(Z_{ik})}\right) \\
&= -\frac{n}{2}\log\sigma_j^2 - \frac{1}{2}\sum_{j=1}^{p}\sum_{i=1}^{n}\frac{Z_{ij}^2}{\sigma_j^2} - \frac{n}{2}\sum_{jk\in T}\log(1-\rho_{jk}^2) \qquad (2.7)\\
&\quad - \frac{1}{2}\sum_{jk\in T}\frac{1}{1-\rho_{jk}^2}\sum_{i=1}^{n}\left(\rho_{jk}^2\frac{Z_{ij}^2}{\sigma_j^2} + \rho_{jk}^2\frac{Z_{ik}^2}{\sigma_k^2} - 2\rho_{jk}\frac{Z_{ij}Z_{ik}}{\sigma_j\sigma_k}\right) + \text{cst}
\end{aligned}$$

where the constant term does not depend on any unknown parameter. In the M step of the EM algorithm, we have to maximize the conditional expectation of Eq. (2.7) with respect to the variances σ_j^2 and the correlation coefficients ρ_{jk}. The resulting estimates take the usual forms, but with the conditional moments of the Z_{ij}, that is

$$\widehat{\sigma}_j^2 = \frac{1}{n}\sum_i \mathbb{E}(Z_{ij}^2 \mid \mathbf{Y}), \qquad \widehat{\rho}_{jk} = \frac{1}{n}\sum_i \mathbb{E}(Z_{ij}Z_{ik} \mid \mathbf{Y}) \,/\, (\widehat{\sigma}_j\widehat{\sigma}_k). \qquad (2.8)$$

which do not depend on T. The maximized conditional expectation of Eq. (2.7) becomes

$$\mathbb{E}\left(\log p_{\widehat{\boldsymbol{\Sigma}}}(\mathbf{Z} \mid T) \mid \mathbf{Y}\right) = -\frac{n}{2}\log\widehat{\sigma}_j^2 - \frac{n}{2}\sum_{jk\in T}\log(1-\widehat{\rho}_{jk}^2) + \text{cst}. \qquad (2.9)$$

We are left with the writing of the conditional expectation of the first two terms of the logarithm of Eq. (2.4), once optimized in $\boldsymbol{\Sigma}$. Combining Eq. (2.2) and Eq. (2.9), and noticing that the probability for an edge to be part of the graph is the sum

of the probability of all the trees than contain this edge, we get (denoting $\widehat{\psi}_{jk} = (1-\widehat{\rho}_{jk}^2)^{-n/2}$)

$$\begin{aligned}\mathbb{E}\left(\log p_{\boldsymbol{\beta}}(T) + \log p_{\widehat{\boldsymbol{\Sigma}}}(\mathbf{Z} \mid T) \mid \mathbf{Y}\right) &= \sum_{T\in\mathcal{T}} p(T \mid \mathbf{Y}) \left(\log p_{\boldsymbol{\beta}}(T) + \log p_{\widehat{\boldsymbol{\Sigma}}}(\mathbf{Z} \mid T)\right) \\ &= -\log B + \sum_{T\in\mathcal{T}} p(T \mid \mathbf{Y}) \sum_{jk\in T} \left(\log \beta_{jk} + \log \widehat{\psi}_{jk}\right) + \text{cst} \\ &= -\log B + \sum_{(j,k)} \mathbb{P}\{jk \in T \mid \mathbf{Y}\} \left(\log \beta_{jk} + \log \widehat{\psi}_{jk}\right) + \text{cst},\end{aligned}$$

which gives Eq. (2.5).

As explained in the section above, we approximate expectations and probabilities conditional on $\mathbf{Y}$ by their variational approximation. This provides us with the approximate conditional distribution of the tree T given the data $\mathbf{Y}$:

$$\widetilde{p}(T \mid \mathbf{Y}) = \prod_{jk\in T} \beta_{jk}\widehat{\psi}_{jk} \Big/ C,$$

where C is the normalizing constant: $C = \sum_T \prod_{jk\in T} \beta_{jk}\widehat{\psi}_{jk}$. The intuition behind this approximation is the following: according to Eq. (2.2), the marginal probability a tree T is proportional to the product of the weights β_{jk} of its edges. The conditional distribution probability of tree is proportional to the same product, the weights β_{jk} being updated as $\beta_{jk}\widehat{\psi}_{jk}$, where $\widehat{\psi}_{jk}$ summarizes the information brought by the data about the edge (j,k).

Steps E and M

E step: From the above computation we get the following approximation:

$$\mathbb{P}\{jk \in T \mid \mathbf{Y}\} \simeq 1 - \sum_{T: jk\notin T} \widetilde{p}(T \mid \mathbf{Y}),$$

and so we define p_{jk} as follows:

$$P_{jk} = 1 - \frac{\sum_{T:jk\notin T} \prod_{jk\in T} \beta_{jk}\widehat{\psi}_{jk}}{\sum_T \prod_{jk\in T} \beta_{jk}\widehat{\psi}_{jk}}.$$

P_{jk} can be computed with Theorem 2.1, letting $[\mathbf{W}^h]_{jk} = \beta_{jk}^h\widehat{\psi}_{jk}$ and $\mathbf{W}^h_{\setminus jk} = \mathbf{W}^h$ except for the entries (j,k) and (k,j) which are set to 0. The modification of $\mathbf{W}^h_{\setminus jk}$ with respect to $\mathbf{W}^h$ amounts to set to zero the weight product, and

so the probability, for any tree T containing the edge (j,k). As a consequence, we get

$$P_{jk}^{h+1} = 1 - \left|Q_{uv}^*(\mathbf{W}_{\setminus jk}^h)\right| / \left|Q_{uv}^*(\mathbf{W}^h)\right| .$$

M step: Applying Lemma 2.1 to the weight matrix $\boldsymbol{\beta}$, the derivative of B with respect to β_{jk} is

$$\partial_{\beta_{jk}} B = [\mathbf{M}(\boldsymbol{\beta})]_{jk} \times B$$

then the derivative of (2.5) with respect to β_{jk} is null for $\beta_{jk}^{h+1} = P_{jk}^{h+1} \Big/ [\mathbf{M}(\boldsymbol{\beta}^h)]_{jk}$.

2.A.3 Matrix tree theorem

For any matrix $\mathbf{W}$, we denote its entry in row u and column v by $[\mathbf{W}]_{uv}$. We define the Laplacian matrix $\mathbf{Q}$ of a symmetric matrix $\mathbf{W} = [w_{jk}]_{1 \le j,k \le p}$ as follows :

$$[\mathbf{Q}]_{jk} = \begin{cases} -w_{jk} & 1 \le j < k \le p \\ \sum_{u=1}^{p} w_{ju} & 1 \le j = k \le p. \end{cases}$$

We further denote $\mathbf{W}^{uv}$ the matrix $\mathbf{W}$ deprived from its uth row and vth column and we remind that the (u,v)-minor of $\mathbf{W}$ is the determinant of this deprived matrix, that is $|\mathbf{W}^{uv}|$.

Theorem 2.1 (Matrix Tree Theorem Chaiken and Kleitman (1978); Meilă and Jaakkola (2006))**.** *For any symmetric weight matrix W, the sum over all spanning trees of the product of the weights of their edges is equal to any minor of its Laplacian. That is, for any $1 \le u, v \le p$,*

$$W := \sum_{T \in \mathcal{T}} \prod_{(j,k) \in T} w_{jk} = |\mathbf{Q}^{uv}|.$$

In the following, without loss of generality, we will choose $\mathbf{Q}^{pp}$. As an extension of this result, Meilă and Jaakkola (2006) provide a close form expression for the derivative of W with respect to each entry of $\mathbf{W}$.

Lemma 2.1 (Meilă and Jaakkola (2006))**.** *Define the entries of the symmetric matrix $\mathbf{M}$ as*

$$[\mathbf{M}]_{jk} = \begin{cases} \left[(\mathbf{Q}^{pp})^{-1}\right]_{jj} + \left[(\mathbf{Q}^{pp})^{-1}\right]_{kk} - 2\left[(\mathbf{Q}^{pp})^{-1}\right]_{jk} & 1 \le j < k < p \\ \left[(\mathbf{Q}^{pp})^{-1}\right]_{jj} & k = p, 1 \le j \le p \\ 0 & 1 \le j = k \le p. \end{cases}$$

it holds that

$$\partial_{w_{jk}} W = [\mathbf{M}]_{jk} \times W.$$

2.A.4 Results

Simulations: dataset dimensions

		SpiecEasi	gCoda	ecoCopula	MRFcov	MInt	EMtree
Easy	Cluster	0.86 (0.20)	0 (0.08)	0.33 (0.14)	0.27 (0.13)	0.38 (0.17)	0.12 (0.09)
	Erdös	0.86 (0.21)	0 (0.15)	0.29 (0.15)	0.25 (0.14)	0.38 (0.15)	0.12 (0.08)
	Scale-free	0.82 (0.15)	0.84 (0.07)	0.85 (0.02)	0.8 (0.03)	0.85 (0.04)	0.19 (0.11)
Hard	Cluster	0.88 (0.12)	0 (0.2)	0.15 (0.18)	0.27 (0.19)	0.77 (0.09)	0.39 (0.09)
	Erdös.	0.88 (0.11)	0 (0.24)	0 (0.15)	0.32 (0.2)	0.77 (0.1)	0.39 (0.09)
	Scale-free	0.83 (0.1)	0.88 (0.05)	0.86 (0.02)	0.8 (0.03)	0.85 (0.04)	0.33 (0.09)

Table 2.3 – Medians and standard-deviation of FDR computed on 100 graphs of each type (*easy*: $n = 100, p = 20$, *hard*: $n = 50, p = 30$)

		SpiecEasi	gCoda	ecoCopula	MRFcov	MInt	EMtree
Easy	Cluster	0.16 (0.11)	0.05 (0.07)	1.04 (0.48)	0.62 (0.25)	0.3 (0.13)	0.81 (0.17)
	Erdös	0.15 (0.09)	0.06 (0.08)	0.95 (0.5)	0.57 (0.26)	0.3 (0.14)	0.65 (0.12)
	Scale-free	0.42 (0.17)	1.97 (0.82)	6.05 (0.4)	4.11 (0.34)	2(0.54)	1.05 (0.1)
Hard	Cluster	0.21 (0.08)	0.02 (0.03)	0.02 (0.17)	0.15 (0.09)	0.68 (0.3)	0.49 (0.11)
	Erdös	0.21 (0.08)	0.02 (0.02)	0 (0.18)	0.15 (0.08)	0.66 (0.25)	0.52 (0.1)
	Scale-free	0.55 (0.12)	2.16 (0.88)	6.14 (0.47)	3.38 (0.29)	1.86 (0.32)	1.03 (0.09)

Table 2.4 – Medians and standard-deviation of density ratio computed on 100 graphs of each type (*easy*: $n = 100, p = 20$, *hard*: $n = 50, p = 30$)

		SpiecEasi	gCoda	ecoCopula	MRFcov	MInt	EMtree
Easy	Cluster	1.77	13.89	1.74	0	0	0
	Erdös	0.68	11.95	0.99	0	0.83	0
	Scale-free	0	0	0	0	0	0
Hard	Cluster	0	14.05	23.40	0	0	0
	Erdös	0	20.85	27.28	0.60	0	0
	Scale-free	0	0.43	0	0	0	0

Table 2.5 – Percentage of empty networks computed on 100 graphs of each type (*easy*: $n = 100, p = 20$, *hard*: $n = 50, p = 30$)

n	Criteria (%)	SpiecEasi	gCoda	ecoCopula	MRFcov	MInt	EMtree
100	FDR	0.93(0.04)	0(0.04)	0.33(0.11)	0.29(0.08)	0.85(0.04)	0.41(0.08)
	density ratio	0.62(0.13)	0.08(0.07)	0.92(0.3)	0.56(0.14)	1.97(0.54)	1.08(0.08)
	empty graphs	0	1.88	0	0	0	0
50	FDR	0.94(0.05)	0(0.13)	0(0.16)	0.33(0.16)	0.85(0.04)	0.62(0.06)
	density ratio	0.61(0.13)	0.04(0.03)	0.08(0.24)	0.22(0.1)	1.86(0.32)	1.14(0.11)
	empty graphs	0	5.97	15.46	0	0	0

Table 2.6 – Medians and standard-deviation of FDR and density ratio critera, as well as percentage of empty networks computed on 100 scale-free graphs with $p = 50$ nodes and $n = 100$ or $n = 50$ samples.

n	p	SpiecEasi	gCoda	ecoCopula	MRFcov	MInt	EMtree
100	20	15.92(0.05)	0.34(0.79)	4.84(0.24)	5.62(0.79)	66.83(36.03)	5.06(0.74)
50	30	16.3(0.06)	21.93(32.44)	11.27(0.84)	5.97(1.45)	73.73(31.57)	3.71(1.11)
100	50	20.48(1.4)	1.07(0.24)	24.99(0.32)	15.53(0.12)	50.51(28.52)	20.54(2.42)
50	50	20.01(2.76)	44.21(14.78)	32.75(1.39)	27.32(1.8)	55.96(23.26)	11.54(0.66)

Table 2.7 – Median and standard-deviation of running times in seconds of methods for the inference of scale-free structures with different values of the number of samples n and of the number of species p.

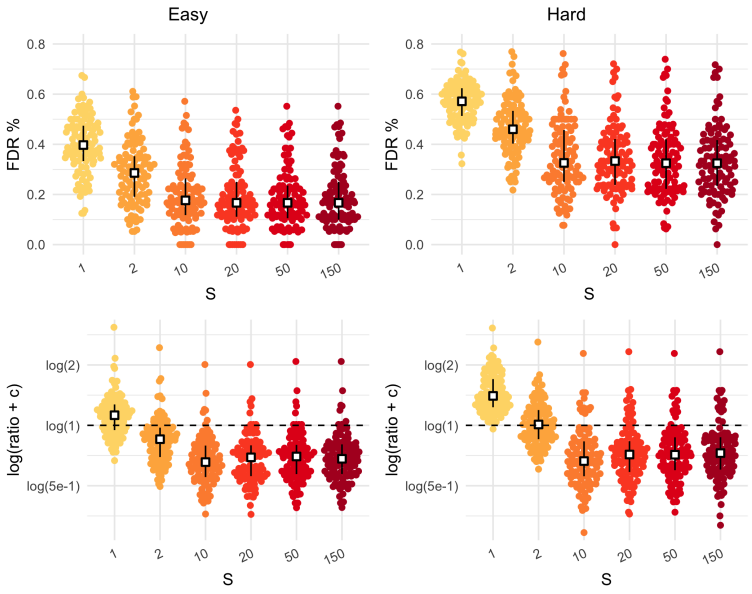


Figure 2.12 – FDR and density ratio measures of EMtree with varying values of number of sub-samples S (Erdös structure).

S	1	2	10	20	50	150
Easy	0.66 (0.15)	1.86 (0.23)	7.00 (0.81)	12.29 (1.27)	29.50 (3.39)	87.30 (10.36)
Hard	0.45 (0.12)	1.44 (0.14)	5.06 (0.78)	8.97 (0.87)	23.35 (2.40)	69.29 (10.83)

Table 2.8 – Median and standard-deviation running-time values in seconds of EMtree with different values of the number of sub-samples S for the inference of Erdös structures.

Simulations: network density

	$n < 50$	$n \geq 50$	$p < 20$	$p \geq 20$
EMtree	0.41 (0.11)	0.60 (0.15)	0.38 (0.12)	0.71 (0.21)
gCoda	0.12 (0.47)	0.07 (0.03)	0.05 (0.03)	0.09 (0.06)
SpiecEasi	2.41 (0.25)	2.41 (0.25)	2.39 (0.25)	2.42 (0.25)

Table 2.9 – Median and standard-deviation of running times for each method in seconds, for n and p parameters. corresponding to Erdös and cluster structures with $5/p$ densities.

Illustrations: edge frequency threshold

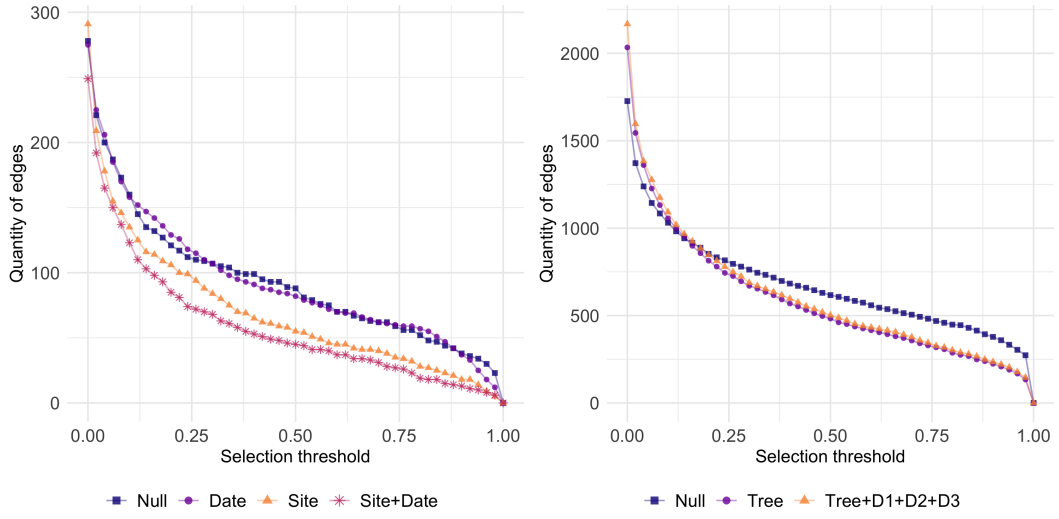


Figure 2.13 – Quantity of selected edges as a function of the selection threshold (*left*: Fatala fishes, *right*: oak mildew.)

The curves displayed on Fig. 2.13 are very smooth, which illustrates the difficulty of setting this threshold.

2.B Vignette for EMtree

EMtree implements an EM algorithm for the inference of interaction networks from abundance data. It uses averages over spanning trees within a Poisson log-normal model. In addition to functions performing the network inference, this package includes functions for count data simulation under the Poisson log-normal model which Gaussian layer of parameters is faithful to a desired graph structure. EMtree also provides with network visualization functions.

2.B.1 Data simulation with EMtree

The EMtree package provides with simulation functions for graphs as well as for count data under the Poisson log-Normal model. Four types of graphs are available in the `generator_graph()` function: erdos (Erdös-Reyni), cluster, scale-free (from the `huge` package) and spanning tree (from the `vegan` package).

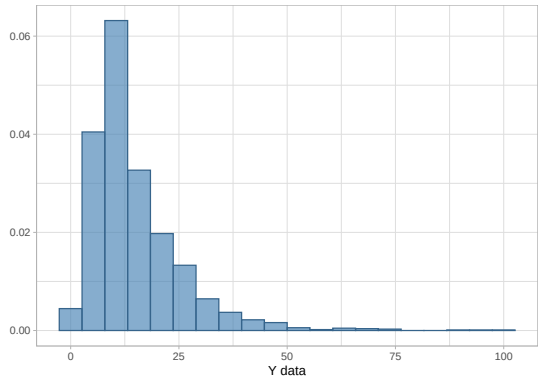
```
p=20
cluster=generator_graph(p, graph="cluster",dens=0.4, r=10)
erdos=generator_graph(p, graph="erdos",dens=0.3)
scaleF=generator_graph(p, graph="scale-free")
tree=generator_graph(p, graph="tree")
```

The output of `generator_graph()` is an adjacency matrix, which can be given as an input to `generator_param()` to define a positive-definite precision matrix `omega` with signed entries, and its corresponding variance-covariance matrix `sigma`.

```
cluster_param=generator_param(cluster, signed = TRUE)
```

Then it is possible to simulate `n` samples of multivariate counts under the PLN model with `generator_PLN()`. Here we simulate Y with the covariate matrix X which includes an intercept and $x_1 \sim \mathcal{N}(0, 0.5)$.

```
n = 100
X = data.frame(int=1,x1=rnorm(n, 0,0.5))
Y = generator_PLN(cluster_param$sigma,covariates = X, n = n)
```



2.B.2 Inference of the Fatala fishes network

This is a basic example detailing how to infer a network, using fishes counts from the Fatala River available in the Barans95 dataset (`ade4` package).

Fatala fishes dataset

The data is composed of 33 species abundances measures in 95 samples. The available covariates are the site and date of the samples.

```
library(ade4)
library(tibble)
data(baran95)
Y = as.matrix(baran95$fau)
X = as_tibble(baran95$plan)
n = nrow(Y)
p = ncol(Y)
```

Network inference

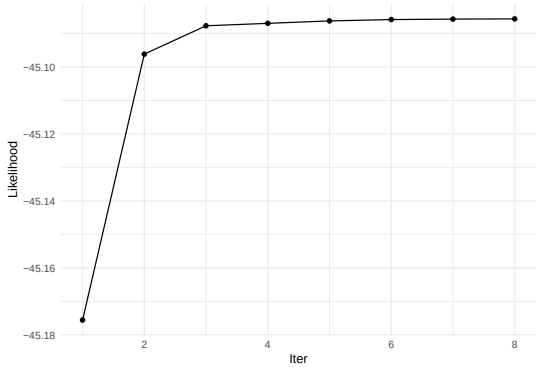
EMtree infers a network from either a correlation matrix of a multivariate Gaussian, or an object created by PLNmodels from count data. Here we first create a `PLNmodels` object with the `PLN()` function:

```
library(PLNmodels)
PLNfit<-PLN(Y ~ X$site)
```

```
##
##  Initialization...
##  Adjusting a PLN model with full covariance model
##  Post-treatments...
##  DONE!
```

And then run `EMtree()`:

```
library(EMtree)
EMtreeFit<-EMtree(PLNfit,  maxIter = 20, plot=TRUE, verbatim=FALSE)
```




```
str(EMtreeFit)
```

```
## List of 6
##  $ edges_prob  : num [1:33, 1:33] 0 0.00696 0.02016 0.14686 0.03192 ...
##  $ edges_weight: num [1:33, 1:33] 0 0.000946 0.000946 0.000948 0.000947 ...
##  $ logpY       : num [1:8] -45.2 -45.1 -45.1 -45.1 -45.1 ...
##  $ maxIter     : num 8
##  $ norm.cst    : num 2.07e-50
##  $ timeEM      : 'difftime' num 0.353778123855591
##   ..- attr(*, "units")= chr "secs"
```

To get a network from a fit of `EMtree()`, the probabilities stored in `edges_prob` can be thresholded. We suggest the $2/p$ threshold, which is the probability of an edge in a tree with uniform weights:

```
probs<- EMtreeFit$edges_prob
net<-1*(probs>2/p)
```

To improve the robstness, the function `ResampleEMtree()` implements a statibility selection of EMtree on S sub-samples. This function uses parallel computations with `mclapply()`. The output `Pmat` gathers all the infered edges probabilities for each sub-sample.

```
ResampEmtreeFit<-ResampleEMtree(counts=Y, covar_matrix = X$site ,
S=10, maxIter=20,cond.tol=1e-8, cores=1)
```

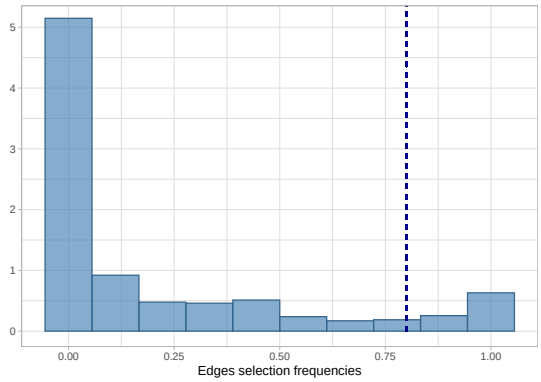
```
## Computing  10 probability matrices with 1 core(s)...  5.78 secs
```

```
str(ResampEmtreeFit$Pmat)
```

```
##  num [1:10, 1:528] 0.01361 0.01728 0.00562 0.00348 0.00364 ...
```

Edges selection frequencies can be derived from the `Pmat` output with the function `freq_selec()`. A final network can then be obtained by thresholding the frequencies, to keep for example edges that are selected in more than 80% of sub-samples:

```
freqs<-freq_selec(ResampEmtreeFit$Pmat,Pt=2/p) # thresh. probabilities
resampNet<-1*(freqs>0.8)# thresh. frequencies
```




```
table(net, resampNet)
```

```
##     resampNet
## net   0   1
##   0 873   0
##   1 112 104
```

Infer networks under several models:

The aim of function `ComparEMtree()` is to run network inference with different covariates specifications. It uses `ResampleEMtree()` and adjust the different models specified in `model_names` as follows:

```
tested_models=list(1,2,c(1,2))
models_names=c("date","site","date + site")
compare_models<-ComparEMtree(Y, X, models=tested_models,
m_names=models_names, Pt=2/p,  S=3, maxIter=5,cond.tol=1e-8,cores=1)
```

```
## model date : Computing  3 probability matrices with 1 core(s)...
##   3.68 secs
## model site : Computing  3 probability matrices with 1 core(s)...
```

```
##   1.5 secs
## model date + site : Computing  3 probability matrices with 1 core(s)...
##   3.95 secs
```

The output of `ComparEMtree()` is a tibble in long format, which gathers information of all interactions in all tested models.

```
head(compare_models,4)
```

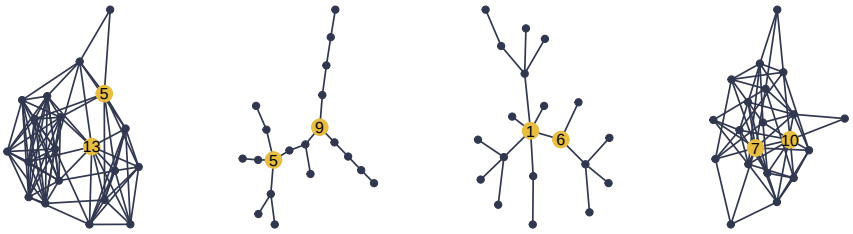
```
## # A tibble: 4 x 4
##   node1 node2 model weight
##   <chr> <chr> <chr>  <dbl>
## 1 1     2     date       0
## 2 1     3     date       0
## 3 2     3     date       0
## 4 1     4     date       0
```

2.B.3 Visualizations

The EMtree package provides with easy plotting functions for network visualizations. They build from the `ggraph` and `tidygraph` packages.

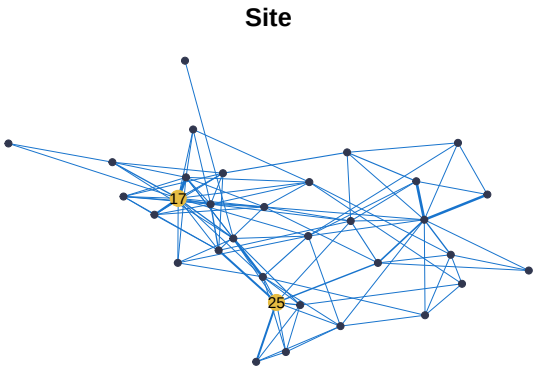
Simple networks

The function `draw_network()` takes a weighted matrix as input, and represents a network with edges widths proportional to the input weights. Several layouts are available (see the `ggraph` documentation). Nodes possessing among the highest betweenness centrality measure can be highlighted with the parameter `btw_rank`. Example from the adjacency matrices simulated earlier (cluster, spanning-tree, scale-free and erdos):



Weighted matrices can be edge probability matrices. For example, the Fatala fishes weighted network adjusted on the covariate *Site* is:

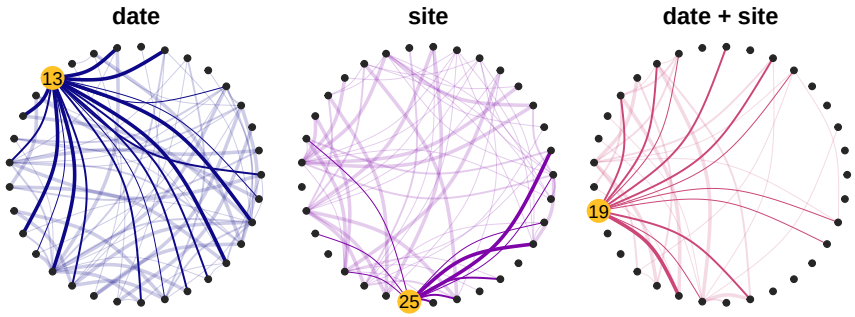
```
probs[probs<2/p]=0 # threshold needed for representation clarity
draw_network(probs,title="Site", pal_edges="dodgerblue3",
layout="nicely",btw_rank=3)$G
```



Several networks

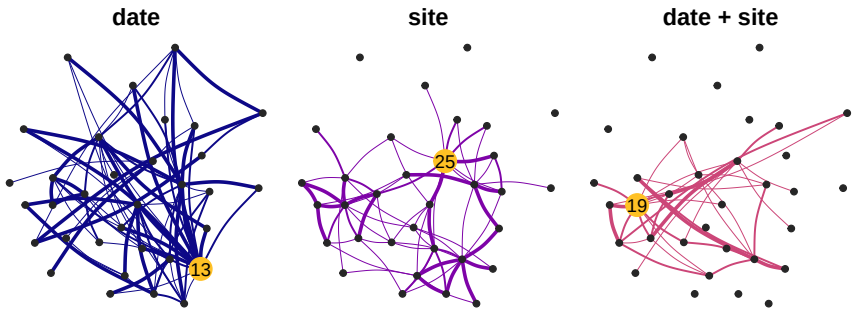
The function `compare_graphs()` draws a facet plot of the output networks from `ComparEMtree()`. Comparing network by eye is difficult, in particular choosing the right layout to do so is often troublesome. Here by default, the circle layout is used so that differences in density and sensitive nodes are easily identified.

```
compare_graphs(compare_models,shade=TRUE)$G
```



However, if another layout is preferred the nodes position is preserved along the facet and defined by choosing the `base_model`.

```
compare_graphs(compare_models,shade=FALSE, layout="nicely", curv=0.1,
base_model="site")$G
```



3

Inference with Missing Actor

Contents

This chapter details a Variational EM algorithm for the inference of missing actors in species interaction networks. The model is similar to the one presented in Chapter 2: counts are modeled with a Poisson log-normal distribution with a latent layer of Gaussian parameters, except here we use its normalized formulation. Therefore in Chapter 2 the Gaussian parameters are $\boldsymbol{Z} \sim \mathcal{N}(0, \Sigma)$, whereas in this chapter we use $\boldsymbol{U} \sim \mathcal{N}(0, \boldsymbol{R})$ with $\boldsymbol{R}$ a correlation matrix, and we have $\boldsymbol{Z}_{ij} = \sigma_j \boldsymbol{U}_{ij}$. The material of this chapter has been submitted to a statistical journal, and is available as a preprint at http://arxiv.org/abs/2007.14299, along with the supplementary

material of the appendix *Article supplements*. Other appendices supplement this work: a vignette detailing the usage of the developed R package nestor, a section presenting three strategies for its initialization, and finally a comparison study of nestor, EMtree and PLN-network on simulated data.

3.1 Introduction

Network inference. Network inference (or structure inference) has become a topical problem in various fields such as biology, ecology, neuro-sciences, social sciences, to name a few. The aim is to unravel the dependency structure that relates a series of variables that can be jointly observed. Graphical models (see e.g. Lauritzen, 1996) provide a natural framework to achieve this task as it allows to encode the dependency structure into a graph, the nodes of which are the variables. Two variables are connected if and only if they are dependent, conditionally on all others. Most methodologies build on the assumption that the network is sparse, meaning that only a small fraction of variable pairs are conditionally dependent. The case of Gaussian graphical models (GGM) is especially appealing as the network corresponds to the support of the precision matrix of the joint Gaussian distribution. The use of a sparsity-inducing penalisation gives raise to the celebrated graphical lasso (Friedman et al., 2008). In a more general context, Chow and Liu (1968) consider a spanning tree structure to impose sparsity to the network, but this drastic form can be alleviated using mixtures of trees (Kirshner, 2008; Meilă and Jaakkola, 2006).
One important aspect of network inference is to distinguish between variables that are marginally dependent (possibly because of their respective dependency with some common other) from variables that are *directly related*, that is conditionally dependent. This distinction requires to account for as many confounding effects as possible, which includes all the other variables but also available covariates. It also requires to consider the existence of some *missing actors* (or missing nodes), that may induce an apparent direct dependency.

Abundance data. Count data is found in a multitude of fields (sociology, biology, economy, ecology...). It results from the counting of events in a given setting such as crime statistics in a state or the number of produced transcripts of a gene in an experiment. The statistical processing of count data cannot always rely on classical methods developed for continuous Gaussian data and appeals for specific methods. It often exhibits specificities such as zero-inflation and a large dispersion. The present work is motivated by the analysis of so-called abundance data, a count data avatar, arising from ecological studies where the number of individuals (the abundance) of a series of living species (plants or animals) is observed in a series of sites. In this context, network inference aims at understanding which pairs of

species are in direct interaction. The covariates are typically environmental descriptors (altitude, temperature, distance to the see, etc.) of each collection site, while the variables are the respective abundances of each species from the community under study.
No nice and generic framework as the GGM exists for count data. A few alternatives rely on copulas (Inouye et al., 2017) or models the node-wise conditional distributions as arising from exponential families. But most joint species distribution models resort to a latent Gaussian layer, which encodes the dependency structure between the species (Popovic et al., 2018, 2019; Warton et al., 2015). The Poisson log-normal model (PLN: Aitchison and Ho, 1989) enters this category: it assumes that a multivariate Gaussian random variable is associated with each species in each site and that the observed abundances are conditionally independent Poisson variables. The PLN model has already been applied to abundance data, both for dimension reduction (Chiquet et al., 2018) and network inference (Chiquet et al., 2019a; Momal et al., 2020).

Missing actors. In many situations, it is likely that not all actors involved in the system have been observed. The term 'actors' refers to either species that were not observed but nonetheless influence the abundance of others, or environmental conditions that were not accounted for.
In the perspective of unravelling the conditional independence structure, this can typically lead to the inference of spurious edges, which are links between observed actors that are not in direct interaction. In the graphical model framework, not accounting for one variable amounts to consider the marginal distribution of the rest of the system, as described in the left panel of Figure 3.1. Missing actors may be quantitative or qualitative. In the latter case it defines a latent group structure (Ambroise et al., 2009).

Figure 3.1 – Example of the marginalization when covariate x is unobserved. *Left*: complete graphical model (including x). *Right*: marginal graphical model of the observed variables (excluding x).

Several approaches have been proposed for network inference accounting for quantitative missing actors in the context of GGM. Many of them (Chandrasekaran et al., 2011; Giraud and Tsybakov, 2012; Lauritzen and Meinshausen, 2012; Meng et al., 2014) adapted the principle of Robust PCA (Candès et al., 2011) to the concentration matrix, assuming it is a sum of two matrices: one low-rank and one sparse. In terms of missing actors in a network, the low-rank part corresponds to missing

actors connected to all variables, whereas the sparse part refers to missing actors having a local effect. Following Robin et al. (2019) (also in the context of GGM), we focus on the later aspect, that is looking for missing actors not necessarily linked to all others. As far as we know, no model has been proposed for the inference of missing actors from abundance data.

Variational inference. The model we consider in this paper involves different types of variables, namely an unknown tree-shaped graphical model, a continuous latent layer (to induce dependence between the species) and unobserved actors. The most popular approach for the inference of such models is the EM algorithm (Dempster et al., 1977), which requires the evaluation of the conditional distribution of all unobserved variables given the data. In the problem we consider, some latent variables are (multivariate) continuous and others are discrete, and their joint conditional distribution turns out to be intractable. In this work we resort to a variational approximation (Wainwright and Jordan, 2008) of this conditional distribution and to a variational EM algorithm for its inference (see e.g. Blei et al., 2017).

Our contribution. In the context of the Poisson log-normal model, we propose a tree-based approach to recover the structure of latent graphical model including actors. The model we consider involves several layers of unobserved variables with intractable conditional distributions, thus we resort to a variational EM algorithm (Blei et al., 2017) for its inference. We introduce the model in Section 3.2 and describe its variational inference in Section 3.3. The performance of the algorithm is assessed via simulations in Section 3.4. The use of the proposed model is illustrated in Section 3.5, where we demonstrate its ability to recover environmental drivers on two ecological datasets. The inference procedure is implemented in the R package nestor, available at `github.com/Rmomal/nestor`.

3.2 Model

3.2.1 Poisson log-normal and tree-shaped graphical models

Poisson log-normal model.

We start with a reminder on the multivariate Poisson log-normal model, with the example of abundance data. The abundances of p species observed on n sites are gathered in the $n \times p$ matrix $\boldsymbol{Y}$ where Y_{ij} is the count of species j in site i, and the row i of $\boldsymbol{Y}$, denoted $\boldsymbol{Y}_i$, is the abundance vector collected on site i. A covariate vector $\boldsymbol{x}_i$ with dimension d is also measured on each site i and all covariates are gathered in the $n \times d$ matrix $\boldsymbol{X}$. The PLN model states that a (latent) Gaussian

vector $\boldsymbol{U}_i$ of size p with variance matrix $\boldsymbol{R} = (\rho_{kl})_{kl}$ is associated with each site:

$$\{\boldsymbol{U}_i\}_{1 \leq i \leq n} \text{ iid}, \qquad \boldsymbol{U}_1 \sim \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{R}), \tag{3.1}$$

the sites being assumed to be independent. To ensure identifiability, we let the diagonal of $\boldsymbol{R}$ be made of 1's, so $\boldsymbol{R}$ is actually a correlation matrix. All latent vectors $\boldsymbol{U}_i$ are gathered in the $n \times p$ matrix $\boldsymbol{U}$. The PLN model further assumes that species abundances in all sites are conditionally independent, and that their respective distribution only depends on the environment and the associated latent variable:

$$\{Y_{ij}\}_{1 \leq i \leq n, 1 \leq j \leq p} \mid \boldsymbol{U} \text{ independent}, \quad Y_{ij} \mid U_{ij} \sim \mathcal{P}\left(\exp(o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + \sigma_j U_{ij})\right), \tag{3.2}$$

where o_{ij} is a known offset term which typically accounts for the sampling effort, and σ_j is the latent standard deviation associated with species j. The vector $d \times 1$ of regression coefficients $\boldsymbol{\theta}_j$ describes the environmental effects on species j. An important feature of the PLN model is that the sign of the correlation between the observed counts is the same as this of correlation between the latent variables (Aitchison and Ho, 1989): $\text{sign}(\text{Cor}(Y_{ij}, Y_{ik})) = \text{sign}(\text{Cor}(U_{ij}, U_{ik}))$.

Tree-shaped graphical models.

Network inference relies on the assumption that few species are directly dependent on one another, meaning that the underlying graphical model is sparse. In the framework of the PLN model, the graphical model of interest rules the distribution of the latent vectors $\boldsymbol{U}_i$ and is encoded in the precision matrix $\boldsymbol{\Omega} := \boldsymbol{R}^{-1}$. A way to foster sparsity is to impose $\boldsymbol{\Omega}$ to be faithful to a spanning tree T, that is: $\boldsymbol{U}_1 \sim \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{\Omega}_T^{-1})$ where the non-zero terms of $\boldsymbol{\Omega}_T$ correspond to the edges of the tree T . However this hypothesis is very restrictive as it allows only $p-1$ links among p species (Chow and Liu, 1968). A more flexible approach consists in assuming that the latent vectors are drawn from a mixture of Gaussian distributions, each faithful to a tree T (Kirshner, 2008; Meilă and Jaakkola, 2006; Meilă and Jordan, 2000; Schwaller et al., 2019):

$$\boldsymbol{U}_1 \sim \sum_{T \in \mathcal{T}_p} p(T) \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{\Omega}_T^{-1}), \tag{3.3}$$

where $\mathcal{T}_p$ is the set of spanning trees with p nodes. We further assume that the tree distribution $\{p(T)\}_{T \in \mathcal{T}_p}$ can be written as a product over the edges:

$$p(T) = B^{-1} \prod_{jk \in T} \beta_{jk}, \qquad \text{with} \quad B = \sum_{T \in \mathcal{T}_p} \prod_{jk \in T} \beta_{jk}. \tag{3.4}$$

The weights β_{jk} are gathered in the $p \times p$ symmetric matrix $\boldsymbol{\beta}$ with diagonal zero. Observe that these weights are defined up to a multiplicative constant, so that only $p(p-1)/2 - 1$ of them may vary independently. This PLN model with latent tree-shaped dependency structure is similar to that considered by Momal et al. (2020).

3.2.2 Introducing the missing actor

PLN model with missing actors.

We now introduce the concept of missing actors, which corresponds to variables that are involved in the graphical model but are not associated with observed variables. To involve such actors in the model, we assume that a complete latent vector $\boldsymbol{U}_i$ with dimension $p + r$ is associated with site i, where r is the number of missing actors. This complete vector can be decomposed as $\boldsymbol{U}_i^\intercal = [\boldsymbol{U}_{Oi}^\intercal \; \boldsymbol{U}_{Hi}^\intercal]$ where $\boldsymbol{U}_{Oi}$ (with dimension p) corresponds to observed species and $\boldsymbol{U}_{Hi}$ (with dimension r) corresponds to the missing actors. The complete $n \times (p + r)$ latent matrix $\boldsymbol{U}$ can be decomposed in the same way as $\boldsymbol{U} = [\boldsymbol{U}_O \; \boldsymbol{U}_H]$, $\boldsymbol{U}_O$ and $\boldsymbol{U}_H$ having dimension $n \times p$ and $n \times r$, respectively.
The model we consider states that

i the complete latent vectors $\boldsymbol{U}_i$ are all iid and distributed according to a mixture similar to (3.3) and (3.4) but with Gaussian distributions (and matrices $\boldsymbol{\Omega}_T$ and $\boldsymbol{\beta}$) of dimension $(p + r)$, and trees drawn from $\mathcal{T}_{p+r}$;

ii the abundances Y_{ij} of the p observed species are distributed according to (3.2), replacing $\boldsymbol{U}$ with $\boldsymbol{U}_O$,

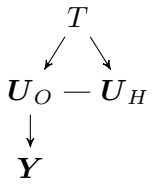


Figure 3.2 – Graphical model linking the count data $\boldsymbol{Y}$, the latent layer of Gaussian parameters $\boldsymbol{U} = (\boldsymbol{U}_O, \boldsymbol{U}_H)$, and the latent tree T.

In the sequel, we shall refer to the elements of $\boldsymbol{U}_O$ and $\boldsymbol{U}_H$ respectively as 'observed' and 'hidden' (or 'missing') latent variables, whereas obviously none of them are actually observed. Figure 3.2 displays the graphical model of the quadruplet $(T, \boldsymbol{U}_O, \boldsymbol{U}_H, \boldsymbol{Y})$. The observed data $\boldsymbol{Y}$ still arise from an PLN model, but the graphical model of the observed latent $\boldsymbol{U}_O$ may not be sparse due to the marginalization over the hidden latent $\boldsymbol{U}_H$. Our main goal is to infer the dependency structure of the complete latent vectors, that is to estimate the elements of the matrices $\boldsymbol{\Omega}_T$

and the edges weights $\boldsymbol{\beta}$. The latent dependency structure is similar to this considered by Robin et al. (2019), but the inference strategy much differs, because of the additional hidden layer.

Identifiability restriction.

The proposed model only makes sense because the graphical model of the complete latent vectors $\boldsymbol{U}_i^\intercal = [\boldsymbol{U}_{Oi}^\intercal \ \boldsymbol{U}_{Hi}^\intercal]$ is supposed to be sparse. Missing actors could obviously not be identified from a regular PLN model, without restriction on the precision matrix $\boldsymbol{\Omega}$, as only the marginal precision matrix of the $\boldsymbol{U}_{Oi}$ could be recovered. Still, to ensure identifiability we impose the same restriction as Robin et al. (2019) that missing latent variables are not connected with each other (the block corresponding to $\boldsymbol{U}_H \times \boldsymbol{U}_H$ is diagonal in each $\boldsymbol{\Omega}_T$).

3.3 Inference

As said in the introduction, we resort to a variational EM algorithm to perform the inference due to the complex latent structure.

3.3.1 Variational inference

The log-likelihood of the so-called *complete* data, that is $(\boldsymbol{Y}, \boldsymbol{U}, T)$, writes

$$\log p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{Y}, \boldsymbol{U}, T) = \log p_{\boldsymbol{\beta}}(T) + \log p_{\boldsymbol{\Omega}}(\boldsymbol{U} \mid T) + \log p_{\boldsymbol{\theta}}(\boldsymbol{Y} \mid \boldsymbol{U}).$$

where $\boldsymbol{\Omega}$ stands for the set of all tree-specific precision matrices: $\boldsymbol{\Omega} = \{\boldsymbol{\Omega}_T, T \in \mathcal{T}_{p+r}\}$. The conditional distributions of the latent variables $\boldsymbol{U}$ and of the tree T given the data $\boldsymbol{Y}$ are both intractable. Variational inference then aims at maximizing a lower bound of the log-likelihood of the observed data, which writes in our context as

$$\begin{aligned} \mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\beta}, \Omega; q) &= \log p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{Y}) - KL\left(q(\boldsymbol{U}, T) \| p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{U}, T \mid \boldsymbol{Y})\right) \\ &= \mathbb{E}_q \log p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{Y}, \boldsymbol{U}, T) + \mathcal{H}(q(\boldsymbol{U}, T)), \end{aligned} \tag{3.5}$$

where $q(\boldsymbol{U}, T)$ stands for the approximate joint conditional distribution of the latent layer and of the tree: $q(\boldsymbol{U}, T) \simeq p(\boldsymbol{U}, T \mid \boldsymbol{Y})$.

Approximate distribution.

The efficiency of variational inference mostly depends on the choice of $q(\boldsymbol{U}, T)$, which is a balance between computational ease and adequation to the target distribution $p(\boldsymbol{U}, T \mid \boldsymbol{Y})$. We adopt here a classical product form for the approximate

distribution: we impose to the latent variables $\boldsymbol{U}$ and to the tree T to be independent according to q (whereas actually they are not conditional on the data), with respective marginals h and g:

$$q(\boldsymbol{U}, T) = h(\boldsymbol{U})g(T).$$

Because the sites are independent, and without further assumption, the distribution h is a product over all sites. Following Chiquet et al. (2018) we approximate the conditional distribution of each latent vector $\boldsymbol{U}_i$ with a Gaussian distribution, that is:

$$h(\boldsymbol{U}) = \prod_i \mathcal{N}_{p+r}(\boldsymbol{U}_i; \boldsymbol{m}_i, \boldsymbol{S}_i).$$

with all $\boldsymbol{S}_i$ diagonal. We gather all the mean vectors $\boldsymbol{m}_i$ in the $n \times (p+r)$ matrix $\mathbf{M}$ and pile up the diagonals of all the variance matrices $\boldsymbol{S}_i$ in the $n \times (p+r)$ matrix denoted $\boldsymbol{S}$.

Variational EM.

The variational EM algorithm then consists in maximizing the lower bound $\mathcal{J}$ defined in (3.5) with respect to the parameters (M step), and to the approximate distributions (VE step), alternatively.

M step: At iteration $t+1$, given the current approximate distribution $q^t(\boldsymbol{U}, T) = g^t(T)h^t(\boldsymbol{U})$, the M step consists in the update of the model parameters, solving

$$\begin{aligned} \boldsymbol{\theta}^{t+1} &= \arg\max_{\boldsymbol{\theta}} \ \mathbb{E}_{h^t}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{Y} \mid \boldsymbol{U})\right], \quad \boldsymbol{\Omega}^{t+1} = \arg\max_{\boldsymbol{\Omega}} \ \mathbb{E}_{q^t}\left[\log p_{\boldsymbol{\Omega}}(\boldsymbol{U} \mid T)\right], \\ \boldsymbol{\beta}^{t+1} &= \arg\max_{\boldsymbol{\beta}} \ \mathbb{E}_{g^t}\left[\log p_{\boldsymbol{\beta}}(T)\right]. \end{aligned} \tag{3.6}$$

Observe that the matrix of edge weights $\boldsymbol{\beta}$ is considered here as a parameter to be estimated, as opposed to Robin et al. (2019), where it was kept fixed and supposed to be given.

VE step: Maximising $\mathcal{J}$ with respect to (wrt) q is equivalent to minimizing the Küllback-Leibler divergence between $q(\boldsymbol{U}, T)$ and $p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{U}, T \mid \boldsymbol{Y})$ that appears in (3.5). Because we adopted a product form for q, the solution of the VE step for both g and h is known to be a mean-field approximation (Wainwright and Jordan, 2008). More specifically, maximising $\mathcal{J}$ gives

$$\begin{aligned} g^{t+1}(T) &\propto \exp\left\{\mathbb{E}_{h^t}\left[\log p_{\boldsymbol{\theta}^{t+1},\boldsymbol{\beta}^{t+1},\Omega^{t+1}}(\boldsymbol{Y}, \boldsymbol{U}, T)\right]\right\} \\ &\propto \exp\left\{\log p_{\boldsymbol{\beta}^{t+1}}(T) + \mathbb{E}_{h^t}\left[\log p_{\boldsymbol{\Omega}^{t+1}}(\boldsymbol{U} \mid T)\right]\right\}, \end{aligned} \tag{3.7}$$

and

$$
\begin{aligned}
h^{t+1}(\boldsymbol{U}) &\propto \exp\left\{\mathbb{E}_{g^{t+1}}\left[\log p_{\boldsymbol{\theta}^{t+1},\boldsymbol{\beta}^{t+1},\Omega^{t+1}}(\boldsymbol{Y},\boldsymbol{U},T)\right]\right\} \\
&\propto \exp\left\{\mathbb{E}_{g^{t+1}}\left[\log p_{\boldsymbol{\Omega}^{t+1}}(\boldsymbol{U} \mid T)\right] + \log p_{\boldsymbol{\theta}^{t+1}}(\boldsymbol{Y} \mid \boldsymbol{U})\right\}. \qquad (3.8)
\end{aligned}
$$

Observing that $\log p_{\boldsymbol{\beta}}(T) + \log p_{\boldsymbol{\Omega}}(\boldsymbol{U} \mid T)$ can be written as a sum over all the edges present in T, we see that $g^{t+1}(T)$ has a product form. So, without any further assumption, we may parametrize $g(T)$ in the same way as $p_{\boldsymbol{\beta}}(T)$:

$$
g(T) = \prod_{jk \in T} \widetilde{\beta}_{jk} / \widetilde{B} \qquad \text{where} \quad \widetilde{B} = \sum_{T \in \mathcal{T}_{p+r}} \prod_{jk \in T} \widetilde{\beta}_{jk}. \qquad (3.9)
$$

We gather the $\widetilde{\beta}_{jk}$'s in the $(p+r) \times (p+r)$ matrix $\widetilde{\boldsymbol{\beta}}$. The parameters $\widetilde{\boldsymbol{\beta}}$, $\mathbf{M}$ and $\boldsymbol{S}$ are called the variational parameters, in the sense that it is equivalent to optimize $\mathcal{J}$ wrt (g, h) or wrt $(\widetilde{\boldsymbol{\beta}}, \mathbf{M}, \boldsymbol{S})$.

3.3.2 Proposed algorithm

The model we consider is an extension of the PLN model, for which an efficient inference algorithm have been implemented in `PLNmodels`, an R package available on CRAN (Chiquet et al., 2018, 2019a).

Prior estimates of $\boldsymbol{\theta}$, $\mathbf{M}_O$ and $\boldsymbol{S}_O$.

To alleviate the computational burden of the inference, we take advantage of this available tool to get an estimate of the regression coefficient matrix $\widehat{\boldsymbol{\theta}}$ and an approximation of the parameters of the observed latent variable conditional distribution $h_O(\boldsymbol{U}_O) \simeq p(\boldsymbol{U}_O \mid \boldsymbol{Y})$. These latter parameters are $\mathbf{M}_O$ and $\boldsymbol{S}_O$ (first p columns of $\mathbf{M}$ and $\boldsymbol{S}$ respectively) and we denote $\widetilde{\mathbf{M}}_O$ and $\widetilde{\boldsymbol{S}}_O$ their approximation. The quantities $\widehat{\boldsymbol{\theta}}$, $\widetilde{\mathbf{M}}_O$ and $\widetilde{\boldsymbol{S}}_O$ are kept fixed in the rest of the algorithm, so the VEM algorithm only deals with the remaining unknown quantities: the model parameters $\boldsymbol{\beta}$, $\boldsymbol{\Omega}$, and the variational parameters $\widetilde{\boldsymbol{\beta}}$, $\mathbf{M}_H$, $\boldsymbol{S}_H$. As a consequence, the final estimates we get yield a lower value of the objective function $\mathcal{J}$ as compared to an optimisation wrt all model and variational parameters.

M step.

This step deals with the update of the model parameters $\boldsymbol{\beta}$ and $\boldsymbol{\Omega}_T$. Some of the calculations are tedious and postponed to Appendix 3.A.

Edges weights $\boldsymbol{\beta}$: As shown in Equation (3.6), the maximization of $\mathcal{J}$ requires the computation of the derivative of $\mathbb{E}_{g^t}[\log p_{\boldsymbol{\beta}}(T)]$ wrt $\boldsymbol{\beta}$, which includes the derivative of the normalizing constant B. The latter can be computed via an extension

of the Matrix Tree theorem (see Meilă and Jaakkola, 2006, Lemma 3.1 reminded in Appendix 3.A). Setting the derivative of the expectation to 0 yields the following update (same as in Momal et al. (2020) and detailed in appendix 3.A):

$$\beta_{kl}^{t+1} = \frac{P_{kl}^t}{M(\boldsymbol{\beta}^t)_{kl}},$$

where $M(\boldsymbol{\beta})$ is defined in Lemma 3.1 and P_{kl}^t is the probability that the edge (k,l) belongs to the tree T according to g^t:

$$P_{kl}^t = \mathbb{P}_{g^t}\{kl \in T\} = \sum_{\substack{T \in \mathcal{T}: \\ T \ni kl}} g^t(T) = \frac{1}{\widetilde{B}^t} \sum_{\substack{T \in \mathcal{T}: \\ T \ni kl}} \prod_{uv \in T} \widetilde{\beta}_{uv}^t.$$

P_{kl}^t is computed using a result from Kirshner (2008) (reminded as Lemma 3.2 in appendix A). We now define the binary variable I_{Tkl} which indicates the presence of the edge kl in tree T, so $P_{kl}^t = \mathbb{E}_{g^t}[I_{Tkl}]$ and $I_T = [I_{Tkl}]_{1 \le k,l \le (p+r)}$ is the adjacency matrix of tree T.

Precision matrices $\boldsymbol{\Omega}_T$: For a given dependency structure in the Gaussian Graphical model framework, Lauritzen (1996) gives maximum likelihood estimates for the precision matrix. These estimators are given as functions of sufficient statistics of the multivariate Gaussian distribution. Indeed in the exponential family framework, the M step of any EM algorithm requires the computation of the expectation of a sufficient statistic, under the current fit of the variational laws (see McLachlan and Krishnan (2007)). Here as $\boldsymbol{U} \mid T$ is centered, a sufficient statistic is $\boldsymbol{U}^\intercal \boldsymbol{U}$. We now let SSD denote the matrix defined as

$$SSD^t = \mathbb{E}_{h^t}(\boldsymbol{U}^\intercal \boldsymbol{U}) = (\mathbf{M}^t)^\intercal \mathbf{M}^t + \boldsymbol{S}_+^t,$$

where $\boldsymbol{S}_+^t = \sum_i \boldsymbol{S}_i^t$. Applying Lauritzen's formulas, we get:

$$\omega_{Tkl}^{t+1} = \begin{cases} \dfrac{-ssd_{kl}^t/n}{1-(ssd_{kl}^t/n)^2} & \text{if } kl \in T \\ 0 & \text{otherwise} \end{cases}, \tag{3.10}$$

$$\omega_{Tkk}^{t+1} = 1 + \sum_l I_{Tkl} \frac{(ssd_{kl}^t/n)^2}{1-(ssd_{kl}^t/n)^2},$$

where ssd_{kl}^t stands for the entry kl of the matrix SSD^t. The calculations are postponed to Appendix 3.A. Observe that estimates of the off-diagonal entries ω_{Tkl}^{t+1} do not depend on T provided that the edge (k,l) belongs to T. Thus the estimates of the off-diagonal terms of the precision matrices $\boldsymbol{\Omega}_T$ are common to all trees sharing a given edge. This does not result from any assumption on the shape

of $\mathbf{\Omega}_T$, but from the properties of the maximum likelihood estimate of Gaussian variance matrix. In the sequel we will simply denote off-diagonal terms by ω_{kl} (as opposed to ω_{Tkk} which still depends on T).

Other quantities are needed for later computations. Lauritzen gives the maximum likelihood estimator of every entry of the correlation matrix $\boldsymbol{R}_T$ corresponding to an edge kl being part of T, which is $\boldsymbol{R}^{t+1}_{Tkl} = ssd^t_{kl}/n$. Hereafter for any matrix A, $A_{[kl]}$ refers to the bloc kl of A: $A_{[kl]} = (a_{ij})_{\{i,j\}\in\{k,l\}}$. The determinant of $\mathbf{\Omega}^{t+1}_T$ factorizes on the edges of T and writes as a function of blocks of the correlation matrix as follows:

$$|\mathbf{\Omega}^{t+1}_T| = \Big(\prod_{kl\in T} |\boldsymbol{R}^{t+1}_{T[kl]}|\Big)^{-1} \quad \text{and for any } kl \in T,\ |\boldsymbol{R}^{t+1}_{T[kl]}| = 1 - (ssd^t_{kl}/n)^2. \tag{3.11}$$

Finally we define the matrix $\overline{\mathbf{\Omega}}^{t+1} = \mathbb{E}_{g^t}[\mathbf{\Omega}^{t+1}_T]$. Noticing that, for $k \neq l$, $\mathbb{E}_{g^t}[\mathbf{\Omega}^{t+1}_T]_{kl} = \mathbb{E}_{g^t}[\mathbf{\Omega}^{t+1} \odot I_T]_{kl}$, edges probabilities appear as follows:

$$\bar{\omega}^{t+1}_{kl} = -P^t_{kl}\frac{ssd^t_{kl}/n}{1-(ssd^t_{kl}/n)^2}, \qquad \bar{\omega}^{t+1}_{kk} = 1 + \sum_l P^t_{kl}\frac{(ssd^t_{kl}/n)^2}{1-(ssd^t_{kl}/n)^2}.$$

VE step.

This step deals with the update of the approximate conditional distributions g and h_H, namely the update of the corresponding variational parameters $\widetilde{\boldsymbol{\beta}}$, $\mathbf{M}_H$ and $\boldsymbol{S}_H$.

Approximate conditional tree distribution $g(T)$: Computing the expression (3.7) yields the following, where the constant term 'cst' does not depend on a specific edge:

$$\begin{aligned}\log g^{t+1}(T) &= \log p_{\boldsymbol{\beta}^{t+1}}(T) + \mathbb{E}_{h^t}\left[\log p_{\mathbf{\Omega}^{t+1}}(\boldsymbol{U} \mid T)\right] + \text{cst} \\ &= \sum_{kl\in T} \log \beta^{t+1}_{kl} - \frac{n}{2}\log|\boldsymbol{R}^{t+1}_{[kl]}| - \omega^{t+1}_{kl}\left[(\mathbf{M}^t)^\intercal \mathbf{M}^t\right]_{kl} + \text{cst}.\end{aligned}$$

Then remembering the product form of g^{t+1} given in (3.9), we obtain the expression for each edge variational weight:

$$\widetilde{\beta}^{t+1}_{kl} = \beta^{t+1}_{kl}\left|\boldsymbol{R}^{t+1}_{[kl]}\right|^{-n/2} \exp\left(-\omega^{t+1}_{kl}\left[(\mathbf{M}^t)^\intercal \mathbf{M}^t\right]_{kl}\right). \tag{3.12}$$

Approximate Gaussian distribution h: According to (3.8), we have that

$$\log h^{t+1}(\boldsymbol{U}) = \mathbb{E}_{g^{t+1}} \log p(\boldsymbol{Y} \mid \boldsymbol{U}_O) - \frac{1}{2}\mathrm{tr}\left(\overline{\boldsymbol{\Omega}}_{\mathrm{T}}^{t+1}(\boldsymbol{U}^\intercal \boldsymbol{U})\right) + \mathrm{cst}.$$

Using the properties of the conditional Gaussian distribution we have that

$$h^{t+1}(\boldsymbol{U}_H \mid \boldsymbol{U}_O) = \mathcal{N}\left(\boldsymbol{U}_H; -\boldsymbol{U}_O \overline{\boldsymbol{\Omega}}_{OH}^{t+1}\left(\overline{\boldsymbol{\Omega}}_H^{t+1}\right)^{-1}, \left(\overline{\boldsymbol{\Omega}}_H^{t+1}\right)^{-1}\right).$$

Now, to get $h_H^{t+1}(\boldsymbol{U}_H)$, it suffices to integrate $h^{t+1}(\boldsymbol{U}_H \mid \boldsymbol{U}_O)$ wrt h_O (the parameter of which are kept fixed along iterations) to get

$$\mathbf{M}_H^{t+1} = -\widetilde{\mathbf{M}}_O \overline{\boldsymbol{\Omega}}_{OH}^{t+1}\left(\overline{\boldsymbol{\Omega}}_H^{t+1}\right)^{-1}, \qquad \boldsymbol{S}_H^{t+1} = \left(\overline{\boldsymbol{\Omega}}_H^{t+1}\right)^{-1}.$$

3.3.3 Algorithm peculiarities

Initialization.

As for any EM algorithm, the choice of the starting point is paramount. The initialization we use here takes the primary estimate $\widetilde{M}_O$ as an input.

Initial clique: As a starting point, we look for a clique of species as potential neighbors of the missing actor h. There are many different ways to do so, and if any prior knowledge exists on that matter it should be used. Otherwise, such a clique can be found using sparse principal component analysis (sPCA; Erichson et al., 2020), where principal components are formed using only a few of the original variables, which is consistent with the assumption that each missing actor is connected only to some actors in the network.

When applying sPCA to $\widetilde{M}_O$, the set of non-zero loadings of each principal components provides us with an initial clique of neighbors of each missing actor.

Parameters initialization: The eigenvectors resulting from the sPCA also provide us with a starting value $\mathbf{M}_H^0$, as well as a first estimate of the latent correlation matrix $\boldsymbol{R}^0$. The parameter $\boldsymbol{\beta}$ is uniformly initialized.

Numerical issues.

Because the Matrix Tree Theorem and Kirshner's formula respectively resort to the calculation of a determinant and a matrix inversion, the proposed algorithm is exposed to numerical instabilities. To circumvent these issues, we rely on both multiple-precision arithmetic and likelihood tempering (via a parameter α, similarly to Schwaller and Robin, 2017). More details are given in Appendix 3.A.

3.4 Simulations

3.4.1 Count datasets

For the simulation study, 300 count datasets of 15 species in total including one missing actor are generated, thus $p = 14$ and $r = 1$. Data is generated as follows. We generate a scale-free structure $\mathcal{G}$ (which degree distribution is a power law) with $p + 1$ nodes using the R package `huge` (Zhao et al., 2012) available on CRAN. The missing species h is chosen as the one with highest degree. We measure the *influence* of the missing actor with its degree, distinguishing three influence classes: *Minor* (degree ≤ 5), *Medium* ($5 <$ degree ≤ 7) and *Major* (degree ≥ 8). For each replicate, the latent layer $\boldsymbol{U}$ and the observed abundances $\boldsymbol{Y}$ are simulated according to the model defined in Section 3.2.

3.4.2 Experiment & Measures

For each simulated dataset, the VEM algorithm is initialized as described in Section 3.3.3. More specifically and because we only look for one missing actor, we consider the cliques corresponding to each of the first two principal components of sPCA, and their respective complements, which provides us with four cliques. Then four VEM algorithms, as described in Section 3.3.2, are run starting from each of the four candidate cliques, and the one yielding the highest lower bound $\mathcal{J}$ is kept. For all simulations, we set the precision of the convergence criterion to $\varepsilon = 10^{-3}$, the tempering parameter to $\alpha = 0.1$ and the maximal number of iterations to 100. The inference quality is assessed regarding the global network inference, the missing actor's position in the network, and its values along the n sites. We refer to this first procedure as the *blind* procedure. Additionally, we define the *oracle* procedure as running the VEM with the set of true neighbors of the missing actor as initial clique.

For each procedure, a general measure of the whole network inference quality is first given by comparing the inferred edge probabilities to the original dependency structure. This is done using the Area Under the ROC Curve (AUC) criteria. Then, to be more specific and target the neighbors of node h specifically, the probabilities of edges involving h are transformed into binary values using the 0.5 threshold. The values are then compared to the original links of h and yield quantities of true/false (T/P) positives/negatives (P/N), from which are built the *precision* (also known as the positive predictive value, $TP/(TP + FP)$) and the *recall* (also known as the true positive rate, $TP/(TP + FN)$) criteria. Finally, we assess the ability to reconstruct the missing actor across the sites by computing the absolute correlation between its inferred vector of means (M_h) and its original latent Gaussian vector $\boldsymbol{U}_h$.

3.4.3 Results

Simulations performance measures are gathered in Table 3.1 and Table 3.2 for blind and oracle procedures respectively. The distributions of the quality measures are displayed in Figure 3.3.

Table 3.1 shows the network is well inferred, as all AUC means are above 0.85, with almost perfect inference when the influence of the missing actor is major. Its neighbors and values per site are very well retrieved in these cases with mean recall values above 0.9 and mean correlation above 0.8, with a great confidence in the algorithm outputs as mean precision is above 0.95. However, there exists a clear deterioration of all performance as the influence decreases with lower means are greater deviations, down to about 0.6 mean values for all measures when the influence is minor. Moreover, the algorithm takes more and more time to converge as the influence decreases, although it stays at about $3s$ for minor cases which is reasonable. Figure 3.3 shows that as the influence decreases, the densities present with several modes and dilute towards 0, illustrating that even if some networks are still well-inferred, there also are more and more cases where the algorithm fails. In particular, the performance decrease of medium cases seems to be only due to a greater number of failed inferences.

All these elements point to minor cases being harder problems to solve, unsurprisingly. Yet as oracle results show in Table 3.2, it is possible to carry out almost-perfect inference in all cases, if the algorithm is initialized with the true clique; the deterioration is still present in all measures, but stays marginal. Thus the harsh decrease in the blind procedures seems to be mainly due to the proposed initialization method failing at correctly finding some of the small cliques of neighbors.

	N	AUC	Precision	Recall	Correlation	Time (s)
Major	100	0.98 (0.06)	0.96 (0.14)	0.94 (0.17)	0.83 (0.10)	2.36 (0.91)
Medium	132	0.93 (0.12)	0.83 (0.26)	0.81 (0.30)	0.73 (0.17)	2.69 (1.15)
Minor	68	0.89 (0.10)	0.61 (0.34)	0.66 (0.36)	0.59 (0.21)	3.08 (1.14)

Table 3.1 – Blind procedure using cliques from initialization. The influence of the missing actor is measured with its degree, distinguishing three influence classes: *Minor* (degree ≤ 5), *Medium* ($5 <$ degree ≤ 7) and *Major* (degree ≥ 8). For each class of influence, the following quantities are reported: number of simulated graphs (N), means and standard deviations of AUC, Precision, Recall, Correlation between missing actor inferred vector of means and original latent vector, and running times in seconds. AUC measures the retrieval of the dependence structure between all variables (observed and missing), whereas precision and recall are specific to the missing actor links.

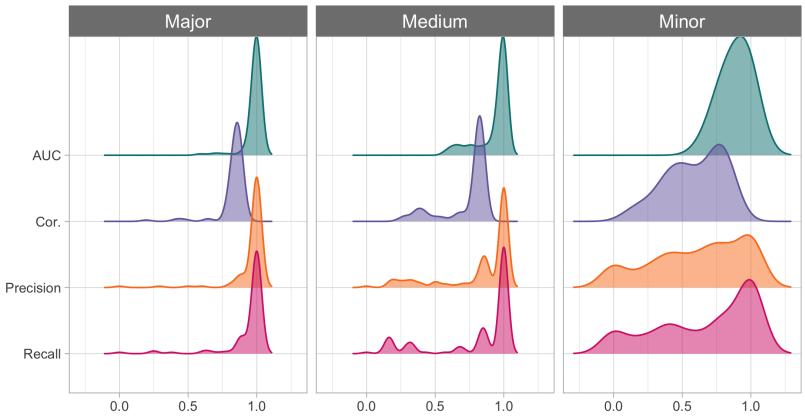


Figure 3.3 – The influence of the missing actor is measured with its degree, distinguishing three influence classes: *Minor* (degree ≤ 5), *Medium* ($5 <$ degree ≤ 7) and *Major* (degree ≥ 8). The distributions of performance measures are displayed for each class of influence: AUC measures the retrieval of the dependence structure between all variables, observed and missing. Precision and recall are specific to the missing actor links.

About intialization. Figure 3.4 compares the initialization quality and the corresponding final inferred neighbors, in terms of initial (-i) and final (-f) false negative (FNR, also 1-TPR) and positive rates (FPR). It clearly appears that final measures mostly increase with false negatives of the initial clique. This means that not including a neighbor in the initialization is much worse for the inference than falsely including a node. The increase of FNR-f is bigger than that of FPR-f, meaning that a wrong initialization leads to a set of inferred neighbors which most part can be trusted, but which will be largely incomplete. This advocates for bigger initialization cliques when no prior information is available.

	N	AUC	Precision	Recall	Cor.	t(s)
Major	100	1 (0.00)	1 (0.00)	1 (0.01)	0.86 (0.02)	1.28 (0.21)
Medium	132	1 (0.02)	1 (0.00)	0.99 (0.04)	0.83 (0.02)	1.38 (0.46)
Minor	68	0.98 (0.04)	0.99 (0.03)	0.96 (0.12)	0.8 (0.04)	1.56 (0.69)

Table 3.2 – Oracle procedure using true clique as starting point. The influence of the missing actor is measured with its degree, distinguishing three influence classes: *Minor* (degree ≤ 5), *Medium* ($5 <$ degree ≤ 7) and *Major* (degree ≥ 8). For each class of influence, the following quantities are reported: number of simulated graphs (N), means and standard deviations of AUC, Precision, Recall, Correlation between missing actor inferred vector of means and original latent vector, and running times in seconds. AUC measures the retrieval of the dependence structure between all variables (observed and missing), whereas precision and recall are specific to the missing actor links.

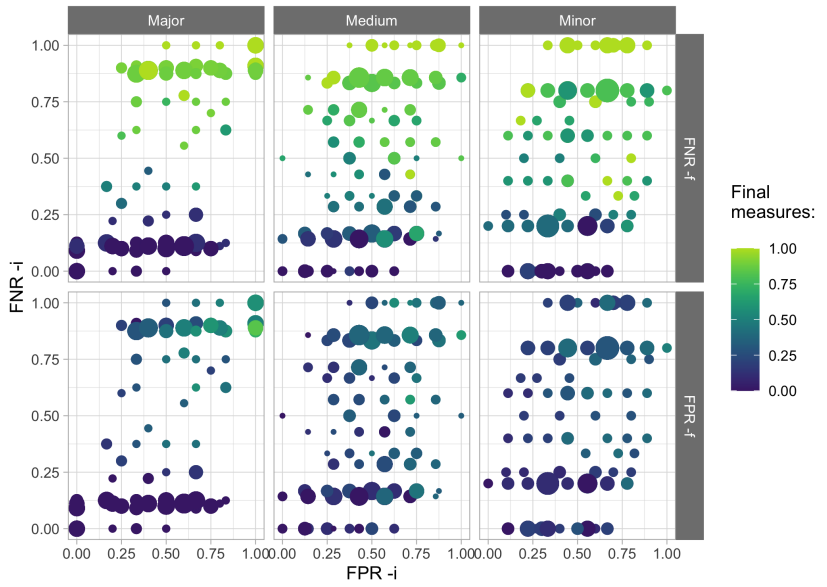


Figure 3.4 – Comparison of initial and final FPR and FNR, for cliques of neighbors of one missing actor obtained with the sparse PCA method. Position of dots are defined according to initial values, their color according to the final FPR and FNR. Sizes are proportional to the density of dots on a given position.

3.5 Applications

Cross validation criterion for model selection.

The proposed model obviously raises the problem of choosing the number of missing actors r (which may be zero). Variational-based inference often relies on approximate versions of the BIC or ICL criteria for model selection. Few theoretical guarantees exist about these approximate criteria and, in the present case, we observed that BIC and ICL penalizations did not yield consistent results. Therefore, we resort to V-fold cross validation to determine the number of missing actors.

More specifically, we split the original dataset $\boldsymbol{Y}$ ($\boldsymbol{X}$ is dropped here for the sake of clarity) into V subsets with almost equal sizes $m_1, \dots m_V$ ($\sum_{v=1}^{V} m_v = n$), which we denote $\{\boldsymbol{Y}^v\}_{v=1,\dots V}$. For each subset v, we define its complement $\boldsymbol{Y}^{-v}$ on which we fit a model with r missing actors and get a parameter estimate $\boldsymbol{\Gamma}_r^{-v} = (\boldsymbol{\theta}_r^{-v}, \boldsymbol{\sigma}_r^{-v}, \boldsymbol{\beta}_r^{-v}, \boldsymbol{\Omega}_r^{-v})$ and measure the fit of $\boldsymbol{\Gamma}_r^{-v}$ to the test dataset $\boldsymbol{Y}^v$.

To avoid the integration over the $(p+r)$-dimensional Gaussian latent layer, we measure the fit with the pairwise composite likelihood (Lindsay, 1988). For any given tree T and parameter $\boldsymbol{\Gamma}$, the bivariate Poisson log-normal pdf $p_{PLN}\left((Y_{ij}, Y_{ik}); \boldsymbol{\Gamma}, T\right)$ can be easily computed for any sample i and pair of species (j, k) with available tools such as the `poilog` R package (Vidar and Steinar, 2008) available on CRAN.

The cross-validation criterion is defined as

$$PCL_r(\boldsymbol{Y}) = \frac{1}{V}\sum_v \frac{1}{B}\sum_{b=1}^{B} \frac{1}{m_v}\sum_{i=1}^{m_v}\sum_{j<k} \log p_{PLN}\left((Y_{ij}^v, Y_{ik}^v); \boldsymbol{\Gamma}_r^{-v}, T_{r,b}^{-v}\right),$$

where the tree samples $\{T_{r,b}^{-v}\}_{b=1\ldots B}$ are iid according to $p_{\boldsymbol{\beta}_r^{-v}}(T)$.
The sampling procedure for spanning trees is given in Appendix 3.A; the complete procedure for the calculation of $PCL_r(\boldsymbol{Y})$ is described by Algorithm 1, given in Appendix 3.A. Note that this criterion measures the fit of the model in terms of abundance prediction, whereas our interest is mostly focused on the inference of the dependency structure. In other words, our goal is identification, that is selecting the smallest model and not the best model in terms of prediction (Arlot and Celisse, 2010).
We did not include this computationally greedy procedure in the simulation study but applied it to the two ecological datasets that will be described in the next two sections. The results, gathered in Figure 3.5, yield $r = 1$ missing actor for the Barents Sea data set, and $r = 2$ missing actors for the Fatala River one.

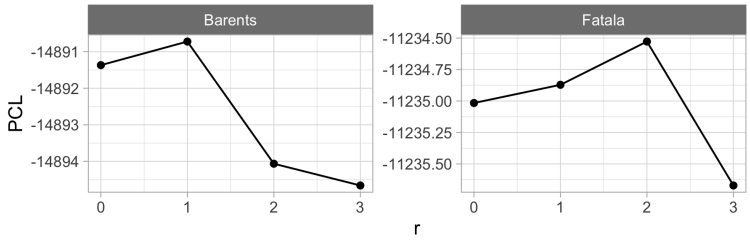


Figure 3.5 – Pairwise composite likelihoods estimates of Barents and Fatala datasets for models including 0 to 3 missing actors.

Initialization.

We performed a wider exploration for the initialization as compared to the simulation study. To enlarge the list of possible cliques, we applied a resampling version of the procedure described in Section 3.3.3, and applied it to 200 sub-samples, each consisting in 80% of the whole data set. This yielded 200 lists of r initial cliques, from which duplicates were removed.

3.5.1 Barents Sea

The dataset was first published by Fossheim et al. (2006) and consists of the abundance of 30 fish species measured in 89 sites in the Barents See in April-May 1997. In addition to abundances, the water temperature was measured in each site. The complete dataset is available at `www.fbbva.es/microsite/multivariate-statistics/data.html`. Fishes

distributions are known to be greatly linked with the temperature. Hence to illustrate our methodology, we present the results of the model fitted without any covariate (that is not accounting for the temperature), but including one missing actor (as suggested by Figure 3.5). To assess the ability of the proposed methodology to retrieve the influence of temperature as a missing actor, we report the empirical correlation between the temperature and the conditional expectation of the missing actor M_h, which we denote $\rho(H, \text{temp})$.

The resampling initialization procedure resulted in 14 different cliques, for each of which a VEM algorithm was run: the mean running time was 6.63mins with deviation 0.70 mins.

The edge probabilities involving node h as an endpoint were either very close to 0 or very close to 1, yielding a total of 6 highly probable neighbors of h. Figure 3.6 shows that many direct interactions are inferred between the corresponding 6 species in absence of a missing actor, which vanish when it is introduced. It also shows that accounting for this actor has only a local effect and that the direct interactions among the other species are preserved, which is consistent with our notion of a missing actor.

In terms of interpretation, Figure 3.7 shows that the missing actor is highly correlated with the temperature. It also appears that the abundances of the species neighbor to the missing actor are much more correlated with the temperature (mean correlation = 0.78, sd = .06) than the abundances of the non-neighbor species (mean correlation = 0.46, sd = .27). This example shows the ability of the method to recover an underlying effect that would not be recorded in the data.

3.5.2 Fatala River

Baran (1995) collected the abundances of 33 fish species in 90 sites along the Fatala River in Guinea between June 1993 and February 1994. The data are available from the R package `ade4` on CRAN (Dray et al., 2007), along with the date and site of collection, from which we deduce the season (dry or rainy). Again the model was fitted without any covariates, but with two missing actors, as suggested by Figure 3.5.

The resampling initialization procedure yielded 60 different cliques, for each of which a VEM algorithm was run: the mean running time was 11.33 min (sd = 1.47 mn). 14 VEM did not reach convergence (with tolerance $\varepsilon = 1e-3$) after 100 iterations. We filtered out the results obtained from the different initializations, when the algorithm obviously ended in a degenerate solution ($\mathbb{V}(M_h) < \exp(-20)$).

Figure 3.8 shows the scatterplot of the estimated conditional mean of the two missing actors (M_{h_1}, M_{h_2}) in each site, colored with either one of the available covariates (site and season). The missing actor h_1 is obviously linked to the site and separates most upstream locations (kilometer 3) from most downstream locations (kilometer

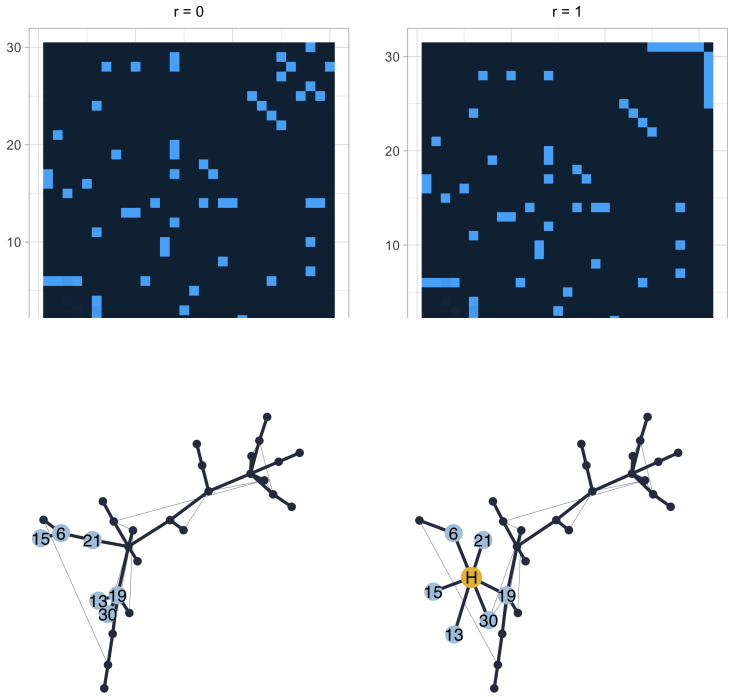


Figure 3.6 – *Top left:* adjacency matrix of the Barents Sea fishes interaction network for $r = 0$ missing actor. The inferred neighbors are gathered in the last 6 columns, so that their interactions are observable in the upper-right corner. *Top right:* adjacency matrix for $r = 1$ missing actor. The last column gathers the interactions of the inferred missing actor. *Bottom*: Inferred interaction network with $r = 0$ (left) and $r = 1$ (right). Colored nodes refer to the inferred neighbors (blue) of the missing actor (yellow). The edges width are proportional to their probability.

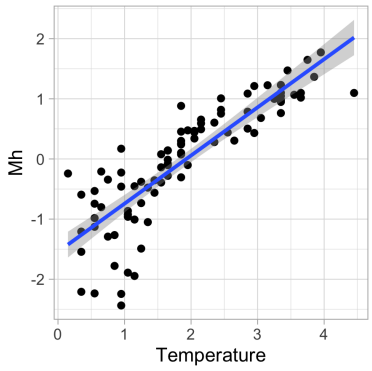


Figure 3.7 – Missing actor estimated vector of means M_h as a function of the temperature. $\rho(H, \text{temp}) = 0.85$.

46). This actor has 11 highly probable neighbor species. Again, this retrieved missing actor corresponds to an underlying effect (in this case: geography) that rules fish species abundances.

The second missing actor seems to be linked with the season but with a less clear separation. Also the variability of M_{h_2} is much smaller than this of M_{h_1}. This effect is therefore questionable, which brings us back to model selection. As mentioned above, we used a procedure based on cross-validation, which may be prone to select too complex models (Arlot and Celisse, 2010; Friedman et al., 2001; Shao, 1993). The definition of a grounded model selection criterion for structure inference in presence of missing actors remains open.

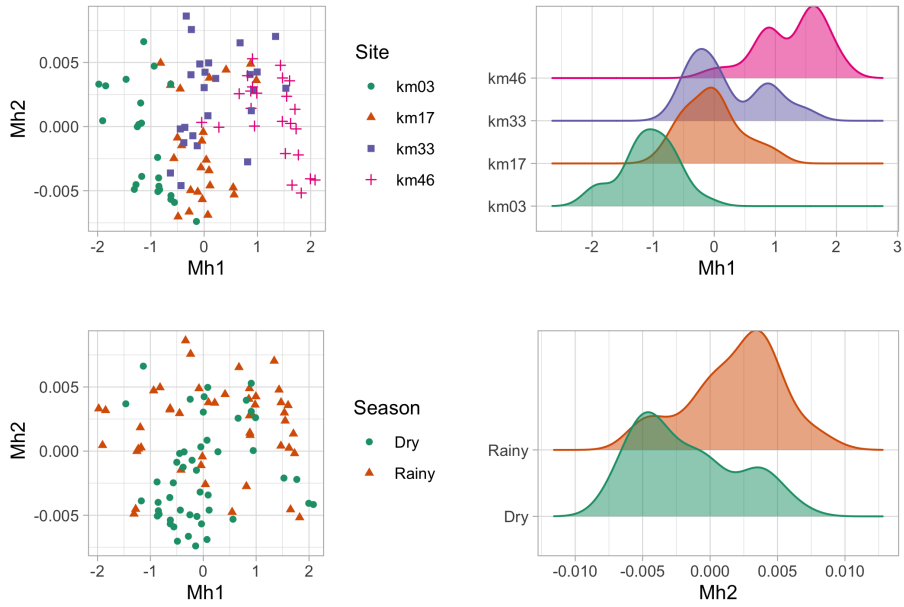


Figure 3.8 – Estimated means M_{h_1} and M_{h_2} of the two inferred missing actors. Left column: scatterplots M_{h_1} vs M_{h_2} with site (top) and season (bottom) color code. Right: distribution of the estimated means across sites. Top right: distribution of M_{h_1} in each location, bottom right: distribution of M_{h_2} in each season.

Appendices

3.A Supplementary material

3.A.1 Algebraic Tools

We here present some algebraic results about spanning tree structures which are used during the computations. Theorem 3.1, Lemma 3.1 as well as Lemma 3.2 use the notion of Laplacian matrix $\mathbf{Q}$ of a symmetric matrix $\mathbf{W} = [w_{jk}]_{1 \leq j,k \leq p}$, which is defined as follows:

$$[\mathbf{Q}]_{jk} = \begin{cases} -w_{jk} & 1 \leq j < k \leq p \\ \sum_{u=1}^{p} w_{ju} & 1 \leq j = k \leq p. \end{cases}$$

We further denote $\mathbf{W}^{uv}$ the matrix $\mathbf{W}$ deprived from its uth row and vth column and we remind that the (u, v)-minor of $\mathbf{W}$ is the determinant of this deprived matrix, that is $|\mathbf{W}^{uv}|$. The following Theorem 3.1 is the extension of Kirchhoff's Theorem to the case of weighted graphs (Chaiken and Kleitman, 1978; Meilă and Jaakkola, 2006).

Theorem 3.1 (Matrix Tree Theorem)**.** *For any symmetric weight matrix W with all positive entries, the sum over all spanning trees of the product of the weights of their edges is equal to any minor of its Laplacian. That is, for any $1 \leq u, v \leq p$,*

$$W := \sum_{T \in \mathcal{T}} \prod_{(j,k) \in T} w_{jk} = |\mathbf{Q}^{uv}|.$$

In the following, without loss of generality, we will choose $\mathbf{Q}^{11}$. As an extension of this result, Meilă and Jaakkola (2006) provide a close form expression for the derivative of W with respect to each entry of $\mathbf{W}$.

Lemma 3.1 (Meilă and Jaakkola (2006))**.** *Define the entries of the symmetric matrix $\mathbf{M}$ as*

$$[\mathbf{M}]_{jk} = \begin{cases} \left[(\mathbf{Q}^{11})^{-1}\right]_{jj} + \left[(\mathbf{Q}^{11})^{-1}\right]_{kk} - 2\left[(\mathbf{Q}^{11})^{-1}\right]_{jk} & 1 < j < k \leq p \\ \left[(\mathbf{Q}^{11})^{-1}\right]_{jj} & k = 1, 1 < j \leq p \\ 0 & j = k. \end{cases}$$

it then holds that

$$\partial_{w_{jk}} W = [\mathbf{M}]_{jk} \times W.$$

Kirshner (2008) build on Lemma 3.1 to provide an efficient computation of all edges probabilities.

Lemma 3.2 (Kirshner (2008))**.** *Let p_W be a distribution on the space of spanning trees, such that $p_W(T) = \prod_{kl \in T} w_{kl}/W$, where W is defined as in Theorem 3.1. Taking the symmetric matrix $\mathbf{M}$ as defined in Lemma 3.1, the probability for an edge kl to be in the tree T^* writes:*

$$\mathbb{P}\{kl \in T^*\} = \sum_{T \in \mathcal{T}} p_W(T) = w_{kl}\, \mathbf{M}_{kl}.$$

3.A.2 Computations

Update of β.

As in Momal et al. (2020), the update of $\boldsymbol{\beta}$ is such that:

$$\boldsymbol{\beta}^{t+1} = \arg\max_{\boldsymbol{\beta}} \; \mathbb{E}_{g^t}\left[\log p_{\boldsymbol{\beta}}(T)\right].$$

By definition of $p_{\boldsymbol{\beta}}(T)$:

$$\mathbb{E}_{g^t}\left[\log p_{\boldsymbol{\beta}}(T)\right] = \sum_{kl} P^t_{kl} \log \beta_{kl} - \log B\,, \qquad B = \sum_{T \in \mathcal{T}} \prod_{kl \in T} \beta_{kl}.$$

Computing the derivative with respect to the edge weight β_{kl} gives:

$$\partial_{\beta_{kl}} \mathbb{E}_{g^t}\left[\log p_{\boldsymbol{\beta}}(T)\right] = \frac{P^t_{kl}}{\beta_{kl}} - \frac{\partial_{\beta_{kl}} B^t}{B^t}.$$

According to Lemma 3.1: $\partial_{\beta_{kl}} B^t = [\boldsymbol{M}]_{kl} \times B$. Finally setting the derivative to 0 yields the update formula $\beta^{t+1}_{kl} = \frac{P^t_{kl}}{M(\boldsymbol{\beta}^t)_{kl}}$.

Update of Ω_T

The update of $\boldsymbol{\Omega}_T$ respects

$$\boldsymbol{\Omega}^{t+1} = \arg\max_{\boldsymbol{\Omega}} \; \mathbb{E}_{q^t}\left[\log p_{\boldsymbol{\Omega}}(\boldsymbol{U} \mid T)\right].$$

This is a problem of parameter optimisation in the context of Gaussian Graphical Models (GGM). In what follows, for any $q \times q$ matrix A, $A_{[kl]}$ will refer to the block kl of A: $A_{[kl]} = (a_{ij})_{\{i,j\} \in \{k,l\}}$. $[A_{[kl]}]^q$ will then denote the matrix obtained by

filling up with zero entries to obtain full dimension $q \times q$, so that:

$$([A_{[kl]}]^q)_{ij} = \begin{cases} a_{ij} & \text{if } \{i,j\} \in \{k,l\} \\ 0 & \text{if } \{i,j\} \in \{1,...,q\}_{\setminus kl} \end{cases}$$

In its proposition 5.9, Lauritzen (1996) states that in a GGM with p variables and associated with the decomposable graph $\mathcal{G}$, the maximum likelihood of the precision matrix exists if and only if $n > \max_{C \in \mathcal{C}} |C|$. It is then given as

$$\widehat{\Omega} = n \left(\sum_{C \in \mathcal{C}} [SSD_{[C]}{}^{-1}]^p - \sum_{S \in \mathcal{S}} \nu(S) \, [SSD_{[S]}{}^{-1}]^p \right)$$

where $\mathcal{C}$ is the set of cliques and $\mathcal{S}$ the set of separators of $\mathcal{G}$, with associated multiplicities $\nu(S)$.

In our context, $\mathcal{G}$ is a spanning tree and so all cliques are edges and separators are nodes. The multiplicity of a given node k as a separator in the graph is $\nu(k) = d(k) - 1$, where $d(k)$ is its degree. Therefore the estimator $\widehat{\Omega}_T$ writes as the following

$$\begin{aligned} \widehat{\Omega}_T &= n \sum_{kl \in T} [(SSD_{[kl]})^{-1}]^{p+r} - n \sum_k (d(k) - 1)[(SSD_{kk})^{-1}]^{p+r} \\ &= n \sum_{kl \in T} [(SSD_{[kl]})^{-1} - (SSD_{kk})^{-1} - (SSD_{ll})^{-1}]^{p+r} + n \sum_k [(SSD_{kk})^{-1}]^{p+r} \end{aligned}$$

As SSD has diagonal n, the expression simplifies. Denoting I_d the identity matrix of dimension d we obtain:

$$\widehat{\Omega}_T = n \sum_{kl \in T} [(SSD_{[kl]})^{-1} - \frac{1}{n} I_2]^{p+r} + I_{p+r}.$$

Detailing each bloc matrices as follows gives the update formulas in (3.10):

$$n \times [(SSD_{[kl]})^{-1} - \frac{1}{n} I_2] = \frac{1}{1 - (ssd_{kl}/n)^2} \begin{pmatrix} (ssd_{kl}/n)^2 & -ssd_{kl}/n \\ -ssd_{kl}/n & (ssd_{kl}/n)^2 \end{pmatrix}$$

Determinant of Ω_T.

The determinant of a precision matrix of a GGM with a decomposable graph is expressed as follows (Lauritzen, 1996):

$$|\Omega| = \frac{\prod_{C \in \mathcal{C}} |\Sigma_C|^{-1}}{\prod_{S \in \mathcal{S}} |\Sigma_S|^{-\nu(S)}},$$

where $\Sigma = \Omega^{-1}$. As $\boldsymbol{\Omega}_T$ is tree-structured, its determinant factorizes on the edges of T. It is expressed with the correlation matrix $\boldsymbol{R}_T$ as follows, denoting $d(k)$ the degree of node k:

$$|\boldsymbol{\Omega}_T| = \frac{\prod_{kl \in T} |\boldsymbol{R}_{Tkl}|^{-1}}{\prod_k |\boldsymbol{R}_{Tkk}|^{1-d(k)}}.$$

Using that $\boldsymbol{R}_T$ has diagonal 1, we obtain for step $t+1$ of the algorithm:

$$|\boldsymbol{\Omega}_T^{t+1}| = \Big(\prod_{kl \in T} |\boldsymbol{R}^{t+1}_{T[kl]}| \Big)^{-1}.$$

Numerical issues.

Exact computations Our algorithm requires the computation of determinants (from the Matrix Tree Theorem) and inverses (in Kirshner's formula) of Laplacian of weight matrices. As we deal with highly variable weights, numerical issues arise: infinite determinants or matrix numerically non-invertible due to either the maximal machine precision (about $1.7 \cdot 10^{308}$), or with machine zero (about $2.2 \cdot 10^{-16}$). To enhance the precision of such computations, we rely on multiple-precision arithmetic which allows the digit of precision of numbers to be limited only by the available memory instead of 64 bits. We implemented matrix inversion and log-determinant computation using both, symbolic computation and multiple precision arithmetic, relying on the `gmp` R package available on CRAN, which uses (Lucas et al., 2020), the C library GMP (GNU Multiple Precision Arithmetic).

Tempering parameter α

Definition: Weights $\widetilde{\beta}$ are mechanically linked to the quantity of data available n. To avoid reaching maximal precision when computing the determinant, a tempering parameter α is applied to every quantity proportional to n, so that the actual update performed is

$$\log \widetilde{\beta}_{kl} = \log \beta_{kl} - \alpha(\frac{n}{2} \log |\widehat{\boldsymbol{R}}_{Tkl}| + \widehat{\omega}_{Tkl}[M^\intercal M]_{kl}).$$

Heuristic for an upper bound: The proposed algorithm requires the computation of the normalizing constant $\widetilde{B}$, which is the determinant of any minor of the Laplacian of the $q \times q$ variational weights matrix $\widetilde{\boldsymbol{\beta}}$. As these weights mechanically increase with the quantity of available data n, this step is numerically very sensitive. Hereafter we denote $|\mathbf{Q}^{uv}|$ this determinant and Δ the maximal machine precision. In order to ease the computations, we define the tempering

parameter α as

$$\log \widetilde{\beta}_{kl} = \log \beta_{kl} - \alpha\left(\frac{n}{2} \log |\widehat{\boldsymbol{R}}_{Tkl}| + \widehat{\omega}_{Tkl}[M^\intercal M]_{kl}\right), \qquad \text{under constraint} \quad |\mathbf{Q}^{uv}| \leq \Delta.$$

Let's first detail the expression for $\widetilde{\beta}_{kl}$. Following the definition of the SSD matrix, and update formulas (3.10) and (3.11), we obtain:

$$\log \widetilde{\beta}_{kl} = \log \beta_{kl} + \alpha\, n \left\{ \frac{(ssd_{kl}/n)^2}{1 - (ssd_{kl}/n)^2} - \frac{1}{2} \log\left[1 - (ssd_{kl}/n)^2\right] \right\}.$$

For large n, we thus have

$$\widetilde{\beta}_{kl} \approx \exp\left[\alpha n \cdot C(ssd_{kl}/n)\right], \qquad \text{with} \quad C(x) = x/(1-x) - \log(\sqrt{1-x}),\ x \in [0, 1[.$$

We then define C_{sup} such that $C_{sup} = C(ssd_{max})$, with $ssd_{max} = \max\{ssd_{kl}, k \neq l\}$. By definition, $\mathbf{Q}^{uv}$ is positive-definite, so its determinant is upper bounded by the product of its diagonal terms (Hadamard's inequality). Namely:

$$\begin{aligned} |\mathbf{Q}^{uv}| &\leq \prod_{i=1}^{q-1} \mathbf{Q}_{ii}^{uv} \leq \prod_{i=1}^{q-1} \sum_{i=1}^{q-1} \exp(\alpha C_{sup} n) \\ &\leq \left[(q-1)\exp(\alpha C_{sup} n)\right]^{q-1}. \end{aligned}$$

Then applying the constraint yields:

$$|\mathbf{Q}^{uv}| \leq \Delta \iff \alpha \leq \frac{1}{C_{sup} n} \left[\frac{1}{q-1} \log \Delta - \log(q-1)\right].$$

For $ssd_{max} = 0.8$, $n = 200$ and $q = 15$, we get $\alpha \leq 1.05 \cdot 10^{-1}$.

3.A.3 Model selection and cross-validation

Sampling spanning trees

Sampling non-uniform spanning trees (i.e. sampling T from $p_{\boldsymbol{\beta}}$) is a research topic by itself, especially for large networks (see Durfee et al., 2017, for a review). For moderate size networks, a rejection algorithm (Devroye, 1986) can be defined in the following way:

1. Sample T from a distribution q, such that there exists a constant M, that ensures that, for all T, $Mq(T) > p_{\boldsymbol{\beta}}(T)$;
2. Keep T with probability $M^{-1} p_{\boldsymbol{\beta}}(T)/q(T)$ or try step 1 again.

The efficiency of such an algorithm strongly relies on the choice of the proposal distribution. Here we adopt the following proposal:

i Sample a connected graph G with independent edges, each drawn with probability $Q_{jk} \propto P_{jk} = \mathbb{P}_{\boldsymbol{\beta}}\{jk \in T\}$;

ii Sample T uniformly among the spanning trees of G.

Evaluation of the proposal. To evaluate the proposal distribution for each sampled tree, we may observe that, the probability for a graph drawn from the proposal to contain a given tree T is approximately

$$\mathbb{P}_q\{G \ni T\} \approx \prod_{jk \in T} Q_{jk},$$

the approximation being due to the connectivity constraint. This constraint can be almost surely satisfied by taking Q_{jk}'s large enough. So, denoting $|\mathcal{T}(G)|$ the number of spanning trees in G, we have that

$$q(T) = \sum_{G \ni T} q(T \mid G) q(G) = \sum_{G \ni T} \frac{q(G)}{|\mathcal{T}(G)|} = \mathbb{P}_q\{G \ni T\}\, \mathbb{E}\left(|\mathcal{T}(G)|^{-1} \mid G \ni T\right).$$

The last expectation can be evaluated via Monte Carlo, by sampling a series of graphs G according to the proposal q but forcing all edges from T to be part of G.

Upper bounding constant M. To evaluate the upper bounding constant M, we may observe that finding the tree T^* such that

$$m_{\boldsymbol{\beta}} := \frac{\mathbb{P}_q\{G \ni T^*\}}{p_{\boldsymbol{\beta}}(T^*)} = \min_{T \in \mathcal{T}} \frac{\mathbb{P}_q\{G \ni T\}}{p_{\boldsymbol{\beta}}(T)} = \min_{T \in \mathcal{T}} \prod_{jk \in T} \frac{Q_{jk}}{\beta_{jk}}.$$

is a minimum spanning tree problem. Then, obviously, for any tree T: $\mathbb{P}_q\{G \ni T\} \geq m_{\boldsymbol{\beta}} p_{\boldsymbol{\beta}}(T)$. Now, because the maximum number of spanning trees within a graph is p^{p-2}, we have

$$M q(T) = M \sum_{G \ni T} \frac{q(G)}{|\mathcal{T}(G)|} \geq \frac{M}{p^{p-2}} \sum_{G \ni T} q(G) = \frac{M}{p^{p-2}} \mathbb{P}_q\{G \ni T\} \geq M \frac{m_{\boldsymbol{\beta}}}{p^{p-2}} p_{\boldsymbol{\beta}}(T).$$

So we may set $M = p^{p-2}/m_{\boldsymbol{\beta}}$. Still, in practice, this bound turns out to be far too large and needs to be tuned down to preserve computational efficiency.

Cross-validation for model selection

The cross-validation procedure to estimate the pairwise composite likelihood is given in Algorithm 1. In practice $V = 10$ and $B = 100$.

Algorithm 1: Cross-validation for model selection with r missing actors

`// 0. INITIALIZATION;`

Divide the dataset $\boldsymbol{Y}$ into V subset $\boldsymbol{Y}^1, \dots \boldsymbol{Y}^V$;

for $v \in \{1, \cdots, V\}$ **do**

`// 1. Apply the VEM algorithm to the train dataset` $\boldsymbol{Y}^{-v}$;

$\boldsymbol{\Gamma}_r^{-v} \leftarrow (\boldsymbol{\theta}_r^{-v}, \boldsymbol{\sigma}_r^{-v}, \boldsymbol{\beta}_r^{-v}, \boldsymbol{\Omega}_r^{-v})$ `// 2. MONTE CARLO APPROXIMATION OF COMPLETE LOG-LIKELIHOOD EXPECTATION;`

for $b \in \{1, \cdots, B\}$ **do**

`// 2.1 Draw tree (see Section 3.A);`

$T_{r,b}^{-v} \sim p_{\boldsymbol{\beta}_r^{-v}}$

`// 2.2. Build the precision matrix having non-nul entries determined by` $T_{r,b}^{-v}$ `and values stored in` $\boldsymbol{\Omega}_r^{-v}$`, and its diagonal terms according to` (3.10);

$\boldsymbol{\Omega}_{T^b} \leftarrow f(T_{r,b}^{-v}, \boldsymbol{\Omega}_r^{-v})$

`// 2.3. Compute the marginal variance matrix;`

$\boldsymbol{\Sigma}_{T^bO} \leftarrow \boldsymbol{\Omega}_{T^bOO} - \boldsymbol{\Omega}_{T^bOH}\boldsymbol{\Omega}_{T^bHH}^{-1}\boldsymbol{\Omega}_{T^bHO}$;

`// 2.4. Compute the bivariate Poisson log-normal density in test sites;`

for site $i \in v$ **do**

for pairs of species (j, k) **do**

$p_{PLN}\left((Y_{ij}^v, Y_{ik}^v); \boldsymbol{\Gamma}_r^{-v}, T_{r,b}^{-v}\right)$ with means $\boldsymbol{x}_i^\intercal \boldsymbol{\theta}_{r,j}^{-v}$ and $\boldsymbol{x}_i^\intercal \boldsymbol{\theta}_{r,k}^{-v}$ and variance matrix $[\boldsymbol{\Sigma}_{T^bO}]_{[jk,jk]}$

`// 2.5. Compute the average;`

$$PCL_{rvb}(\boldsymbol{Y}^v, \boldsymbol{\Gamma}_r^{-v}, T^b) = \frac{1}{m_v}\sum_{i=1}^{m_v}\sum_{j<k}\log p_{PLN}\left((Y_{ij}^v, Y_{ik}^v); \boldsymbol{\Gamma}_r^{-v}, T_{r,b}^{-v}\right)$$

`// 3. AVERAGE OVER SUBSETS;`

$$PCL_r(\boldsymbol{Y}) = \frac{1}{V}\sum_v PCL_{rv}(\boldsymbol{Y}^v, \boldsymbol{\Gamma}_r^{-v}).$$

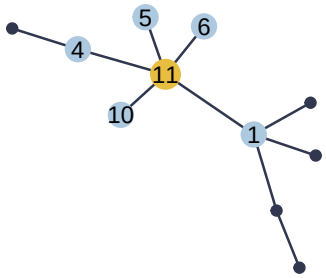
3.B Vignette for nestor

nestor (Network inference from Species counTs with missing actORs) is an R package for the inference of species interaction networks from their observed abundances, accounting for possible unobserved missing actors in the network.

3.B.1 Simulation and preparation

The function `generate_missing_data()` of `nestor` simulates data with missing actors. It requires the desired `type` of network structure (scale-free, erdos, tree or cluster) and the number of missing actors `r`. Here is an example with `r=1` for the scale-free structure:

```
library(nestor)
p=10
r=1
n=100
data=generate_missing_data(n, p, r,type="scale-free", plot=TRUE)
```



The true original set of the missing actor neighbors is available in the value `TC`:

```
data$TC
#> [[1]]
#> [1]  1  4  5  6 10
```

The data is then prepared for analysis with the first step of the procedure: fit the PLN model. The `norm_PLN()` function is a wraper to `PLNmodels::PLN()` which normalizes all the necessary outputs, namely the mean, variance and correlation matrices of the model latent Gaussian layer corresponding to observed species.

```
PLNfit<-norm_PLN(data$Y)
MO<-PLNfit$MO
SO<-PLNfit$SO
sigma_obs=PLNfit$sigma_O
```

3.B.2 Inference

Single clique initialization

The VEM algorithm then needs to be initialized. One way is to find an initial clique of neighbors for the missing actor, for example using the `FitSparsePCA()` function:

```
initClique = FitSparsePCA(data$Y,r=1, min.size = 3)$cliques
initClique
#> [[1]]
#> [1]  2  5  7  9 10
```

The `min.size` parameter defines the minimal size of the output clique. The function `init_mclust()` is also available for finding a single clique, it uses `mclust::Mclust()`.
Once an initial clique has been found, the algorithm can be initialized. This is the aim of the function `initVEM()`, which initializes all required parameters. This function builds one initialization from one initial clique. We initialize with the clique previously identified:

```
initList = initVEM(data$Y, cliqueList=initClique, sigma_obs, M0, r=1 )
```

Then to set the tempering parameter `alpha`, we can look at the output of the `alphaMax()` function.

```
alphaMax(p+r, n)
#> [1] 0.3000768
```

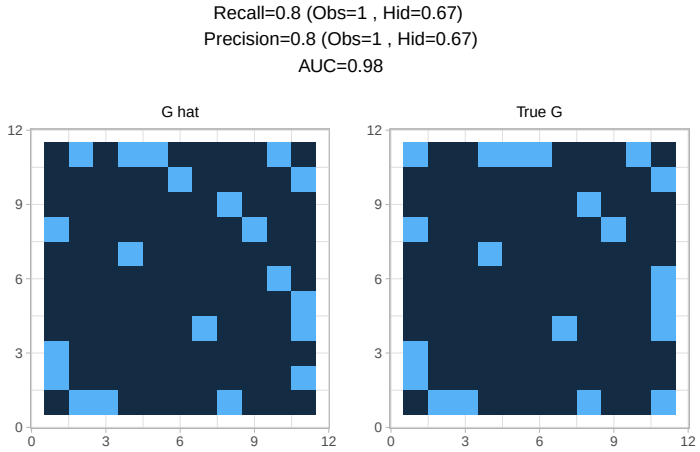
The actual tempering parameter should be lower than the upper bound given by `alphaMax()`. Here we set `alpha` to 0.1. The core function `nestor()` can now be run as follows:

```
fit = nestor(data$Y, M0,S0, initList=initList, alpha=0.1, eps=1e-3,
             maxIter=30)
#>
#> nestor ran in 1.078secs and 24 iterations.
```

The object `fit` contains inferred means and variances of the complete data, as well as edges weight and probability matrices.

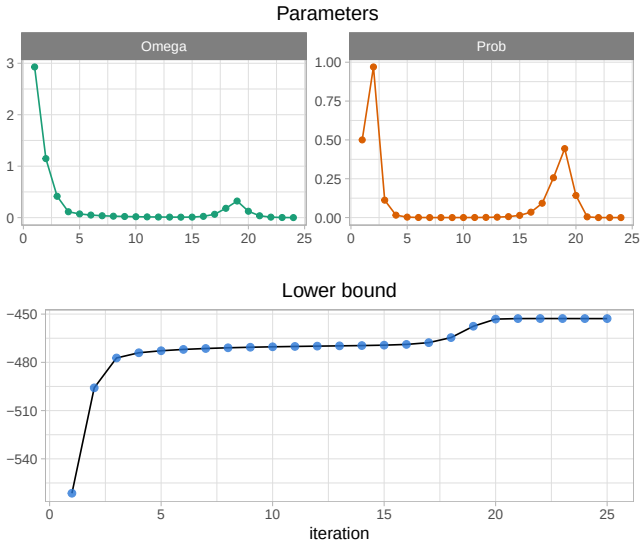
This package contains several visualization functions. `plotPerf()` gives a quick overview of the fit performance compared to initial graph:

```
plotPerf(fit$Pg, data$G,r=1)
```



The convergence of `nestor()` can be checked with the plotting function `plotConv()`:

```
plotConv(nestorFit = fit)
```



Initialization with a list of cliques

The fit of the `nestor()` function is very sensitive to the initialization, and so it is recommended to try several initial cliques. Several functions are available for finding a list of possible starting points:

— `init_blockmodels()` uses the R package blockmodels,
— `boot_FitSparsePCA()` is a bootstraped version using `sparsepca::spca()`,
— `complement_spca()` looks in the complement of the `sparsepca::spca()` output.

Note that several fits of the function `init_mclust()` will also yield a list of cliques. Here we use the `complement_spca()` function, which uses the sparsepca package and returns the cliques corresponding to the `k` first principal components as well as their complement.

```
six_cliques = complement_spca(data$Y, k=3)
six_cliques
#> [[1]]
#> [[1]][[1]]
#> [1] 2 9
#>
#> [[2]]
#> [[2]][[1]]
#> [1]  7 10
#>
#> [[3]]
#> [[3]][[1]]
#> [1]  2  5 10
#>
#> [[4]]
#> [[4]][[1]]
#> [1]  1  3  4  5  6  7  8 10
#>
#> [[5]]
#> [[5]][[1]]
#> [1] 1 2 3 4 5 6 8 9
#>
#> [[6]]
#> [[6]][[1]]
#> [1] 1 3 4 6 7 8 9
```

This package provides with a parallel procedure for the computation of several fits of `nestor()` corresponding to a list of possible cliques, with the function `List.nestor()`. Below is an example with the list of six cliques previously obtained with the `complement_spca()` function:

```
fitList=List.nestor(six_cliques, data$Y, sigma_obs, M0,S0,r=1,
eps=1e-3, maxIter = 50, alpha=0.1, cores=1)
```

The object `fitList` is simply the list of all the `nestor()` fits. This procedure aborts in case of degenerated behaviour, which happens when the provided clique is too far from truth. Wrong fits can be identified by their small ouput size:

```
do.call(rbind,lapply(fitList, length))
#>      [,1]
#> [1,]    3
#> [2,]   12
#> [3,]    3
#> [4,]   12
#> [5,]   12
#> [6,]   12
```

Finally, as this is an example on simulated data we can assess the performance of each converged fit with their AUC, Precision and Recall regarding the hidden node `h`, and the correlation between the inferred means and the original latent Gaussian vector of `h`.

```
do.call(rbind,lapply(fitList, function(vem){
  if(length(vem)>4){
    perf=ppvtpr(vem$Pg, data$G, r=r)
    c(auc=auc(vem$Pg, data$G),precH=perf$PPVH, recH=perf$TPRH,
    corMH=cor(vem$M[,p+r], data$UH))
  }
})) %>% as_tibble()
#> # A tibble: 4 x 4
#>     auc precH  recH  corMH
#>   <dbl> <dbl> <dbl>  <dbl>
#> 1  0.81  0.6   0.5  -0.785
#> 2  0.93  0.75  1    -0.815
#> 3  0.76  0.5   0.67 -0.715
#> 4  0.69  0.43  0.5  -0.585
```

3.C Clique initialization

It has been previously shown that the initial clique of the missing actor neighbors is paramount for the developed method. Many initialization strategies can be designed, in this section we present three of them that are implemented in the nestor package.

3.C.1 Sparse PCA

The sparse PCA is a specific kind of PCA where the principal components are linear combinations of only a few other variables. This principle is coherent with the idea of missing actors with a local effect, whereas a classic PCA would look for a global missing actor affecting all other species. The sparse PCA is run directly on the

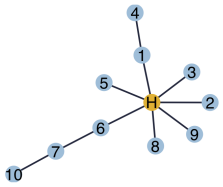


Figure 3.9 – A scale-free graph with 11 nodes.

observed counts. The possible clique of neighbors for the missing actor is then the set of variables associated with an axis of the PCA.
As it is a sparse method, the resulting clique might be too small and it is interesting to control the minimal size. For example for a dataset simulated from the scale-free graph in Figure 3.9 with the true clique $C^* = \{1, 2, 3, 5, 6, 8, 9\}$, running a sparse PCA with no constraint yields the clique $C = \{3, 9\}$, whereas imposing a minimal size of 4 yields $C = \{1, 2, 3, 4, 5, 6, 9\}$.

3.C.2 Correlation clustering

A missing actor creates correlation among the other variables, which can be identified by looking for groups in the correlation structure. One way to observe the latter is to consider the plan defined by the two first eigen vectors of the correlation matrix, and to look for clusters in this space. Most clustering methods are designed to separate the variables into k groups, whereas here the aim is to find k groups (for k missing actors) which do not involve all the variables. To do this, we use the mclust package which allows to specify a model for the clustering while including some noise.
The aforementioned vector space is actually the correlation circle of the variables. Noise points are uniformly simulated on the circle, then the desired number of groups is found using mclust. The clustering is performed in polar coordinates to take negatively correlated variables into account. In the example of Figure 3.10, the clique of original neighbors is correctly identified, however this result is subject to the randomness of the noise. It is therefore best to run the method a few times and identify a list of cliques.

3.C.3 Structure clustering

Inferring the network from observed data only is equivalent to marginalizing the complete network on the missing actors. The neighbors of the missing actors then form a clique which can be identified as a block in the adjacency matrix of the marginal graph. The package blockmodels is an implementation of the Stochastic

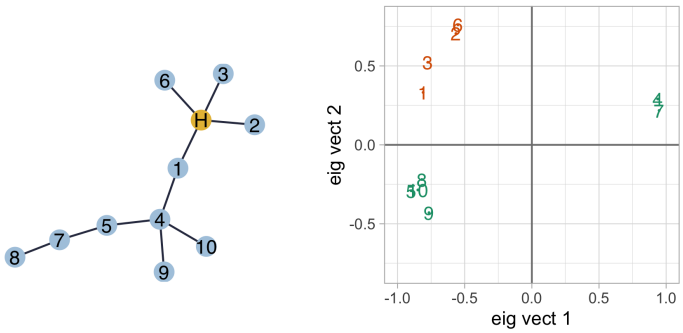


Figure 3.10 – *Left*: original scale-free graph with 11 nodes among which one is unobserved, with clique of neighbors $C^* = \{1, 2, 3, 6\}$. *Right*: corresponding variable correlation circle, the group found by mclust is highlighted in orange.

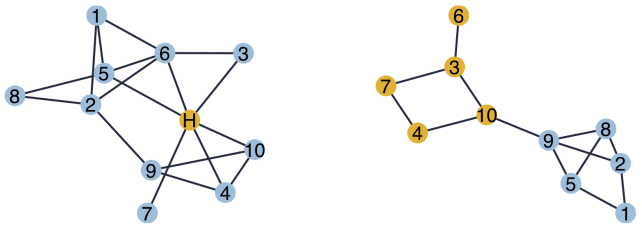


Figure 3.11 – *Left*: original cluster graph with 11 nodes among which one is unobserved, with clique of neighbors $C^* = \{3, 4, 5, 6, 7, 10\}$. *Right*: inferred marginal network and the two groups identified by blockmodels.

Block Model. Running it on the adjacency matrix obtained with nestor or EMtree (when nestor degenerates) with a Bernoulli model provides with a separation of the nodes into a given number of blocks, which are the possible cliques.

Figure 3.11 is a simulated example on a cluster graph with 11 nodes. Count data with one missing actor is generated with this conditional dependency structure, and the inferred marginal network is given as an input to blockmodels. We ask for a separation in 2 groups; the first group identified consists of 5 of the 6 original neighbors.

This strategy gives a list of disjoint cliques and it is likely that some of them will cause nestor to degenerate, for they would not contain original neighbors. On real datasets, it is recommended to test different number of groups and consider the list of all the resulting cliques.

3.D Comparison with PLN-network and EMtree

This section compares the network inference methods from count data which build on the estimation of the PLN model as described in Chiquet et al. (2018): PLN-network (Chiquet et al., 2019a), EMtree and nestor without missing actors. The R package PLN-network performs a sparse estimation of the precision matrix using the glasso. Details about EMtree and nestor are available respectively in Chapter 2 and Chapter 3 of this work.

3.D.1 EMtree and nestor

Different models

The aim of EMtree is the network inference, and as such it focuses on the update of the tree distribution parameters (the edges weights $\boldsymbol{\beta}$). It thus performs network inference without ever considering the precision matrix, which is the main difference with nestor. Indeed, the inference of missing actors requires the estimation of their means M_H, which directly depends on terms of the proxy for the precision matrix ($\overline{\boldsymbol{\Omega}}$). M_H is actually the parameter through which passes the information from the data to the network parameters $\widetilde{\boldsymbol{\beta}}$, therefore the computation of the edges probabilities is also different.
EMtree approximates the edges probabilities conditional on $\boldsymbol{Y}$ by applying the Matrix Tree theorem on the matrix $\boldsymbol{\beta} \odot \boldsymbol{\psi}$, where $\boldsymbol{\psi} = \left((1-\rho_{kl}^2)^{-n/2}\right)_{kl}$ with ρ_{kl} the estimated correlation between variables k and l. Transposed in the variational inference framework, edges probabilities conditional on $\boldsymbol{Y}$ are approximated by the edges probabilities computed from the distribution $g(T)$, which approximates $p(T \mid \boldsymbol{Y})$. The distribution $g(T)$ is parametrized with the variational edges weights $\widetilde{\boldsymbol{\beta}}$, which update formula for the edges kl at step $t+1$ is given in equation (3.12):

$$\widetilde{\beta}_{kl}^{t+1} = \beta_{kl}^{t+1} \left|\boldsymbol{R}_{[kl]}^{t+1}\right|^{-n/2} \exp\left(-\omega_{kl}^{t+1}\left[(\mathbf{M}^t)^\intercal \mathbf{M}^t\right]_{kl}\right).$$

Now noticing that $\left|\boldsymbol{R}_{[kl]}\right|^{-n/2} = \psi_{kl}$, we can link this formula to that of the quantity used by EMtree to compute probabilities. We see that the difference between the two strategies is the term $\exp\left(-\omega_{kl}^{t+1}\left[(\mathbf{M}^t)^\intercal \mathbf{M}^t\right]_{kl}\right)$, which stems from the modeling difference of EMtree and nestor.

Different behaviors

The modeling difference induces differences in behaviors between the two algorithms. The extra exponential term in the formula for the edges weights of nestor is directly dependent on the data dimensions, and mechanically implies a high variability and a wide range of values for the weights. This causes numerical instabilities, especially

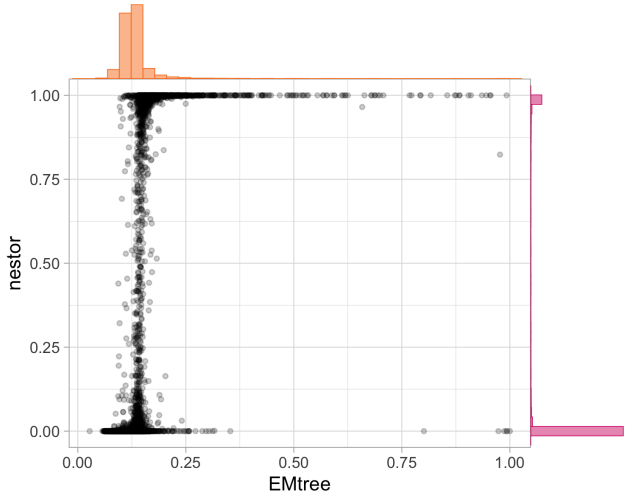


Figure 3.12 – Relationship between edges probabilities computed by nestor and EMtree for cluster graphs of dimension $p = 15$ and density $5/p$.

when manipulating the determinant and inverse of the weights Laplacian matrix. That's the reason why nestor is very sensitive to the initialization and value of the tempering parameter.
On the other hand EMtree is a robust procedure. However its estimation involves less information and the output probabilities are less discriminant than that of nestor. Figure 3.12 illustrates this fact in the case of dense cluster graphs.

3.D.2 Performance comparison

Experiment

We compared the three methods ability to infer networks based on AUC, precision and recall criteria. We simulated 100 datasets with $n = 200$ samples and $p = 15$ species for erdos (Erdös-Reyni) and cluster structures, and tested two density levels ($3/p$ and $5/p$). To give an efficient presentation of Precision-Recall curves, mean curves were computed, that is all graphs of a certain type and density level were pooled together and precision and recall measures computed on the corresponding pooled results of each method.
The edges scores of PLN-network are extracted from its regularization path as the sum of all penalties that do not cancel an edge. The grid of penalties is parametrized with $min.ratio = 1 \cdot 10^{-3}$, after checking that the PLN-network fit on some typical simulated graphs visited the best model (reach of a maximum on the BIC curve). For EMtree we consider edges probabilities obtained without the resampling step, rather than edges selection frequencies. The nestor method is parametrized with the tempering parameter set to 0.1. With density $3/p$, nestor degenerated for 25 clusters and 5 erdos graphs, and for 2 cluster graphs with density $5/p$. Those datasets were

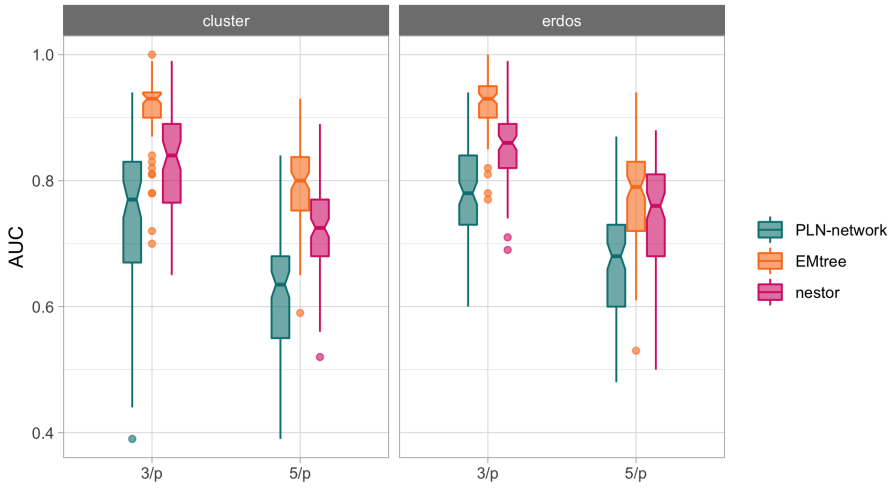


Figure 3.13 – AUC measures obtained with PLN-network, EMtree and nestor on 100 datasets of each type and density level.

removed from the following results.

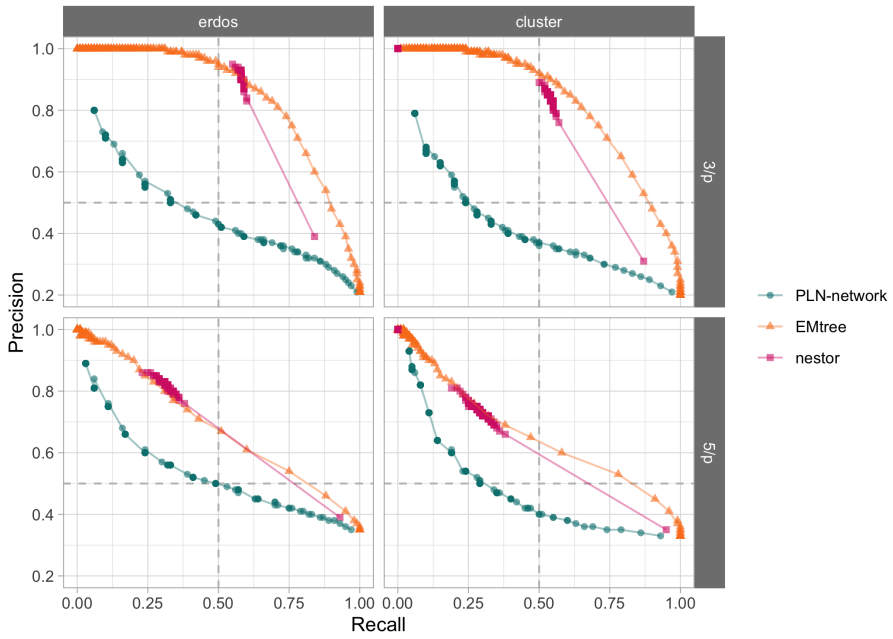


Figure 3.14 – Mean Precision-Recall curves obtained with PLN-network, EMtree and nestor on 100 datasets of each type and density level.

Results

AUC measures are gathered in Figure 3.13. All types and density levels show the same ranking of methods: the best AUC are obtained with EMtree, and the

weakest with PLN-network which however stays above 60% in median. The median values of nestor are closer to those of EMtree than to PLN-network. The higher density levels represent a greater difficulty for all methods, and cluster graphs seem slightly harder than erdos ones for nestor and PLN-network.

From the mean Precision-Recall curves available in Figure 3.14, we see than EMtree also shows the best trajectory. PLN-network seems to never visit the value pairs above 0.5, and its trajectory is not affected by the density level. The trajectory of nestor is close to that of EMtree, and they both greatly deteriorate with the density level. Due to its highly discriminant probabilities, nestor only visits a few value pairs, which are above 0.5 for the $3/p$ density level with a Precision of at least 0.8 and a Recall of 0.5 to 0.6. For denser graphs, the Recall decreases below 0.5 and the Precision remains above 0.7, making nestor a very conservative method.

4

PERSPECTIVES

Contents

The previous chapters detailed the proposed methodology for network inference from incomplete abundance data. The species abundances $\boldsymbol{Y}$ are jointly modeled in a GLMM using the Poisson log-normal distribution, thus taking advantage of its properties to account for offsets and experimental and/or environmental covariates $\boldsymbol{X}$. The problem of network inference is then transposed to the Gaussian latent layer of model parameters $\boldsymbol{Z}$, where it is performed using averaging on spanning-tree structures with decomposable distribution on trees parametrized with edges weights $\boldsymbol{\beta}$. Missing actors are inferred in the Gaussian layer as well, simultaneously with the network. We remind the mathematical formulation of this model:

$$\begin{cases} T \sim \prod_{kl \in T} \beta_{kl} / B, \quad B = \sum_{T \in \mathcal{T}} \prod_{kl \in T} \beta_{kl}, \\ \boldsymbol{Z}_i \mid T \sim \mathcal{N}(0, \boldsymbol{\Omega}_T), \quad \{\boldsymbol{Z}_i\}_i \text{ iid}, \\ Y_{ij} \mid \boldsymbol{Z}_i \sim \mathcal{P}(\exp(o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + Z_{ij})), \ (Y_{ij} \perp\!\!\!\perp) \mid \boldsymbol{Z}_i. \end{cases}$$

The qualities of this approach have been demonstrated on simulated examples as well as empirical datasets. But why does it work ? Mostly because exact computations are possible in the Gaussian layer, thanks to its maniability and the structure and algebraic properties of spanning trees. This flexibility calls for some natural extensions which were not developed due to time constraints, that we now present together with specific details of this model.

4.1 Unresolved questions

Offset modeling

The presented methodology allows to take into account covariates and offsets, which is paramount in the modeling of experiments. A change in the covariates included can greatly modify the inferred network, however this is an information that can be controlled. On the other hand, not including offsets amounts to model incoherent data and therefore yields inconsistent results. Unfortunately, offsets are not always obvious or easy to get, and evaluating them possibly represents a modeling step in itself. For example in an ecological census of fauna, the offset has to at least account for the period of time of observation, the experience of the observer and the species detectability, which is obviously the most difficult to get. Detectability depends on the relative position of the species to the observer, and can depend on the environmental conditions as well, making its modeling a delicate step.

Algorithm initialization

The approach can also infer missing actors as detailed in Chapter 3. The initialization of the algorithm for this inference requires an initial clique of neighbors for each missing actor. This parameter is critical, as it defines all the initialization and if it is too far from the true clique, that is if not a minimum of true neighbors are included, the procedure can degenerate and abort. Several methods for setting this parameter are proposed in the nestor package. They are based on finding groups of highly correlated variables (sparse PCA and model-based clustering of the estimated correlation matrix), or finding clusters in the structure of the marginal graph (using Stochastic Bloc Models). The estimated lower bound of the likelihood can be used to choose between a large set of results with different initializations, thus one way to go is simply to sample several starting points (possibly by boostrapping the previous methods) and test all of them. Many other methods could be used, as well as prior knowledge on species, and we do not conclude on the best way to initialize the cliques. However we observed that it is less important for the inference quality to be precise than to include some true neighbors in the clique. Therefore we advise to choose the initial clique also based on its size.

Model selection

One key question when working with possibly incomplete data is to know if it is indeed incomplete, and to what extent. This amounts to model selection to chose

the number of missing actors r. Usually model selection is carried out with computations on the likelihood of each model. The model developed involves several latent parameters, and the inference is therefore performed in a variational framework which yields a lower bound of the likelihood. Moreover as the method takes advantage of the variational estimation of the PLN model described in Chiquet et al. (2018), the quantity that is computed is actually an approximation of this lower bound. This results in the usual model selection tools, which originally apply to likelihoods, to not work (Akaike/Bayesian/Extended Bayesian Information Criterion). Then the only strategy left is to resort to cross-validation, which is greedy and not very satisfying either. There is a need of adapted criteria for models with variational inference, which is still an open research question of statistical theory.

4.2 Extensions of the adopted approach

In this section, we present direct extensions of the developed methodology regarding the network in itself (estimation of its parameters, network comparison), and the data nature (processing of different data types and spatialized data).

4.2.1 Characterizing species interactions

Generally, species interactions are characterized by their sign and strength. This information is contained in the partial correlation matrix, defined as $\boldsymbol{R}_p = (\rho_{jk})_{1 \leq j,k \leq p}$ with

$$\rho_{jk} = \frac{-\omega_{jk}}{\sqrt{\omega_{jj}\omega_{kk}}},$$

which is simply the opposite of the correlation matrix built from the precision matrix $\boldsymbol{\Omega}$. Some methods are thus dedicated to the estimation of the partial correlation matrix to build knowledge on an ecosystem functioning. In the context of Gaussian Graphical Models, Lauritzen (1996) develops maximum likelihood estimators for the precision matrix which depends on that of the covariance matrix, and the structure of the graph, as detailed in section 1.2.2. It thus requires the precise knowledge of the graph, and even more than that: the only terms of the covariance matrix that are correctly estimated are those corresponding to edges in the graph. Therefore the prior inference of the structure is necessary. Moreover, these MLE are only valid for decomposable graphs, therefore using Proposition 1.1 the network might need to be triangularized as illustrated in Figure 4.1. Therefore following these three steps:

1. Infer the conditional dependency structure $\mathcal{G}$,
2. Make $\mathcal{G}$ a chordal graph,
3. Compute $\widehat{\boldsymbol{\Omega}}_{MLE}$.

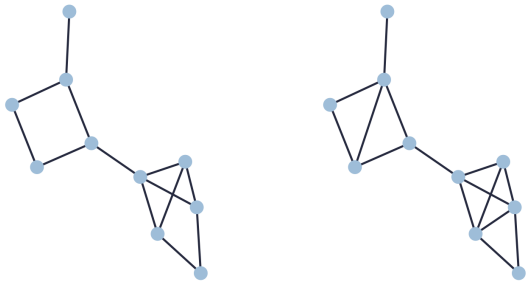


Figure 4.1 – Example of triangularization. *Left*: graph $\mathcal{G}$. *Right*: a chordal version of $\mathcal{G}$: maximal loops are of size 3.

would lead to an approximation of $\mathbf{\Omega}$. Note that there is not a unique solution to step 2. If the inferred network is already chordal, this yields the MLE of $\mathbf{\Omega}$, available after the network inference.

4.2.2 Network comparison

Comparing networks has many applications and yields results of interest in various fields, regarding for example the evolution of a system across seasons, or how a pathogen modifies the organization of a microbiome.

The comparison from a statistical point of view is linked to testing differences, or questioning the independence of two networks for example. Rather than defining statistical tests based on networks distributions, current strategies consists in summarizing networks with vectors of measures on nodes. These are called embeddings, which aim to capture the structure of the graph and the relationships between the nodes in a concise manner (e.g. Chen and Koga (2019); Tsitsulin et al. (2018)). Embeddings can then be compared using PCA for example. The presented approach assumes the graph is a latent tree, which estimated distribution can be seen as another available graph summary.

A first step toward statistical network comparison using trees could be to compare the estimated laws on trees, for example using the Küllback-Leibler divergence which is easily written with tree distributions and gives two interesting quantities.

First, it is possible to compute the contribution of a specific edge kl to the distribution. We denote $p_{\boldsymbol{\beta}_{\setminus kl}}$ the decomposable tree distribution where the weight β_{kl} is set to zero. Its normalization constant is $B_{\setminus kl} = \sum_{T \in \mathcal{T}} \prod_{uv \neq kl} \beta_{uv}$. Computing the Küllback-Leibler divergence between $p_{\boldsymbol{\beta}}$ and $p_{\boldsymbol{\beta}_{\setminus kl}}$ yields the contribution of kl

to $p_{\boldsymbol{\beta}}(T)$. It writes:

$$\begin{aligned} KL(p_{\boldsymbol{\beta}}(T) \,||\, p_{\boldsymbol{\beta}_{\setminus kl}}(T)) &= \mathbb{E}_{p_{\boldsymbol{\beta}}(T)}[\log p_{\boldsymbol{\beta}}(T) - \log p_{\boldsymbol{\beta}_{\setminus kl}}(T)] \\ &= P_{kl} \log \beta_{kl} + \log(B_{\setminus kl}/B). \end{aligned}$$

Where $P_{kl} = \mathbb{E}_{p_{\boldsymbol{\beta}}}[\mathbb{1}\{kl \in T\}]$ is the probability for the edge kl to be in tree T under the tree distribution $P_{\boldsymbol{\beta}}$. Now noticing that $B_{\setminus kl}/B = 1 - P_{kl}$, we finally get the following contribution of kl:

$$KL(p_{\boldsymbol{\beta}}(T) \,||\, p_{\boldsymbol{\beta}_{\setminus kl}}(T)) = P_{kl} \log \beta_{kl} + \log(1 - P_{kl}).$$

As the distribution under which the expectation is taken must be dominant over the other, the divergence $KL(p_{\boldsymbol{\beta}_{\setminus kl}}(T) \,||\, p_{\boldsymbol{\beta}}(T))$ is not defined.

Then, if all edges weights are finite and non-null, the divergence can be symmetrized into a so-called dissimilarity measure to compare two networks. Assuming the data is observed under two conditions A and B, we consider the edges weights matrices estimated from each subsets $\boldsymbol{\beta}^A$ and $\boldsymbol{\beta}^B$. The dissimilarity measure between the tree distributions $p_{\boldsymbol{\beta}^A}(T)$ and $p_{\boldsymbol{\beta}^B}(T)$ writes:

$$D(p_{\boldsymbol{\beta}^A}(T), p_{\boldsymbol{\beta}^B}(T)) = \frac{1}{2}\left[KL\big(p_{\boldsymbol{\beta}^B}(T) \,||\, p_{\boldsymbol{\beta}^A}(T)\big) + KL\big(p_{\boldsymbol{\beta}^A}(T) \,||\, p_{\boldsymbol{\beta}^B}(T)\big)\right].$$

After computation, this actually simplifies into the following expression, where $P^A_{kl} = \mathbb{E}_{p_{\boldsymbol{\beta}^A}}[\mathbb{1}\{kl \in T\}]$ is the probability for the edge kl to be in tree T under the tree distribution $p_{\boldsymbol{\beta}^A}$:

$$D(p_{\boldsymbol{\beta}^A}(T), p_{\boldsymbol{\beta}^B}(T)) = \sum_{kl} \log(\beta^A_{kl}/\beta^B_{kl}) \left(\frac{P^A_{kl} - P^B_{kl}}{2}\right).$$

Finally if we consider K different conditions, the K tree distributions could be compared by computing the $K \times K$ matrix of dissimilarity measures, provided that all edges weights are finite and non-null.

4.2.3 Other data types

This work focuses on abundance data, however the network inference method developed here could be extended to other data types. Indeed, the strategy of tree averaging is based on the Gaussian Layer of the $\boldsymbol{Z}$. Therefore it could also be used with a model which would keep this layer and add a different emission law than the

Poisson to generate data $\boldsymbol{Y}$, that is considering the model

$$\begin{cases} T \sim \prod_{kl \in T} \beta_{kl}/B, & B = \sum_{T \in \mathcal{T}} \prod_{kl \in T} \beta_{kl}, \\ \boldsymbol{Z}_i \mid T \sim \mathcal{N}(0, \boldsymbol{\Omega}_T), & \{\boldsymbol{Z}_i\}_i \text{ iid}, \\ Y_{ij} \mid \boldsymbol{Z}_i \sim F(o_{ij}, \boldsymbol{x}_i, Z_{ij}), & (Y_{ij} \perp\!\!\!\perp) \mid \boldsymbol{Z}_i \end{cases} .$$

For example, choosing the emission law F as binomial, multinomial or a Tweedie distribution (Tweedie, 1984) would respectively yield presence/absence, ordinal, or positive continuous data (e.g. to model biomass). Mixed data could also be generated, as long as the emission law parameters are stored in the Gaussian layer $\boldsymbol{Z}$.

Another interesting extension is for F to be multidimensional. Nodes of the network would then be vectors of information known as multi-attribute variables. This can be the case when multiple measures on the same system are available. For example in microbiology, different information can be measured on a set of genes such as transcriptomics, proteomics, or metabolomics data. Then these multiple sources can be integrated to infer the gene regulatory network (Chiquet et al., 2019b; Siahpirani et al., 2019).

Choosing another emission law would require to estimate its parameters, which can be a sensitive task. However the network inference mechanics in the Gaussian layer using tree averaging would remain the same.

4.2.4 Spatial dependence

In ecological surveys, environmental conditions might be spatially heterogeneous (e.g. different climate at different altitudes). This translates in spatial correlation in the observed data, which has to be accounted for in the network inference.

Spatial correlation is due to spatially close environments being similar. Therefore a first and quick way to correct for spatial dependence is to include environmental covariates encoding the spatial proximity (e.g. spatial coordinates) in the model. For example Figure 4.2 shows how a variogram may be flattened by adjusting for latitude and longitude. However such correction might not be enough and the residuals from the adjustment might still be spatially correlated. Spatial coordinates could be adjusted in a more complicated way than the linear relationship, but another solution is to consider this dependence to be the result of an unobserved variable. As a missing variable, the spatial dependency can be taken into account to unravel the direct dependencies among species by using the method detailed in Chapter 3. Adjusting for spatial covariates corrects for the spatial effects on the means, whereas including spatial dependence as a node in the interaction network corresponds to a correction of the variances.

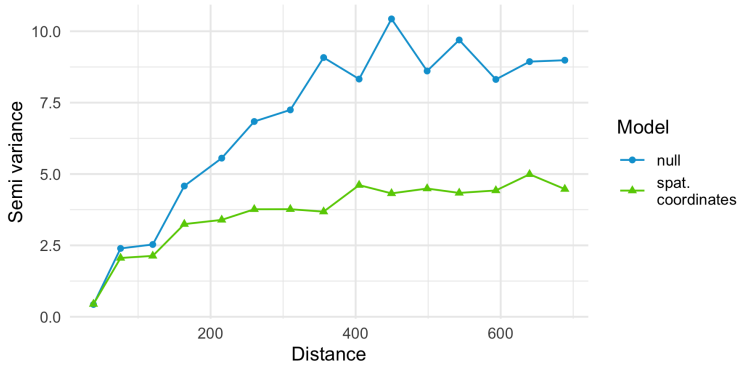


Figure 4.2 – Variogram of the species Trisopterus esmarkii (Norway pout) from the Barents dataset with and without adjusting linearly for the spatial coordinates.

Another way of correcting the variances for spatial dependencies is to include spatial variances directly in the model. To do this a first approach is to assume a separation of the dependencies due to the spatial effects stored in the matrix $\Gamma = (\Gamma_{st})_{1\leq s,t\leq n}$ on the one hand, and due to the species interactions stored in the matrix $\Sigma = (\sigma_{jk})_{1\leq j,k\leq p}$ on the other hand. To simplify we consider either pairs of species j and k on the same site s or a same species j on pair of sites s and t. The Gaussian parameters are then modeled as follows:

$$(Z_{sj}, Z_{sk}) \sim \mathcal{N}(0, \gamma_{ss}\Sigma_{[j,k]}),$$
$$(Z_{sj}, Z_{tj}) \sim \mathcal{N}(0, \sigma_{jj}\Gamma_{[s,t]}).$$

That way we obtain $\mathbb{C}\mathrm{ov}\,(Z_{sj}, Z_{sk}) = \gamma_{ss}\sigma_{jk}$ and $\mathbb{C}\mathrm{ov}\,(Z_{sj}, Z_{tj}) = \sigma_{jj}\gamma_{st}$. This is equivalent to consider the transformation of the matrix $\boldsymbol{Z}$ into the vector $Vec(\boldsymbol{Z}) = (Z_{11}, ..., Z_{1p}, Z_{21}, ..., Z_{np}) \in \mathbb{R}^{n\times p}$ to be distributed as:

$$Vec(\boldsymbol{Z}) \sim \mathcal{N}(0, \Gamma \otimes \Sigma).$$

As the covariance structure is different for a pair of sites or a pair of species, this model requires to use the bivariate Poisson log-normal distribution for the count data $\boldsymbol{Y}$. It is then possible to write a pairwise composite log-likelihood which involves all the variance parameters:

$$c\ell_\theta(\boldsymbol{Y}) = \sum_{j=1}^{p}\sum_{s<t} \log p_\theta(Y_{sj}, Y_{tj}) + \sum_{s=1}^{n}\sum_{j<k} \log p_\theta(Y_{sj}, Y_{sj}).$$

The number of parameters for the variance is $\frac{1}{2}(n^2+p^2)$, which is quite large. Classically to reduce this number, the Γ matrix is parametrized with a spatial covariance function. For example the Matérn functions (which include the exponential covariance function and others) define the covariance between two points distant of d

units from each other as a function of d and three other parameters (Cressie and Wikle, 2015).
Performing network inference using tree averaging in this case would simply amount to consider

$$Vec(\boldsymbol{Z}) \sim \mathcal{N}(0, \Gamma \otimes \Sigma_T),$$

then the inference of edges probabilities would go as previously detailed.

4.3 Network inference in the observed layer

All of the above is an adaptation of the presented model to perform network inference in the latent Gaussian layer of parameters. However it can be shown that if the graphical model in the latent Gaussian layer $\boldsymbol{Z}$ is a connected graph, then the marginal graphical model of the observed counts $\boldsymbol{Y}$ is a clique.

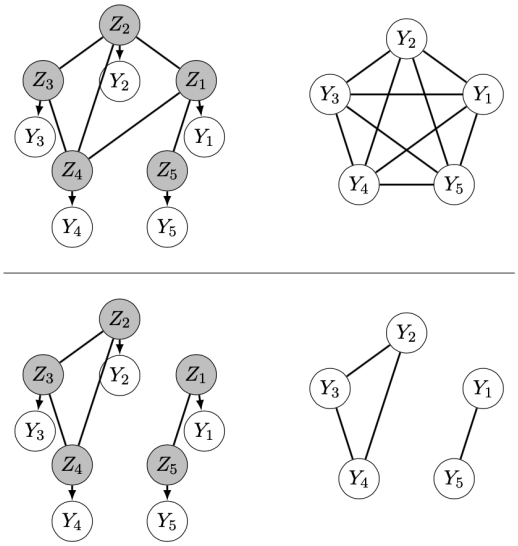


Figure 4.3 – Graphical representation of the joint distribution $p(Z_i, Y_i)$ (left) and marginal distribution $p(Y_i)$ (right) when the graphical model of the latent variables is connected (top) or not (bottom).

As Figure 4.3 illustrates, only a separation in the latent space results in a separation in the observed space.
It is possible to write a model which meets the primary goal of inferring the network directly in the space of observed data $\boldsymbol{Y}$. Considering any marginal and bivariate discrete distribution for counts, assuming a tree-shaped dependency structure for the counts writes:

$$p_\theta(\boldsymbol{Y}_i \mid T) = \prod_{j=1}^{p} p_\theta(Y_{ij}) \prod_{jk \in T} \frac{p_\theta(Y_{ij}, Y_{ik})}{p_\theta(Y_{ij}) p_\theta(Y_{ik})}.$$

Covariates and offsets could be involved as parameters on the distributions means. Then the joint distribution of counts would be a mixture on trees:

$$p_{\beta,\theta}(\boldsymbol{Y}) = \sum_{T \in \mathcal{T}} p_\beta(T) p_\theta(\boldsymbol{Y} \mid T).$$

This model involves only one latent layer: the tree T. With a decomposable distribution on T, the quantity $\mathbb{E}_{\beta,\theta}[\log p_{\beta,\theta}(\boldsymbol{Y}, T) \mid \boldsymbol{Y}]$ of the E step of an EM algorithm could be estimated using the Matrix Tree Theorem. Regarding the M step, the estimation of the weights β would be the same as for the PLN model with network inference in the Gaussian layer. However the estimation of the θ parameter is complex, as its estimator is usually not explicit and is common to several terms of the product on the edges in $p_\theta(\boldsymbol{Y} \mid T)$. The M step is thus the main difficulty for the estimation of this model.

5

Résumé

This chapter is a summary of the present work, written in French.

Les réseaux sont des objets qui permettent de graphiquement représenter des liens entre des entités. Ces outils sont utilisés dans des domaines très variés, allant de l'informatique aux neurosciences, aux sciences sociales et à la biologie. Ce travail s'intéresse aux réseaux d'espèces en écologie et microbiologie (ou méta-génomique), et plus particulièrement à l'inférence des réseaux de dépendances conditionnelles entre espèces d'une même communauté partageant le même environnement. Les réseaux représentent alors les espèces par des noeuds, et leurs liens de dépendances par des arêtes.

L'inférence de réseau considérée ici a pour point de départ des mesures répétées de comptages des espèces d'intérêt, soit un tableau de données de n mesures discrètes de p espèces. Il est important de distinguer ce cas de figure où le réseau n'est pas connu ni observé, de celui où il est possible d'observer les liens du réseau directement, et donc de reconstruire ce dernier via des comptages d'interactions observées, comme c'est classiquement le cas en écologie par exemple. Considérer les relations de dépendances conditionnelles permet à la fois d'obtenir des réseaux parcimonieux et interprétables en ne représentant que des liens directs entre espèces, et d'inférer des liens qui ne sont pas directement observables comme par exemple les dépendances entre des (pseudo-)espèces microscopiques (bactéries, champignons, protéines, virus, gènes, etc.) ou des liens dont la nature les rend difficilement identifiables (par exemple les relations de coopération ou de communication).

Les modèles graphiques sont le cadre mathématique des réseaux de dépendances conditionnelles. Notamment, les modèles graphique gaussiens possèdent des propriétés particulières facilitant leur inférence. Ce cas particulier est un cadre très utilisé pour l'inférence de réseaux en biologie, cependant il n'est pas directement applicable à des données discrètes. Le premier objectif de ce travail est ainsi de développer une méthodologie pour l'inférence de réseaux de dépendances conditionnelles à partir de données de comptages. Par ailleurs, pour assurer la validité des résultats il est nécessaire que les covariables expérimentales ainsi que les offsets mesurés (durées d'observation, profondeur de séquençage) soient inclus dans la modélisation des comptages. L'inférence de réseau en elle-même tire parti des propriétés algébriques des arbres couvrants pour réaliser une exploration efficace et exhaustive de l'espace

des graphes réduit à celui de ces structures particulières.
Il est en outre possible que toutes les espèces ou covariables n'aient pas été mesurées lors de l'expérience. Un réseau inféré à partir de données incomplètes est alors un réseau marginal, qui présente mécaniquement des formations denses entre les espèces liées à un acteur non observé menant à des interprétations biaisées et partielles. Le second objectif de ce travail est d'inclure de possibles acteurs manquants dans l'inférence, afin d'obtenir à la fois le réseau complet et des informations permettant de caractériser et de mieux comprendre les acteurs manquants.

Chapitre 1

Le premier chapitre expose en détails les éléments de théorie invoqués pour la modélisation et l'inférence développées dans les chapitres suivants. Le cadre des modèles graphiques est tout d'abord abordé de manière générale et inspirée de Lauritzen (1996), avant d'introduire les arbres couvrants et leurs propriétés algébriques. Dans un second temps, les particularités des modèles graphiques gaussiens sont présentées, anisi que deux méthodes pour leur inférence. Après un bref exposé de l'inférence dans le cadre de données incomplètes, la dernière partie traite de l'inférence de réseaux à partir de données de comptage et présente plusieurs stratégies issues de la littérature en écologie des communautés et microbiologie.

Modèles graphiques

Définitions générales.

Un graphe $\mathcal{G}$ est constitué d'un ensemble de noeuds V et d'un ensemble d'arêtes E. Pour un triplet (A, B, S) de sous-ensembles disjoints de V, l'ensemble S sépare A et B dans $\mathcal{G}$ si tout chemin allant de A à B intersecte S. La notion de séparation est essentielle pour établir le lien entre graphe et relations d'indépendances conditionnelles au sein d'une variable aléatoire multi-variée.
Soit $\mathcal{G}$ un graphe non-dirigé et $X = (X_v)_{v\in V}$ un vecteur aléatoire à valeurs dans un espace produit $\mathcal{X} = \otimes_{v\in V}\mathcal{X}_v$. Pour tout sous-ensemble A de V, X_A dénote $(X_v)_{v\in A}$.
Propriété (Markov globale). *Une mesure de probabilité sur $\mathcal{X}$ satisfait la propriété de Markov globale par rapport à $\mathcal{G}$ si pour tout triplet (A, B, S) de sous-ensembles disjoints de V, on a*

$$S \text{ sépare } A \text{ et } B \Rightarrow X_A \perp\!\!\!\perp X_B \mid X_S.$$

Un modèle graphique pour X est alors tout graphe tel que la distribution de X soit globale Markov par rapport à ce graphe. La propriété de Markov globale est seulement une implication, ce qui signifie qu'il est autorisé qu'un modèle graphique

n'inclue pas toutes les relations d'indépendances conditionnelles. Lorsque que la relation est une équivalence, la distribution de X est dite fidèle Markov, et le graphe associé représente exactement toutes les relations d'indépendances conditionnelles dans X.
Si S sépare A et B et si de plus tous les noeuds dans S sont liés enter eux (S est complet), le triplet (A, B, S) est alors une décomposition propre de $\mathcal{G}$. Cette notion permet de définir un graphe décomposable.
Définition (Graphe décomposable). *Un graphe $\mathcal{G}$ est dit décomposable s'il existe une décomposition propre (A, B, C) sur $\mathcal{G}$, et si les sous-graphes définis par $A \cup B$ et $B \cup C$ sont eux-mêmes décomposables.*
Les modèles graphiques à graphes décomposables permettent de structurer l'écriture des paramètres de variance sur des ensembles de variables complets, ce qui facilite leur estimation.

Arbres couvrants.

Parmi l'ensemble des graphes, les arbres couvrants sont les structures les plus parcimonieuses, et les structures sans cycles les plus denses. Naturellement décomposables, ils sont pratiques à manipuler grâce à leurs propriétés algébriques. L'espace des graphes non-dirigés est de taille super-exponentielle ($2^{p(p-1)/2}$ graphes possibles pour p noeuds). L'espace des arbres couvrants, bien que plus petit, reste combinatoirement grand (p^{p-2} arbres pour p noeuds). Il est cependant possible de sommer sur cet espace en $\mathcal{O}(p^3)$ opérations. Ce théorème est l'extension aux réels du théorème de Kirchhoff (Chaiken and Kleitman, 1978; Meilă and Jaakkola, 2006), connu sous le nom de théorème arbre-matrice ou des mineurs égaux.

Théorème (Arbre-matrice). *Soit une matrice de poids symétrique $\mathbf{W} = (w_{jk})_{jk}$ dont les entrées sont dans $\mathbb{R}^+$. Alors la somme sur l'ensemble des arbres couvrants du produit des poids de leurs arêtes est égal à n'importe quel mineur du Laplacien $\mathbf{Q}$ de $\mathbf{W}$. Formellement, pour tout $1 \leq u, v \leq p$:*

$$W := \sum_{T \in \mathcal{T}} \prod_{jk \in T} w_{jk} = |\mathbf{Q}^{uv}|.$$

Ce théorème fait apparaître une forme somme-produit sur l'espace des arbres couvrants. Le lemme technique qui suit, établi par Meilă and Jaakkola (2006), permet de dériver une telle somme. On rappelle que le Laplacien d'une matrice est la matrice dont les termes diagonaux (resp. extra-diagonaux) sont les sommes par lignes (resp. l'opposé des termes extra-diagonaux) de la matrice de départ.

Lemme (Dérivation d'une somme-produit sur les arbres). *Soit une matrice de poids symétrique $\mathbf{W} = (w_{jk})_{jk}$ dont les entrées sont dans $\mathbb{R}^+$. $\mathbf{Q}^{11}$ dénote le mineur [1,1]*

de son Laplacien. Soit alors la matrice $\mathbf{M}$*, définie termes à termes comme suit :*

$$[\mathbf{M}(\mathbf{W})]_{jk} = \begin{cases} \left[(\mathbf{Q}^{11})^{-1}\right]_{jj} + \left[(\mathbf{Q}^{11})^{-1}\right]_{kk} - 2\left[(\mathbf{Q}^{11})^{-1}\right]_{jk} & 1 < j < k \leq p \\ \left[(\mathbf{Q}^{11})^{-1}\right]_{jj} & k = 1, 1 < j \leq p \\ 0 & j = k. \end{cases}$$

Alors la dérivée partielle de $W = \sum_{T \in \mathcal{T}} \prod_{jk \in T} w_{jk}$ *vaut :*

$$\partial_{w_{jk}} W = [\mathbf{M}]_{jk} \times W.$$

Ces deux résultats rendent possible l'utilisation des arbres couvrants au sein de procédures d'inférence.

Cas gaussien

Un cadre populaire.

Les modèles graphiques sont particulièrement utilisés dans le cas gaussien, notamment pour leur propriété de fidélité.

Propriété (Fidélité). *Soit* $X \sim \mathcal{N}(\mu, \mathbf{\Sigma})$ *une variable aléatoire gaussienne multi-variée. Alors sa distribution est fidèle Markov au graphe* $\mathcal{G}$ *dont les arêtes sont les termes non nuls de sa matrice de précision* $\mathbf{\Omega} = \mathbf{\Sigma}^{-1}$*.*

Ainsi dans le cas gaussien le graphe représentant exactement les relations d'indépendances conditionnelles est facilement défini. Dans le cadre des modèles graphiques gaussiens, Lauritzen (1996) donne l'estimateurs du maximum de vraisemblance des termes de $\mathbf{\Sigma}$ correspondant aux arêtes du graphe. Si le graphe est de plus décomposable il existe un estimateur du maximum de vraisemblance pour $\mathbf{\Omega}$, et une estimation simplifiée de son déterminant en découle. Ces estimateurs ont une forme complexe, mais sont manipulables si la structure de graphe considérée est simple.

Inférence pénalisée.

L'inférence d'un modèle graphique gaussien revient à identifier les éléments non-nuls de sa matrice de précision $\mathbf{\Omega}$. Classiquement, une estimation par régularisation pénalisée de $\mathbf{\Omega}$ est réalisée, connue sous le nom de *graphical lasso* (Friedman et al., 2008). Cette approche maximise la log-vraisemblance pénalisée par la norme ℓ_1 de

$\boldsymbol{\Omega}$:

$$\underset{\boldsymbol{\Omega}\geq 0}{\arg\max}\left\{\log|\boldsymbol{\Omega}| + \operatorname{tr}\left(\mathbf{Y}^\intercal\mathbf{Y}\boldsymbol{\Omega}\right) - \lambda||\boldsymbol{\Omega}||_1\right\}, \qquad ||\boldsymbol{\Omega}||_1 = \sum_{\mathrm{j}\neq\mathrm{k}}|\omega_{\mathrm{jk}}|.$$

Cette stratégie vise une estimation parcimonieuse en introduisant des termes nuls dans $\boldsymbol{\Omega}$. Très utilisée, elle nécessite cependant une attention particulière pour la sélection du niveau de parcimonie, c'est à dire du paramètre de régularisation λ.

Inférence par arbres.

Ce travail utilise une stratégie à base d'arbres couvrants. Une exploration de l'espace des arbres couvrants suppose que le graphe sous-jacent est un arbre aléatoire T. Ainsi contrairement à l'inférence pénalisée, le graphe est ici une variable latente du modèle. En définissant une distribution de probabilité sur cet espace, il est possible de définir un mélange d'arbre (Meilă and Jordan, 2000) pour des variables gaussiennes.

Définition (Mélange d'arbres gaussien). *La distribution d'une variable aléatoire $\boldsymbol{Y}$ est un mélange d'arbres gaussien centré sur l'espace des arbres couvrants si elle s'écrit :*

$$p(\boldsymbol{Y}) = \sum_{T\in\mathcal{T}} p(T)p(\boldsymbol{Y}\mid T), \qquad \boldsymbol{Y}\mid T \sim \mathcal{N}(0,\Sigma_T).$$

Dans l'expression d'un mélange d'arbres gaussien, la distribution gaussienne $p(\boldsymbol{Y}\mid T)$ s'écrit naturellement comme un produit sur les paires de noeuds. Donc, en choisissant une distribution $p(T)$ qui se factorise de la même manière une forme somme-produit apparaît, qui est calculable grâce au théorème arbre-matrice.

Une telle distribution sur l'espace des arbres est par exemple la distribution décomposable, qui attribue un poids à chaque arête et pour laquelle la probabilité d'un arbre est proportionnelle au produit de ces poids. Cette distribution permet notamment de calculer facilement des probabilités d'arêtes. En utilisant la définition d'une probabilité d'arête comme la somme des probabilités des arbres contenant cette arête, il devient clair qu'adopter une distribution décomposable mène à une nouvelle forme somme-produit sur l'espace des arbres. Pour une matrice de poids $\mathbf{W} = (w_{uv})_{uv}$, la probabilité que les noeuds j et k soient reliés dans T est :

$$\mathbb{P}\{jk\in T\} = \sum_{\substack{T\in\mathcal{T}\\ T\ni jk}} p(T) \propto \sum_{\substack{T\in\mathcal{T}\\ T\ni jk}}\prod_{uv\in T} w_{uv}.$$

Chaque probabilité peut ensuite être calculée par le théorème arbre-matrice, ou en utilisant un résultat de Kirshner (2008) permettant de les calculer toutes en une seule opération.

Inférence de variables latentes

L'algorithme Espérance-Maximisation (EM, Dempster et al. (1977)) permet de mener une inférence sur des données $\boldsymbol{Y}$ en présence de variables latentes $\boldsymbol{Z}$. C'est un algorithme itératif qui alterne deux étapes : une étape d'estimation (étape E) de l'espérance conditionnelle de la log-vraisemblance jointe des variables observées et latentes (complète), et une étape de maximisation (étape M) de cette quantité en les paramètres du modèle.

Lorsque la distribution des variables latentes conditionnellement aux observées $p(\boldsymbol{Z} \mid \boldsymbol{Y})$ n'est pas disponible ou calculable, l'étape E a besoin d'être modifiée. L'approche variationnelle permet alors d'approximer $p(\boldsymbol{Z} \mid \boldsymbol{Y})$ en définissant un ensemble de distributions autorisées Q, et une mesure de distance de la distribution approchée à la vraie distribution conditionnelle. L'algorithme est alors Variationnel EM (VEM), dont l'étape VE est un problème d'optimisation visant à trouver la distribution $q \in Q$ la plus proche de $p(\boldsymbol{Z} \mid \boldsymbol{Y})$. L'algorithme VEM peut aussi être vu comme l'optimisation d'une borne inférieure de la vraisemblance, pénalisée par une mesure de la distance entre la distribution conditionnelle $p(\boldsymbol{Z} \mid \boldsymbol{Y})$ et son approximation variationnelle $q(\mathbf{Z})$.

Propriété (Approximation en champs moyen). *Soit un algorithme VEM avec Q l'ensemble des distributions factorisables et la divergence de Küllback-Leibler, appliqué aux données observées $\boldsymbol{Y}$ et à K variables latentes $\boldsymbol{Z} = \{z_1, ..., z_K\}$. Alors la solution du problème d'optimisation de l'étape VE à l'itération $t+1$ pour la distribution marginale q_k de Z_k est telle que:*

$$q_k^{(t+1)}(z_k) \propto \exp\left\{\mathbb{E}_{q^t_{\setminus k}}\left[\log p_{\boldsymbol{\theta}^{t+1}}(\boldsymbol{Y}, \boldsymbol{Z})\right]\right\}.$$

Ce résultat, dû à Beal and Ghahramani (2003), est connu sous le nom d'approximation en champ moyen et permet une écriture claire des algorithmes VEM sous les conditions précisées.

Inférence à partir de données de comptages

L'inférence de réseaux à partir de données de comptages nécessite la définition d'une distribution jointe permettant de modéliser les dépendances entre variables discrètes. Il existe peu d'options dans le domaine, et une manière de faire est d'utiliser les modèles linéaires mixtes généralisés. Pour chaque échantillon i et espèce j, un effet aléatoire Z_{ij} gaussien est associé au comptage moyen m_{ij} par une fonction de lien g. Une écriture générale de ce modèle prenant en compte l'offset o_{ij} et le vecteur

de covariables $\boldsymbol{x}_i$ de coefficient $\boldsymbol{\theta}_j$ est la suivante :

$$\begin{cases} g(m_{ij}) = o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + Z_{ij}, \\ \mathbf{Z}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}), \\ \boldsymbol{Y}_{ij} \mid \boldsymbol{Z}_i \sim F(m_{ij}). \end{cases}$$

$\boldsymbol{Y}_{ij}$ est distribué selon F, de moyenne m_{ij}. Par la suite, c'est la matrice de variance-covariance $\boldsymbol{\Sigma}$ des effets aléatoires gaussiens $\boldsymbol{Z}$ qui est étudiée, plutôt que les dépendances de $\boldsymbol{Y}$ directement. Il existe différentes stratégies pour se ramener au cadre gaussien en utilisant les modèles linéaires mixtes généralisés, et dans les chapitres suivants nous utilisons la distribution Poisson log-normale (PLN). Dans ce cas précis, F est une distribution de Poisson et g la fonction log.

Chapitre 2

Ce deuxième chapitre présente la méthode développée pour l'inférence de réseaux d'interactions d'espèces à partir de données de comptages. Les comptages $\boldsymbol{Y}$ sont modélisés avec la distribution Poisson log-normale. L'inférence du réseau en elle-même a lieu dans la couche des paramètres gaussiens $\boldsymbol{Z}$, et utilise un mélange d'arbres. Cela signifie que le graphe des dépendances conditionnelles sous-jacent à $\boldsymbol{Z}$ est supposé être un arbre latent T. Ce modèle hiérarchique à deux couches latentes peut se résumer ainsi : un arbre est d'abord tiré aléatoirement, puis les paramètres gaussiens sont tirés conditionnellement à T, et enfin les comptages Y sont tirés conditionnellement aux $\boldsymbol{Z}$ selon une loi de Poisson.

$$T \rightarrow \boldsymbol{Z} \rightarrow \boldsymbol{Y}$$

La stratégie d'inférence de réseaux mise en place vise la mise à jour de la distribution de l'arbre latent T au travers des poids $\boldsymbol{\beta}$, et le calcul de probabilités d'arêtes.

Modèle

Les comptages $\boldsymbol{Y}$ sont modélisés avec la distribution PLN. En notant $\boldsymbol{Z}$ les paramètres latents, $\boldsymbol{x}_i$ le vecteur de covariables correspondant à l'échantillon i, $\boldsymbol{\theta}_j$ son coefficient pour l'espèce j et o_{ij} l'offset associé, le modèle s'écrit comme suit :

$$Y_{ij} \mid \boldsymbol{Z}_i \sim \mathcal{P}(\exp(o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + Z_{ij})), \quad (Y_{ij} \perp\!\!\!\perp) \mid \boldsymbol{Z}_i.$$

Conditionnellement à l'arbre T, les paramètres latents $\boldsymbol{Z}$ sont distribués selon la loi normale. La loi de $\boldsymbol{Z} \mid T$ est donc fidèle Markov à T et s'écrit :

$$\boldsymbol{Z}_i \mid T \sim \mathcal{N}(0, \boldsymbol{\Omega}_T), \quad \{\boldsymbol{Z}_i\}_i \text{ iid},$$

où les termes non-nuls de $\mathbf{\Omega}_T$ correspondent aux arêtes de T. Enfin, l'arbre T est supposé suivre une loi décomposable sur ses arêtes, avec $\boldsymbol{\beta}$ la matrice des paramètres de poids et B la constante de normalisation :

$$T \sim \prod_{kl \in T} \beta_{kl}/B, \quad B = \sum_{T \in \mathcal{T}} \prod_{kl \in T} \beta_{kl}.$$

L'inférence du graphe des dépendances de la couche des $\boldsymbol{Z}$ repose alors sur un mélange d'arbres gaussien. C'est à dire que les paramètres latents suivent une distribution de mélange de gaussiennes indépendantes dont chaque composante est fidèle à un arbre de l'espace des arbres couvrants $\mathcal{T}$:

$$\mathbf{Z}_i \sim \sum_{T \in \mathcal{T}} p(T) \mathcal{N}(\boldsymbol{Z}_i \mid T; 0, \mathbf{\Omega}_T).$$

Inference

Les paramètres concernant directement le réseau sont les poids rassemblés dans $\boldsymbol{\beta}$. L'inférence du modèle comprend deux étapes qui séparent l'estimation de $\boldsymbol{\beta}$ du reste des paramètres. La première étape utilise l'algorithme variationnel développé par Chiquet et al. (2018) pour avoir accès aux estimateurs variationnels de $\boldsymbol{\theta}$ et des statistiques exhaustives relatives à $\boldsymbol{Z}$. Ces paramètres sont fixés pour la suite de l'inférence. La seconde étape est un algorithme EM qui a pour but l'estimation de $\boldsymbol{\beta}$, et le calcul des probabilités d'arêtes.
L'algorithm EM requiert le calcul de l'espérance conditionnelle de la log-vraisemblance complète. En notant $\widehat{\rho}_{jk}$ la corrélation estimée entre les variables j et k, et $P_{jk} \simeq \mathbb{P}\{jk \in T \mid \boldsymbol{Y}\}$ la probabilité d'arête conditionnelle estimée entre j and k, cette espérance est approximée par la quantité ci-dessous, où le terme cst ne dépend pas de $\boldsymbol{\beta}$:

$$\sum_{1 \leq j < k \leq p} P_{jk} \log\left(\beta_{jk}(1 - \widehat{\rho}_{jk}^2)^{-n/2}\right) - \log B + cst.$$

Étape E : Les estimateurs des statistiques exhaustives de $\boldsymbol{Z}$ obtenus en première étape de l'inférence donnent accès aux $\widehat{\rho}_{jk}$. L'étape E consiste donc en le calcul des probabilités approchées P_{jk}. On considère pour cela la probabilité conditionnelle à $\boldsymbol{Y}$ de l'ensemble des arbres contenant l'arête jk, ce qui revient à appliquer le théorème arbre-matrice à une nouvelle matrice de poids, de terme général $\beta_{jk}(1 - \widehat{\rho}_{jk}^2)^{-n/2}$.

Étape M : Cette étape maximise la quantité précédente en $\boldsymbol{\beta}$. La forme close de la formule de mise à jour pour $\boldsymbol{\beta}$ est obtenue en utilisant un lemme de Meilă and Jaakkola (2006), qui définit une matrice $\mathbf{M}(\boldsymbol{\beta})$, fonction de $\boldsymbol{\beta}$. À l'itération

$t+1$ de l'algorithme EM, la mise à jour est telle que:

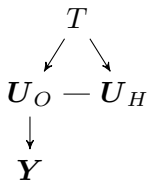
$$\beta_{jk}^{t+1} = P_{jk}^{t+1} \Big/ \left[\mathbf{M}(\boldsymbol{\beta}^t)\right]_{jk} .$$

Simulations et applications

La procédure d'inférence est implémentée dans le package R EMtree, disponible sur GitHub (`https://github.com/Rmomal/EMtree`). Les performances de cette méthode ont été comparées à celles d'approches venant de la micro-biologie (SpiecEasi (Kurtz et al., 2015), gCoda (Fang et al., 2017), MInt (Biswas et al., 2016)) et de l'écologie des communautés (MRFcov (Clark et al., 2018), ecoCopula (Popovic et al., 2019)). Ces méthodes diffèrent sur la manière qu'elles ont de se ramener au cadre gaussien. Toutes ces méthodes infèrent ensuite le réseau en ayant recourt à une estimation pénalisée de la matrice de précision de la loi normale.
Lors de l'étude de simulations sur des graphes de densité et structure variables (cluster, Erdös-Reyni, scale-free), EMtree s'illustre comme la méthode donnant le moins de faux positifs (fausses arêtes) et conservant une densité de réseau comparable à l'originale, tout en figurant parmi les algorithmes les plus rapides.
Deux exemples d'application d'EMtree, un en écologie et un en méta-génomique, montrent l'importance de la prise en compte des covariables expérimentales dans le modèle. Concernant la deuxième application, EMtree retrouve des résultats obtenus précédemment dans Jakuschkin et al. (2016).

Chapitre 3

Le chapitre 3 considère le modèle exposé précédemment dans le chapitre 2, dans le cas où la couche latente des paramètres gaussiens $\boldsymbol{Z}$ comprend des dimensions supplémentaires qui ne correspondent à aucune donnée observée. Ces dimensions supplémentaires représentent des acteurs, espèces ou covariables, qui ont une influence sur les données observées mais n'ont pas été mesurés. Ce sont des acteurs manquants, qui s'ils ne sont pas pris en compte génèrent des liens de dépendances conditionnelles entre les espèces dépendantes de chacun des acteurs. Le modèle précédent est modifié pour prendre en compte les dimensions supplémentaires indexées par H de la couche latente, et considère la version normalisée de cette dernière, notée $\boldsymbol{U}$. Le graphe suivant synthétise les dépendances entre variables, où $\boldsymbol{U}_O$ sont les variables latentes des données observées, et $\boldsymbol{U}_H$ celles des acteurs manquants :



C'est un modèle plus complexe que précédemment, comprenant trois couches latentes. Un algorithme VEM est développé pour l'inférence, qui permet d'obtenir des informations concernant les acteurs manquants, en plus d'inférer le réseau complet des dépendances conditionnelles.

Modèle

Les comptages sont modélisés avec la distribution PLN comme détaillé au chapitre 2, mais en considérant la version normalisée de la couche latente gaussienne :

$$Y_{ij} \mid U_{ij} \sim \mathcal{P}\left(\exp(o_{ij} + \boldsymbol{x}_i^\intercal \boldsymbol{\theta}_j + \sigma_j U_{ij})\right), \qquad (Y_{ij} \perp\!\!\!\perp) \mid \boldsymbol{U}_i.$$

La couche latente gaussienne $\boldsymbol{U}$ est composée de $\boldsymbol{U}_O$, de dimension p qui correspond aux $\boldsymbol{Y}$ observés, et $\boldsymbol{U}_H$ qui rassemble les r variables supplémentaires non-observées. Le reste du modèle est le même que précédemment, si ce n'est pour les dimensions supplémentaires : l'arbre T est paramétré par une matrice de poids $\boldsymbol{\beta}$ de dimension $(p+r)\times(p+r)$, la distribution marginale de $\boldsymbol{U}$ est un mélange gaussien sur l'espace des arbres couvrants de dimension $p+r$, dont chaque composante est fidèle à un arbre. On note $\boldsymbol{\Omega}$ l'ensemble des matrices de précision des gaussiennes fidèles à un arbre : $\boldsymbol{\Omega} = \{\boldsymbol{\Omega}_T, T \in \mathcal{T}\}$.

Inférence

La complexité supplémentaire de la structure latente de ce modèle demande une inférence variationnelle. L'inférence présentée ici maximise la borne inférieure suivante qui utilise la divergence de Küllback-Leibler:

$$\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\Omega}; q) = \log p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{Y}) - KL\left(q(\boldsymbol{U}, T) \| p_{\boldsymbol{\theta},\boldsymbol{\beta},\Omega}(\boldsymbol{U}, T \mid \boldsymbol{Y})\right).$$

Ci-dessus, $q(\boldsymbol{U}, T)$ est la distribution approchée des variables latentes conditionnellement aux données observées : $q(\boldsymbol{U}, T) \approx p(\boldsymbol{U}, T \mid \boldsymbol{Y})$. On suppose que q est une distribution produit, ce qui permet de séparer les distributions marginales variationnelles de l'arbre T et des variables gaussiennes $\boldsymbol{U}$ comme suit:

$$q(\boldsymbol{U}, T) = h(\boldsymbol{U})\, g(T).$$

La distribution h est de plus elle-même supposée être un produit de n gaussiennes à matrices de variance-covariances diagonales, à l'instar de Chiquet et al. (2018). Les paramètres de h sont les matrices $M = [M_O, M_H]$ et $S = [S_O, S_H]$ de dimensions $n \times (p+r)$, rassemblant respectivement les moyennes et les variances de chaque composante du produit. La distribution g de l'arbre T se factorise sur les arêtes de T, et ses paramètres de poids sont rassemblés dans la matrice $\widetilde{\boldsymbol{\beta}}$.

La procédure d'inférence tire à nouveau parti de l'estimation variationnelle du modèle PLN développée dans Chiquet et al. (2018). Cette estimation donne une approximation des paramètres $\boldsymbol{\theta}$, M_O et S_O qui sont considérés fixes pour la suite de l'inférence.
L'algorithme VEM développé a pour but l'estimation des poids $\boldsymbol{\beta}$ et itère les étapes suivantes:

Étape VE : Cette étape maximise $\mathcal{J}$ par rapport à q, ce qui revient à maximiser $\mathcal{J}$ en les paramètres variationnels de g et ceux restants de h, soit en $\widetilde{\boldsymbol{\beta}}$, M_H et S_H. L'écriture de q sous forme factorisée permet d'aboutir à une approximation en champs moyen pour les estimations de h et g.

Étape M : Cette étape maximise $\mathcal{J}$ en les paramètres restants de la log-vraisemblance complète, soit en $\boldsymbol{\beta}$ et $\boldsymbol{\Omega}$. La mise à jour de β est la même que dans le chapitre 2 et nécessite le calcul des probabilités d'arêtes. Dans le cas présent la distribution conditionnelle des arbres aux données est approchée par g, ce qui justifie de calculer les probabilités d'arêtes en appliquant la formule de Kirshner (2008) aux poids variationnels $\widetilde{\boldsymbol{\beta}}$.

L'estimation des matrices $\boldsymbol{\Omega}_T$ est plus complexe et applique au contexte des arbres couvrants les formules du maximum de vraisemblance de Lauritzen (1996), établies dans le cadre des modèles graphiques gaussiens décomposables. Ces formules sont définies à partir de l'espérance de statistiques exhaustives, calculée à partir de $\mathbf{M}$ et $\boldsymbol{S}$.

Cette procédure est implémentée dans le package R nestor (*Network inference from Species counTs with missing actORs*, `https://github.com/Rmomal/nestor`). Une étude de simulations révèle son efficacité et l'importance fondamentale de son initialisation.

Applications

La procédure d'inférence donne une estimation du réseau complet et des paramètres du modèle. Cela rend disponible deux informations clés pour caractériser les acteurs manquants supposés du réseau, à savoir leur position dans le réseau, et l'estimation de leur moyenne variationnelle M_H sur les différents sites d'échantillonnage.
Un premier exemple d'application utilise des données de recensement de poissons dans la mer de Barents. Là, l'inférence de réseau avec $r = 1$ acteur manquant donne un noeud dont le vecteur de moyennes M_H est fortement corrélé à la covariable de température, tout comme ceux de ses voisins directs dans le réseau. Dans un second exemple sur des poissons du fleuve Fatala en Guinée, l'inférence est réalisée avec deux acteurs manquants. L'étude de leur moyenne variationnelle montre que le premier est lié à la nature spatiale de ces donnée, et le second semble lié à la

dimension temporelle de l'échantillonnage.
Ces applications illustrent à la fois la validité de la méthode en comparant les acteurs inférés à des covariables d'importance, et une première approche pour la caractérisation des acteurs manquants.

Chapitre 4

Le dernier chapitre présente des perspectives de ce travail. Après avoir résumé les spécificités et discuté des points sensibles de la méthodologie développée, des extensions naturelles du modèle sont présentées. Les premières portent sur le réseau inféré. Plus précisément, une méthode d'estimation de la matrice de précision $\mathbf{\Omega}$ de la couche latente est proposée, qui utilise les estimateurs de Lauritzen. L'estimation précise de cette matrice permet des interprétations intéressantes dans le domaine d'application en question, à propos de la force et du sens des interactions détectées.
Le sujet de la comparaison de réseaux est abordé dans un second temps. La manière de faire générale est de résumer les réseaux à des vecteurs de mesures caractéristiques, puis de comparer ces vecteurs. Ici nous proposons de comparer les distributions d'arbres inférées sur les réseaux.
Des perspectives sur les spécificités des données disponibles sont ensuite discutées. La procédure d'inférence de réseaux développée a lieu au sein d'une couche gaussienne latente. Tant qu'elle est présente dans le modèle l'inférence de réseau reste la même, aussi la loi d'émission des données à partir de ces paramètres latents peut être différente. Ceci permet d'inférer des réseaux à partir de données de natures variées, pourvu que l'estimation des paramètres de cette loi soit disponible. Par ailleurs, les données peuvent présenter des dépendances spatiales, comme c'est souvent le cas en écologie. Il est possible de prendre en compte ces effets en moyenne, en ajustant par exemple des coordonnées spatiales au modèle. Cette première correction peut ne pas être suffisante. Nous présentons une manière d'introduire des paramètres de variance des effets spatiaux dans le modèle d'inférence de réseau directement, afin d'ajuster les effets spatiaux en variance.
Enfin, une perspective plus générale sur l'inférence de réseau à partir de données de comptages sans avoir recours à une couche latente gaussienne est présentée. Comme la vraisemblance d'une variable aléatoire conditionnellement à une structure d'arbre se factorise sur les arêtes de cet arbre, l'idée est d'inférer le réseau par un mélange d'arbres en utilisant une loi bivariée sur les comptages. La difficulté de cette approche réside dans l'estimation des paramètres.

Liste des publications

— Momal, Raphaëlle, Stéphane Robin, and Christophe Ambroise. "Tree-based inference of species interaction networks from abundance data." Methods in Ecology and Evolution 11.5 (2020): 621-632.

— Momal, Raphaëlle, Stéphane Robin, and Christophe Ambroise. "Accounting for missing actors in interaction network inference from abundance data." arXiv preprint arXiv:2007.14299 (2020).

Bibliography

J. Aitchison. The statistical analysis of compositional data. Journal of the Royal Statistical Society: Series B (Methodological), 44(2):139–160, 1982.

J. Aitchison and C.H Ho. The multivariate Poisson-log normal distribution. Biometrika, 76(4):643–653, 1989.

J Aitchison and Sheng M Shen. Logistic-normal distributions: Some properties and uses. Biometrika, 67(2):261–272, 1980.

C. Ambroise, J. Chiquet, and C. Matias. Inferring sparse gaussian graphical models with latent structure. Electronic Journal of Statistics, 3:205–238, 2009.

M. J. Anderson, P. de Valpine, A. Punnett, and A. E. Miller. A pathway for multivariate analysis of ecological communities using copulas. Ecology and evolution, 9(6):3276–3294, 2019.

S. Arlot and A. Celisse. A survey of cross-validation procedures for model selection. Statistics surveys, 4:40–79, 2010.

M. Austin. Species distribution models and ecological theory: a critical assessment and some possible new approaches. Ecological modelling, 200(1-2):1–19, 2007.

E. Baran. Dynamique spatio-temporelle des peuplements de Poissons estuariens en Guinée (Afrique de l'Ouest). PhD thesis, Thèse de Doctorat, Université de Bretagne Occidentale, 1995.

I. Bartomeus, D. Gravel, J. M Tylianakis, M. A Aizen, I. A Dickie, and M. Bernard-Verdier. A common framework for identifying linkage rules across different types of interactions. Functional Ecology, 30(12):1894–1903, 2016.

MJ Beal and Z Ghahramani. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. In Bayesian Statistics 7: Proceedings of the Seventh Valencia International Meeting, volume 7, pages 453–464. Oxford University Press, 2003.

P. Bickel, D. Choi, X. Chang, H. Zhang, et al. Asymptotic normality of maximum likelihood and its variational approximation for stochastic blockmodels. The Annals of Statistics, 41(4):1922–1943, 2013.

S. Biswas, M. McDonald, D. S Lundberg, J. L. Dangl, and V. Jojic. Learning microbial interaction networks from metagenomic count data. Journal of Computational Biology, 23(6):526–535, 2016.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. Journal of the American Statistical Association, 112(518):859–877, 2017.

E.J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? Journal of the ACM (JACM), 58(3):1–37, 2011.

P. Carbonetto, M. Stephens, et al. Scalable variational inference for bayesian variable selection in regression, and its accuracy in genetic association studies. Bayesian analysis, 7(1):73–108, 2012.

A. Celisse, J.-J. Daudin, L. Pierre, et al. Consistency of maximum-likelihood and variational estimators in the stochastic block model. Electronic Journal of Statistics, 6:1847–1899, 2012.

S. Chaiken and D. J. Kleitman. Matrix tree theorems. Journal of combinatorial theory, Series A, 24(3):377–381, 1978.

V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. Rank-sparsity incoherence for matrix decomposition. SIAM J. Optim, 21:572–596, 2011.

H. Chen and H. Koga. Gl2vec: Graph embedding enriched by line graphs with edge features. In International Conference on Neural Information Processing, pages 3–14. Springer, 2019.

J. Chiquet, M. Mariadassou, and S. Robin. Variational inference for probabilistic poisson pca. The Annals of Applied Statistics, 12(4):2674–2698, 2018.

J. Chiquet, M. Mariadassou, and S. Robin. Variational inference for sparse network reconstruction from count data. In International Conference on Machine Learning, 2019a.

J. Chiquet, G. Rigaill, and M. Sundqvist. A multiattribute gaussian graphical model for inferring multiscale regulatory networks: an application in breast cancer. In Gene Regulatory Networks, pages 143–160. Springer, 2019b.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory, 14(3):462–467, May 1968.

N. J Clark, K. Wells, and O. Lindberg. Unravelling changing interspecific interactions across environmental gradients using markov random fields. Ecology, 99(6): 1277–1283, 2018.

N. Cressie and C. K. Wikle. Statistics for spatio-temporal data. John Wiley & Sons, 2015.

T. Dallas, A. W Park, and J. M Drake. Predicting cryptic links in host-parasite networks. PLoS computational biology, 13(5):e1005557, 2017.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. J. Royal Statist. Soc., series B, 39:1–38, 1977.

P. Desjardins-Proulx, I. Laigle, T. Poisot, and D. Gravel. Ecological interactions and the netflix problem. PeerJ, 5:e3644, 2017.

L. Devroye. Non-uniform random variate generation. Springer, 1986.

C. F Dormann, M. Bobrowski, D M. Dehling, D. J Harris, F. Hartig, H. Lischke, M. D Moretti, J. Pagel, S. Pinkert, M. Schleuning, et al. Biotic interactions in species distribution modelling: 10 questions to guide interpretation and avoid false conclusions. Global ecology and biogeography, 27(9):1004–1016, 2018.

S. Dray, A.-B. Dufour, et al. The ade4 package: implementing the duality diagram for ecologists. Journal of statistical software, 22(4):1–20, 2007.

D. Durfee, R. Kyng, J. Peebles, A. B Rao, and S. Sachdeva. Sampling random spanning trees faster than matrix multiplication. In Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, pages 730–742, 2017.

J. Elith and J. R. Leathwick. Species distribution models: ecological explanation and prediction across space and time. Annual review of ecology, evolution, and systematics, 40, 2009.

N. B. Erichson, P. Zheng, K. Manohar, S. L. Brunton, J. N. Kutz, and A. Y. Aravkin. Sparse principal component analysis via variable projection. SIAM Journal on Applied Mathematics, 80(2):977–1002, 2020.

D. M Evans, J. JN Kitson, D. H Lunt, N. A Straw, and M. JO Pocock. Merging dna metabarcoding and ecological network analysis to understand and build resilient terrestrial ecosystems. Functional ecology, 30(12):1904–1916, 2016.

H. Fang, C. Huang, H. Zhao, and M. Deng. gCoda: conditional dependence network inference for compositional data. Journal of Computational Biology, 24(7):699–708, 2017.

O. P. Faugeras. Inference for copula modeling of discrete data: a cautionary tale and some facts. Dependence Modeling, 5(1):121–132, 2017.

K. Faust and J. Raes. Microbial interactions: from networks to models. Nature Reviews Microbiology, 10(8):538, 2012.

M. Fossheim, E. M Nilssen, and M. Aschan. Fish assemblages in the Barents Sea. Marine Biology Research, 2(4):260–269, 2006.

L. C Freeman. Centrality in social networks conceptual clarification. Social networks, 1(3):215–239, 1978.

J. Friedman, T. Hastie, and R. Tibshirani. The elements of statistical learning, volume 1. Springer series in statistics New York, 2001.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9(3):432–441, 2008.

N. Friedman. Inferring cellular networks using probabilistic graphical models. Science, 303(5659):799–805, 2004.

C. Giraud and A. Tsybakov. Discussion of "latent variable graphical model selection via convex optimization". Annals of Statistics, 2012.

G. B. Gloor, J. M. Macklaim, V. Pawlowsky-Glahn, and J. J. Egozcue. Microbiome datasets are compositional: and this is not optional. Frontiers in microbiology, 8: 2224, 2017.

C. H Graham and B. G Weinstein. Towards a predictive model of species interaction beta diversity. Ecology letters, 21(9):1299–1310, 2018.

L. Guidi, S. Chaffron, L. Bittner, D. Eveillard, A. Larhlimi, S. Roux, S. Darzi, Y.and Audic, L. Berline, J. R. Brum, et al. Plankton networks driving carbon export in the oligotrophic ocean. Nature, 532(7600):465–470, 2016.

D. J. Harris. Generating realistic assemblages with a joint species distribution model. Methods in Ecology and Evolution, 6(4):465–473, 2015.

T. J Hastie. Generalized additive models. In Statistical models in S, pages 249–307. Routledge, 2017.

D. Inouye, P. Ravikumar, and I. Dhillon. Square root graphical models: Multivariate generalizations of univariate exponential families that permit positive dependencies. In International conference on machine learning, pages 2445–2453, 2016.

D. I. Inouye, E. Yang, G. I. Allen, and P. Ravikumar. A review of multivariate distributions for count data derived from the poisson distribution. Wiley Interdisciplinary Reviews: Computational Statistics, 9(3):e1398, 2017.

B. Jakuschkin, V. Fievet, L. Schwaller, T. Fort, C. Robin, and C. Vacher. Deciphering the pathobiome: Intra- and interkingdom interactions involving the pathogen erysiphe alphitoides. Microb Ecol, 72(4):870–880, 2016.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. Machine learning, 37(2):183–233, 1999.

P. Jordano. Sampling networks of ecological interactions. Functional Ecology, 30 (12):1883–1893, 2016.

D. Karlis. An em algorithm for multivariate poisson distribution and related models. Journal of Applied Statistics, 30(1):63–77, 2003.

S. Kirshner. Learning with tree-averaged densities and distributions. In Advances in Neural Information Processing Systems, pages 761–768, 2008.

Z. D. Kurtz, C. L. Müller, E. R. Miraldi, D. R. Littman, M. J. Blaser, and R. A. Bonneau. Sparse and compositionally robust inference of microbial ecological networks. PLoS computational biology, 11(5):e1004226, 2015.

S. L. Lauritzen. Graphical Models. Oxford Statistical Science Series. Clarendon Press, 1996.

S. L. Lauritzen and N. Meinshausen. Discussion: Latent variable graphical model selection via convex optimization. The Annals of Statistics, 2012.

B. G. Lindsay. Composite likelihood methods. Contemporary mathematics, 80(1): 221–239, 1988.

H. Liu, J. Lafferty, and L. Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. Journal of Machine Learning Research, 10(10), 2009.

H. Liu, K. Roeder, and L. Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. In Advances in neural information processing systems, pages 1432–1440, 2010a.

Han Liu, Kathryn Roeder, and Larry Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. In Advances in neural information processing systems, pages 1432–1440, 2010b.

A. Lucas, I. Scholz, R. Boehme, S. Jasson, and M. Maechler. gmp: Multiple precision arithmetic, 2020. URL https://CRAN.R-project.org/package=gmp. R package version 0.5-13.6.

G. J. McLachlan and T. Krishnan. The EM algorithm and extensions, volume 382. John Wiley & Sons, 2007.

M. Meilă and T. Jaakkola. Tractable bayesian learning of tree belief networks. Statistics and Computing, 16(1):77–92, 2006.

M. Meilă and M. I. Jordan. Learning with mixtures of trees. Journal of Machine Learning Research, 1:1–48, 2000.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. The annals of statistics, pages 1436–1462, 2006.

Z. Meng, B. Eriksson, and A. O. Hero III. Learning latent variable gaussian graphical models. Proceedings of the 31 International Conference on Machine Learning, 32, 2014.

T. Minka. Divergence measures and message passing. Technical report, Technical report, Microsoft Research, 2005.

R. Momal, S. Robin, and C. Ambroise. Tree-based inference of species interaction networks from abundance data. Methods in Ecology and Evolution, 11 (5):621–632, 2020. doi: 10.1111/2041-210X.13380. URL https://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.13380.

N. Morueta-Holme, B. Blonder, B. Sandel, B. J. McGill, R. K. Peet, J. E. Ott, C. Violle, B. J. Enquist, P. M. Jørgensen, and J-C. Svenning. A network approach for inferring species associations from co-occurrence data. Ecography, 39(12): 1139–1150, 2016.

R. M Neal and G. E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In Learning in graphical models, pages 355–368. Springer, 1998.

C. Olito and J. W Fox. Species traits and abundances predict metrics of plant–pollinator network structure, but not pairwise interactions. Oikos, 124(4):428–436, 2015.

J. T Ormerod and M. P Wand. Explaining variational approximations. The American Statistician, 64(2):140–153, 2010.

O. Ovaskainen, J. Hottola, and J. Siitonen. Modeling species co-occurrence by multivariate logistic regression generates new hypotheses on fungal interactions. Ecology, 91(9):2514–2521, 2010.

O. Ovaskainen, N. Abrego, P. Halme, and D. Dunson. Using latent variable models to identify large networks of species-to-species associations at different spatial scales. Methods in Ecology and Evolution, 7(5):549–555, 2016.

O. Ovaskainen, G. Tikhonov, A. Norberg, F Guillaume B., L. Duan, D. Dunson, T. Roslin, and N. Abrego. How to make more out of community data? a conceptual framework and its implementation as models and software. Ecology Letters, 20(5):561–576, 2017.

A. Panagiotelis, C. Czado, and H. Joe. Pair copula constructions for multivariate discrete data. Journal of the American Statistical Association, 107(499):1063–1072, 2012.

T. Poisot, E. Canard, D. Mouillot, N. Mouquet, and D. Gravel. The dissimilarity of species interaction networks. Ecology letters, 15(12):1353–1361, 2012.

T. Poisot, D. B Stouffer, and D. Gravel. Beyond species: why ecological interaction networks vary through space and time. Oikos, 124(3):243–251, 2015.

T. Poisot, D. B Stouffer, and S. Kéfi. Describe, understand and predict: why do we need networks in ecology? Functional Ecology, 30(12):1878–1882, 2016.

L. J. Pollock, R. Tingley, W. K. Morris, N. Golding, R. B. O'Hara, K. M. Parris, P. A. Vesk, and M. A. McCarthy. Understanding co-occurrence by modelling species simultaneously with a joint species distribution model (jsdm). Methods in Ecology and Evolution, 5(5):397–406, 2014.

G. C Popovic, F. KC Hui, and D. I Warton. A general algorithm for covariance modeling of discrete data. Journal of Multivariate Analysis, 165:86–100, 2018.

G. C. Popovic, D. I. Warton, F. J. Thomson, F. K. C. Hui, and A. T. Moles. Untangling direct species associations from indirect mediator species effects with graphical models. Methods in Ecology and Evolution, 10(9):1571–1583, 2019.

A. Raj, M. Stephens, and J. K. Pritchard. faststructure: variational inference of population structure in large snp data sets. Genetics, 197(2):573–589, 2014.

S. A Richards. Dealing with overdispersed count data in applied ecology. Journal of Applied Ecology, 45(1):218–227, 2008.

G. Robin, C. Ambroise, and S. Robin. Incomplete graphical model inference via latent tree aggregation. Statistical Modelling, 19(5):545–568, 2019.

L. Schwaller and S. Robin. Exact bayesian inference for off-line change-point detection in tree-structured graphical models. Statistics and Computing, 27(5): 1331–1345, 2017.

L. Schwaller, S. Robin, and M. Stumpf. Bayesian Inference of Graphical Model Structures Using Trees. J. Soc. Franc. Stat., 160(2):1–23, 2019.

J. Shao. Linear model selection by cross-validation. Journal of the American statistical Association, 88(422):486–494, 1993.

A. F. Siahpirani, D. Chasman, and S. Roy. Integrative approaches for inference of genome-scale gene regulatory networks. In Gene Regulatory Networks, pages 161–194. Springer, 2019.

M. Stock, T. Poisot, W. Waegeman, and B. De Baets. Linear filtering reveals false negatives in species interaction data. Scientific reports, 7:45908, 2017.

H. Teicher. On the multivariate poisson distribution. Scandinavian Actuarial Journal, 1954(1):1–9, 1954.

R. Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1):267–288, 1996.

A. Tsitsulin, D. Mottin, P. Karras, and E. Müller. Verse: Versatile graph embeddings from similarity measures. In Proceedings of the 2018 World Wide Web Conference, pages 539–548, 2018.

M. CK. Tweedie. An index which distinguishes between some important exponential families. In Statistics: Applications and new directions: Proc. Indian statistical institute golden Jubilee International conference, volume 579, pages 579–604, 1984.

W. Ulrich and N. J. Gotelli. Null model analysis of species associations using abundance data. Ecology, 91(11):3384–3397, 2010.

A. Valiente-Banuet, M. A Aizen, J. M. Alcántara, J. Arroyo, A. Cocucci, M. Galetti, M. B García, D. García, J. M Gómez, P. Jordano, et al. Beyond species loss: the extinction of ecological interactions in a changing world. Functional Ecology, 29 (3):299–307, 2015.

G. Vidar and E. Steinar. poilog: Poisson lognormal and bivariate Poisson lognormal distribution, 2008. R package version 0.4.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Found. Trends Mach. Learn., 1(1–2):1–305, 2008.

D. I. Warton, F. G. Blanchet, R. B. O'Hara, O. Ovaskainen, S. Taskinen, S. C Walker, and F. KC. Hui. So many variables: joint modeling in community ecology. Trends in Ecology & Evolution, 30(12):766–779, 2015.

B. G Weinstein and C. H Graham. Persistent bill and corolla matching despite shifting temporal resources in tropical hummingbird-plant interactions. Ecology letters, 20(3):326–335, 2017.

K Wells and R. B O'Hara. Species interactions: estimating per-individual interaction strength and covariates before simplifying data into per-species ecological networks. Methods in Ecology and Evolution, 4(1):1–8, 2013.

E. Yang, P. K. Ravikumar, G. I. Allen, and Z. Liu. On poisson graphical models. In Advances in neural information processing systems, pages 1718–1726, 2013.

T. Zhao, H. Liu, K. Roeder, J. Lafferty, and L. Wasserman. The huge package for high-dimensional undirected graph estimation in R. Journal of Machine Learning Research, 13(Apr):1059–1062, 2012.

Y. Zhu and I. Cribben. Sparse graphical models for functional connectivity networks: best methods and the autocorrelation issue. Brain connectivity, 8(3):139–165, 2018.

université PARIS-SACLAY

ÉCOLE DOCTORALE de mathématiques Hadamard (EDMH)

Titre : Inférence de réseaux à partir de données d'abondances incomplètes.

Mots clés : Réseaux, Modèles Graphiques, Données d'abondances, Acteurs manquants, Algorithme VEM

Résumé : Les réseaux sont utilisés comme outils en microbiologie et en écologie pour représenter des relations entre espèces. Les modèles graphiques gaussiens sont le cadre mathématique dédié à l'inférence des réseaux de dépendances conditionnelles, qui permettent une séparation claires des effets directs et indirects. Cependant, les données observées sont souvent des comptages discrets qui ne permettent pas l'utilisation de ce modèle. Cette thèse développe une méthodologie pour l'inférence de réseaux à partir de données d'abondance d'espèces. La méthode repose sur une exploration efficace et exhaustive de l'espace des arbres couvrants dans un espace latent des comptages observés, rendue possible par les propriétés algébriques de ces structures.
Par ailleurs, il est probable que les comptages observés dépendent d'acteurs non mesurés (espèces ou covariable). Ce phénomène produit des arêtes supplémentaires dans le réseau marginal entre les espèces liées à l'acteur manquant dans le réseau complet, ce qui fausse la suite des analyses. Le second objectif de ce travail est de prendre en compte les acteurs manquants lors de l'inférence de réseau. Les paramètres du modèle proposé sont estimés par une approche variationnelle, qui fournit des éléments d'information pertinents à propos des données non observées.

Title: Network inference from incomplete abundance data.

Keywords: Networks, Graphical Models, Abundance data, Missing Actors, VEM algorithm

Abstract: Networks are tools used to represent species relationships in microbiology and ecology. Gaussian Graphical Models provide with a mathematical framework for the inference of conditional dependency networks, which allow for a clear separation of direct and indirect effects. However observed data are often discrete counts and the inference cannot be directly performed with this model. This work develops a methodology for network inference from species observed abundances. The method relies on specific algebraic properties of spanning tree structures to perform an efficient and complete exploration of the space of spanning trees. The inference takes place in a latent space of the observed counts.
Then, observed abundances are likely to depend on unmeasured actors (e.g. species or covariate). This results in spurious edges in the marginal network between the species linked to the latter in the complete network, causing inaccurate further analysis. The second objective of this work is to account for missing actors during network inference. To do so we adopt a variational approach yielding valuable insights about the missing actors.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France