Technische Universität München

TUM School of Computation, Information and Technology

Causal Inference of Extremes:
Recursive Max-Linear Models Under Observational Noise

Johannes Ernst-Emanuel Buck

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz:
Prof. Dr. Matthias Scherer

Prüfer*innen der Dissertation:
1. Prof. Dr. Claudia Klüppelberg
2. Prof. Dr. Carlos Enrique Améndola Cerón,
   Technische Universität Berlin

Die Dissertation wurde am 24.01.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 16.03.2023 angenommen.

Abstract

Recursive max-linear models have gained increasing interest for modeling extremal dependence. The causal structure of such models is represented by a Bayesian network, where node variables are defined as a max-linear function of parental node variables and an independent innovation. In this dissertation, we study recursive max-linear models under observational noise. Particularly, we are interested in structural properties and causal inference of noisy max-linear models.

In Chapter 1, we introduce recursive max-linear models and summarize important concepts that are useful throughout the dissertation. For ease of reading, we also summarize the main results of each of the following chapters.

In Chapter 2, we investigate the influence of recursive max-linear models under propagating, one-sided noise. We study structural properties and prove that the minimum DAG that preserves the distribution remains unchanged. Moreover, if we use a minimum ratio estimator for the ML coefficient matrix, which determines the dependence structure of the graph, we can show that this estimator at the left limit of support is completely determined by the distribution of the noise variables up to a positive constant. If the noise terms are regularly varying, we show that the estimated ML coefficient matrix derived from minimum ratios converges to a matrix of independent Weibull entries after proper centering and rescaling. This extends previous results about the minimum ratio estimator.

In Chapter 3, we introduce QTree , a novel technique to learn root-directed spanning trees, a special class of Bayesian networks. Inspired but not strictly limited to max-linear networks, the QTree algorithm uses a pair-wise score for nodes based on a lower quantile gap to measure the concentration of the distribution around its minimum. Then, we use Chu–Liu/Edmonds' algorithm to find the root-directed spanning tree of minimum score. We also introduce a parameter selection method based on bootstrap aggregation. Applied to various data sets of river discharge data, the algorithm strongly outperforms existing state-of-the-art methods. If the data follows a noisy recursive max-linear model, under a mild assumption on the noise tail, we show asymptotic consistency of the estimated root-directed spanning tree.

In Chapter 4, we introduce a machine learning approach to learn Bayesian networks from extreme data. Based on gradient descent, we find the parameters that optimally fit extreme data to a max-linear model. Using existing literature, we take a characterization of acyclicity in terms of a smooth function to ensure that the solution is supported on a Bayesian network. We show asymptotic consistency under mild assumptions and illustrate the performance in a simulation study as well as two real-world data sets: data on river discharges from Chapter 2 and data on foreign exchange rates.

Zusammenfassung

Rekursive max-lineare Modelle haben zunehmendes Interesse zur Modellierung extremaler Abhängigkeit geweckt. Die kausale Struktur solcher Modelle wird durch ein Bayes'sches Netzwerk dargestellt, in dem Knotenvariablen als eine max-lineare Funktion von übergeordneten Knotenvariablen und einer unabhängigen Innovation definiert sind. In dieser Dissertation untersuchen wir rekursive max-lineare Modelle unter Beobachtungsrauschen. Insbesondere sind wir an strukturellen Eigenschaften und kausaler Inferenz verrauschter max-linearer Modelle interessiert.

In Kapitel 1 führen wir rekursive max-lineare Modelle ein und fassen wichtige Konzepte zusammen, die für die gesamte Dissertation nützlich sind. Zur besseren Lesbarkeit fassen wir auch die wichtigsten Ergebnisse der folgenden Kapitel zusammen.

In Kapitel 2 untersuchen wir den Einfluss rekursiver max-linearer Modelle unter einseitigem Rauschen. Wir untersuchen strukturelle Eigenschaften und beweisen, dass der minimale DAG $\mathcal{D}^B$, der die Verteilung erhält, unverändert bleibt. Danach verwenden wir einen Minimum-Ratio-Schätzer für die ML-Koeffizientenmatrix, die die Abhängigkeitsstruktur des Graphen bestimmt. Damit können wir zeigen, dass dieser Schätzer an dem linken Limit des Trägers vollständig durch die Verteilung der Rauschvariablen bis zu einer positiven Konstante bestimmt wird. Wenn die Rauschterme regulär variieren, zeigen wir, dass die geschätzte ML-Koeffizientenmatrix, basierend auf dem Minimum-Ratio-Schätzer nach ordnungsgemäßer Zentrierung und Skalierung zu einer Matrix unabhängiger Weibull-Einträge konvergiert. Dies erweitert frühere Ergebnisse über den Minimum-Ratio-Schätzer.

In Kapitel 3 stellen wir QTree vor, eine neuartige Technik zum Erlernen wurzelgerichteter Spannbäume, einer speziellen Klasse von Bayes'schen Netzen. Inspiriert, aber nicht auf max-lineare Netzwerke beschränkt, verwendet der QTree-Algorithmus einen paarweisen Score für Knoten basierend auf der Dispersion im unteren Quantil, um die Konzentration der Verteilung an ihrem Minimum zu messen. Danach wenden wir Chu-Liu/Edmonds Algorithmus an, um den wurzelgerichteten Spannbaum mit minimalem Score zu finden. Wir führen auch eine Parameterauswahlmethode ein, die auf Bootstrap-Aggregation basiert. Angewendet auf verschiedene Datensätze von Flussabflussdaten übertrifft der Algorithmus bestehende State-of-the-Art-Methoden. Wenn die Daten einem verrauschten rekursiven max-linearen Modell folgen, zeigen wir unter einer milden Annahme des Rauschterms die asymptotische Konsistenz des geschätzten wurzelgerichteten Spannbaums $\hat{\mathcal{T}}$.

In Kapitel 4 stellen wir einen Ansatz vor, der auf Machine Learning beruht, um Bayes'sche Netze aus extremen Daten zu lernen. Basierend auf dem Gradientenverfahren finden wir die Parameter, die extreme Daten optimal an ein max-lineares Modell anpassen. Unter Verwendung vorhandener Literatur nutzen wir eine Charakterisierung der Azyklizität im Sinne einer glatten Funktion, um sicherzustellen, dass die Lösung auf einem Bayes'schen Netzwerk unterstützt wird. Wir zeigen die asymptotische Konsistenz unter milden Annahmen und veranschaulichen die Leistung in einer Simulationsstudie sowie an zwei realen Datensätzen: Daten zu Flussabflüssen aus Kapitel 2 und Daten zu Wechselkursen.

Acknowledgement

I would like to thank everyone that has accompanied me on this journey.

First and foremost my supervisor, Claudia Klüppelberg, for introducing me to this research topic, for guiding, motivating, mentoring, and, if necessary, to bring me back to the right track. My gratitude also goes to the Chair of Mathematical Statistics and every former and current member I met along the way for creating an open environment of fruitful discussions that helped to spark new ideas and exchange knowledge.

A special thanks goes to Ngoc Tran for enlightening discussions and constructive feedback which greatly helped me to progress in this thesis. Also, her hospitality during my stay at UT Austin was greatly appreciated.

I would like to thank the Hanns Seidel Foundation for not just financially supporting me but also for promoting a network for exchanging ideas and meeting people from various different fields. I would also like to thank them for their guidance and conceptual support.

Finally, I would like to thank my family, particularly my mother, for always supporting, helping, and believing in me.

Contents

List of Figures

List of Tables

Notation and Acronyms

$\bigvee_{k\in S} k$	Maximum over all elements in S.
$\bigwedge_{k\in S} k$	Minimum over all elements in S.
$\mathrm{An}(i)$	Set of ancestors of i, including i.
$\mathrm{an}(i)$	Set of ancestors of i, excluding i.
B	ML coefficient matrix/Kleene-star matrix.
C	Edge weight matrix.
C^*	ML coefficient matrix/Kleene-star matrix.
DAG	Directed Acyclic Graph.
$\mathcal{D}^B$	Minimum ML DAG.
$\mathrm{de}(i)$	Set of descendants of i, excluding i.
FDR	False Discovery Rate.
FPR	False Positive Rate.
ML	Max-Linear.
nSHD	Normalized Structural Hamming Distance.
$\mathrm{pa}(i)$	Set of parents of i.
$\mathrm{supp}(X)$	Support of X.
TPR	True Positive Rate.

Chapter 1

Introduction

In this thesis, we investigate recursive max-linear (ML) models under observational noise. We first introduce the key concepts that we are going to use subsequently. A *recursive ML model* is of the form

$$X_i = \bigvee_{j \in \mathrm{pa}(i)} c_{ij} X_j \vee Z_i, \quad i = 1, \ldots, d \tag{1.1}$$

where the dependence structure between random variables is represented by a DAG $\mathcal{D} := (V, E)$ with node set $V := \{1, \ldots, d\}$ and edge set $E = E(\mathcal{D}) \subseteq V \times V$, and each variable X_i for $i \in V$ has a representation in terms of ML functions of its parental nodes $\mathrm{pa}(i) = \{j \in V : (j,i) \in E\}$ and an independent innovation Z_i. The model has been defined first in Gissibl and Klüppelberg [2018] and subsequently considered in the literature as a tool for modeling extremal dependence(Buck and Klüppelberg [2021], Gissibl and Klüppelberg [2018], Gissibl et al. [2018, 2021], Klüppelberg and Krali [2021], Tran et al. [2021]).

Observe that recursive ML models are a special class of recursive *structural equation models (SEM)*, where X_i is given as a function f_i of its parental nodes $X_{\mathrm{pa}(i)}$ and an independent innovation Z_i, i.e.

$$X_i = f_i(X_{\mathrm{pa}(i)}, Z_i), \quad i = 1, \ldots, d.$$

The edge weight matrix $C = (c_{ij})_{d \times d}$ is generally not identifiable, see Example 4.3.1 in Gissibl et al. [2021]. Therefore, for a path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ from j to i, we define the *path weight*

$$d_{ij}(p) := \prod_{l=0}^{n-1} c_{k_{l+1} k_l}. \tag{1.2}$$

Denoting the set of all paths from j to i by P_{ij}, we introduce the *ML coefficient matrix* $B := (b_{ij})_{d \times d}$ as

$$b_{ij} := \bigvee_{p \in P_{ij}} d_{ij}(p) \quad \text{for } j \in \mathrm{an}(i), \quad b_{ii} = 1, \quad \text{and} \quad b_{ij} = 0 \quad \text{for } j \in V \setminus \mathrm{An}(i). \tag{1.3}$$

Here, $\mathrm{an}(i)$ denotes the set of ancestors of i, i.e. $\mathrm{an}(i) = \{j \in V : P_{ij} \neq \emptyset\}$ and $\mathrm{An}(i) = \mathrm{an}(i) \cup \{i\}$. Then, (1.1) can be written in terms of the ML coefficient matrix as

$$X_i = \bigvee_{j \in \mathrm{An}(i)} b_{ji} Z_j, \quad i = 1, \ldots, d. \tag{1.4}$$

Unlike the edge weight matrix C, the ML coefficient matrix is identifiable (see Theorem 3.1. in Gissibl et al. [2021]). In tropical algebra, the ML coefficient matrix is also known as the Kleene star matrix and is usually denoted by C^*. Within this dissertation, we will use the two terms interchangeably.

Given equation (1.2), we can see that B is solely determined by edge weights c_{ji} that are part of a max-weighted path. For this reason, we call a path p from j to i *critical*, if $d_{ij}(p) = b_{ij}$. Edge weights that are not part of any max-weighted path are redundant as they do not contribute to the distribution. This motivates the *minimum ML DAG* $\mathcal{D}^B$ given as

$$\mathcal{D}^B = (V, E^B) := \Big(V, \Big\{(j,i) \in E : b_{ij} > \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{pa}(i)} b_{kj} b_{ik}\Big\}\Big),$$

where $\mathrm{de}(j)$ denotes the set of descendants, i.e. $\mathrm{de}(j) = \{k \in V : P_{kj} \neq \emptyset\}$. Since the Kleene star matrix B solely determines the dependence structure of a recursive ML model, estimation of B became a central interest in research. Mainly two approaches have been considered. The first involves the estimation of scaling parameters. For a regularly varying vector X that follows a recursive ML model and $1 \leq j, i \leq d$ define the scaling parameters

$$\sigma_{ij}^2 = \int_{\Theta_+^{d-1}} \omega_i \omega_j dH_x(\omega), \quad \omega = (\omega_1, \ldots, \omega_d) \in \Theta_+^{d-1}.$$

Here, X is multivariate regularly varying, H_X is the spectral measure, Θ_+^{d-1} is the positive unit sphere and (R, ω) is the polar representation of X, i.e. $\omega_i = X_i/R$. Then, by a connection between σ_{ij}^2 and B, it is possible to estimate the scaling parameters and then to recursively estimate B. For more information, see Klüppelberg and Krali [2021]. However, the method bears significant drawbacks. First, the approach requires knowledge of a topological order of the nodes, i.e. an order of nodes π such that $\pi(j) < \pi(i)$ for all $(j,i) \in E$ which needs to be estimated separately. Moreover, consistency can only be guaranteed for a recursive ML model *not* polluted with any noise.

The second approach is based on pairwise minimum ratios. Observe that (1.4) is equivalent to

$$X_i = \bigvee_{j \in \mathrm{An}(i)} b_{ij} X_j, \quad i = 1, \ldots, d,$$

see e.g. (4.8). Therefore, X_i/X_j is lower bounded by $b_{ij} > 0$ if and only if $j \in \mathrm{An}(i)$. Moreover, it is easy to verify, that for an i.i.d. sample $\{x^1, \ldots, x^n\}$, the estimator

$$\bigwedge_{t=1}^{n} x_i^t / x_j^t$$

converges almost surely to b_{ij} for $n \to \infty$. Indeed, Gissibl et al. [2021] show that a version of this estimator is the generalized maximum likelihood estimator in the sense of Kiefer and Wolfowitz [1956]. However, also this approach has serious setbacks. Similarly to the first approach, the minimum ratio is sensitive towards misspecifications and consistency can only be guaranteed for a recursive ML model *not* polluted with any noise.

In recent years, recursive max-linear experienced increasing interest in the literature. In Klüppelberg and Sönmez [2022], the authors propose an extension of recursive max-linear models by introducing them on the oriented square lattice $\mathbb{Z}^2$. In a first step, for two points $j = (j_1, j_2)$ and $i = (i_1, i_2) \in \mathbb{Z}^2$, the authors assume an edge $j \to i$, whenever the Manhattan metric $\delta(j,i) = |i_1 - j_1| + |i_2 - j_2|$ is equal to one and $i_1 - j_1 \geq 0$ and $i_2 - j_2 \geq 0$. This introduces a special class of recursive ML models that is supported on Bayesian networks with an infinite number of nodes.

In a next step, the authors introduce a model where the edge set is determined by a realization of a sequence of i.i.d. Bernoulli random variables making the graph itself random. Then, they study basic properties, particularly the probability of independence for pairs of nodes.

In Hollering and Sullivant [2021], the authors consider a special class of discrete max-linear models where edge weights c_{ij} are equal to one for all $(j,i) \in E$ and the innovations $Z_1, \ldots, Z_d$ are k-state discrete random variables. The authors investigate structural properties and show that these models are isomorphic to conjunctive Bayesian network models.

In Améndola et al. [2022], the authors investigate conditional independence relations for recursive ML models. They propose $*$-separation for max-linear models, an alternative to the standard d-separation.

For a max-linear network over a DAG $\mathcal{D} = (V, E)$ and subsets, $I, J, K \subseteq V$, the authors show that if $I \perp\!\!\!\perp_* J | K$, then $X_I \perp\!\!\!\perp X_J | X_K$. Moreover, in Améndola et al. [2021], the authors show that $*$-separation and d-separation induce the same Markov equivalence classes on a DAG $\mathcal{D}$.

Finally, in Asenova and Segers [2022], the authors show that max-linear models are identifiable for a special class of Bayesian networks which they refer to as trees of transitive tournaments.

This thesis builds upon the estimation of recursive ML models. In particular, we are interested how to extend the existing literature to learn recursive max-linear noise under observational noise. We will now briefly address the approach and scope of each of the three following chapters.

Chapter 2: Recursive max-linear models with propagating noise

In Chapter 2, we are going to investigate the influence of propagating one-sided noise. We call a vector $\boldsymbol{U} \in \mathbb{R}_+^d$ a *recursive ML vector with propagating noise* on a DAG $\mathcal{D} = (V, E)$, if

$$U_i := \Big(\bigvee_{j \in \mathrm{pa}(i)} c_{ij} U_j \vee Z_i \Big) \varepsilon_i, \quad i \in 1, \ldots, d, \tag{1.5}$$

with edge weight matrix $C := (c_{ij}\mathbf{1}_{\mathrm{pa}(i)}(j))_{d\times d}$. Observe that we use U instead of X to better distinguish the progagating noise model from the original model. The noise variables $\varepsilon_1, \dots, \varepsilon_d$ are i.i.d. and atom-free random variables with $\varepsilon_i \geq 1$ and unbounded above for all $i \in V$, and independent of the innovations vector $\boldsymbol{Z} := (Z_1, \dots, Z_d)$.

This setting has interesting structural features. For the noise-free model, b_{ij} is defined as the maximum path weight from j to i, see (1.3). However, for propagating noise, the maximum path weight becomes random and different paths can become critical, depending on the realization. This gives the motivation to define *possible critical path realizations*.

However, interestingly, the minimum DAG $\mathcal{D}^B$, i.e. the minimum DAG that preserves the distribution remains unchanged. Moreover, even though there might be many possible critical paths, up to a constant the left limit of support of a ratio U_i/U_j is entirely determined by the path $d_{ij}(p)$ where $d_{ij}(p) = b_{ij}$. For this reason, given a regular variation condition, we prove that the estimated Kleene star matrix $\hat{B}$ derived from minimum ratios decomposes and, hence, converges to a matrix of independent Weibull entries after proper centering and rescaling, see Theorem 2.5.2.

As for the estimation, we consider three settings.

(1) All ancestral relations are known; i.e., we know the set of edges E, hence the DAG. This might be the case when modeling networks that contain natural information about edges. The problem then reduces to finding appropriate estimates $\hat{b}_{ij}$ for $j \in \mathrm{an}(i)$.

(2) The ancestral relations are unknown; however, we know a topological order of the nodes. Then, in contrast to setting 1, we need to decide if a path from j to i with $j < i$ exists.

(3) Neither the underlying DAG nor a topological order of the nodes is known.

For each setting, we provide algorithms to ensure a consistent estimator. We show its performance in a simulation study as well as a real-world data set on dietary supplements.

Chapter 3: Estimating a Directed Tree for Extremes

While Chapter 2 gives interesting insights into the properties of recursive ML models, the minimum ratio estimator is very sensitive to misspecifications and, therefore, often not applicable to causal inference. In Chapter 3, we introduce QTree, a simple and efficient algorithm to learn a graph over the class of root-directed spanning trees. A root-directed spanning tree $\mathcal{T} = (V, E)$ is a directed graph such that each node $i \in V$ except the root r has exactly one child, the root r has none, and there is a path from every node $i \neq r$ to r.

For the theoretical properties of the algorithm, particularly the proof of consistency, we assume that the data follows a recursive ML model. The idea is to introduce a measure of concentration between ratios of random variables. This idea, however, applies for a wide variety of SEMs such that the algorithm is inspired but generally not limited to recursive ML models. Moreover, for numerical stability and unlike in Chapter 2, we work with the logarithm of the extreme data. Therefore, the equivalent log model is given as

$$X_i = \bigvee_{j \in \mathrm{pa}(i)} (c_{ij} + X_j) \vee Z_i, \quad c_{ij}, Z_i \in \mathbb{R}, \quad i \in V. \tag{1.6}$$

Our aim is to learn $\mathcal{T}$ from an i.i.d. sample $\mathcal{X} = \{x^1+\varepsilon^1, \dots, x^n+\varepsilon^n\} \in \mathbb{R}^d$, where the x_i are generated via (1.6), and the ε^t are independent noise variables in $\mathbb{R}^d$. Similarly to the minimum-difference estimator, the approach relies on pairwise observations. In the first step, we define the pairwise sample $\mathcal{X}_{ij}$ as

$$\mathcal{X}_{ij}(\alpha) := \{x_i - x_j : x \in \mathcal{X}, x_j > Q_{\mathcal{X}_j}(\alpha)\}, \tag{1.7}$$

where $Q_{\mathcal{X}_j}(\alpha)$ is the α-th quantile of the empirical distribution of $\mathcal{X}$ in the j-th coordinate. An illustration of the resulting distribution can be found in Figure 3.1. By restricting to such observations where X_j is large, we filter for strong signals such that the noise is small relative to the signal.

Given pairwise samples $\mathcal{X}_{ij}(\alpha)$, we propose the following score

$$w_{ij}(\underline{r}) := \Big(\mu(\mathcal{X}_{ij}(\alpha)) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r})\Big)^2,$$

where $\underline{r} \in (0,1)$ is a fixed, small empirical quantile and $\mu(\mathcal{X}_{ij}(\alpha))$ the empirical mean. Here, the mean works as a normalization, and the strict minimum is replaced by a small quantile level α. For this reason, the scores can be seen as a generalization of the minimum-difference estimator.

We apply the algorithm to four data sets of river discharge data, the *Upper Danube Basin* and three parts of the *Lower Colorado River* in Texas. A separation of the Lower Colorado River is necessary due to dams dividing the network. Compared to state-of-the-art methods, we get highly significant results, uniformly outperforming all algorithms. We also suggest a stabilizing subsampling procedure for the two parameters $(\alpha, \underline{r})$. Assuming that the noise follows a light tail, we prove asymptotic consistency of the estimated root-directed spanning tree $\mathcal{T}$ under a max-linear network.

Finally, we also conduct a simulation study to show that QTree is robust with respect to different dependence structures (given by edge weights) and different node distributions entailed from different innovation distributions.

Chapter 4: Learning Bayesian Networks from Extreme Data

Inspired by the success in Chapter 3, we want to extend the theory of QTree. Particularly, the assumption of an underlying root-directed spanning tree is strict and often does not hold.

In Chapter 4, we introduce a machine learning approach to learn recursive ML models without any assumption on the class of Bayesian network. This work improves upon the previous chapter by introducing a novel estimator based on the optimization framework of Zheng et al. [2018]. The authors achieve a novel characterization of acyclicity in terms of a smooth function

$$\mathrm{tr}(\exp(C \circ C)),$$

where $\circ$ is the Hadamard product, $\mathrm{tr}(A)$ denotes the trace and $\exp(A)$ is the matrix exponential of A. A matrix $C \in \mathbb{R}^{d\times d}$ is supported on a Bayesian network if and only if the upper expression equals d.

Using this result, we introduce an optimization framework to estimate the Kleene star matrix defined in (1.3) that best fits the data. Moreover, we show that the recursive ML model can be seen as a special case of a max-out network. We can therefore write recursive max-linear models in terms of a neural network and use gradient descent to find the estimator that best fits the data.

We call this algorithm MLDAG and we also introduce a subsampling procedure to automatically choose the set of parameters. We apply the estimator to the data set of river discharges of the Upper Danube Basin which we already considered in Chapter 3, leading to competitive results. We also apply the algorithm to data on foreign exchange rates in terms of the British Pound sterling leading to a sensible estimation of the causal structure.

Finally, for an i.i.d. sample $\mathcal{X} = \{x^1 + \varepsilon^1, \ldots, x^n + \varepsilon^n\}$ in $\mathbb{R}^d$, where the x_i are generated via (1.6), and the ε^t are independent noise variables in $\mathbb{R}^d$, under some mild assumption, we prove the asymptotic consistency of the estimator.

Each chapter of the thesis is based on a paper or a manuscript that is very close to being submitted:

Chapter 2 J. Buck and C. Klüppelberg. Recursive max-linear models with propagating noise. Electronic Journal of Statistics, 15(2):4770 – 4822, 2021. doi: 10.1214/21-EJS1903.

Chapter 3 N. M. Tran, J. Buck, and C. Klüppelberg. Estimating a directed tree for extremes, arXiv preprint: 2102.06197, Revision under review at The Journal of the Royal Statistical Society, Series B, 2021

Chapter 4 J. Buck and N. M. Tran. Learning Bayesian Networks from Extreme Data. In preparation, 2023.

Each chapter is self-contained, i.e. it contains an introduction which introduces the necessary notation, methodology and literature to understand the respective chapter. Different notations, abbreviations, and model assumptions on a recursive ML model seem reasonable in different settings; therefore, they might differ from chapter to chapter.

Chapter 2

Recursive max-linear models with propagating noise

2.1 Introduction

Graphical modeling has shown to be a powerful tool for understanding causal dependencies in a multivariate random vector. However, most models are linear and limited to discrete or Gaussian distributions (see e.g. Koller and Friedman [2009] and Lauritzen [1996]). Such models lead to severe underestimation of large risks and, therefore, are not suitable in the context of extreme risk assessment. First examples combining extreme value methods with graphical models include flooding in river networks (Engelke and Hitz [2020]), financial risk (Einmahl et al. [2017], Klüppelberg and Krali [2021]), and nutrients (Klüppelberg and Krali [2021]).

We consider the class of recursive max-linear (ML) models, which has been defined in Gissibl and Klüppelberg [2018]. A recursive ML model is defined by a structural equation model (SEM) of the form

$$X_i = \bigvee_{j \in \mathrm{pa}(i)} c_{ij} X_j \vee Z_i, \quad i = 1, \dots, d \tag{2.1}$$

where the dependence structure between random variables is represented by a DAG $\mathcal{D} := (V, E)$ with node set $V := \{1, \dots, d\}$ and edge set $E = E(\mathcal{D}) \subseteq V \times V$, and each variable X_i for $i \in V$ has a representation in terms of ML functions of its parental nodes $\mathrm{pa}(i) = \{j \in V : (j, i) \in E\}$ and an independent innovation Z_i.

Both, SEMs (e.g. Bollen [1989], Pearl [2009]) and directed graphical models (e.g. Koller and Friedman [2009], Lauritzen [1996], Spirtes et al. [2000]) are well-established models and widely used to understand causality.

ML models similar to (2.1) have been proposed and studied in a time series context (e.g. Davis and Resnick [1989]), in terms of moving maxima processes (e.g. Hall et al. [2002]), or as tropical models in algebra (e.g. Joswig [2020], Maclagan and Sturmfels [2015]) with applications to various optimization problems (e.g. Baccelli et al. [1992], Butkovič [2010], Tran and Yu [2019]).

As shown in Klüppelberg and Lauritzen [2020], recursive ML models respect the basic Markov properties associated with DAGs (e.g. Lauritzen [2001], Lauritzen et al. [1990]).

Moreover, the equation system (2.1) has the solution

$$X_i = \bigvee_{j \in \mathrm{pa}(i)} b_{ij} Z_j, \quad i = 1, \ldots, d, \tag{2.2}$$

with ML coefficient matrix (also known as Kleene star matrix) $\boldsymbol{B} := (b_{ij})_{d \times d}$, see Butkovič [2010], Corollary 1.6.16. Unlike the edge weight matrix $\boldsymbol{C} = (c_{ij})_{d \times d}$, $\boldsymbol{B}$ is identifiable and completely determines the distribution of $\boldsymbol{X} := (X_1, \ldots, X_d)$ (see Gissibl et al. [2021], Theorem 1). Also, $\boldsymbol{B}$ is idempotent with respect to the tropical matrix multiplication defined in (2.9) below, and defines a graphical model on a DAG with node set V and an edge $j \to i$ whenever there is a path from j to i in $\mathcal{D}$. Furthermore, Gissibl et al. [2021] proposes a minimum ratio estimator for $\boldsymbol{B}$, which itself is idempotent, and is a generalized maximum likelihood estimator in the sense of Kiefer and Wolfowitz [1956].

The model (2.1) states that an extreme node observation X_i in the DAG is either the result of a large external innovation Z_i, or the weighted maximum of observations from the parent nodes of i in $\mathcal{D}$. As we see from the solution (2.2), all past innovations drive this observation. Our aim is to generalize this rather restricted recursive structure as to allow for certain observation errors, independent of this model.

More precisely, we extend the original model (2.1) by allowing for multiplicative observation errors and define

$$U_i = \Big(\bigvee_{j \in \mathrm{pa}(i)} c_{ij} U_j \vee Z_i \Big) \varepsilon_i, \quad i = 1, \ldots, d, \tag{2.3}$$

with $\varepsilon_i \geq 1$ and i.i.d. for $i = 1, \ldots, d$. By taking advantage of tropical algebra, we present in Theorem 2.3.2 a solution of (2.3) which represents each node variable U_i in terms of a ML function of its ancestral nodes and an independent innovation Z_i given by

$$U_i = \bigvee_{j \in \mathrm{an}(i) \cup \{i\}} \bar{b}_{ij} Z_j, \quad i = 1, \ldots, d,$$

where an(i) denotes the ancestors of i and $\bar{b}_{ij}$ are random variables involving the edge weights and the noise variables.

It comes as no surprise that the true DAG and edge weights for a recursive ML model with propagating noise inherit the non-identifiability property from the non-noisy model. However, as we will prove in Section 2.4, the ML coefficient matrix $\boldsymbol{B} = (b_{ij})_{d \times d}$ remains identifiable in spite of the observational noise and even if we do not know the underlying DAG.

To link up our new model (2.3) with existing literature, observe that a log-transformation of (2.3) yields

$$\tilde{U}_i = \bigvee_{j \in \mathrm{pa}(i)} (\tilde{c}_{ij} + \tilde{U}_j) \vee \tilde{Z}_i + \tilde{\varepsilon}_i, \quad i = 1, \ldots, d \tag{2.4}$$

with $\tilde{\varepsilon}_i \geq 0$. However, due to the maximum operator, the class of max-linear models is highly non-smooth such that most standard methods do not apply. For every $j \in \mathrm{pa}(i)$, the difference $\tilde{U}_i - \tilde{U}_j$ is lower-bounded by $\tilde{c}_{ij}$ and

$$\mathbb{P}(\tilde{U}_i - \tilde{U}_j \leq \tilde{c}_{ij} + x \mid \tilde{U}_i = \tilde{c}_{ij} + \tilde{U}_j + \tilde{\varepsilon}_i) = \mathbb{P}(\tilde{\varepsilon}_i \leq x).$$

Example 1.2.2 of Butkovič [2010] might then serve as motivating example for model (2.4). Assume that i is a priority flight and there are several feeder flights to i. Assume that the departure of a feeder flight j is delayed by $\tilde{U}_j$, and there are passengers, who have to catch flight i. Moreover, let $\tilde{c}_{ij}$ be the departure time of flight i minus arrival time of flight j minus transit time according to schedule. There may be a further delay $\tilde{Z}_i$ of flight i caused by non-connecting passengers. Assuming that flight i also waits for such non-connecting passengers, the delay of flight i is given by $\tilde{U}_i = \bigvee_{j\in \mathrm{pa}(i)}(\tilde{c}_{ij} + \tilde{U}_j) \vee \tilde{Z}_i$. Some other delays are independent of possible passenger delays like bad weather conditions, delayed start clearance etc. Then we find exactly model (2.4), the max-linear model with positive noise.

The estimation of (linear) functions with one-sided errors has been considered in the literature before. For instance, in Hall and Van Keilegom [2009] and Jirak et al. [2014] observations are given by $Y_j = f(X_j) + \varepsilon_j$ for $j = 1, \dots, n$ with observation errors $\varepsilon_j > 0$, with density given conditionally or unconditionally on $X_j = x$, and f describes some frontier or boundary curve, which has to be estimated. To present an archetypical example, consider the linear regression problem stated in Smith [1985] and Smith [1994] as $Y_i = \beta + \varepsilon_i$ for $i = 1, \dots, n$ and observation errors, which have density $g(x) \sim \alpha c x^{\alpha-1}$ as $x \downarrow 0$ for $\alpha, c > 0$. In these papers, the focus is on the non-regular case, when $\alpha < 2$. Then β can be estimated by the sample minimum $Y_{1,n}$ which has a Weibull limit law:

$$\lim_{n\to\infty} \mathbb{P}\big((nc)^{-1/\alpha}(Y_{1,n} - \beta_0) \le x\big) = 1 - \exp(-x^{-\alpha}), \quad 0 < x < \infty. \tag{2.5}$$

The work in Smith [1985] has been used in Davis and McCormick [1989] to estimate the coefficient ϕ of a first order autoregressive time series with positive innovations. They propose the minimum ratio estimator $\hat{\phi} = \bigwedge_{j=1}^{n} X_j/X_{j-1}$ and show in their Corollary 2.4 that it also has a Weibull limit law similar to (2.5).

In our model (2.3) we find two interpretations for the noise variables. Firstly, in the log-transformed version (2.4), we consider a ML model as baseline model, which is observed with some additive noise. A second representation is given in Corollary 2.3.3 below, where the edge and path weights become noisy by the noise variables. This gives rise to the interpretation that we observe the model parameters with noise similarly as in the regression examples above. As a consequence, a path from j to i realizing the ML coefficient b_{ij} is no longer deterministic but depends on the individual realizations of the noise variables. However, in Theorem 2.3.12 we show that at the left limit of support the distribution of the ratio of two model components is determined by all noise variables along the path between the two nodes. Assuming noise variables with regularly varying distribution in their left limit of support, we propose a minimum ratio estimator and show in Theorem 2.5.2 that the estimated ML coefficient matrix converges to a matrix of independent Weibull entries after proper centering and rescaling.

The chapter is organized as follows. In Section 2.2, we summarize the properties of recursive ML models as defined in (2.1) and state the most important results relevant for this chapter. In Section 2.3 we consider the extension of the recursive ML model given in (2.3), which we coin the *max-linear model with propagating noise* and present its solution and the main properties of this new model. In Section 2.4 we address the identifiability of the ML model with propagating noise. Similarly as in (2.5) we suggest minimum ratio estimators for the model parameters $\boldsymbol{B}$. In Section 2.5 we assume regular variation of the noise variables. Under this assumption, we show that the minimum ratios are asymptotically independent and Weibull distributed. Finally, in Section 2.6, we provide a data example and apply the theory that we have derived in the previous sections. All proofs are postponed to a section of Supplementary Material.

Throughout we use the following notation. $\mathbb{R}_+ = (0, \infty)$ and $\overline{\mathbb{R}}_+ = [0, \infty)$, $x \wedge y = \min\{x, y\}$ and $x \vee y = \max\{x, y\}$ with $\bigwedge_{i\in\emptyset} x_i = \infty$ and $\bigvee_{i\in\emptyset} x_i = 0$ for $x_i \in \mathbb{R}_+$. Bold letters denote vectors and matrices, e.g. $\boldsymbol{I}_d$ denotes the $d \times d$ identity matrix. Moreover, all vectors are column vectors unless stated otherwise. For two functions f, g we write $f(x) \sim g(x)$ as $x \downarrow c$ if $\lim_{x\downarrow c} f(x)/g(x) = 1$ and $\mathbf{1}$ denotes the indicator function. For a random variable Y with distribution function F_Y, the symbol $F_Y^{\leftarrow}$ denotes its *quantile function.*

2.2 Preliminaries — Recursive max-linear models

2.2.1 Graph terminology

We use the same graph notation as in Gissibl and Klüppelberg [2018]. A *directed graph* is a pair (V, E) of a *node set* $V = \{1, \ldots, d\}$ and an *edge set* $E = \{j \to i : i, j \in V, i \neq j\}$. A node j is called a *parent* of i if $j \to i \in E$ and write $(j, i) \in E$. A *(directed) path* from j to i is a sequence of distinct nodes $[j = k_0, k_1, \ldots, k_n = i]$ such that $k_{r-1} \to k_r$ for each $r \in \{1, \ldots, n\}$, and a directed cycle is a path where $j = i$. A node j is called an *ancestor* of i, if there exists a path from j to i, then i is called a *descendant* of j. The node sets $\mathrm{pa}(i)$, $\mathrm{an}(i)$ and $\mathrm{de}(i)$ denote the parents, ancestors, and the descendants of node i, respectively, and we abbreviate $\mathrm{An}(i) := \mathrm{an}(i) \cup \{i\}$.

Finally, for a path $p = [k_0, k_1, \ldots, k_n]$ we define the *node set on the path (excluding the initial node)* by $S_p := \{k_1, \ldots, k_n\}$ and its *path length* by $|S_p|$.

Throughout this chapter $\mathcal{D} = (V, E)$ is a directed acyclic graph (DAG), and we recall that a complete DAG is a complete graph with directed edges.

A matrix $\boldsymbol{C} \in \overline{\mathbb{R}}_+^{d\times d}$ defines a *weighted directed graph*, where $j \to i \in \mathcal{D}$ if and only if its *edge weight* c_{ij} is positive. The *path weight* of a path in $\mathcal{D}$ is then the product of its edge weights.

For a DAG $\mathcal{D}$ on V with edge weight matrix $\boldsymbol{C}$, its *reachability DAG* is defined as a DAG on V having edge $j \to i$ if and only if $\mathcal{D}$ has a path from $j \to i$; moreover, $\boldsymbol{B}$ represents the edge weight matrix of the reachability DAG. We call $\boldsymbol{B}$ the *ML coefficient matrix* and remark that it is a *weighted reachability matrix* for $\mathcal{D}$.

2.2.2 Recursive max-linear models

We first formally introduce the class of recursive ML models and state their most important results for this chapter. Let $\mathcal{D} = (V, E)$ be a DAG. Then a random vector $\boldsymbol{X} := (X_1, \ldots, X_d)$ is a *recursive max-linear vector* or follows a *max-linear Bayesian* *network* on $\mathcal{D}$ if

$$X_i := \bigvee_{j \in \mathrm{pa}(i)} c_{ij} X_j \vee Z_i, \quad i \in 1, \dots, d, \tag{2.6}$$

with positive edge weights c_{ij} for $i \in V$ and $j \in \mathrm{pa}(i)$, and independent positive random variables $Z_1, \dots, Z_d$ with support $\mathbb{R}_+$ and atom-free distributions. We shall refer to $\boldsymbol{Z} := (Z_1, \dots, Z_d)$ as the *vector of innovations.*

For a path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ from j to i we define the *path weight*

$$d_{ij}(p) := \prod_{l=0}^{n-1} c_{k_{l+1}k_l}. \tag{2.7}$$

Denoting the set of all paths from j to i by P_{ij}, we define the ML coefficient matrix $\boldsymbol{B} = (b_{ij})_{d \times d}$ of $\boldsymbol{X}$ with entries

$$b_{ij} := \bigvee_{p \in P_{ij}} d_{ij}(p) \quad \text{for } j \in \mathrm{an}(i), \quad b_{ii} = 1, \quad \text{and} \quad b_{ij} = 0 \quad \text{for } j \in V \setminus \mathrm{An}(i).$$

The components of $\boldsymbol{X}$ can also be expressed as ML functions of their ancestral innovations and an independent one; the corresponding ML coefficients are the entries of $\boldsymbol{B}$:

$$X_i = \bigvee_{j \in \mathrm{An}(i)} b_{ij} Z_j, \quad i \in 1, \dots, d, \tag{2.8}$$

which can be shown by a path analysis as in Theorem 2.2 in Gissibl and Klüppelberg [2018] or by tropical algebra as in (2.11) below, and as we explain now.

For two non-negative matrices $\boldsymbol{F}$ and $\boldsymbol{G}$, where the number of columns in $\boldsymbol{F}$ is equal to the number of rows in $\boldsymbol{G}$, we define the matrix product $\odot : \overline{\mathbb{R}}_+^{m \times n} \times \overline{\mathbb{R}}_+^{n \times p} \to \overline{\mathbb{R}}_+^{m \times p}$ by

$$(\boldsymbol{F} = (f_{ij})_{m \times n}, \boldsymbol{G} = (g_{ij})_{n \times p}) \mapsto \boldsymbol{F} \odot \boldsymbol{G} := \Big(\bigvee_{k=1}^{n} f_{ik} g_{kj} \Big)_{m \times p}. \tag{2.9}$$

The triple $(\overline{\mathbb{R}}_+, \vee, \cdot)$, is an idempotent semiring with 0 as 0-element and 1 as 1-element and the operation $\odot$ is therefore a matrix product over this semiring; see for example Butkovič [2010]. Denoting by $\mathcal{M}$ all $d \times d$ matrices with non-negative entries and by $\vee$ the componentwise maximum between two matrices, $(\mathcal{M}, \vee, \odot)$ is also a semiring with the null matrix as 0-element and the $d \times d$ identity matrix $\boldsymbol{I}_d$ as 1-element.

The matrix product $\odot$ allows us to represent the ML coefficient matrix $\boldsymbol{B}$ of $\boldsymbol{X}$ in terms of the edge weight matrix $\boldsymbol{C} := (c_{ij} \mathbf{1}_{\mathrm{pa}(i)}(j))_{d \times d}$ of $\mathcal{D}$, since (2.6) can be rewritten as

$$\boldsymbol{X} = (\boldsymbol{C} \odot \boldsymbol{X}) \vee \boldsymbol{Z} \tag{2.10}$$

with unique solution (equivalent to (2.8)) given by

$$\boldsymbol{B} = (\boldsymbol{I}_d \vee \boldsymbol{C})^{\odot(d-1)} = \bigvee_{k=0}^{d-1} \boldsymbol{C}^{\odot k}, \qquad \boldsymbol{X} = \boldsymbol{B} \odot \boldsymbol{Z}, \tag{2.11}$$

where $\boldsymbol{B}$ is the *Kleene star matrix* and $\boldsymbol{A}^{\odot 0} = \boldsymbol{I}_d$ and $\boldsymbol{A}^{\odot k} = \boldsymbol{A}^{\odot(k-1)} \odot \boldsymbol{A}$ for $\boldsymbol{A} \in \overline{\mathbb{R}}_+^{d\times d}$ and $k \in \mathbb{N}$; see Proposition 1.6.15 of Butkovič [2010] as well as Theorem 2.4 and Corollary 2.5 of Gissibl and Klüppelberg [2018]. For more information on the max-times (tropical) algebra in ML models, see Section 2.2 in Améndola et al. [2022].

We have seen that a recursive ML vector $\boldsymbol{X}$ has two representations, one in terms of parental nodes X_j and edge weights c_{ij} and another in terms of innovations Z_j and ML coefficients b_{ij}. However, while the ML coefficient matrix $\boldsymbol{B}$ of $\boldsymbol{X}$ is identifiable from the distribution of $\boldsymbol{X}$, the edge weight matrix $\boldsymbol{C}$ is generally not, see Theorem 5.4(b) in Gissibl and Klüppelberg [2018]. Theorem 5.3 in that paper and Theorem 2 in Gissibl et al. [2021] show that an edge with edge weight c_{ij} is identifiable from $\boldsymbol{B}$ if and only if it is the unique path from j to i with $d_{ij}(p) = b_{ij}$.

For a recursive ML vector $\boldsymbol{X}$ on a DAG $\mathcal{D} = (V, E)$ and ML coefficient matrix $\boldsymbol{B}$ this result leads to the following definition.

Definition 2.2.1. Let $\boldsymbol{X} \in \mathbb{R}_+^d$ be a recursive ML vector on the DAG $\mathcal{D} = (V, E)$ with ML coefficient matrix $\boldsymbol{B}$. We define the *minimum ML DAG of* $\boldsymbol{X}$ as

$$\mathcal{D}^B = (V, E^B) := \Big(V, \Big\{(j,i) \in E : b_{ij} > \bigvee_{k \in \mathrm{de}(j)\cap \mathrm{pa}(i)} \frac{b_{kj} b_{ik}}{b_{kk}}\Big\}\Big).$$

Moreover, it has been shown that the support of a ratio of components of a recursive ML vector $\boldsymbol{X}$ satisfies

$$\mathrm{supp}(X_i/X_j) = \begin{cases} [b_{ij}, \infty) & \text{for } j \in \mathrm{an}(i), \\ [0, 1/b_{ji}] & \text{for } i \in \mathrm{an}(j), \\ \{1\} & \text{for } i = j, \\ \mathbb{R}_+ & \text{otherwise,} \end{cases} \tag{2.12}$$

with $\mathbb{P}(X_i/X_j = b_{ij}) > 0$ for all $j \in \mathrm{an}(i)$; see Lemma 1 of Gissibl et al. [2021]. Hence, for a given i.i.d. sample $\boldsymbol{X}^1, \dots, \boldsymbol{X}^n$ from $\boldsymbol{X}$ define a minimum ratio estimator $\hat{\boldsymbol{B}}$ of $\boldsymbol{B}$ by $\hat{b}_{ij} := \bigwedge_{k=1}^n (X_i^k / X_j^k)$ for $i, j \in V$. Moreover, when the DAG $\mathcal{D}$ is known, we define $\boldsymbol{B}_0$ by

$$\boldsymbol{B}_0 = (B_0(i,j))_{d\times d} := \Big(\bigwedge_{k=1}^n \frac{X_i^k}{X_j^k} \mathbf{1}_{\mathrm{pa}(i)}(j)\Big)_{d\times d} \quad \text{and set} \quad \hat{\boldsymbol{B}} = (\boldsymbol{I}_d \vee \boldsymbol{B}_0)^{\odot(d-1)}.$$

Theorem 4 of Gissibl et al. [2021] ensures that $\hat{\boldsymbol{B}}$ is a generalized maximum likelihood estimate (GMLE) in the sense of Kiefer and Wolfowitz [1956].

2.2.3 Minimum domain of attraction and regular variation

Introducing propagating noise into the recursive ML model will smooth out the atoms in (2.12). The minimum ratio estimators will still estimate the left endpoints $\boldsymbol{B} = (b_{ij})$ and we will be able to provide distributional limit results. These will be based on minimum domain of attraction results and regular variation. Extreme value theory is more focused nowadays on running maximima (e.g. Embrechts et al. [1997]), but results for running minima are obtained by noting that $\bigwedge_{i=1}^n Y_i = -\bigvee_{i=1}^n (-Y_i)$.

From this we obtain that the family of Weibull distributions are limit distributions of running minima of i.i.d. random variables, see equation (2.14).

Definition 2.2.2. A positive random variable Ψ_α is *Weibull distributed with left endpoint $x_L > -\infty$, shape* $\alpha > 0$ *and scale* $s > 0$ and we write $Y \sim \text{Weibull}(\alpha, x_L, s)$ if the distribution function of Y is given by

$$\Psi_{\alpha,x_L,s}(x) = 1 - \exp\left(-\left(\frac{x - x_L}{s}\right)^\alpha\right), \quad x \geq x_L.$$

Random variables, whose running minima have such a Weibull limit satisfy certain conditions. Here the following definition is essential and we refer to Bingham et al. [1987] for details.

Definition 2.2.3. Let Y be a random variable with distribution function F and left endpoint x_L. Then we call Y or F *regularly varying at x_L with exponent* $\alpha > 0$, if

$$\lim_{t\downarrow 0} \frac{F(x_L + tx)}{F(x_L + t)} = x^\alpha, \quad x > 0. \tag{2.13}$$

We abbreviate this by $Y \in RV_\alpha^{x_L}$ or $F \in RV_\alpha^{x_L}$, respectively. We also note that $Y \in RV_\alpha^{x_L}$ is equivalent to $Y - x_L \in RV_\alpha^0$.

Then, adapting Theorem 3.3.12 of Embrechts et al. [1997] to the minimum of i.i.d. random variables $X_1, \dots, X_d$ with distribution function F, we obtain

$$\begin{aligned} &\exists (a_n > 0) \text{ s.t. } \frac{1}{a_n}\Big(\bigwedge_{i=1}^n X_i - x_L\Big) \xrightarrow{d} \Psi_\alpha,\ n \to \infty \\ \iff\ & x_L > -\infty, F(x_L + \cdot) \in RV_\alpha^0. \end{aligned} \tag{2.14}$$

Let ε be a random variable with left endpoint $x_L = 1$, and $\tilde{\varepsilon} := \ln(\varepsilon)$. Then for $x > 0$,

$$\begin{aligned} \lim_{t\downarrow 0} \frac{P(\ln(\varepsilon) \leq tx)}{P(\ln(\varepsilon) \leq t)} &= \lim_{t\downarrow 0} \frac{P(\varepsilon \leq e^{tx})}{P(\varepsilon \leq e^t)} = \lim_{t\downarrow 0} \frac{P(\varepsilon - 1 \leq tx(1 + o(1))}{P(\varepsilon - 1 \leq t(1 + o(1))} \\ &= \lim_{t\downarrow 0} \frac{P(\varepsilon - 1 \leq tx)}{P(\varepsilon - 1 \leq t)}, \end{aligned} \tag{2.15}$$

such that $\varepsilon \in RV_\alpha^1$ if and only if $\tilde{\varepsilon} \in RV_\alpha^0$. Two relevant families of distribution functions are given in the next example.

Example 2.2.4. *(a) [Weibull distribution] Let $\tilde{\varepsilon}$ have distribution function as in Definition 2.2.2 with $x_L = 0$. Then by a l'Hospital argument,*

$$\lim_{t\downarrow 0} \frac{\Psi_{\alpha,s}(tx)}{\Psi_{\alpha,s}(t)} = x^\alpha,$$

which implies that $\tilde{\varepsilon} \in RV_\alpha^0$ and $\varepsilon \in RV_\alpha^1$.

(b) [Gamma distribution] Let $\tilde{\varepsilon}$ have density $g(x) = \lambda^\alpha e^{-\lambda x} x^{\alpha-1}/\Gamma(\alpha)$ for $x > 0$ and parameters $\lambda > 0, \alpha > 0$. Then by a l'Hospital argument,

$$\lim_{t\downarrow 0} \frac{G(tx)}{G(t)} = \lim_{t\downarrow 0} \frac{e^{-\lambda tx} t^{\alpha-1} x^\alpha}{e^{-\lambda t} t^{\alpha-1}} = x^\alpha, \quad x > 0,$$

which implies that $\tilde{\varepsilon} \in RV_\alpha^0$ and $\varepsilon \in RV_\alpha^1$. □

We provide here also some preliminary results.

Proposition 2.2.5 (Karamata's Tauberian Theorem 1.7.1, Bingham et al. [1987])**.** *Let U be a non-decreasing function on $\mathbb{R}$ with $U(x) = 0$ for all $x < 0$, and Laplace-Stieltjes transform $\hat{U}(s) = \int_{[0,\infty)} e^{-sx}$*
$dU(x) < \infty$ for all large s. For $l \in RV_0^\infty$ and $c \geq 0, \rho \geq 0$, the following are equivalent

$$\begin{aligned} U(x) &\sim cx^\rho l(1/x)/\Gamma(1+p), && x \downarrow 0 \\ \hat{U}(s) &\sim cs^{-\rho} l(s), && s \to \infty. \end{aligned} \tag{2.16}$$

From this, we obtain the following corollary.

Corollary 2.2.6. 1. *Let $X \in RV_{\alpha_1}^0, Y \in RV_{\alpha_2}^0$ be independent, then $X + Y \in RV_{\alpha_1+\alpha_2}^0$,*

2. *Let $X, Y \geq 1$ be independent and such that $\tilde{X} = \ln(X) \in RV_{\alpha_1}^0, \tilde{Y} = \ln(Y) \in RV_{\alpha_2}^0$. Then $(XY - 1) \in RV_{\alpha_1+\alpha_2}^0$.*

Proof of Corollary 2.2.6 (a) We use Proposition 2.2.5 a) for U being F_X or F_Y, the distribution function of X or Y, respectively. Since the Laplace-Stieltjes transforms $\hat{F}_X(s) = \int_{[0,\infty)} e^{-sx} dF_X(x) \leq 1$ and $\hat{F}_Y(s) = \int_{[0,\infty)} e^{-sx} dF_Y(x) \leq 1$ for all $s \geq 0$, by Proposition 2.2.5, they are both regularly varying at ∞ in the sense of (2.16); i.e., $\hat{F}_X \in RV_{\alpha_1}^\infty, \hat{F}_Y \in RV_{\alpha_2}^\infty$. By independence, the convolution theorem for Laplace-Stieltjes transforms gives $\hat{F}_{X+Y}(s) = \hat{F}_X(s)\hat{F}_Y(s)$ and, therefore, $\hat{F}_{X+Y} \in RV_{\alpha_1+\alpha_2}^\infty$. Applying again Proposition 2.2.5 we find that $X_1 + X_2 \in RV_{\alpha_1+\alpha_2}^0$.
(b) This follows from a Taylor expansion. □

2.3 Recursive ML model with propagating noise

In this section we define the recursive ML model with propagating noise, present structural results, investigate which properties of the non-noisy model prevail, and derive distributional results for component ratios of the model in preparation for the structure learning results to follow.

2.3.1 Definitions and representations

Definition 2.3.1. A vector $\boldsymbol{U} \in \mathbb{R}_+^d$ is a *recursive ML vector with propagating noise* on a DAG $\mathcal{D} = (V, E)$, if

$$U_i := \Big(\bigvee_{j \in \mathrm{pa}(i)} c_{ij} U_j \vee Z_i \Big) \varepsilon_i, \quad i \in 1, \ldots, d, \tag{2.17}$$

with edge weight matrix $\boldsymbol{C} := (c_{ij} \mathbf{1}_{\mathrm{pa}(i)}(j))_{d \times d}$. The noise variables $\varepsilon_1, \ldots, \varepsilon_d$ are i.i.d. and atom-free random variables with $\varepsilon_i \geq 1$ and unbounded above for all $i \in V$, and independent of the innovations vector $\boldsymbol{Z} := (Z_1, \ldots, Z_d)$. For simplicity, we denote by ε a generic noise variable and by Z a generic innovation.

Although the noise variables act on the observations, formally we can view them as random scalings of edge weights. More precisely, for a path $p = [j = k_0 \to k_1 \to \ldots \to$

$k_n = i]$ from j to i we define the *random path weight* $\bar{d}_{ij}$ similarly to the definition of d_{ij} in (2.7) as

$$\bar{d}_{ij}(p) := \varepsilon_j \prod_{l=0}^{n-1} c_{k_{l+1}k_l}\varepsilon_{k_{l+1}} = d_{ij}(p)\varepsilon_j \prod_{l=0}^{n-1} \varepsilon_{k_{l+1}}. \tag{2.18}$$

If we define the *random edge weight matrix*

$$\bar{\boldsymbol{C}} = (\bar{c}_{ij})_{d\times d} := (c_{ij}\varepsilon_i \mathbf{1}_{\mathrm{pa}(i)}(j))_{d\times d} \tag{2.19}$$

we can rewrite (2.18) as

$$\bar{d}_{ij}(p) := \varepsilon_j \prod_{l=0}^{n-1} \bar{c}_{k_{l+1}k_l}$$

for every path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ from j to i. Hence, we can view the noise variables as random scalings for the edge weights c_{ij}. Since $\varepsilon \geq 1$, the edge weights c_{ij} of the non-noisy model are lower bounds for the random edge-weights $\bar{c}_{ij}$ of the propagating noise model.

Again denoting the set of all paths from j to i by P_{ij}, we define the *random ML coefficient matrix* $\bar{\boldsymbol{B}} = (\bar{b}_{ij})_{d\times d}$ of $\boldsymbol{U}$ with entries

$$\bar{b}_{ij} := \bigvee_{p\in P_{ij}} \bar{d}_{ij}(p) \quad \text{for } j \in \mathrm{an}(i), \quad \bar{b}_{ii} = \varepsilon_i, \quad \text{and} \quad \bar{b}_{ij} = 0 \quad \text{for } j \in V \setminus \mathrm{An}(i). \tag{2.20}$$

We next show that there exists a solution of (2.17) in terms of the ancestral innovations $\boldsymbol{Z}$ and $\bar{\boldsymbol{B}}$. All proofs of this section are postponed to Section 2.7.1.

Theorem 2.3.2. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as in* (2.17)*. Define $(\boldsymbol{E}_d)_{d\times d}$ as the diagonal matrix given by*

$$E_d(i,i) = \varepsilon_i \quad \textit{for } i \in V \quad \textit{and} \quad E_d(i,j) = 0 \quad \textit{for } i,j \in V \textit{ and } i \neq j.$$

We rewrite (2.17) *in matrix form by means of the matrix multiplication* (2.9) *as*

$$\boldsymbol{U} = \boldsymbol{E}_d \odot \big(\boldsymbol{C} \odot \boldsymbol{U} \vee \boldsymbol{Z}\big).$$

Then $\boldsymbol{U}$ has a unique solution in terms of the tropical matrix multiplication with random matrix $\bar{\boldsymbol{B}}$ given by

$$\bar{\boldsymbol{B}} = (\boldsymbol{I}_d \vee \bar{\boldsymbol{C}})^{\odot(d-1)} \odot \boldsymbol{E}_d, \quad \boldsymbol{U} = \bar{\boldsymbol{B}} \odot \boldsymbol{Z}, \tag{2.21}$$

with $\bar{\boldsymbol{C}}$ as defined in (2.19)*.*

Since $\bar{b}_{ij} = 0$ whenever $j \notin \mathrm{An}(i)$, the representation (2.21) can be rewritten as follows.

Corollary 2.3.3. *Let $\boldsymbol{U}$ be as in Theorem 2.3.2 and $\bar{b}_{ij}$ be the random ML coefficients defined in* (2.20)*. Then* (2.21) *is equivalent to*

$$U_i = \bigvee_{j\in\mathrm{An}(i)} \bar{b}_{ij} Z_j, \quad i \in 1,\ldots,d. \tag{2.22}$$

Note that the definition in (2.17) is equivalent to

$$U_i = \tilde{U}_i \, \varepsilon_i \quad \text{with} \quad \tilde{U}_i := \bigvee_{j \in \mathrm{pa}(i)} c_{ij} U_j \vee Z_i, \quad i \in 1, \ldots, d. \tag{2.23}$$

Since $\boldsymbol{B}$ is idempotent, the solution of $\boldsymbol{X}$ as in (2.11) can also be written as $\boldsymbol{X} = \boldsymbol{B} \odot \boldsymbol{X}$. This is no longer the case for the solution $\boldsymbol{U}$ in (2.21). However, from the above result we can compute the following representation, which is used in Definition 2.3.5(e) below.

Corollary 2.3.4. *Let $\boldsymbol{U}$ and $\bar{b}_{ij}$ be as in Corollary 2.3.3. Then* (2.22) *is equivalent to*

$$U_i = \bigvee_{j \in \mathrm{An}(i)} \bar{b}_{ij} \tilde{U}_j, \quad i = 1, \ldots, d, \tag{2.24}$$

with $\tilde{U}_j$ as in (2.23).

2.3.2 Paths classification and graph reduction in the noisy model

We define critical and generic paths which play an essential role for the distributional properties of the model. Similarly as shown for the non-noisy model in Section 5 of Gissibl and Klüppelberg [2018], the vector $\boldsymbol{U}$ may also be a recursive ML model on a subgraph of $\mathcal{D}$. This subgraph depends on the ML coefficients, which are now random. Hence, we start by comparing the ML coefficient matrices $\boldsymbol{B}$ and $\bar{\boldsymbol{B}}$ of the non-noisy and noisy models.

Definition 2.3.5. Let $\mathcal{D}$ be a DAG with edge weight matrix $\boldsymbol{C}$ and let $\boldsymbol{B}$ be the corresponding ML coefficient matrix (i.e., the Kleene star of $\boldsymbol{C}$). Let p be a path from j to i with node set S_p.

(a) p is called a *(non-random) critical path* if $d_{ij}(p) = b_{ij}$.

(b) p is called a *generic path* if it is the only path satisfying $d_{ij}(p) = b_{ij}$.

(c) We call $\boldsymbol{C}$ *generic*, if two nodes are connected by at most one critical path.

(d) For a fixed $\omega \in \Omega$, we call p a *random critical path* if $\bar{d}_{ij}(p) = \bar{b}_{ij}$.

(e) p is called a *possible critical path realization*, if $U_i = U_j d_{ij}(p) \prod_{k \in S_p} \varepsilon_k = \tilde{U}_j \bar{d}_{ij}(p)$ happens with positive probability.

Remark 2.3.6. *We have defined a non-random critical path and a random critical path. We want to emphasize, however, that while the first path property is simply inherited from $\boldsymbol{C}$ via $\boldsymbol{B}$, the second one is inherited from $\boldsymbol{C}$ and the noise variables. We also note that by continuity of the innovations and the noise variables, any random critical path between a pair of nodes must be a.s. unique, although it may vary with the realizations of the noise variables.*

We explain the model and the notions of Definition 2.3.5 in an example.

Example 2.3.7. *Consider the DAG:*

$$\begin{tikzcd} 1 \arrow[r] \arrow[rr, bend left] & 2 \arrow[r] & 3 \end{tikzcd}$$

Then, $\boldsymbol{C}$ is generic if and only if $c_{31} \neq c_{21}c_{32}$. Moreover, we have

$$U_3 = (\bar{c}_{31} \vee \bar{c}_{21}\bar{c}_{32})\varepsilon_1 Z_1 \vee \bar{c}_{32}\varepsilon_2 Z_2 \vee \varepsilon_3 Z_3,$$

with $\bar{c}_{ij} = c_{ij}\varepsilon_i$ as defined in (2.19)*.*

Now assume that $c_{31} > c_{21}c_{32}$. In that case, $[1 \to 3]$ is the critical path, while the path $[1 \to 2 \to 3]$ is not critical. However, $\mathbb{P}(\bar{c}_{31} < \bar{c}_{21}\bar{c}_{32}) = \mathbb{P}(\varepsilon_2 > c_{31}/(c_{21}c_{32})) > 0$. If $\mathbb{P}(\bar{c}_{31} > \bar{c}_{21}\bar{c}_{32})$, then the edge $1 \to 3$ is random critical, otherwise $1 \to 2 \to 3$ is random critical. Since both paths can be random critical with positive probability, all paths in $\mathcal{D}$ can be possible critical path realizations.

In contrast, if $c_{31} < c_{21}c_{32}$ we have $\mathbb{P}(\bar{c}_{31} > \bar{c}_{21}\bar{c}_{32}) = \mathbb{P}(\varepsilon_2 < c_{31}/(c_{21}c_{32})) = 0$. In this case, the path $[1 \to 3]$ can be random critical only on a null set and therefore $[1 \to 3]$ is not a possible critical path realization.

This illustrates that a path p from j to i with path weight $d_{ij}(p) < b_{ij}$ may as well contribute to the distribution of U_i. However, an edge $p = [j \to i]$ with $d_{ij}(p) < b_{ij}$ is still not identifiable and does not change the distribution of $\boldsymbol{U}$.

Recall from (2.11) and (2.21) that

$$\boldsymbol{X} = \boldsymbol{B} \odot \boldsymbol{Z} \quad \text{and} \quad \boldsymbol{U} = \bar{\boldsymbol{B}} \odot \boldsymbol{Z}.$$

We present some useful properties of $\boldsymbol{B}$ and $\bar{\boldsymbol{B}}$ providing a link between the noisy and non-noisy model as defined in (2.6) and (2.17), respectively. Such properties have been shown for $\boldsymbol{B}$ in Gissibl [2018], Gissibl and Klüppelberg [2018], Gissibl et al. [2021], and we investigate here which of them remain valid for $\bar{\boldsymbol{B}}$.

Lemma 2.3.8. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as defined in* (2.17) *with $\boldsymbol{B}$ and $\bar{\boldsymbol{B}}$ defined in* (2.11) *and* (2.21)*, respectively. Then the following assertions hold:*

1. $\bar{b}_{ij} = \bigvee_{k \in V} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}} \geq \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$*, where the inequality is strict, whenever the random critical path from j to i is the edge $j \to i$, or $j = i$.*
2. *There exists some path $p := [j \to \ldots \to k \to \ldots \to i]$ from j to i that passes through k such that*
$$\bar{d}_{ij}(p) = \bar{b}_{ij} \quad \text{if and only if} \quad \bar{b}_{ij} = \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}.$$
3. $\frac{U_i}{U_j} \geq \frac{\bar{b}_{ij}}{\bar{b}_{jj}} \geq b_{ij}$ *with $b_{ij} = 0$ for $j \notin \mathrm{An}(i)$*

4. $\operatorname{supp}(U_i/U_j) = \begin{cases} [b_{ij}, \infty) & \textit{for } j \in \operatorname{an}(i), \\ [0, 1/b_{ji}] & \textit{for } i \in \operatorname{an}(j), \\ \{1\} & \textit{for } i = j, \\ \mathbb{R}_+ & \textit{otherwise.} \end{cases}$

Moreover, for $j \neq i$, neither the distribution of U_i/U_j nor the distribution of U_j/U_i have any atoms.

5. If $b_{ij} = \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{an}(i)} \frac{b_{kj} b_{ik}}{b_{kk}}$, *then* $\bar{b}_{ij} = \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{an}(i)} \frac{\bar{b}_{kj} \bar{b}_{ik}}{\bar{b}_{kk}}$.

6. If $b_{ij} > \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{an}(i)} \frac{b_{kj} b_{ik}}{b_{kk}}$ *and* $\operatorname{de}(j) \cap \operatorname{an}(i) \neq \emptyset$, *then*

$$\mathbb{P}\Big(\bar{b}_{ij} > \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{an}(i)} \frac{\bar{b}_{kj} \bar{b}_{ik}}{\bar{b}_{kk}}\Big) > 0 \quad \textit{and} \quad \mathbb{P}\Big(\bar{b}_{ij} = \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{an}(i)} \frac{\bar{b}_{kj} \bar{b}_{ik}}{\bar{b}_{kk}}\Big) > 0.$$

Definition 5.1 of Gissibl and Klüppelberg [2018] presents the smallest subgraph of $\mathcal{D}$ such that $\boldsymbol{X}$ is a recursive ML model on this DAG as

$$\mathcal{D}^B = (V, E) := \left(V, \left\{ (j,i) \in E : b_{ii} > \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{pa}(i)} \frac{b_{kj} b_{ik}}{b_{kk}} \right\} \right).$$

This means that $\mathcal{D}^B$ contains an edge $j \to i$ of $\mathcal{D}$ if and only if this edge is the only critical path from $j \to i$ in $\mathcal{D}$.

Lemma 2.3.8 b) and f) motivate the following definition as the random analog of $\mathcal{D}^B$.

Definition 2.3.9. Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on the DAG $\mathcal{D} = (V, E)$ as defined in (2.17). Then we define the *minimum ML DAG* $\bar{\mathcal{D}}^B$ as

$$\bar{\mathcal{D}}^B = (V, \bar{E}) := \left(V, \left\{ (j,i) \in E : \mathbb{P}\Big(\bar{b}_{ij} > \bigvee_{k \in \operatorname{de}(j) \cap \operatorname{pa}(i)} \frac{\bar{b}_{kj} \bar{b}_{ik}}{\bar{b}_{kk}}\Big) > 0 \right\} \right).$$

This means that $\bar{\mathcal{D}}^B$ contains an edge $j \to i$ of $\mathcal{D}$ if and only if this edge is a possible critical path realization from $j \to i$ in $\mathcal{D}$.

In addition, applying first Lemma 2.3.8 e) and f), and in the second part Lemma 2.3.8 b) yields the following result.

Corollary 2.3.10. *Let $\boldsymbol{X} \in \mathbb{R}_+^d$ be a recursive ML vector on a DAG $\mathcal{D} = (V, E)$ as defined in* (2.6) *and $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise as defined in* (2.17) *on the same DAG $\mathcal{D}$ with the same edge weight matrix $\boldsymbol{C}$. Then*

$$\mathcal{D}^B = \bar{\mathcal{D}}^B,$$

which is the smallest DAG that preserves the distributions of $\boldsymbol{X}$ and of $\boldsymbol{U}$.

We will henceforth only use the term $\mathcal{D}^B$.

The next lemma summarizes properties of possible critical path realizations from Definition 2.3.5(e).

Lemma 2.3.11. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as defined in* (2.17). *Then the following assertions hold:*

1. *A path $p = [j = k_0 \to \ldots \to k_n = i]$ in $\mathcal{D}$ is a possible critical path realization from j to i if and only if all edges of p belong to the minimum ML DAG $\mathcal{D}^B$.*
2. *Let p_1 and p_2 be two possible critical path realizations from j to i and from l to m, respectively. Then*

$$\Big\{U_i = U_j d_{ij}(p_1) \prod_{k \in S_{p_1}} \varepsilon_k, U_m = U_l d_{ml}(p_2) \prod_{k \in S_{p_2}} \varepsilon_k\Big\} \tag{2.25}$$

has positive probability if and only if $S_{p_1} \cap S_{p_2} = \emptyset$, or for every $r \in S_{p_1} \cap S_{p_2}$ the sub-path of p_1 from j to r is a sub-path of p_2 or the sub-path of p_2 from l to r is a sub-path of p_1.

We illustrate part b) with Figure 2.1 and Figure 2.2.

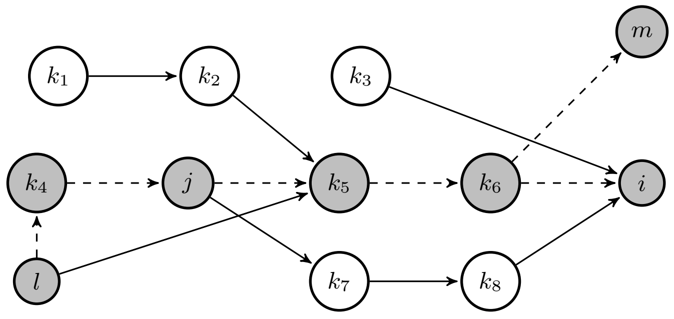


Figure 2.1: Both dashed paths $p_1 := [j \to k_5 \to k_6 \to i]$ and $p_2 := [l \to k_4 \to j \to k_5 \to k_6 \to m]$ can be possible critical path realizations from the same realized noise variables along the nodes.

2.3.3 Distributions of component ratios of the noisy model

The next result is important as it not only helps us to understand the model better, but is also an important step for learning the model.

Theorem 2.3.12. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as defined in* (2.17). *Suppose that $p_{\max} := [j = k_0 \to \cdots \to k_n = i]$ is generic. Let $S_{p_{\max}} = \{k_1, \ldots, k_n\}$ be the set of nodes on $p_{\max}$. Then*

$$\begin{aligned} &\mathbb{P}\Big(\frac{U_i}{U_j} \le b_{ij} x\Big) \\ &\sim \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \le x, \frac{U_i}{U_j} = b_{ij} \prod_{k \in S_{p_{\max}}} \varepsilon_k\Big) \sim c\,\mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \le x\Big), \quad x \downarrow 1, \end{aligned}$$

for some constant $c \in (0,1)$.

Remark 2.3.13. *If the distributions of the noise variables and the innovations as well as the path weights of the underlying DAG $\mathcal{D}$ are given, the constant c in Theorem 2.3.12 can be calculated explicitly.*

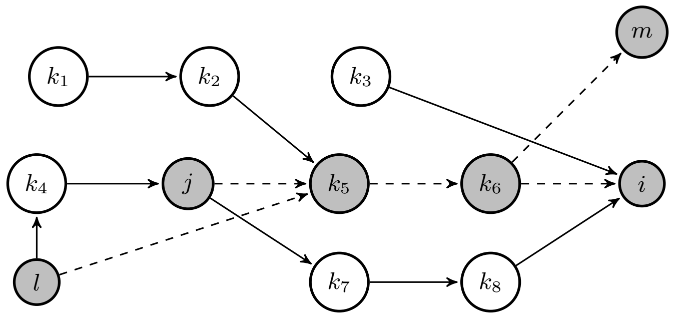


Figure 2.2: Both dashed paths $p_1 := [j \to k_5 \to k_6 \to i]$ and $p_2 := [l \to k_5 \to k_6 \to m]$ can only on a null-set be possible critical path realizations from the same realized noise variables along the nodes.

Theorem 2.3.12 also shows that, while a path p from j to i with $d_{ij}(p) < b_{ij}$ can contribute to the distribution of $\boldsymbol{U}$ (as we have seen in Example 2.3.7), they influence the distribution of U_i/U_j at their left limit of support only by the constant $c \in (0,1)$.

We now extend the result to situations with several critical paths.

Corollary 2.3.14. *Let $\boldsymbol{U}$ be as in Theorem 2.3.12. Suppose that exactly the paths $p_1, \dots, p_n$ from j to i are critical; i.e., $d_{ij}(p_1) = \ldots = d_{ij}(p_n) = b_{ij}$. Then*

$$\mathbb{P}\Big(\frac{U_i}{U_j} \le b_{ij}x\Big) \sim c\,\mathbb{P}\Big(\bigcap_{p\in\{p_1,\dots,p_n\}} \Big\{\prod_{k\in S_p} \varepsilon_k \le x\Big\}\Big), \quad x \downarrow 1,$$

for some constant $c \in (0,1)$.

For simplicity, we assume from now on that $\boldsymbol{C}$ is generic in the sense of Definition 2.3.5. However, we want to remark that all such results can be extended to the case of several non-random critical paths between two nodes. The proofs of such results work similarly as the proof of Corollary 2.3.14.

We continue with another consequence of Theorem 2.3.12.

Corollary 2.3.15. *Let $\boldsymbol{U}$ be as in Theorem 2.3.12 and suppose that $p := [j = k_0 \to \cdots \to k_n = i]$ is generic. Let $\boldsymbol{U}^1, \dots, \boldsymbol{U}^n$ for $n \in \mathbb{N}$ be an i.i.d. sample from $\boldsymbol{U}$. Then, for the same constant $c \in (0,1)$ as in Theorem 2.3.12, we have*

$$\mathbb{P}\Big(\bigwedge_{k=0}^{n} \frac{U_i^k}{U_j^k} \le b_{ij}x\Big) \sim c\,n\,\mathbb{P}\Big(\prod_{i=1}^{n} \varepsilon_{k_i} \le x\Big), \quad x \downarrow 1.$$

We conclude this section by extending Theorem 2.3.12 to multivariate distributions, which is an important structural result of the new model. We only formulate and prove the bivariate case, the general case is then obvious. Recall that in Lemma 2.3.11 we gave a necessary and sufficient condition for (2.26) below.

Theorem 2.3.16. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as defined in (2.17). Suppose generic paths p_1 from j to i and p_2 from l to m. Assume that*

$$\mathbb{P}\Big(U_i = U_j b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k, U_m = U_l b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k\Big) > 0. \tag{2.26}$$

Then

$$\mathbb{P}\left(\frac{U_i}{U_j} \leq b_{ij}x_1, \frac{U_m}{U_l} \leq b_{ml}x_2\right) \sim c\,\mathbb{P}\Big(\prod_{k\in S_{p_1}} \varepsilon_k \leq x_1, \prod_{k\in S_{p_2}} \varepsilon_k \leq x_2\Big)$$
$$\sim \mathbb{P}\Big(\prod_{k\in S_{p_1}} \varepsilon_k \leq x_1, \prod_{k\in S_{p_2}} \varepsilon_k \leq x_2, \frac{U_i}{U_j} = b_{ij}\prod_{k\in S_{p_1}} \varepsilon_k, \frac{U_m}{U_l} = b_{ml}\prod_{k\in S_{p_2}} \varepsilon_k\Big),$$

for $x_1, x_2 \downarrow 1$ and some constant $c \in (0,1)$.

2.4 Identification and estimation

We first address the question of identifiability of $\boldsymbol{B}$ from the distribution of $\boldsymbol{U}$. In particular, we are going to show that even though innovations and noise variables are generally not identifiable, $\boldsymbol{B}$ remains identifiable also in the propagating noise model.

We discuss three settings (1)-(3) below. For each setting, we propose an appropriate minimum ratio estimator for $\boldsymbol{B}$. Afterwards, we will show the almost sure convergence of each of the estimators.

2.4.1 Identifiability of the model

In this section we discuss the question of identifiability of the DAG $\mathcal{D}$ and the edge weights $\boldsymbol{C}$ of a ML model with recursive noise from the distribution of $\boldsymbol{U}$. As we have already seen in Example 2.3.7, the true DAG $\mathcal{D}$ and the edge weight matrix $\boldsymbol{C}$ underlying $\boldsymbol{U}$ in representation (2.17) are generally not identifiable from the distribution of $\boldsymbol{U}$. The smallest DAG with a chance to be identified from the distribution of $\boldsymbol{U}$ is the minimum ML DAG $\bar{\mathcal{D}}^B$ of Definition 2.3.9, which in turn can be identified from $\boldsymbol{B}$ by Corollary 2.3.10.

By the equivalence of $\mathcal{D}^B$ and $\bar{\mathcal{D}}^B$, Theorem 2 in Gissibl et al. [2021] also holds for the propagating noise model defined in (2.17), i.e., the theorem defines the class of DAGs that preserve the distribution of $\boldsymbol{U}$. As by Lemma 2.3.8 d) the ML coefficients are limits of supports of component ratios of $\boldsymbol{U}$, the following is immediate.

Corollary 2.4.1. *Let $\boldsymbol{U} \in \mathbb{R}^d_+$ be a recursive ML model with propagating noise on a DAG $\mathcal{D}$ as defined in* (2.17). *Then the ML coefficient matrix $\boldsymbol{B}$ is identifiable from the distribution of $\boldsymbol{U}$.*

Since we can identify $\boldsymbol{B}$ from the distribution of $\boldsymbol{U}$, we can also identify the minimum ML DAG $\mathcal{D}^B$ from Definition 2.2.1 (which by Definition 2.3.9 and Corollary 2.3.10 is the minimum DAG preserving the distribution of $\boldsymbol{U}$). Therefore, since $\varepsilon \geq 1$, Theorem 2 of Gissibl et al. [2021] also holds for the propagating noise model as defined in (2.17). Therefore, as exemplified in Example 2.3.7, we can identify the class of all DAGs and edge weights that could have generated $\boldsymbol{U}$.

However, unlike for the non-noisy model, we can generally not identify innovations or noise variables. To see this assume a source node U_i in a DAG $\mathcal{D}$ such that $\mathrm{an}(i) = \emptyset$. If $\boldsymbol{U}$ follows a recursive ML model with propagating noise, then $U_i := Z_i\varepsilon_i$. In particular, we can not identify Z_i or ε_i.

When estimating a recursive ML model with propagating noise, we distinguish between three settings. We point out that all algorithms below work equally, if the data is generated from a noise-free max-linear model as defined in (2.6).

(1) All ancestral relations are known; i.e., we know the set of edges E, hence the DAG. This might be the case when modeling networks that contain natural information about edges. The problem then reduces to finding appropriate estimates $\hat{b}_{ij}$ for $j \in \mathrm{an}(i)$.

(2) The ancestral relations are unknown; however, we know a topological order of the nodes. Then, in contrast to setting 1, we need to decide if a path from j to i with $j < i$ exists.

(3) Neither the underlying DAG nor a topological order of the nodes is known. Then we need to find a topological order of the nodes and proceed then as in setting 2.

We next want to estimate $\boldsymbol{B}$ for each of the three settings (1)-(3).

2.4.2 Known DAG structure with unknown edge weights

Given an i.i.d. sample $\boldsymbol{U}^1, \ldots, \boldsymbol{U}^n$ from a recursive ML model with propagating noise on a known DAG $\mathcal{D}$ as defined in (2.17) and knowing all ancestral relations of $\mathcal{D}$, we could choose the simple estimate

$$\check{\boldsymbol{B}} := (\check{b}_{ij})_{d\times d} = \Big(\bigwedge_{k=1}^{n} \frac{U_i^k}{U_j^k} \mathbf{1}_{\mathrm{An}(i)}(j) \Big)_{d\times d}. \tag{2.27}$$

However, as in the non-noisy model, the estimate (2.27) may not define any recursive ML model on the given DAG $\mathcal{D}$, cf. Example 3 of Gissibl et al. [2021].

We use instead

$$\boldsymbol{B}_0 = (B_0(i,j))_{d\times d} := \Big(\bigwedge_{k=1}^{n} \frac{U_i^k}{U_j^k} \mathbf{1}_{\mathrm{pa}(i)}(j) \Big)_{d\times d} \quad \text{and set} \quad \hat{\boldsymbol{B}} = (\boldsymbol{I}_d \vee \boldsymbol{B}_0)^{\odot(d-1)}. \tag{2.28}$$

Applying Lemma 2 in Gissibl et al. [2021] to $\boldsymbol{B}_0$, the estimator $\hat{\boldsymbol{B}}$ yields a valid estimate of the given DAG in the sense that $\hat{\boldsymbol{B}}$ defines a recursive ML model and for any pair $(j,i) \notin E(\mathcal{D})$ we have $\hat{b}_{ij} = \bigvee_{k\in\{1,\ldots,d\}\setminus\{j,i\}} \hat{b}_{kj}\hat{b}_{ik}$. Moreover, by the idempotency of $\hat{\boldsymbol{B}}$ and Lemma 2.3.8 c), similarly to the non-noisy model, it also holds that

$$b_{ij} \le \hat{b}_{ij} \le \check{b}_{ij}, \quad j \in \mathrm{an}(i). \tag{2.29}$$

2.4.3 Known topological order

Given an i.i.d. sample $\boldsymbol{U}^1, \ldots, \boldsymbol{U}^n \in \mathbb{R}_+^d$ from a recursive ML model with propagating noise without knowing $\mathcal{D}$, but knowing the topological order of nodes, we adapt the estimator (2.27) to this situation and define

$$\hat{\boldsymbol{B}} := (\hat{b}_{ij})_{d\times d} = \Big(\bigwedge_{k=1}^{n} \frac{U_i^k}{U_j^k}\mathbf{1}_{(j<i)}\Big)_{d\times d}. \tag{2.30}$$

2.4.4 Unknown DAG and unknown topological order

Given an i.i.d. sample $\boldsymbol{U}^1, \dots, \boldsymbol{U}^n \in \mathbb{R}_+^d$ from a recursive ML model with propagating noise without knowing $\mathcal{D}$ or the topological order, we will recover a topological order first and then proceed as in Section 2.4.3.

Estimating the topological order of an underlying DAG is often done by learning algorithms that successively identify source nodes and succeeding generations. For additive models, usually regression techniques are applied (see e.g. Chen et al. [2019] or Peters et al. [2014]). In the recursive ML model, the noise is not additive and the model is highly non-linear. Hence, such regression methods cannot be applied. However, under the condition of multivariate regular variation, the paper Klüppelberg and Krali [2021] suggests a learning algorithm for the model without noise as given in (2.1). We propose a different approach, which to the best of our knowledge has not been considered in the literature before. It applies to the propagating noise model without any distributional assumptions on the innovations and noise variables and learns the DAG by using minimum ratios. We first consider the matrix of all minimum ratios given by

$$\check{\boldsymbol{B}} := (\check{b}_{ij})_{d\times d} = \Big(\bigwedge_{k=1}^{n} \frac{U_i^k}{U_j^k}\Big)_{d\times d}. \tag{2.31}$$

Let Π denote the set of all topological orders of V. Furthermore, denote an equivalence class of topological orders induced by the underlying (unknown) DAG $\mathcal{D} = (V, E)$ by

$$R_{\mathcal{D}} := \{\pi \in \Pi : \pi(j) < \pi(i) \quad \text{for all } (j,i) \in E\}. \tag{2.32}$$

By Lemma 2.3.8 d), $\check{b}_{ij}$ is lower bounded by b_{ij} for $j \in \operatorname{an}(i)$ and $\check{b}_{ij} \to 0$ a.s. as $n \to \infty$ for $j \notin \operatorname{An}(i)$. This is a direct result from Lemma 2.3.8 c) and the fact that the minimum is non-increasing. Hence, for any $\pi \in R_{\mathcal{D}}$ it holds that $\check{b}_{ij} \to 0$ a.s. as $n \to \infty$ whenever $\pi(j) > \pi(i)$. Therefore, also

$$\max_{\substack{(j,i)\in V\times V:\\ \pi(j)>\pi(i)}} \check{b}_{ij} \to 0 \quad \text{a.s. for } n \to \infty. \tag{2.33}$$

In contrast, for any $\pi \notin R_D$, there is a pair of nodes (j, i) such that $b_{ij} > 0$ although $\pi(j) > \pi(i)$. For this reason,

$$\max_{\substack{(j,i)\in V\times V:\\ \pi(j)>\pi(i)}} \check{b}_{ij} \to c_\pi > 0 \quad \text{a.s. for } n \to \infty. \tag{2.34}$$

As a consequence, for a given topological order π, by (2.33) and (2.34), the maximum converges almost surely to zero if and only if $\pi \in R_{\mathcal{D}}$. Hence we propose a topological order that minimizes this expression, i.e.,

$$\arg\min_{\pi\in\Pi} \max_{\substack{(j,i)\in V\times V:\\ \pi(j)>\pi(i)}} \check{b}_{ij}. \tag{2.35}$$

A topological order found by (2.35) generally is not unique. Algorithm 1 returns a unique topological order for any fixed estimated matrix $\check{\boldsymbol{B}}$.

Algorithm 1 Estimating a topological order

```
Input: A matrix of minimum ratios $\check{\boldsymbol{B}}$ as in (2.31)
Output: An estimated topological order $\hat{\pi}$
1: Set $\check{\mathcal{D}} = (V, E)$ with $V = \{1, \dots, d\}$ and $E = \emptyset$.
2: Set $S := \{(j,i) \in V \times V : j \neq i\}$ and sort the elements $(j,i)$ of $S$ by the size of
   $\check{b}_{ij}$ from large to small.
3: for $(j,i)$ in S do
4:     if $i \notin \mathrm{an}(j)$ in $\check{\mathcal{D}}$ then
5:         $E = E \cup (j,i)$
6:     end if
7: end for
8: return the topological order $\hat{\pi}$ of the DAG $\check{\mathcal{D}}$
```

Proposition 2.4.2. *Algorithm 1 solves the optimization problem in equation* (2.35).

Proof. Let π be the topological order from Algorithm 1 and denote by $S(\pi)$ the objective function in (2.35). Algorithm 1 sorts all pairs (j,i) by the size of $\check{b}_{ij}$ and draws an edge from j to i whenever there is no path from i to j. Now consider the first pair (j,i) in the algorithm where we do not draw an edge since $i \in \mathrm{an}(j)$. Consider a permutation π' that is obtained by exchanging the order of nodes i and j in π. Since there exists already a path from i to j, it follows that $S(\pi) \leq S(\pi')$. □

The DAG $\check{\mathcal{D}}$ constructed in Algorithm 1 works as an auxiliary instrument to infer a topological order. Observe that $\check{\mathcal{D}}$ is a complete DAG, with directed edges between every node pair in V and, hence, there is a unique topological order representing $\check{\mathcal{D}}$. Moreover, since we sort the weights by size, the algorithm solves (2.35) in an optimal way for given $\check{B}$. At first sight the algorithm bears some similarity to Kruskal's classical algorithm for finding a minimum spanning tree; see Kruskal [1956]. However, Algorithm 1 works with directed edges and, of course, the optimization problem itself is very different.

Adding an edge and checking the presence of a path between any pair of nodes both can be implemented in $O(d)$ amortized complexity (see Italiano [1986]). Hence, since S as computed in line 2 of Algorithm 1 contains $d(d-1)$ pairs of nodes, we have an overall amortized complexity of $O(d^3)$. After Algorithm 1, we can again use the minimum ratio estimator

$$\hat{\boldsymbol{B}} := (\hat{b}_{ij})_{d\times d} = \Big(\bigwedge_{k=1}^{n} \frac{U_i^k}{U_j^k} \mathbf{1}_{(\hat{\pi}(j)<\hat{\pi}(i))} \Big)_{d\times d}. \tag{2.36}$$

2.4.5 Strong consistence of $\hat{B}$ and learning the minimum ML DAG $\mathcal{D}^B$

We first want to formally state the a.s. convergence of the proposed estimators for the ML coefficient matrix $\boldsymbol{B}$. Afterwards, we discuss how to learn the minimum ML DAG $\mathcal{D}^B$. The proofs of Proposition 2.4.3 and Lemma 2.4.4 can be found in Section 2.7.2.

Proposition 2.4.3. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise as defined in* (2.17) *and let $\boldsymbol{U}^1, \dots, \boldsymbol{U}^n \in \mathbb{R}_+^d$ be an i.i.d. sample from $\boldsymbol{U}$. Then the estimates* (2.28), (2.30) *and* (2.36) *of $\boldsymbol{B}$ are strongly consistent, i.e., it holds a.s. for $n \to \infty$ that*

$$\hat{b}_{ij} \longrightarrow b_{ij} \quad \text{for } j \in \operatorname{an}(i), \quad \hat{b}_{ii} = 1, \quad \text{and} \quad \hat{b}_{ij} \longrightarrow 0 \quad \text{for } j \in V \setminus \operatorname{An}(i).$$

In Sections 2.4.2-2.4.4 we have been discussing how to estimate $\boldsymbol{B}$ under the settings (1)-(3). However, as we know from Corollary 2.3.10, only critical edges of $\mathcal{D}$ contribute to the distribution of $\boldsymbol{U}$. Asymptotically, we can almost surely identify $\mathcal{D}^B$ since there is an edge $j \to i$ in $\mathcal{D}^B$ if and only if $b_{ij} > b_{kj} b_{ik}$ for all $k \in \operatorname{de}(j) \cap \operatorname{an}(i)$.

However, in real life we estimate the edges of $\mathcal{D}^B$ for a finite data set. Since $\bigwedge_{k=1}^n (U_i^k / U_j^k) > 0$ holds for all $n \in \mathbb{N}$ and all $i, j \in V$, the estimators (2.30) or (2.36) would result in a matrix representing a complete DAG.

Since small estimated values $\hat{b}_{ij}$ may well be 0 in the true model, we use a threshold $\delta_1 > 0$ with the aim to set an estimator $\hat{b}_{ij} < \delta_1$ equal to 0. However, setting single values $\hat{b}_{ij} := 0$ may destroy the idempotency of $\hat{\boldsymbol{B}}$ since idempotency requires for any triple of nodes (j, k, i),

$$\hat{b}_{kj} \hat{b}_{ik} = \bigwedge_{t=1}^n \frac{U_k^t}{U_j^t} \bigwedge_{t=1}^n \frac{U_i^t}{U_k^t} \le \bigwedge_{t=1}^n \frac{U_k^t}{U_j^t} \frac{U_i^t}{U_k^t} = \hat{b}_{ij}. \tag{2.37}$$

For the estimates however, it might be possible that $\hat{b}_{ij} < \delta_1$, while $\hat{b}_{lj} > \delta_1$ and $\hat{b}_{il} > \delta_1$. In this case, setting $\hat{b}_{ij} = 0$ would result in $\hat{b}_{ij} < \hat{b}_{lj} \hat{b}_{il}$ violating (2.37). To preserve the idempotency of $\hat{\boldsymbol{B}}$ while setting some small values to 0, we propose a simple adapted thresholding algorithm.

Lemma 2.4.4. *Algorithm 2 with threshold $\delta_1 > 0$ outputs an idempotent matrix, i.e., $\hat{\boldsymbol{B}} \odot \hat{\boldsymbol{B}} = \hat{\boldsymbol{B}}$ and there is no other idempotent matrix $\boldsymbol{B}'$ such that $b'_{ij} = \hat{b}_{ij}$ whenever $\hat{b}_{ij} > \delta_1$ that contains more zero entries than $\hat{\boldsymbol{B}}$.*

Remark 2.4.5. *If we choose $\delta_1 \le \min\{\check{b}_{ij} : j < i\}$ no entry is set to 0, and if $\delta_1 > \max\{\check{b}_{ij} : j < i\}$ all entries are set to 0 except for the diagonal. So in the first case, we obtain the complete DAG and in the second case the DAG consists of isolated nodes only.*

In order to estimate the minimum ML DAG $\mathcal{D}^B$ it is not sufficient to decide if a path from j to i exists, i.e., if $b_{ij} > 0$. We need in particular to decide if the edge $j \to i$ belongs to $\mathcal{D}^B$. By continuity of the noise variables we may observe for the estimated path weights

$$\hat{b}_{ij} > \hat{b}_{lj} \hat{b}_{il}$$

Algorithm 2 Thresholding while maintaining idempotency

Input: A (known or estimated) topological order $\pi : 1, \dots, d$ and an idempotent estimate $\hat{\boldsymbol{B}}$ as in (2.30) or (2.36) and a threshold value $\delta_1 > 0$ **Output**: An idempotent estimate $\hat{\boldsymbol{B}}$

```
1: $E := \{(j,i) \in V \times V : \operatorname{sgn}(\hat{b}_{ij}) = 1 \text{ and } i \neq j\}$
2: $\mathcal{D} := (V, E)$
3: $S := \{(j,i) \in E : 0 < \hat{b}_{ij} < \delta_1\}$
4: Sort the pairs $(j,i)$ in $S$ by the distance $i - j$ from low to high
5: for $(j,i)$ in S do
6:     if $(j - i) == 1$ then
7:         $\hat{b}_{ij} = 0$
8:     end if
9:     if for every $l$ with $j < l < i$: $(j,l)$ or $(l,i) \in S$ then
10:        $\hat{b}_{ij} = 0$
11:    else
12:        $S = S \setminus \{(j,i)\}$
13:    end if
14: end for
15: return $\hat{\boldsymbol{B}}$
```

even if $b_{ij} = b_{lj} b_{il}$. However, by Proposition 2.4.3, in this situation the difference $(\hat{b}_{ij} - \hat{b}_{lj}\hat{b}_{il}) \to 0$ a.s. as $n \to \infty$. Therefore, we introduce another threshold $\delta_2 > 0$ enforcing an edge in $\mathcal{D}^B$ if this difference is greater than δ_2. In Theorem 2.3.12 we have seen that the distribution of the ratio $\mathbb{P}(U_i/U_j \le b_{ij}x)$ is asymptotically determined by $\mathbb{P}(\prod_{k \in S_p} \varepsilon_k - 1 \le x)$ for $x \downarrow 0$. Hence, the rate of convergence of $(\hat{b}_{ij} - \hat{b}_{lj}\hat{b}_{il})$ depends crucially on the path length $m = |S_p|$. Ideally, we therefore choose $\delta_2 = \delta_2(n,m)$ depending not only on the sample size n, but also on the path length m.

More precisely, since $F^{\leftarrow}_{\sum_{k \in S_p} \tilde{\varepsilon}_k}(1/n) \sim F^{\leftarrow}_{\prod_{k \in S_p} \varepsilon_k - 1}(1/n)$ (see Theorem 2.5.2 and its proof below), and assuming that $\boldsymbol{C}$ is generic, we find that Algorithm 3 asymptotically identifies $\mathcal{D}^B$, if

$$F^{\leftarrow}_{\sum\limits_{k \in S_p} \tilde{\varepsilon}_k}(1/n) = o(\delta_2(n,m)) \quad \text{for} \quad n \to \infty.$$

In real life we do not know the number of critical edges in either of the three settings. We distinguish between setting (1) and settings (2)-(3) and propose Algorithm 3 with $\delta_2(m) := \delta_2(n,m)$, i.e., for a fixed sample size n we focus on the path length m. For setting (1) we do know the underlying unweighted DAG $\mathcal{D}$. Therefore, we do not need to decide whether some small value $\hat{b}_{ij}$ corresponds to a path from j to i. However, we do not know the minimum ML DAG $\mathcal{D}^B$ such that we would apply Algorithm 3 to estimate $\mathcal{D}^B$. For settings (2) and (3) we would apply first Algorithm 2 and afterwards Algorithm 3.

To further illustrate this, observe the diagram below. In setting 3, we start with $\check{\boldsymbol{B}}$, while in setting 2 with $\hat{\boldsymbol{B}}$ and for setting 1, we start with $\tilde{\boldsymbol{B}}$.

$$\boxed{\check{\boldsymbol{B}} \xrightarrow{Alg.\ 1} \hat{\pi} \xrightarrow{(2.36)} \hat{\boldsymbol{B}};\ (\hat{\pi},\ \hat{\boldsymbol{B}}) \xrightarrow{Alg.\ 2} \tilde{\boldsymbol{B}};\ (\hat{\pi},\ \tilde{\boldsymbol{B}}) \xrightarrow{Alg.\ 3} D^{\tilde{B}}}$$

Algorithm 3 Approximating max-weighted paths

Input: Threshold sequences $\delta_2(1), \dots, \delta_2(d)$ and settings
(1): a known underlying DAG $\mathcal{D} := (V, E)$ and an estimate $\hat{\boldsymbol{B}}$ as in (2.28), or
(2-3): a (known or estimated) topological order $\pi : 1, \dots, d$ and a thresholded matrix $\hat{\boldsymbol{B}}$ obtained from Algorithm 2.

Output: An estimated minimum DAG $\mathcal{D}^{\hat{B}} = (V, E^{\hat{B}})$

```
$E^{\hat{B}} := \emptyset$ and $\mathcal{D}^{\hat{B}} := (V, E^{\hat{B}})$
(1):     $S := \{(j,i) \in V \times V : j \in \text{pa}(i)\}$ and infer a topological order $\pi : 1, \dots, d$ from $\mathcal{D}$
(2)-(3): $S := \{(j,i) \in V \times V : j < i\}$
Sort pairs $(j,i)$ in $S$ by their distance $(i - j)$ according to the topological order from low to high
**for** $(j,i)$ in S **do**
    **if** $\exists$ path $p$ from $j$ to $i$ in $\mathcal{D}^{\hat{B}}$ **then**
        Set $m$ as the maximum path length in $\mathcal{D}^{\hat{B}}$
        Set $l := \text{argmax}_{l \in V \setminus \{j,i\}} \left(\check{b}_{lj} \check{b}_{il}\right)$
        **if** $(\check{b}_{ij} - \check{b}_{lj} \check{b}_{il}) > \delta_2(m)$ **then**
            $E^{\hat{B}} := E^{\hat{B}} \cup \{(j,i)\}$
        **end if**
    **else**
        **if** $\check{b}_{ij} > 0$ **then**
            $E^{\hat{B}} := E^{\hat{B}} \cup \{(j,i)\}$
        **end if**
    **end if**
**end for**
**return** $\mathcal{D}^{\hat{B}} = (V, E^{\hat{B}})$
```

In the next section we derive the asymptotic distribution of the estimators.

2.5 Asymptotic distribution of the minimum ratio estimators

With the goal of proving asymptotic distributional properties of the minimum ratio estimators for the different settings (1)-(3), we require regular variation of the noise variable ε in its left endpoint. Under this condition we first prove that also the minimum ratio estimators $\bigwedge_{k=1}^{n}(U_i^k/U_j^k)$ are regularly varying. Moreover, we show that their joint limit distribution is the product of Weibull distributions. In this section we assume $\boldsymbol{C}$ is generic in the sense of Definition 2.3.5. The results can be extended to a non-generic model by similar methods as used in Corollary 2.3.14.

In what follows we assume that the random variables $\tilde{\varepsilon}_i := \ln(\varepsilon_i) > 0$ for $i = 1, \dots, d$ are i.i.d. regularly varying at zero with exponent $\alpha > 0$ and recall from Corollary 2.2.6(b) that this is equivalent to $(\varepsilon - 1) \in RV_\alpha^0$ or $\varepsilon \in RV_\alpha^1$.

We first prove that $\ln(U_i/U_j) - \ln(b_{ij})$ is regularly varying at zero which will be a consequence of Theorem 2.3.12. In this auxiliary result as well as in the theorems below we need that $\boldsymbol{C}$ is generic. Further, for a path p we denote by $\zeta(p) = |S_p|$ its path length.

Lemma 2.5.1. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as defined in* (2.17) *and assume that the path $p := [j \to \ldots \to i]$ from j to i is generic. If $\ln(\varepsilon) \in RV_\alpha^0$, then $\ln(U_i/U_j) - \ln(b_{ij}) \in RV_{\zeta(p)\alpha}^0$.*

The following is the main result of this section and describes the asymptotic distribution of the minimum ratio estimator $\hat{\boldsymbol{B}}$ from (2.36). In particular, it shows that its entries are asymptotically independent.

Theorem 2.5.2. *Let $\boldsymbol{U} \in \mathbb{R}_+^d$ be a recursive ML vector with propagating noise as defined in* (2.17)*. Assume that $\boldsymbol{C}$ is generic and that $\tilde{\varepsilon} = \ln(\varepsilon) \in RV_\alpha^0$. For every path p_{ij} from j to i and node set $S_{p_{ij}}$ choose $a_n^{(ij)} \sim F^{\leftarrow}_{\sum_{k \in S_{p_{ij}}} \tilde{\varepsilon}_k}(1/n)$ as $n \to \infty$. If $\boldsymbol{U}^1, \ldots, \boldsymbol{U}^n$ is an i.i.d. sample from $\boldsymbol{U}$, then*

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{1}{a_n^{(ij)} b_{ij}} \Big(\bigwedge_{k=1}^n \frac{U_i^k}{U_j^k} - b_{ij} \Big) \le x_{ij} \, \forall (j,i) \in V \times V \text{ with } b_{ij} > 0 \right)$$
$$= \prod_{\substack{(j,i) \in V \times V: \\ b_{ij} > 0}} \Psi_{\zeta(p_{ij})\alpha, (c^{(ij)})^{1/(\zeta(p_{ij})\alpha)}}(x_{ij}), \quad x_{ij} > 0,$$

where $c^{(ij)} \in (0,1)$ is defined as in Theorem 2.3.12.

If we know the minimum ML DAG $\mathcal{D}^B = (V, E(\mathcal{D}^B))$, it is preferable to estimate b_{ij} as in (2.28). Then Theorem 2.5.2 reduces as follows.

Corollary 2.5.3. *Let the assumptions of Theorem 2.5.2 hold and assume that the minimum ML DAG $\mathcal{D}^B(V, E(\mathcal{D}^B))$ is known. Then*

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{1}{a_n^{ij} b_{ij}} \Big(\bigwedge_{k=1}^n \frac{U_i^k}{U_j^k} - b_{ij} \Big) \le x_{ij} \, \forall (j,i) \in E(\mathcal{D}^B) \right)$$
$$= \prod_{(j,i) \in E(\mathcal{D}^B)} \Psi_{\alpha, (c^{(ij)})^{1/\alpha}}(x_{ij}), \quad x_{ij} > 0.$$

2.6 Data analysis and simulation study

We want to apply the methods that we have developed over the past sections and consider a data example. For a quality assessment we also perform a simulation study.

2.6.1 Data example

We consider dietary supplement data of $n = 8327$ independent interviews taken from the NHANES report for the year 2015-2016, which is available at https://wwwn.cdc.gov/Nchs/Nhanes/2015-2016/DR1TOT_I.XPT. They have been part of a questionnaire as to “What We Eat in America”, which recorded the food and beverage consumed by all participants during the 24 hours period prior to the interview. The data contains 168 food components with the object of estimating the total intake of calories, macro and micro nutrients from foods and beverages. More details can be found on the website.

In Janßen and Wan [2020], the data set has been considered in terms of an adapted k-means clustering algorithm for extremal observations. Moreover, assuming a recursive ML model and standardizing the marginal data to regular variation at ∞ with $\alpha = 2$, Klüppelberg and Krali [2021] investigated the causal relationship between four nutrients using a different estimation method based on scalings.

In our data example we consider the same four nutrients, namely vitamin A (DR1TVARA), α-carotene (DR1TACAR), β-carotene (DR1TBCAR) and lutein+zeaxanthin (DR1TLZ) as in Klüppelberg and Krali [2021]. We abbreviate them by VA, AC, BC and LZ. In order to make results comparable to those of Klüppelberg and Krali [2021], we also use the empirical integral transform to standardize the data to Fréchet(2) margins (see e.g. Beirlant et al. [2004], p. 381) by setting for $i = 1, 2, 3, 4$,

$$U_{li} := \Big(-\log\Big(\frac{1}{n+1}\sum_{j=1}^{n}\mathbf{1}_{\{\bar{U}_{ji}\le \bar{U}_{li}\}}\Big)\Big)^{-1/2}, \quad l = 1, \ldots, n = 8327,$$

where multiple ranks are uniformly randomly ordered.

We first consider the full matrix of minimum ratios $\check{\boldsymbol{B}} = (\check{b}_{ij})_{d\times d}$ with $\check{b}_{ij} = \bigwedge_{t=1}^{n}(X_i^t/X_j^t)$ given by

$$\begin{array}{cc} \begin{array}{cccc} VA & AC & BC & LZ \end{array} & \\ \begin{pmatrix} 1 & 0.146 & 0.321 & 0.132 \\ 0.014 & 1 & 0.010 & 0.007 \\ 0.011 & 0.177 & 1 & 0.168 \\ 0.007 & 0.019 & 0.025 & 1 \end{pmatrix} & \begin{array}{c} VA \\ AC \\ BC \\ LZ \end{array} \end{array}$$

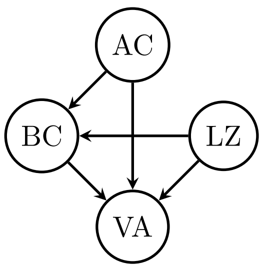
We next apply Algorithm 1 to obtain an estimated topological order $\hat{\pi} := (AC, LZ, BC, VA)$. First we want to assess the quality of the estimated topological order $\hat{\pi}$, which also supports or contradicts the model assumption of a Bayesian network. Motivated by the coefficient R^2 of determination in regression we define the following.

Definition 2.6.1. For a given topological order π and an estimator $\check{B}$ of the ML coefficient matrix we define the *ML coefficient of determination*

$$R_{\max}(\pi) = \frac{\sum\limits_{\substack{(j,i)\in V\times V:\\ \pi(j)<\pi(i)}} \check{b}_{ij}}{\sum\limits_{\substack{(j,i)\in V\times V:\\ j\neq i}} \check{b}_{ij}}.$$

The coefficient $R_{\max}(\pi)$ can take any value in the interval $[0, 1]$. Large $R_{\max}(\pi)$ supports the hypothesis that the underlying graph is a DAG and the estimated topological order lies in the equivalence class of topological orders defined in (2.32).

In our data example, we have $R_{\max}(\hat{\pi}) = 0.929$, strongly supporting the hypothesis of a recursive ML model. Now using the estimator (2.36), and applying Algorithms 2 and 3 with $\delta_1 = 0.02$ and $\delta_2(k) = 0.02$ for $k \in \{1,2,3\}$, we get the estimated minimum ML DAG $\mathcal{D}^{\hat{B}}$ and ML coefficient matrix $\hat{\boldsymbol{B}}$, where we sorted the matrix according to $\hat{\pi}$. These are shown in Figure 2.3.



$$\hat{\boldsymbol{B}} = \begin{array}{c} \begin{array}{cccc} \mathrm{A}C & \mathrm{L}Z & \mathrm{B}C & \mathrm{V}A \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0.177 & 0.168 & 1 & 0 \\ 0.146 & 0.132 & 0.321 & 1 \end{pmatrix} \end{array} \begin{array}{l} \\ \mathrm{A}C \\ \mathrm{L}Z \\ \mathrm{B}C \\ \mathrm{V}A \end{array} \tag{2.38}$$

Figure 2.3: Estimated minimum ML DAG $\mathcal{D}^{\hat{B}}$ with estimated ML coefficient matrix $\hat{B}$.

Observe that, since we estimate the edge from AC to LZ to be absent, there are two possible topological orders.

From the estimates we observe that both, α-carotene and β-carotene lead to high amounts of vitamin A. This is in line with our expectation since β-carotene is a precursor to vitamin A and can be converted by β-carotene 15,15'-monoxygenase by many animals including humans. Similarly, also α-carotene can be converted to vitamin A. However, it is only half as active as β-carotene which explains that the edge weight from α-carotene to vitamin A is approximately half compared to the edge weight from β-carotene to vitamin A (0.146 compared to 0.321). Moreover, we can see that high amounts of lutein+zeaxanthin also lead to high amounts of β-carotene and high amounts of α-carotene also lead to high amounts of β-carotene. However, we did not find a significant connection between α-carotene and lutein+zeaxanthin. Observe that Klüppelberg and Krali [2021] inferred the same topological order, yet with one additional edge from α-carotene to lutein+zeaxanthin. However, it is also the edge with the smallest estimated edge weight. Similarly as in Klüppelberg and Krali [2021], we plot bivariate extremes in Figure 2.4 to underline our finding. The first 5 plots in Figure 2.4 look rather similar. For every large value of the substance on the vertical axis, we can see a large value of the substance on the horizontal axis. Moreover, these observations are shaped closely to a line. In contrast, a large value of the substance on the horizontal axis might as well coincide with a small value of the substance on the vertical axis. Therefore, e.g. a high amount of α-carotene leads to a high amount of vitamin A but a high amount of vitamin A does not necessarily lead to a high amount of α-carotene. This also supports that the dependence is not mutual and hence we can model it by a DAG. The same can be seen for any pair given in the plots 1-5. Moreover, since parts of the observations are shaped closely along a line, which we would expect for a recursive ML model, we can conclude that the recursive ML model fits the data very well.

The sixth plot is different from the other 5 plots, since for most large observations of α-carotene the level of lutein+zeaxanthin is not increased as most large observations in lutein+zeaxanthin do also not result in a high level of α-carotene. Therefore, the two substances do not seem to affect each other and we rightly concluded that there is no edge.

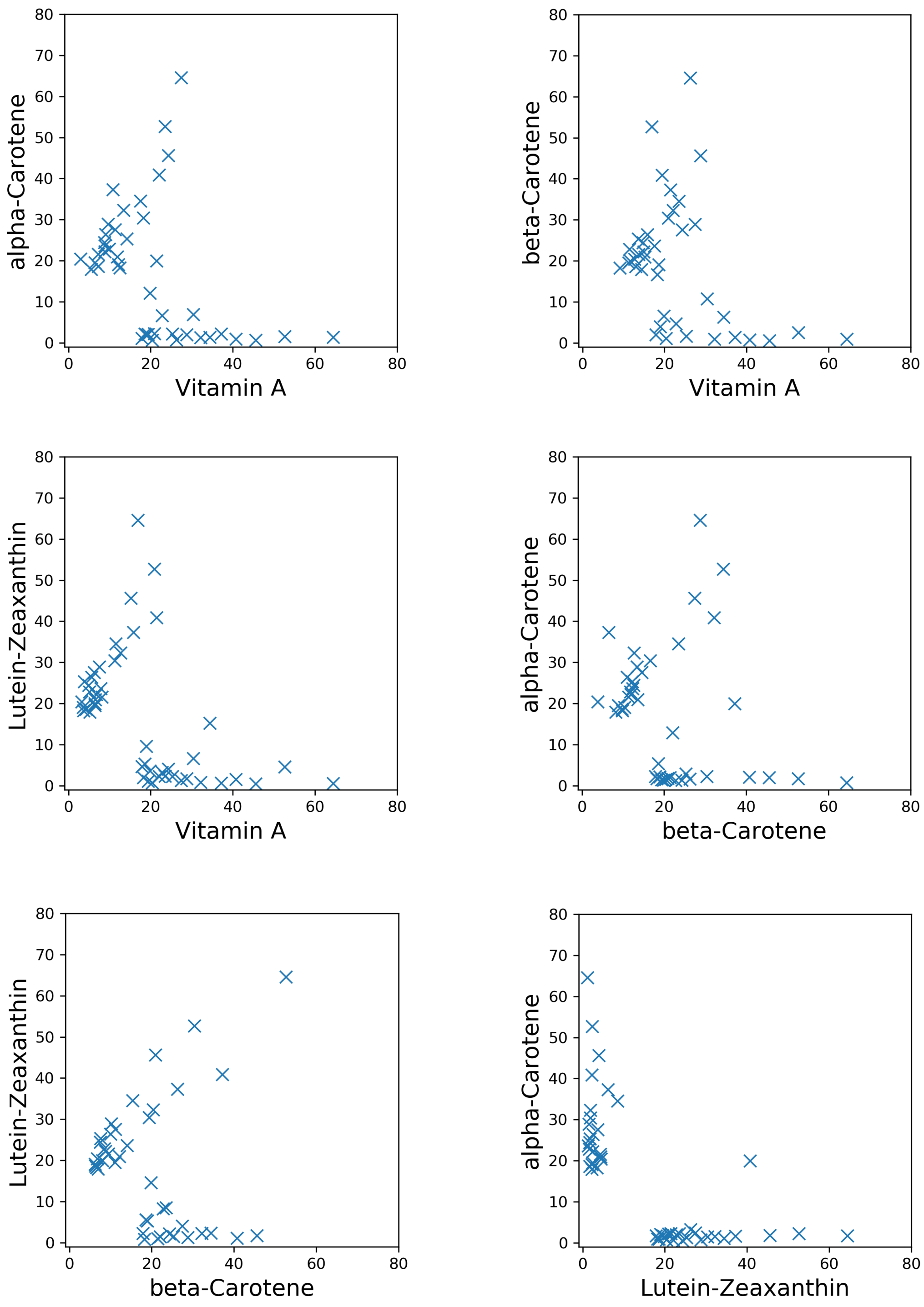


Figure 2.4: The empirical bivariate extremes (25 largest observations).

Sample Size	Correct Runs(1)	Correct Runs(2)	Correct Runs(3)
50	555	946	998
200	995	1000	1000
500	1000	1000	1000
1000	1000	1000	1000

Table 2.1: Empirical success probability for estimated topological order being in the equivalence class of topological orders for (1) No noise, (2): Gamma(1,2), (3); Gamma(2,2).

2.6.2 Simulation study

We want to illustrate the effect of observational noise in the ML model. We simulate recursive ML vectors with propagating noise, where the innovations $Z_1, \dots, Z_4$ are Fréchet(2) distributed and we use the estimated $\hat{\boldsymbol{B}}$ from (2.38) from the data analysis above for the ML coefficient matrix $\boldsymbol{B}$. Moreover, we simulate three different scenarios. In the first scenario, we assume the non-noisy model as given in (2.1), while for the second scenario we choose the propagating noise model with a medium sized noise and in the third setting we choose a noise variable which is stochastically larger. The scenarios are given as follows:

(1) No noise
(2) $\ln(\varepsilon_i) \sim \text{Gamma}(\lambda = 1, \alpha = 2)$ for $i \in \{1, 2, 3, 4\}$, which corresponds to $\mathbb{E}(\varepsilon_i) = 2$
(3) $\ln(\varepsilon_i) \sim \text{Gamma}(\lambda = 2, \alpha = 2)$ for $i \in \{1, 2, 3, 4\}$, which corresponds to $\mathbb{E}(\varepsilon_i) = 4$

We assume to have no information on the underlying DAG and we only consider the quality of the estimator $\check{b}_{ij}$ given in (2.31). We choose the sample sizes $n \in \{50, 200, 500, 1000\}$ and 1000 simulation runs for each sample size. We first assess the success probabilities for Algorithm 1. Table 2.1 shows that the topological order can be correctly estimated even for small sample sizes. Moreover, the number of correct runs increases for larger noise variables. This is expected since the noise variables are one-sided. Therefore, for a path p from j to i the ratio $U_i/U_j \geq d_{ij}(p) \prod_{k \in S_p} \varepsilon_k$ increases, while the ratio $U_j/U_i \leq 1/(d_{ij}(p) \prod_{k \in S_p} \varepsilon_k)$ decreases. Therefore, it is easier to identify the paths in $\mathcal{D}$ for larger noise.

Next, we want to assess the quality of the estimated ML coefficient matrix $\check{\boldsymbol{B}}$. To do so, for every pair (j, i) with $b_{ij} > 0$ and every simulation run $k \in \{1, \dots, 1000\}$, we denote the minimum ratio estimator given in (2.31) by $\check{b}_{ij}^k$.

We consider the empirical RMSE, standard deviation and bias for each $b_{ij} > 0$ in each model (1)-(3). In what follows we compare the three classical quantities

$$\mathrm{bias}(\check{b}_{ij}) := \frac{1}{1000}\sum_{k=1}^{1000} \check{b}_{ij}^{k} - b_{ij}, \tag{2.39}$$

$$\mathrm{SD}(\check{b}_{ij}) := \sqrt{\frac{1}{1000}\sum_{k=1}^{1000}(\check{b}_{ij}^{k} - \bar{\check{b}}_{ij})^2} \quad \text{with} \quad \bar{\check{b}}_{ij} = \frac{1}{1000}\sum_{k=1}^{1000}\check{b}_{ij}^{k}, \tag{2.40}$$

$$\mathrm{RMSE}(\check{b}_{ij}) := \sqrt{\frac{1}{1000}\sum_{k=1}^{1000}(\check{b}_{ij}^{k} - b_{ij})^2}, \tag{2.41}$$

All three quantities are comparatively small even for small sample sizes and decrease whenever the sample size increases. Moreover, they are larger in the propagating noise model and larger noise terms also increase the three quantities. This is in line with what we can expect from the model as noise terms increase the ratios U_i/U_j and hence also increase the minimum ratio estimator. On the other hand, recall from above that with increasing noise the estimation of the DAG improves.

Sample Size	Edge	Edge Weight	Bias(1)	Bias(2)	Bias(3)
50	$AC \to BC$	0.177	0.012	0.020	0.046
50	$AC \to VA$	0.146	0.028	0.029	0.063
50	$LZ \to BC$	0.168	0.013	0.020	0.045
50	$LZ \to VA$	0.132	0.027	0.029	0.064
50	$BC \to VA$	0.321	0	0.014	0.053
200	$AC \to BC$	0.177	0	0.005	0.019
200	$AC \to VA$	0.146	0	0.007	0.026
200	$LZ \to BC$	0.168	0	0.005	0.020
200	$LZ \to VA$	0.132	0.001	0.007	0.026
200	$BC \to VA$	0.321	0	0.004	0.023
500	$AC \to BC$	0.177	0	0.002	0.012
500	$AC \to VA$	0.146	0	0.003	0.016
500	$LZ \to BC$	0.168	0	0.002	0.012
500	$LZ \to VA$	0.132	0	0.003	0.016
500	$BC \to VA$	0.321	0	0.001	0.015
1000	$AC \to BC$	0.177	0	0.001	0.008
1000	$AC \to VA$	0.146	0	0.001	0.011
1000	$LZ \to BC$	0.168	0	0.001	0.008
1000	$LZ \to VA$	0.132	0	0.001	0.010
1000	$BC \to VA$	0.321	0	0.001	0.010

Table 2.2: Empirical Bias (2.39) for (1): No noise, (2): Gamma(1,2), (3): Gamma(2,2)

Sample Size	Edge	Edge Weight	Std(1)	Std(2)	Std(3)
50	$AC \to BC$	0.177	0.031	0.023	0.031
50	$AC \to VA$	0.146	0.046	0.033	0.041
50	$LZ \to BC$	0.168	0.031	0.022	0.031
50	$LZ \to VA$	0.132	0.046	0.033	0.043
50	$BC \to VA$	0.321	0.006	0.015	0.031
200	$AC \to BC$	0.177	0.001	0.005	0.011
200	$AC \to VA$	0.146	0.002	0.006	0.015
200	$LZ \to BC$	0.168	0.002	0.005	0.012
200	$LZ \to VA$	0.132	0.005	0.008	0.017
200	$BC \to VA$	0.321	0	0.004	0.013
500	$AC \to BC$	0.177	0	0.002	0.007
500	$AC \to VA$	0.146	0	0.003	0.009
500	$LZ \to BC$	0.168	0	0.002	0.007
500	$LZ \to VA$	0.132	0.001	0.003	0.009
500	$BC \to VA$	0.321	0	0.001	0.008
1000	$AC \to BC$	0.177	0	0.001	0.004
1000	$AC \to VA$	0.146	0	0.001	0.006
1000	$LZ \to BC$	0.168	0	0.001	0.004
1000	$LZ \to VA$	0.132	0	0.001	0.006
1000	$BC \to VA$	0.321	0	0.001	0.005

Table 2.3: Empirical Standard Deviation (2.40) for (1): No noise, (2): Gamma(1,2), (3): Gamma(2,2)

Sample Size	Edge	Edge Weight	RMSE(1)	RMSE(2)	RMSE(3)
50	$AC \to BC$	0.177	0.033	0.030	0.056
50	$AC \to VA$	0.146	0.054	0.044	0.075
50	$LZ \to BC$	0.168	0.033	0.029	0.054
50	$LZ \to VA$	0.132	0.053	0.044	0.077
50	$BC \to VA$	0.321	0.006	0.021	0.061
200	$AC \to BC$	0.177	0.001	0.007	0.023
200	$AC \to VA$	0.146	0.002	0.009	0.030
200	$LZ \to BC$	0.168	0.002	0.007	0.023
200	$LZ \to VA$	0.132	0.006	0.011	0.031
200	$BC \to VA$	0.321	0	0.005	0.027
500	$AC \to BC$	0.177	0	0.003	0.014
500	$AC \to VA$	0.146	0	0.004	0.018
500	$LZ \to BC$	0.168	0	0.003	0.014
500	$LZ \to VA$	0.132	0.001	0.004	0.018
500	$BC \to VA$	0.321	0	0.002	0.017
1000	$AC \to BC$	0.177	0	0.001	0.090
1000	$AC \to VA$	0.146	0	0.002	0.120
1000	$LZ \to BC$	0.168	0	0.001	0.090
1000	$LZ \to VA$	0.132	0	0.002	0.012
1000	$BC \to VA$	0.321	0	0.001	0.011

Table 2.4: Empirical RMSE (2.41) for (1): No noise, (2): Gamma(1,2), (3): Gamma(2,2)

2.7 Supplementary Material

2.7.1 Proofs of Section 2.3

Proof of Theorem 2.3.2 Rewrite (2.17) in matrix form by means of the tropical matrix multiplication (2.9) as

$$\boldsymbol{U} = \boldsymbol{E}_d \odot \big(\boldsymbol{C} \odot \boldsymbol{U} \vee \boldsymbol{Z}\big).$$

The associative law implies

$$\boldsymbol{U} = (\boldsymbol{E}_d \odot \boldsymbol{C} \odot \boldsymbol{U}) \vee (\boldsymbol{E}_d \odot \boldsymbol{Z}) \quad \Leftrightarrow \quad \boldsymbol{U} = (\bar{\boldsymbol{C}} \odot \boldsymbol{U}) \vee \bar{\boldsymbol{Z}}, \tag{2.42}$$

with $\bar{\boldsymbol{C}} = \boldsymbol{E}_d \odot \boldsymbol{C}$, which is identical to (2.19), and $\bar{\boldsymbol{Z}} = \boldsymbol{E}_d \odot \boldsymbol{Z}$. The right-most equation in (2.42) is of the same form as the non-noisy model in (2.10), so that analogously to its solution given in (2.11), we get the solution

$$\boldsymbol{B}^* = (\boldsymbol{I}_d \vee \bar{\boldsymbol{C}})^{\odot(d-1)}, \qquad \boldsymbol{U} = \boldsymbol{B}^* \odot \bar{\boldsymbol{Z}} = \boldsymbol{B}^* \odot \boldsymbol{E}_d \odot \boldsymbol{Z},$$

where $\boldsymbol{B}^*$ is the Kleene star matrix of $\bar{\boldsymbol{C}}$. Therefore, defining $\bar{\boldsymbol{B}} = \boldsymbol{B}^* \odot \boldsymbol{E}_d$ yields the result. □

Proof of Corollary 2.3.4 From (2.22) and the continuity of Z and ε we have

$$U_i = \bigvee_{j \in \mathrm{An}(i)} \bar{b}_{ij} Z_j = \bar{b}_{ik} Z_k \tag{2.43}$$

for some unique $k \in \mathrm{An}(i)$. We want to show that this implies $U_i = \bar{b}_{ik}\tilde{U}_k$, i.e., $\tilde{U}_k = Z_k$. Applying first (2.23), then (2.22) and finally (2.20), we obtain

$$\tilde{U}_k = \frac{U_k}{\varepsilon_k} = \frac{\bigvee_{l \in \mathrm{An}(k)} \bar{b}_{kl} Z_l}{\varepsilon_k} \geq \frac{\bar{b}_{kk} Z_k}{\varepsilon_k} = Z_k.$$

Now assume that $\tilde{U}_k > Z_k$. Then there exists an $l \in \mathrm{an}(k) \subset \mathrm{An}(i)$ with $\bar{b}_{kl} Z_l > \varepsilon_k Z_k$. Note also that the maximum random path weight from l to i must be greater or equal than the maximum random path weight from l to i passing through node k. These two facts lead to

$$U_i = \bigvee_{j \in \mathrm{An}(i)} \bar{b}_{ij} Z_j \geq \bar{b}_{il} Z_l \geq \frac{\bar{b}_{kl}\bar{b}_{ik}}{\varepsilon_k} Z_l > \bar{b}_{ik} Z_k,$$

The above inequality, however, contradicts (2.43). Therefore, since $k \in \mathrm{An}(i)$, we have

$$U_i \leq \bigvee_{j \in \mathrm{An}(i)} \bar{b}_{ij} \tilde{U}_j.$$

Now assume that $U_i < \vee_{j \in \mathrm{An}(i)} \bar{b}_{ij} \tilde{U}_j$. Then, with the same arguments as above, $U_i < \vee_{j \in \mathrm{An}(i)} \bar{b}_{ij} Z_j$ which is a contradiction. □

Proof of Lemma 2.3.8 (a) We first assume that $j = i$. Since $\mathcal{D}$ is a DAG, $\mathrm{de}(i) \cap \mathrm{pa}(i) = \emptyset$ and $\bar{b}_{ki}\bar{b}_{ik} = 0$ for all $k \neq i$. Therefore, the equality holds and the inequality is equivalent to $\bar{b}_{ii} \geq 0$ which obviously holds.

Next, assume $j \neq i$ and $j \notin \mathrm{an}(i)$. Then by (2.20) $\bar{b}_{ij} = 0$ and there is no path from j to i. Therefore, $\mathrm{de}(j) \cap \mathrm{pa}(i) = \emptyset$. Hence, the right-hand side of the inequality equals zero. Moreover, the equality holds as well, otherwise $\bar{b}_{kj} > 0$ and $\bar{b}_{ik} > 0$ for some $k \in V$ and therefore, by (2.20) there would be a path from j to k and from k to i which contradicts $j \notin \mathrm{an}(i)$.

For $j \in \mathrm{pa}(i)$ with $\mathrm{de}(j) \cap \mathrm{pa}(i) = \emptyset$, the critical path must be the edge $j \to i$ since it is the only path from j to i. Furthermore, the equality $\bar{b}_{ij} = \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$ holds for $k = i$ and $k = j$ while for all $k \notin \{i, j\}$ it must hold that $\frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}} = 0$. Therefore, the equality holds. Moreover, the right-hand side of the inequality again equals zero and we have strict inequality.

Now assume $j \in \mathrm{an}(i)$ and $\mathrm{de}(j) \cap \mathrm{pa}(i) \neq \emptyset$. Then for every path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ with $n \geq 2$ from j to i and every $k_m \in \{k_1, \ldots, k_{n-1}\}$, by (2.18),

$$\begin{aligned}
\bar{d}_{ij}(p) &= \varepsilon_j \prod_{l=0}^{n-1} c_{k_{l+1}k_l} \varepsilon_{k_{l+1}} \\
&= \frac{\varepsilon_j \prod_{l=0}^{m-1} c_{k_{l+1}k_l}\varepsilon_{k_{l+1}} \cdot \varepsilon_{k_m} \prod_{l=m}^{n-1} c_{k_{l+1}k_l}\varepsilon_{k_{l+1}}}{\varepsilon_{k_m}} \\
&= \frac{\bar{d}_{k_m j}(p_1)\bar{d}_{ik_m}(p_2)}{\bar{b}_{k_m k_m}},
\end{aligned} \tag{2.44}$$

with $p_1 = [j = k_0 \to k_1 \to \ldots \to k_m]$ and $p_2 = [k_m \to \ldots \to k_n = i]$, where in the last step we have used that $\varepsilon_{k_m} = \bar{b}_{k_m k_m}$. Therefore, for the random critical path p

with $\bar{b}_{ij} = \bar{d}_{ij}(p)$ it holds that every sub-path of this path is itself critical, otherwise we could find a path of larger random path weight by replacing the sub-path by a path of larger random weight. It follows that

$$\bar{b}_{ij} \geq \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$$

with equality whenever the critical path p from j to i contains a node $k \in \mathrm{de}(j) \cap \mathrm{an}(i)$. Since for $k = i$ or $k = j$ we have $\bar{b}_{ij} = \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$ and for $k \in V \setminus \big((\mathrm{an}(i) \cap \mathrm{de}(j)) \cup \{j, i\}\big)$ we have $\frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}} = 0$, the equality holds as well.

(b) First assume that there is a path $p := [j \to \ldots \to k \to \ldots \to i]$ with $\bar{d}_{ij}(p) = \bar{b}_{ij}$. Then by (2.44) we have $\bar{b}_{ij} = \frac{\bar{d}_{kj}(p_1)\bar{d}_{ik}(p_2)}{\bar{b}_{kk}}$. Now every sub-path of a random critical path must be itself critical, as explained in the proof of part a). Hence, $\bar{b}_{kj} = \bar{d}_{kj}(p_1)$ and $\bar{b}_{ik} = \bar{d}_{ik}(p_2)$ and for this reason $\bar{b}_{ij} = \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$.

In contrast, let $\bar{d}_{ij}(p) < \bar{b}_{ij}$ for all $p \in P_{ikj}$, where P_{ikj} denotes all paths from j to i that pass through k. Now choose $p_1 = [j \to \ldots \to k]$ and $p_2 = [k \to \ldots \to i]$ such that $\bar{d}_{kj}(p_1) = \bar{b}_{kj}$ and $\bar{d}_{ik}(p_2) = \bar{b}_{ik}$. Then, for the path $p \in P_{ikj}$ that results from concatenation of p_1 and p_2 we have by (2.20)

$$\bar{b}_{ij} > \bar{d}_{ij}(p) = \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}},$$

which proves the reverse direction.

(c) For $j = i$ the inequality obviously holds, since $b_{ii} = 1$. If $j \notin \mathrm{An}(i)$, then by definition $\bar{b}_{ij} = b_{ij} = 0$ and $\bar{b}_{jj} = \varepsilon_j \geq 1$. Therefore, the inequality is equivalent to $U_i/U_j \geq 0$, which is true. Now let $j \in \mathrm{an}(i)$. Then by (2.18) and (2.20) the center ratio can be written as

$$\frac{\bar{b}_{ij}}{\bar{b}_{jj}} := \bigvee_{p \in P_{ij}} \frac{\bar{d}_{ij}(p)}{\bar{b}_{jj}} = \bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{l=0}^{n-1} \varepsilon_{k_{l+1}} \geq \bigvee_{p \in P_{ij}} d_{ij}(p) = b_{ij}, \tag{2.45}$$

since $\bar{b}_{jj} = \varepsilon_j$ and $\varepsilon_i \geq 1$. Now we use (2.24) and obtain by (2.45)

$$\frac{U_i}{U_j} = \frac{\bigvee_{k \in \mathrm{An}(i)} \bar{b}_{ik}\tilde{U}_k}{U_j} \geq \frac{\bar{b}_{ij}\tilde{U}_j}{U_j} = \frac{\bar{b}_{ij}U_j}{\varepsilon_j U_j} = \frac{\bar{b}_{ij}}{\bar{b}_{jj}} \geq b_{ij}.$$

(d) We first prove by contradiction that there is no lower bound for U_i/U_j of larger value than the one given in part (c). Assume $j \in \mathrm{an}(i)$ and there is a lower bound $c > b_{ij}$. Since $Z_1, \ldots, Z_d$ are i.i.d., every innovation Z_l can realize the maximum with positive probability, such that for every $l \in \mathrm{An}(j)$,

$$\mathbb{P}\Big(\Big\{U_j = \bigvee_{k \in \mathrm{An}(j)} \bar{b}_{jk}Z_k = \bar{b}_{jl}Z_l\Big\} \cap \Big\{U_i = \bigvee_{k \in \mathrm{An}(i)} \bar{b}_{ik}Z_k = \bar{b}_{il}Z_l\Big\}\Big) > 0. \tag{2.46}$$

Hence, without loss of generality we assume that this holds for $l = j$. Denote the random critical path $p := [j = k_0 \to \ldots \to k_n = i]$ such that $\bar{d}_{ij}(p) = \bar{b}_{ij}$. Then, it follows on the event in (2.46) with $l = j$ from (2.18) that

$$\mathbb{P}\left(\frac{U_i}{U_j} < c\right) = \mathbb{P}\left(\frac{\bar{b}_{ij}}{\bar{b}_{jj}} < c\right) = \mathbb{P}\left(\frac{\varepsilon_j d_{ij}(p) \prod_{l=0}^{n-1} \varepsilon_{k_{l+1}}}{\varepsilon_j} < c\right)$$
$$= \mathbb{P}\left(d_{ij}(p) \prod_{l=0}^{n-1} \varepsilon_{k_{l+1}} < c\right) > 0,$$

since $d_{ij}(p) \le b_{ij} < c$ and $\varepsilon \ge 1$. Hence, c is no lower bound and together with part c) this entails the support for $j \in \mathrm{an}(i)$.

Now assume $j \notin \mathrm{An}(i)$ such that $b_{ij} = 0$. Assume that U_i/U_j is lower bounded by some $c > 0$. Then by (2.22),

$$\frac{U_i}{U_j} \ge c \quad \Leftrightarrow \quad \bigvee_{k \in \mathrm{An}(i)} \bar{b}_{ik} Z_k \ge c \bigvee_{k \in \mathrm{An}(j)} \bar{b}_{jk} Z_k,$$

which is equivalent to

$$\bigvee_{k \in \mathrm{An}(i)} \bar{b}_{ik} Z_k \ge c \Big(\bigvee_{k \in \mathrm{An}(i) \cap \mathrm{An}(j)} \bar{b}_{jk} Z_k \vee \bigvee_{k \in \mathrm{An}(j) \setminus \mathrm{An}(i)} \bar{b}_{jk} Z_k \Big).$$

Therefore, it holds in particular, that

$$\bigvee_{k \in \mathrm{An}(i)} \bar{b}_{ik} Z_k \ge c \, \bar{b}_{jl} Z_l \tag{2.47}$$

for every $l \in \mathrm{An}(j) \setminus \mathrm{An}(i)$. This set is non-empty since $j \notin \mathrm{An}(i)$, so it contains at least j. However, since the innovation and the noise variables are all independent and unbounded above, we have for every $l \in \mathrm{An}(j) \setminus \mathrm{An}(i)$

$$\mathbb{P}\Big(Z_l \ge \frac{\bigvee_{k \in \mathrm{An}(i)} \bar{b}_{ik} Z_k}{c \, \bar{b}_{jl}}\Big) > 0,$$

contradicting (2.47) and, hence, the assumption of a lower positive bound c for U_i/U_j.

The upper interval limits of U_i/U_j for $j \in \mathrm{an}(i)$ and and $j \notin \mathrm{An}(i)$ follow from changing the roles of i and j. For $j \ne i$, the ratio U_i/U_j always contains ε_i or ε_j and both random variables are atom-free and independent of all innovations $Z_1, \dots, Z_d$ and ε_k for $k \ne i$ and $k \ne j$. Therefore, the ratio inherits the continuity of the noise variables and part d) follows.

(e) For $j = i$ we have $b_{ij} = 1 \ne 0 = \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{b_{kj} b_{ik}}{b_{kk}}$. If $j \notin \mathrm{An}(i)$ we have $b_{ij} = \bar{b}_{ij} = 0$ by (2.20).

Next assume that $j \in \mathrm{an}(i)$, and $b_{ij} = \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{b_{kj} b_{ik}}{b_{kk}} \ne 0$. Then there is a path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ from j to i with non-random path weight $d_{ij}(p) = b_{ij}$, which is not the edge $j \to i$.

For a contradiction, assume that $\bar{b}_{ij} > \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj} \bar{b}_{ik}}{\bar{b}_{kk}}$. This is equivalent to the edge $j \to i$ being the random critical path. However, every path $p \in P_{ij}$ has random path weight, which depends on both noise variables ε_i and ε_j, so in particular, the non-random critical path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ from j to i with path weight $d_{ij}(p) = b_{ij}$ is one of these paths. Therefore, by (2.18) and since $b_{ij} > c_{ij}$, the random path weight of p is

$$\bar{d}_{ij}(p) = b_{ij}\varepsilon_j \prod_{l=0}^{n-1} \varepsilon_{k_{l+1}} \geq b_{ij}\varepsilon_j\varepsilon_i > c_{ij}\varepsilon_j\varepsilon_i = \bar{b}_{ij},$$

where we have used that $\varepsilon \geq 1$. This is a contradiction and hence $\bar{b}_{ij} = \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$.

(f) The assumptions $b_{ij} > \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{b_{kj}b_{ik}}{b_{kk}}$ and $\mathrm{de}(j) \cap \mathrm{an}(i) \neq \emptyset$ are equivalent to the edge $p_{\max} = [j \to i]$ being the only non-random critical path.

Let $p' = [j = k_0 \to k_1 \to \ldots \to k_n = i] \neq p_{\max}$ be the path such that $\bigvee_{p \in P_{ij} \setminus \{p_{\max}\}} \bar{d}_{ij}(p) = \bar{d}_{ij}(p')$. Then

$$\bigvee_{p \in P_{ij} \setminus \{p_{\max}\}} \bar{d}_{ij}(p) = \bar{d}_{ij}(p') = \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}},$$

otherwise we can construct a path of larger random path weight from j to i passing through k as explained in the proof of part a). First assume that $\bar{b}_{ij} = \bar{d}_{ij}(p_{\max})$. Then, $\bar{b}_{ij} > \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$ and $\mathrm{de}(j) \cap \mathrm{an}(i) \neq \emptyset$ is by (2.18) and (2.20) equivalent to

$$\bar{b}_{ij} > \bar{d}_{ij}(p') = \varepsilon_i\varepsilon_j d_{ij}(p') \prod_{l=0}^{n-2} \varepsilon_{k_{l+1}} \quad \Longleftrightarrow \quad \frac{b_{ij}}{d_{ij}(p')} > \prod_{l=0}^{n-2} \varepsilon_{k_{l+1}}. \tag{2.48}$$

Since $\varepsilon \geq 1$, also $b_{ij}/d_{ij}(p') > 1$. Hence, the event given by (2.48) has positive probability which is however, strictly smaller than one, since the noise variables do not have an upper bound. Therefore, since $\bar{b}_{ij} \geq \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}$, by part a), the complementary event

$$\Big\{\bar{b}_{ij} = \bigvee_{k \in \mathrm{de}(j) \cap \mathrm{an}(i)} \frac{\bar{b}_{kj}\bar{b}_{ik}}{\bar{b}_{kk}}\Big\}$$

is also having positive probability. □

Proof of Lemma 2.3.11 (a) Suppose there is an edge $k_l \to k_{l+1}$ in p such that $c_{k_{l+1}k_l} \notin \mathcal{D}^B$. Then, $\mathrm{de}(j) \cap \mathrm{an}(i) \neq \emptyset$ and by Lemma 2.3.8 e) $\mathbb{P}(\bar{b}_{k_{l+1}k_l} = c_{k_{l+1}k_l}\varepsilon_{k_l}\varepsilon_{k_{l+1}}) = 0$, so we can replace the edge $k_l \to k_{l+1}$ by some other path to get a new path from j to i of larger random path weight than p. Hence, p is not a possible critical path realization. The same argument can be used for the reverse.

(b) First consider $\neg(S_{p_1} \cap S_{p_2} = \emptyset$ or for every $r \in S_{p_1} \cap S_{p_2}$ the sub-path of p_1 from j to r is a sub-path of p_2 or the sub-path of p_2 from l to r is a sub-path of $p_1)$. Then there exists some node $r \in S_{p_1} \cap S_{p_2}$ such that $p_1 = [j \to \ldots \to s \to r \to \ldots \to i]$ and $p_2 = [l \to \ldots \to t \to r \to \ldots \to m]$ with $s \neq t$. Denote by $p_{11} := [j \to \ldots \to s]$ the sub-path of p_1 from j to s. We want to show by contradiction that the event (2.25) has probability zero. Therefore, we consider the subset of Ω such that (2.25) holds and show that it is a null-set. Since on this subset, p_1 is the random critical path and passes through s, by Lemma 2.3.8 b) we have $\bar{b}_{ij} = \frac{\bar{b}_{sj}\bar{b}_{is}}{\bar{b}_{ss}}$ and $U_s = U_j\bar{b}_{sj}/\varepsilon_j =$

$U_j d_{sj}(p_{11}) \prod_{k\in S_{p_{11}}} \varepsilon_k$. With the same argument it also holds that $\bar{b}_{ij} = \frac{\bar{b}_{rj}\bar{b}_{ir}}{\bar{b}_{rr}}$ and $U_r = U_j \bar{b}_{rj}/\varepsilon_j = U_j d_{sj}(p_{11}) \prod_{k\in S_{p_{11}}} \varepsilon_k c_{rs}\varepsilon_r$. Hence, it must holds that $U_r = U_s c_{rs}\varepsilon_r$. By the same arguments, we also must have $U_r = U_t c_{rt}\varepsilon_r$, which together leads to

$$U_s c_{rs} = U_t c_{rt}.$$

This is by (2.20) and (2.22) equivalent to

$$c_{rs} \bigvee_{l\in \mathrm{An}(s)} \varepsilon_l \bigvee_{p\in P_{sl}} d_{sl}(p) \prod_{k\in S_p} \varepsilon_k Z_l = c_{rt} \bigvee_{l\in \mathrm{An}(t)} \varepsilon_l \bigvee_{p\in P_{tl}} d_{tl}(p) \prod_{k\in S_p} \varepsilon_k Z_l.$$

Now since $\mathcal{D}$ is acyclic, there cannot be a path from s to t and from t to s; so without loss of generality we can assume that there is no path from t to s. However, the right-hand side of the equation always contains ε_t which is not part of the left-hand side. Since $Z_1, \dots, Z_d$ as well as $\varepsilon_1, \dots, \varepsilon_i$ are atom-free and independent random variables, this can only happen on a null-set.

Next consider the reverse, i.e., $S_{p_1} \cap S_{p_2} = \emptyset$ or for every $r \in S_{p_1} \cap S_{p_2}$ the sub-path of p_1 from j to r is a sub-path of p_2 or the sub-path of p_2 from l to r is a sub-path of p_1.

If $S_{p_1} \cap S_{p_2} = \emptyset$, then the probability of (2.25) is obviously positive. Without loss of generality we now assume that for every $r \in S_{p_1} \cap S_{p_2}$ the sub-path of p_2 from l to r is a sub-path of p_1. We now define r to be the last common node of the two paths p_1 and p_2. Then, p_1 and p_2 induce the paths $p' = [j \to \dots \to l \to \dots r]$, $p'' = [r \to \dots \to i]$ and $p''' = [r \to \dots \to m]$. Then

$$\begin{aligned}
&\Big\{U_i = U_j d_{ij}(p_1) \prod_{k\in S_{p_1}} \varepsilon_k, U_m = U_l d_{ml}(p_2) \prod_{k\in S_{p_2}} \varepsilon_k\Big\} \\
&= \Big\{U_r = U_j d_{rj}(p') \prod_{k\in S_{p'}} \varepsilon_k, U_i = U_r d_{ir}(p'') \prod_{k\in S_{p''}} \varepsilon_k, U_m = U_r d_{mr}(p''') \prod_{k\in S_{p''}} \varepsilon_k\Big\},
\end{aligned}$$

which has positive probability, since $S_{p'} \cap S_{p''} \cap S_{p'''} = \emptyset$. □

Proof of Theorem 2.3.12 By the law of total probability we have for $x \geq 1$,

$$\begin{aligned}
I(x) &:= \mathbb{P}\Big(\frac{U_i}{U_j} \leq b_{ij}x\Big) = \mathbb{P}\Big(\frac{U_i}{U_j} \leq b_{ij}x, U_i = \tilde{U}_j \bar{b}_{ij}\Big) + \mathbb{P}\Big(\frac{U_i}{U_j} \leq b_{ij}x, U_i \neq \tilde{U}_j \bar{b}_{ij}\Big) \\
&=: I_1(x) + I_2(x)
\end{aligned}$$

We denote all paths from j to i by $P_{ij} = \{p_1, \dots, p_r, p_{\max}\}$. There are two situations, either $r = 0$ (where we interpret the above set of paths as $\{p_{\max}\}$), or $r \geq 1$. We first give a proof for $r \geq 1$. We start with $I_1(x)$. Since $p_{\max}$ is generic, every path $p \neq p_{\max}$ from j to i has non-random edge weight $d_{ij}(p) < b_{ij}$. Therefore, with (2.23) in the first line, (2.20) in the third and (2.18) in the last, we have for $x > 1$,

$$\begin{aligned}
I_1(x) &= \mathbb{P}(U_i/(\tilde{U}_j\varepsilon_j) \leq b_{ij}x, U_i = \tilde{U}_j \bar{b}_{ij}) \\
&= \mathbb{P}(\bar{b}_{ij}/\varepsilon_j \leq b_{ij}x, U_i = \tilde{U}_j \bar{b}_{ij}) \\
&= \mathbb{P}\Big(\bigvee_{p\in P_{ij}} \bar{d}_{ij}(p)/\varepsilon_j \leq b_{ij}x, U_i = \tilde{U}_j \bar{b}_{ij}\Big) \\
&= \mathbb{P}\Big(\bigvee_{p\in P_{ij}} d_{ij}(p) \prod_{k\in S_p} \varepsilon_k \leq b_{ij}x, U_i = \tilde{U}_j \bar{b}_{ij}\Big). \qquad (2.49)
\end{aligned}$$

By definition of $\bar{b}_{ij}$ in (2.20), there is a path $p \in \{p_{\max}, p_1, \ldots, p_r\}$ such that $\bar{d}_{ij}(p) = \bar{b}_{ij}$ and by continuity of ε the probability that multiple paths satisfy the equation is equal to 0. Therefore, again applying the law of total probability, we find

$$\begin{aligned} I_1(x) &= \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le b_{ij} x, \bar{d}_{ij}(p_{\max}) = \bar{b}_{ij}, U_i = \tilde{U}_j \bar{b}_{ij} \Big) \qquad (2.50) \\ &+ \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le b_{ij} x, \bigvee_{p \in \{p_1, \ldots, p_r\}} \bar{d}_{ij}(p) = \bar{b}_{ij}, U_i = \tilde{U}_j \bar{b}_{ij} \Big) \\ &= \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le b_{ij} x, \bar{d}_{ij}(p_{\max}) = \bar{b}_{ij}, U_i = \tilde{U}_j \bar{b}_{ij} \Big) \\ &+ \sum_{s=1}^{r} \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le b_{ij} x, \bar{d}_{ij}(p_s) = \bar{b}_{ij}, U_i = \tilde{U}_j \bar{b}_{ij} \Big) \\ &=: I_{11}(x) + I_{12}(x). \end{aligned}$$

We first find upper and lower bounds for $I_{11}(x)$. We denote by P_{ijk} all paths from k to i which pass through j. Using the simple identity

$$\{z_1 \vee z_2 \le a, z_1 \vee z_2 = z_1\} = \{z_1 \le a, z_2 \le z_1\}, \qquad (2.51)$$

(2.20) and (2.22) imply

$$\begin{aligned} &\Big\{ \bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le b_{ij} x \Big\} \bigcap \Big\{ \bar{d}_{ij}(p_{\max}) = \bar{b}_{ij} \Big\} \bigcap \Big\{ U_i = \tilde{U}_j \bar{b}_{ij} \Big\} \qquad (2.52) \\ &= \Big\{ \bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le \bar{d}_{ij}(p_{\max}) \Big\} \bigcap \Big\{ \bar{d}_{ij}(p_{\max}) \le b_{ij} x \Big\} \bigcap \Big\{ U_i = \tilde{U}_j \bar{b}_{ij} \Big\} \\ &= \Big\{ \bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \le b_{ij} \prod_{k \in S_{p_{\max}}} \varepsilon_k \Big\} \bigcap \Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \le x \Big\} \bigcap \Big\{ U_i = \tilde{U}_j \bar{b}_{ij} \Big\} \\ &= \bigcap_{p \in P_{ij} \setminus \{p_{\max}\}} \Big\{ \prod_{k \in S_p} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)} \prod_{k \in S_{p_{\max}}} \varepsilon_k \Big\} \bigcap \Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \le x \Big\} \bigcap \\ &\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in S_p \cup \{l\}} \varepsilon_k Z_l \le b_{ij} \prod_{k \in S_{p_{\max}}} \varepsilon_k \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\} \Big\}. \qquad (2.53) \end{aligned}$$

Cancelling all noise variables possible, and since $\varepsilon > 1$, we find a lower bound

$$\begin{aligned}
I_{11}(x) = \mathbb{P}\Big(& \bigcap_{p \in P_{ij} \setminus \{p_{\max}\}} \Big\{ \prod_{k \in S_p \setminus S_{p_{\max}}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} \prod_{k \in S_{p_{\max}} \setminus S_p} \varepsilon_k \Big\} \bigcap \\
& \Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \\
& \leq b_{ij} \prod_{k \in S_{p_{\max}} \setminus (S_p \cup \{l\})} \varepsilon_k \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\}\Big\}\Big) \qquad (2.54) \\
& \geq \mathbb{P}\Big(\bigcap_{p \in P_{ij} \setminus \{p_{\max}\}} \Big\{ \prod_{k \in S_p \setminus S_{p_{\max}}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} \Big\} \bigcap \Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \\
& \bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\}\Big\}\Big) \\
& = \mathbb{P}\Big(\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\}\Big\} \\
& \bigcap \bigcap_{p \in P_{ij} \setminus \{p_{\max}\}} \Big\{ \prod_{k \in S_p \setminus S_{p_{\max}}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} \Big\}\Big) \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big) \\
& =: c_1 \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big),
\end{aligned}$$

for some constant $c_1 \in [0, 1]$ by independence of the noise variables.

We show that $c_1 > 0$. To do so, recall that $b_{ij}/d_{ij}(p) > 1$ for every $p \neq p_{\max}$. Therefore, since $\{p \in P_{ij} \setminus \{p_{\max}\}\} \neq \emptyset$ and $\varepsilon > 1$,

$$\mathbb{P}\Big(\bigcap_{p \in P_{ij} \setminus \{p_{\max}\}} \Big\{ \prod_{k \in S_p \setminus S_{p_{\max}}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} \Big\}\Big) > 0. \qquad (2.55)$$

Next, we want to show that also

$$\mathbb{P}\Big(\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\}\Big\}\Big) > 0. \qquad (2.56)$$

For this, observe that the left-hand side of the inequality in (2.56) does not contain Z_j, since all paths from j to i pass through j. Since $\bar{b}_{jl}$ and the left-hand side of the inequality in (2.56) is independent of Z_j for all $l \in \{1, \ldots, d\}$ and Z_j has unbounded support, Z_j can become arbitrarily large with positive probability such that (2.56) holds.

The intersection of the two events has also positive probability since (2.55) is independent of Z_j. This implies that $c_1 > 0$ and a positive lower bound for $I_{11}(x)$.

To get an upper bound, observe that $\varepsilon \geq 1$ and, hence, for every set S_p we have

$$\Big\{ \varepsilon_k : k \in S_{p_{\max}} \text{ and } \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \subseteq \Big\{ \varepsilon_k : k \in S_{p_{\max}} \text{ and } \prod_{k \in S_{p_{\max}} \setminus S_p} \varepsilon_k \leq x \Big\}.$$

Therefore, starting with (2.54) we find the upper bound

$$
\begin{aligned}
I_{11}(x) &\leq \mathbb{P}\Big(\bigcap_{p \in P_{ij}\setminus\{p_{\max}\}} \Big\{ \prod_{k \in S_p \setminus S_{p_{\max}}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} x \Big\} \bigcap \Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \\
&\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il}\setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\})\setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} x \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\}\Big\}\Big) \\
&= \mathbb{P}\Big(\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il}\setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\})\setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} x \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\}\Big\} \\
&\bigcap \bigcap_{p \in P_{ij}\setminus\{p_{\max}\}} \Big\{ \prod_{k \in S_p \setminus S_{p_{\max}}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} x \Big\}\Big) \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big) \\
&= c_2(x)\, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big). \qquad (2.57)
\end{aligned}
$$

Since the innovations and the noise variables are atom-free, it follows that $\lim_{x\downarrow 1} c_2(x) = c_1$ and, therefore,

$$
I_{11}(x) \sim c_1 \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big), \quad x \downarrow 1. \qquad (2.58)
$$

We next show that $I_{12}(x) = o(I_{11}(x))$ as $x \downarrow 1$. We have for each summand $m \in \{1, \ldots, r\}$, using the simple identity (2.51) to obtain the third line,

$$
\begin{aligned}
&\mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \leq b_{ij} x, \bar{d}_{ij}(p_m) = \bar{b}_{ij}, U_i = \tilde{U}_j \bar{b}_{ij} \Big) \\
&\leq \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \leq b_{ij} x, \bar{d}_{ij}(p_m) = \bar{b}_{ij} \Big) \\
&= \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \leq \bar{d}_{ij}(p_m), \bar{d}_{ij}(p_m) \leq b_{ij} x \Big) \\
&= \mathbb{P}\Big(\bigcap_{p \in P_{ij}\setminus\{p_m\}} \Big\{ \prod_{k \in S_p} \varepsilon_k \leq \frac{d_{ij}(p_m)}{d_{ij}(p)} \prod_{k \in S_{p_m}} \varepsilon_k \Big\} \bigcap \Big\{ \prod_{k \in S_{p_m}} \varepsilon_k \leq \frac{b_{ij} x}{d_{ij}(p_m)} \Big\}\Big) \\
&\leq \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq \frac{d_{ij}(p_m)}{b_{ij}} \prod_{k \in S_{p_m}} \varepsilon_k \Big\} \bigcap \Big\{ \prod_{k \in S_{p_m}} \varepsilon_k \leq \frac{b_{ij} x}{d_{ij}(p_m)} \Big\}\Big) \qquad (2.59)
\end{aligned}
$$

Now the first event rewrites as $\{\frac{b_{ij}}{d_{ij}(p_m)} \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq \prod_{k \in S_{p_m}} \varepsilon_k\} \subseteq \{\frac{b_{ij}}{d_{ij}(p_m)} \leq \prod_{k \in S_{p_m}} \varepsilon_k\}$, since $\varepsilon > 1$. Moreover,

$$
\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq \frac{d_{ij}(p_m)}{b_{ij}} \prod_{k \in S_{p_m}} \varepsilon_k \Big\} \bigcap \Big\{ \prod_{k \in S_{p_m}} \varepsilon_k \leq \frac{b_{ij} x}{d_{ij}(p_m)} \Big\} \subseteq \Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\}.
$$

Hence,

$$
(2.59) \leq \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \Big\{ \prod_{k \in S_{p_m}} \varepsilon_k \in \Big[\frac{b_{ij}}{d_{ij}(p_m)}, \frac{b_{ij} x}{d_{ij}(p_m)} \Big] \Big\}\Big),
$$

Moreover, since $\varepsilon \geq 1$, we have for every subset $S \subseteq S_{p_{\max}}$ that $1 \leq \prod_{k \in S} \varepsilon_k \leq x$, whenever $1 \leq \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x$. Therefore, for another node set $\tilde{S}$ with $S \cap \tilde{S} = \emptyset$ we have

$$\prod_{k \in S} \varepsilon_k \prod_{k \in \tilde{S}} \varepsilon_k \in [a, b] \quad \Rightarrow \quad \prod_{k \in \tilde{S}} \varepsilon_k \in [a/x, b]. \tag{2.60}$$

Finally, since $\bar{d}_{ij}(p_m) = \bar{b}_{ij}$ and $d_{ij}(p_m) < d_{ij}(p_{\max})$ we have $S_{p_m} \setminus S_{p_{\max}} \neq \emptyset$. In total, we obtain

$$\begin{aligned}
(2.59) &\leq \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \Big\{ \prod_{k \in S_{p_m}} \varepsilon_k \in \Big[\frac{b_{ij}}{d_{ij}(p_m)}, \frac{b_{ij} x}{d_{ij}(p_m)} \Big] \Big\} \Big) \\
&\leq \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big) \mathbb{P}\Big(\prod_{k \in S_{p_m} \setminus S_{p_{\max}}} \varepsilon_k \in \Big[\frac{b_{ij}}{x d_{ij}(p_m)}, \frac{b_{ij} x}{d_{ij}(p_m)} \Big] \Big) \\
&= \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big) o(1), \quad x \downarrow 1,
\end{aligned}$$

as the interval in the second probability gets arbitrarily small and the distribution of ε is atom-free. Comparing this upper bound with (2.58) we can see that every summand of $I_{12}(x)$ is negligible with respect to $I_{11}(x)$ as $x \downarrow 1$. Since there are only finitely many nodes and hence finitely many paths from j to i, we have proved that $I_{12}(x) = o(I_{11}(x))$ as $x \downarrow 1$. Hence,

$$I_1(x) \sim c_1 \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big), \quad x \downarrow 1. \tag{2.61}$$

Next, we assume that $r = 0$, i.e., that there is only one path $p_{\max}$ from j to i. Then from (2.50) we find that $I_1(x) = I_{11}(x)$ and simplifies (2.54) to

$$\begin{aligned}
I_1(x) &= \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \\
&\quad \bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in S_p \cup \{l\}} \varepsilon_k Z_l \leq b_{ij} \prod_{k \in S_{p_{\max}}} \varepsilon_k \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\} \Big\} \Big) \\
&\geq \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \\
&\quad \bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\} \Big\} \Big) \\
&= \mathbb{P}\Big(\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\} \Big\} \Big) \\
&\quad \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big) = c_1 \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big)
\end{aligned}$$

for $c_1 > 0$. On the other hand,

$$\begin{aligned}
I_1(x) &\leq \mathbb{P}\Big(\bigcap_{l \in \mathrm{An}(i)} \Big\{ \bigcap_{p \in P_{il} \setminus P_{ijl}} \Big\{ d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \leq b_{ij} x \bigvee_{l \in \mathrm{An}(j)} \bar{b}_{jl} Z_l \Big\} \Big\} \Big) \\
&\quad \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big) = c_2(x) \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big),
\end{aligned}$$

and, since Z and ε are atom-free it again follows that $\lim_{x\downarrow 1} c_2(x) = c_1$ and therefore (2.61) holds also for $r = 0$.

We next show that $I_2(x) = o(I_1(x))$ as $x \downarrow 1$. Since $I_{12}(x) = o(I_{11}(x)$ as $x \downarrow 1$, we can and do assume that

$$\bar{b}_{ij} = b_{ij}\varepsilon_j \prod_{k\in S_{p_{\max}}} \varepsilon_k. \tag{2.62}$$

Moreover, since for all paths $p \in P_{ijl}$ we have $l \in \mathrm{An}(i)$ if and only if $l \in \mathrm{An}(j)$,

$$\begin{aligned}U_i &= \bigvee_{l\in\mathrm{An}(i)} \bigvee_{p\in P_{il}\setminus P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l \vee \bigvee_{l\in\mathrm{An}(j)} \bigvee_{p\in P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l \\ &= \bigvee_{l\in\mathrm{An}(i)} \bigvee_{p\in P_{il}\setminus P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l \vee b_{ij} \prod_{k\in S_{p_{\max}}} \varepsilon_k U_j\end{aligned}$$

by (2.62). If $\bigvee_{l\in\mathrm{An}(i)} \bigvee_{p\in P_{il}\setminus P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l > b_{ij} \prod_{k\in S_{p_{\max}}} \varepsilon_k U_j$, then it follows that

$$U_i = \bigvee_{l\in\mathrm{An}(i)} \bigvee_{p\in P_{il}\setminus P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l. \tag{2.63}$$

Moreover, it holds by (2.24), (2.20) and (2.18)

$$U_i = \bigvee_{k\in\mathrm{An}(i)} \bar{b}_{ik}\tilde{U}_k \geq \bar{b}_{ij}\tilde{U}_j \geq b_{ij} \prod_{k\in S_{p_{\max}}} \varepsilon_k U_j. \tag{2.64}$$

Hence, $U_i/U_j \leq b_{ij}x$ implies that $\prod_{k\in S_{p_{\max}}} \varepsilon_k \leq x$. Therefore, using (2.23), (2.63) and (2.62) we get

$$\begin{aligned}I_2(x) &= \mathbb{P}\Big(\frac{U_i}{U_j} \leq b_{ij}x, U_i > \tilde{U}_j\bar{b}_{ij}\Big) = \mathbb{P}\Big(U_i \in (\tilde{U}_j\bar{b}_{ij}, \tilde{U}_j\varepsilon_j b_{ij}x]\Big) \\ &\leq \mathbb{P}\Big(\Big\{\prod_{k\in S_{p_{\max}}} \varepsilon_k \leq x\Big\} \bigcap \\ &\Big\{\bigvee_{l\in\mathrm{An}(i)} \bigvee_{p\in P_{il}\setminus P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l \in \Big(b_{ij}\tilde{U}_j\varepsilon_j \prod_{k\in S_{p_{\max}}} \varepsilon_k, b_{ij}\tilde{U}_j\varepsilon_j x\Big]\Big\}\Big) \\ &\leq \mathbb{P}\Big(\Big\{\prod_{k\in S_{p_{\max}}} \varepsilon_k \leq x\Big\} \bigcap \\ &\Big\{\bigvee_{l\in\mathrm{An}(i)} \bigvee_{p\in P_{il}\setminus P_{ijl}} d_{il}(p) \prod_{k\in S_p\cup\{l\}} \varepsilon_k Z_l \in \Big(b_{ij}\tilde{U}_j\varepsilon_j, b_{ij}\tilde{U}_j\varepsilon_j x\Big]\Big\}\Big),\end{aligned}$$

since $\varepsilon \geq 1$. Using that $U_j = \tilde{U}_j \varepsilon_j$ and $j \notin S_{p_{\max}}$ and the same argument as in (2.60), we get

$$
\begin{aligned}
&I_2(x) \leq \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\} \bigcap \\
&\Big\{ \bigvee_{l \in \mathrm{An}(i)} \bigvee_{p \in P_{il} \setminus P_{ijl}} d_{il}(p) \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \in \Big(\frac{b_{ij}\tilde{U}_j \varepsilon_j}{x}, b_{ij}\tilde{U}_j \varepsilon_j x \Big] \Big\}\Big) \\
&= \mathbb{P}\Big(\Big\{ \prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big\}\Big) \\
&\mathbb{P}\Big(\Big\{ \bigvee_{l \in \mathrm{An}(i)} \bigvee_{p \in P_{il} \setminus P_{ijl}} \frac{d_{il}(p)}{\varepsilon_j} \prod_{k \in (S_p \cup \{l\}) \setminus S_{p_{\max}}} \varepsilon_k Z_l \in \Big(\frac{b_{ij}\tilde{U}_j}{x}, b_{ij}\tilde{U}_j x \Big] \Big\}\Big),
\end{aligned}
$$

since $\mathcal{D}$ being acyclic implies that $\tilde{U}_j$ and ε_j are independent of ε_k for every $k \in S_{p_{\max}}$. For $x \downarrow 1$ the interval in the second probability gets arbitrarily small. Since the distribution of the noise-variables is atom-free and the left-hand side contains ε_j that is not included in $\tilde{U}_j$, this probability tends to zero as $x \downarrow 1$. Comparing this upper bound with (2.61) we can see that $I_2(x) = o(I_1(x))$ as $x \downarrow 1$. Since $I_{12}(x) = o(I_{11}(x))$, we have

$$
I(x) \sim I_1(x) \sim I_{11}(x) \sim c_1 \, \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x \Big), \quad x \downarrow 1, \tag{2.65}
$$

holds, where the last asymptotic equivalence follows from (2.61). Moreover, we have by (2.50), using (2.18),(2.20) and (2.23),

$$
\begin{aligned}
I(x) \sim I_{11}(x) &= \mathbb{P}\Big(\bigvee_{p \in P_{ij}} d_{ij}(p) \prod_{k \in S_p} \varepsilon_k \leq b_{ij}x, \bar{d}_{ij}(p_{\max}) = \bar{b}_{ij}, U_i = \tilde{U}_j \bar{b}_{ij} \Big) \\
&= \mathbb{P}\Big(\bar{d}_{ij}(p_{\max})/\varepsilon_j \leq b_{ij}x, U_i = \tilde{U}_j \bar{d}_{ij}(p_{\max}) \Big) \\
&= \mathbb{P}\Big(\prod_{k \in S_{p_{\max}}} \varepsilon_k \leq x, U_i = U_j b_{ij} \prod_{k \in S_{p_{\max}}} \varepsilon_k \Big), \quad x \downarrow 1,
\end{aligned} \tag{2.66}
$$

which, together with (2.65), proves the result. □

Proof of Corollary 2.3.14. We first show the result for $P_{ij} \setminus \{p_1, \dots, p_n\} \neq \emptyset$, i.e., there exists a path p from j to i with $d_{ij} < b_{ij}$. We start as in the proof of Theorem 2.3.12 for $x \geq 1$

$$
I(x) = I_1(x) + I_2(x)
$$

and similarly to (2.49), we again apply the law of total probability to $I_1(x)$

$$\begin{aligned}
I_1(x) &= \mathbb{P}\Big(\bigvee_{p\in P_{ij}} d_{ij}(p)\prod_{k\in S_p}\varepsilon_k \le b_{ij}x, U_i=\tilde U_j\bar b_{ij}\Big)\\
&= \mathbb{P}\Big(\bigvee_{p\in P_{ij}} d_{ij}(p)\prod_{k\in S_p}\varepsilon_k \le b_{ij}x, \bigvee_{p\in\{p_1,\dots,p_n\}} \bar d_{ij}(p)=\bar b_{ij}, U_i=\tilde U_j\bar b_{ij}\Big)\\
&+\mathbb{P}\Big(\bigvee_{p\in P_{ij}} d_{ij}(p)\prod_{k\in S_p}\varepsilon_k \le b_{ij}x, \bigvee_{p\in P_{ij}\setminus\{p_1,\dots,p_n\}} \bar d_{ij}(p)=\bar b_{ij}, U_i=\tilde U_j\bar b_{ij}\Big)\\
&= \mathbb{P}\Big(\bigvee_{p\in P_{ij}} d_{ij}(p)\prod_{k\in S_p}\varepsilon_k \le b_{ij}x, \bigvee_{p\in P_{ij}\setminus\{p_1,\dots,p_n\}} \bar d_{ij}(p)\le\\
&\bigvee_{p\in\{p_1,\dots,p_n\}} \bar d_{ij}(p), U_i=\tilde U_j\bar b_{ij}\Big)+\mathbb{P}\Big(\bigvee_{p\in P_{ij}} d_{ij}(p)\prod_{k\in S_p}\varepsilon_k \le b_{ij}x,\\
&\bigvee_{p\in P_{ij}\setminus\{p_1,\dots,p_n\}} \bar d_{ij}(p)>\bigvee_{p\in\{p_1,\dots,p_n\}} \bar d_{ij}(p), U_i=\tilde U_j\bar b_{ij}\Big)\\
&=: \tilde I_{11}(x)+\tilde I_{12}(x).
\end{aligned}$$

With the same arguments as in the proof of Theorem 2.3.12 we find upper and lower bounds for $\tilde I_{11}(x)$. Analogously to (2.53) and (2.54) we find

$$\begin{aligned}
&\tilde I_{11}(x) = \mathbb{P}\Big(\bigcap_{p\in P_{ij}\setminus\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p}\varepsilon_k\le\frac{b_{ij}}{d_{ij}(p)}\bigvee_{\tilde p\in\{p_1,\dots,p_n\}}\prod_{k\in S_{\tilde p}}\varepsilon_k\Big\}\bigcap\\
&\bigcap_{p\in\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p}\varepsilon_k\le x\Big\}\bigcap\bigcap_{l\in \mathrm{An}(i)}\Big\{\bigcap_{p\in P_{il}\setminus P_{ijl}}\Big\{d_{il}(p)\prod_{k\in S_p\cup\{l\}}\varepsilon_k Z_l\le\\
&b_{ij}\bigvee_{\tilde p\in\{p_1,\dots,p_n\}}\prod_{k\in S_{\tilde p}}\varepsilon_k\bigvee_{l\in\mathrm{An}(j)}\bar b_{jl}Z_l\Big\}\Big\}\Big)\\
&\ge\mathbb{P}\Big(\bigcap_{p\in P_{ij}\setminus\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p\setminus(\cup_{i=1}^n S_{p_i})}\varepsilon_k\le\frac{b_{ij}}{d_{ij}(p)}\Big\}\bigcap\bigcap_{p\in\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p}\varepsilon_k\le x\Big\}\\
&\bigcap\bigcap_{l\in\mathrm{An}(i)}\Big\{\bigcap_{p\in P_{il}\setminus P_{ijl}}\Big\{d_{il}(p)\prod_{k\in S_p\cup\{l\}\setminus(\cup_{i=1}^n S_{p_i})}\varepsilon_k Z_l\le b_{ij}\bigvee_{l\in\mathrm{An}(j)}\bar b_{jl}Z_l\Big\}\Big\}\Big)\\
&= c_1\,\mathbb{P}\Big(\bigcap_{p\in\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p}\varepsilon_k\le x\Big\}\Big)
\end{aligned}$$

and analogously to (2.57) we have

$$\begin{aligned}
\tilde I_{11}(x) &\le \mathbb{P}\Big(\bigcap_{p\in P_{ij}\setminus\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p\setminus(\cup_{i=1}^n S_{p_i})}\varepsilon_k\le\frac{b_{ij}}{d_{ij}(p)}x\Big\}\bigcap\\
&\bigcap_{p\in\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p}\varepsilon_k\le x\Big\}\\
&\bigcap\bigcap_{l\in\mathrm{An}(i)}\Big\{\bigcap_{p\in P_{il}\setminus P_{ijl}}\Big\{d_{il}(p)\prod_{k\in S_p\cup\{l\}\setminus(\cup_{i=1}^n S_{p_i})}\varepsilon_k Z_l\le b_{ij}x\bigvee_{l\in\mathrm{An}(j)}\bar b_{jl}Z_l\Big\}\Big\}\Big)\\
&= c_2(x)\,\mathbb{P}\Big(\bigcap_{p\in\{p_1,\dots,p_n\}}\Big\{\prod_{k\in S_p}\varepsilon_k\le x\Big\}\Big)
\end{aligned}$$

With the same arguments as in the previous proof, we can show that $c_1 \in (0,1)$ and $c_2(x) \to c_1$ for $x \downarrow 1$ and $\tilde{I}_{12}(x) = o(\tilde{I}_{11}(x))$ and $I_2(x) = o(I_1(x))$. Hence, the result follows. If $P_{ij} \setminus \{p_1, \dots, p_n\} = \emptyset$ the result follows analogously. $\square$

Proof of Corollary 2.3.15 From Theorem 2.3.12 we have as $x \downarrow 1$,

$$\begin{aligned}
&\mathbb{P}\Big(\bigwedge_{k=0}^{n} \frac{U_i^k}{U_j^k} \le b_{ij}x\Big) = 1 - \Big(1 - \mathbb{P}\Big(\frac{U_i}{U_j} \le b_{ij}x\Big)\Big)^n \\
&= 1 - \Big(1 - c(1+o(1))\,\mathbb{P}\Big(\prod_{k\in S_p} \varepsilon_k \le x\Big)\Big)^n \\
&= 1 - \sum_{k=0}^{n} \binom{n}{k} \Big(-c(1+o(1))\,\mathbb{P}\Big(\prod_{k\in S_p} \varepsilon_k \le x\Big)\Big)^k \sim c\, n\, \mathbb{P}\Big(\prod_{i=1}^{n} \varepsilon_{k_i} \le x\Big),
\end{aligned}$$

where we have used the binomial theorem and the fact that the summands for $k \ge 2$ are negligible when n is fixed. $\square$

Proof of Theorem 2.3.16. We give a proof for $P_{ij} \setminus \{p_1\} \ne \emptyset$ and $P_{ml} \setminus \{p_2\} \ne \emptyset$, i.e., p_1 and p_2 are not the only paths from j to i and from l to m, respectively. All other cases follow analogously. By the law of total probability, we have

$$\begin{aligned}
&\mathbb{P}\Big(\frac{U_i}{U_j} \le b_{ij}x_1, \frac{U_m}{U_l} \le b_{ml}x_2\Big) \\
&= \mathbb{P}\Big(\prod_{k\in S_{p_1}} \varepsilon_k \le x_1, \prod_{k\in S_{p_2}} \varepsilon_k \le x_2, U_i = U_j b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k, U_m = U_l b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k\Big) \\
&+ \mathbb{P}\Big(\prod_{k\in S_{p_1}} \varepsilon_k \le x_1, \frac{U_m}{U_l} \le b_{ml}x_2, U_i = U_j b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k, U_m \ne U_l b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k\Big) \\
&+ \mathbb{P}\Big(\frac{U_i}{U_j} \le b_{ij}x_1, \prod_{k\in S_{p_2}} \varepsilon_k \le x_2, U_i \ne U_j b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k, U_m = U_l b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k\Big) \\
&+ \mathbb{P}\Big(\frac{U_i}{U_j} \le b_{ij}x_1, \frac{U_m}{U_l} \le b_{ml}x_2, U_i \ne U_j b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k, U_m \ne U_l b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k\Big) \\
&=: I_1(x_1,x_2) + I_2(x_1,x_2) + I_3(x_1,x_2) + I_4(x_1,x_2)
\end{aligned}$$

We first consider $I_1(x_1,x_2)$. Observe that for $I_{11}(x)$ defined in (2.50) we have by (2.66)

$$I_{11}(x) = \mathbb{P}\Big(\prod_{k\in S_{p_{\max}}} \varepsilon_k \le x, U_i = U_j b_{ij} \prod_{k\in S_{p_{\max}}} \varepsilon_k\Big)$$

and hence, $I_1(x_1,x_2)$ is the bivariate extension to $I_{11}(x)$. For this reason, we can follow the proof of Theorem 2.3.12 at (2.52), we again find upper and lower bounds based on the decomposition

$$\Big\{ \bigvee_{p\in P_{ij}\backslash\{p_1\}} \prod_{k\in S_p} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)} \prod_{k\in S_{p_1}} \varepsilon_k \Big\} \bigcap \Big\{ \bigvee_{p\in P_{ml}\backslash\{p_2\}} \prod_{k\in S_p} \varepsilon_k \le \frac{b_{ml}}{d_{ml}(p)} \prod_{k\in S_{p_2}} \varepsilon_k \Big\} \bigcap$$
$$\Big\{ \prod_{k\in S_{p_1}} \varepsilon_k \le x_1 \Big\} \bigcap \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\backslash P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_p\cup\{n\}} \varepsilon_k Z_n \le$$
$$b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k \bigvee_{n\in \mathrm{An}(j)} \bar{b}_{jn} Z_n \Big\}\Big\} \bigcap \Big\{ \prod_{k\in S_{p_2}} \varepsilon_k \le x_2 \Big\} \bigcap$$
$$\bigcap_{n\in \mathrm{An}(m)} \Big\{ \bigcap_{p\in P_{mn}\backslash P_{mln}} \Big\{ d_{mn}(p) \prod_{k\in S_p\cup\{n\}} \varepsilon_k Z_n \le b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k \bigvee_{n\in \mathrm{An}(l)} \bar{b}_{ln} Z_n \Big\}\Big\}.$$

Now for three paths p, p_1 and p_2 and a node i we denote

$$S_{p+i\backslash p_1+p_2} := (S_p \cup \{i\}) \setminus (S_{p_1} \cup S_{p_2}) \quad \text{and} \quad S_{p\backslash p_1+p_2} := S_p \setminus (S_{p_1} \cup S_{p_2}).$$

On the set $\{\prod_{k\in S_{p_1}} \varepsilon_k \le x_1\} \cap \{\prod_{k\in S_{p_2}} \varepsilon_k \le x_2\}$ we have for $x_1, x_2 > 1$, since $\varepsilon > 1$,

$$\Big\{ \bigvee_{p\in P_{ij}\backslash\{p_1\}} \prod_{k\in S_p} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)} \prod_{k\in S_{p_1}} \varepsilon_k \Big\}$$
$$= \Big\{ \bigvee_{p\in P_{ij}\backslash\{p_1\}} \prod_{k\in S_p\backslash S_{p_1}} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)} \prod_{k\in S_{p_1}\backslash S_p} \varepsilon_k \Big\}$$
$$\supseteq \Big\{ \bigvee_{p\in P_{ij}\backslash\{p_1\}} \prod_{k\in S_p\backslash S_{p_1}} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)} \Big\} \supseteq \Big\{ \bigvee_{p\in P_{ij}\backslash\{p_1\}} \prod_{k\in S_{p\backslash p_1+p_2}} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)x_2} \Big\}$$

as well as

$$\bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\backslash P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_p\cup\{n\}} \varepsilon_k Z_n \le b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k \bigvee_{n\in \mathrm{An}(j)} \bar{b}_{jn} Z_n \Big\}\Big\}$$
$$= \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\backslash P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in (S_p\cup\{n\})\backslash S_{p_1}} \varepsilon_k Z_n \le b_{ij} \prod_{k\in S_{p_1}\backslash S_p} \varepsilon_k$$
$$\bigvee_{n\in \mathrm{An}(j)} \bar{b}_{jn} Z_n \Big\}\Big\} \supseteq \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\backslash P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in (S_p\cup\{n\})\backslash S_{p_1}} \varepsilon_k Z_n \le$$
$$b_{ij} \bigvee_{n\in \mathrm{An}(j)} \bigvee_{\tilde{p}\in P_{jn}} d_{jn}(\tilde{p}) \prod_{k\in S_{\tilde{p}+n\backslash p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\}$$
$$\supseteq \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\backslash P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_{p+n\backslash p_1+p_2}} \varepsilon_k Z_n \le$$
$$\frac{b_{ij}}{x_2} \bigvee_{n\in \mathrm{An}(j)} \bigvee_{\tilde{p}\in P_{jn}} d_{jn}(\tilde{p}) \prod_{k\in S_{\tilde{p}+n\backslash p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\}.$$

Therefore,

$$\begin{aligned}
&I_1(x_1,x_2) \geq \mathbb{P}\Big(\Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)x_2}\Big\} \bigcap \Big\{ \bigvee_{p\in P_{ml}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \\
&\varepsilon_k \leq \frac{b_{ml}}{d_{ml}(p)x_1}\Big\} \bigcap \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\setminus P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \varepsilon_k Z_n \leq \frac{b_{ij}}{x_2} \bigvee_{n\in\mathrm{An}(j)} \\
&\bigvee_{\tilde p\in P_{jn}} d_{jn}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\} \bigcap \bigcap_{n\in\mathrm{An}(m)} \Big\{ \bigcap_{p\in P_{mn}\setminus P_{mln}} \Big\{ d_{mn}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \\
&\varepsilon_k Z_n \leq \frac{b_{ml}}{x_1} \bigvee_{n\in\mathrm{An}(l)} \bigvee_{\tilde p\in P_{ln}} d_{ln}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\} \\
&\bigcap \Big\{ \prod_{k\in S_{p_1}} \varepsilon_k \leq x_1 \Big\} \bigcap \Big\{ \prod_{k\in S_{p_2}} \varepsilon_k \leq x_2 \Big\}\Big) \\
&=: c_3(x_1,x_2)\, \mathbb{P}\Big(\Big\{ \prod_{k\in S_{p_1}} \varepsilon_k \leq x_1 \Big\} \bigcap \Big\{ \prod_{k\in S_{p_2}} \varepsilon_k \leq x_2 \Big\}\Big).
\end{aligned}$$

For an upper bound, observe that on $\{\prod_{k\in S_{p_1}} \varepsilon_k \leq x_1\} \cap \{\prod_{k\in S_{p_2}} \varepsilon_k \leq x_2\}$ we have

$$\begin{aligned}
&\Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_p} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} \prod_{k\in S_{p_1}} \varepsilon_k \Big\} = \Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_p\setminus S_{p_1}} \varepsilon_k \leq \frac{b_{ij}}{d_{ij}(p)} \\
&\prod_{k\in S_{p_1}\setminus S_p} \varepsilon_k \Big\} \subseteq \Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_p\setminus S_{p_1}} \varepsilon_k \leq \frac{b_{ij}x_1}{d_{ij}(p)} \Big\} \\
&\subseteq \Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \varepsilon_k \leq \frac{b_{ij}x_1}{d_{ij}(p)} \Big\}
\end{aligned}$$

as well as

$$\begin{aligned}
&\bigcap_{n\in\mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\setminus P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_p\cup\{n\}} \varepsilon_k Z_n \leq b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k \bigvee_{n\in\mathrm{An}(j)} \bar b_{jn} Z_n \Big\}\Big\} \\
&\subseteq \bigcap_{n\in\mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\setminus P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in (S_p\cup\{n\})\setminus S_{p_1}} \varepsilon_k Z_n \leq \\
&\quad b_{ij}x_1x_2 \bigvee_{n\in\mathrm{An}(j)} \bigvee_{\tilde p\in P_{jn}} d_{jn}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\} \\
&\subseteq \bigcap_{n\in\mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\setminus P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \varepsilon_k Z_n \leq \\
&\quad b_{ij}x_1x_2 \bigvee_{n\in\mathrm{An}(j)} \bigvee_{\tilde p\in P_{jn}} d_{jn}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\}.
\end{aligned}$$

For this reason,

$$I_1(x_1,x_2) \leq c_4(x_1,x_2)\, \mathbb{P}\Big(\Big\{ \prod_{k\in S_{p_1}} \varepsilon_k \leq x_1 \Big\} \bigcap \Big\{ \prod_{k\in S_{p_2}} \varepsilon_k \leq x_2 \Big\}\Big),$$

with

$$c_4(x_1,x_2) := \mathbb{P}\Big(\Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \varepsilon_k \le \frac{b_{ij}x_1}{d_{ij}(p)}\Big\} \bigcap \Big\{ \bigvee_{p\in P_{ml}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \varepsilon_k \le \frac{b_{ml}x_2}{d_{ml}(p)}\Big\} \bigcap \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\setminus P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \varepsilon_k Z_n \le b_{ij}x_1x_2 \bigvee_{n\in \mathrm{An}(j)} \bigvee_{\tilde p\in P_{jn}} d_{jn}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\} \bigcap \bigcap_{n\in \mathrm{An}(m)} \Big\{ \bigcap_{p\in P_{mn}\setminus P_{mln}} \Big\{ d_{mn}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \varepsilon_k Z_n \le b_{ml}x_1x_2 \bigvee_{n\in \mathrm{An}(l)} \bigvee_{\tilde p\in P_{ln}} d_{ln}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\}\Big).$$

Since all random variables are continuous, $c_4(x_1,x_2)$ tends to $c_3(x_1,x_2)$ for $x_1,x_2 \downarrow 1$, and

$$c_3(x_1,x_2) \le c \le c_4(x_1,x_2)$$

with

$$c := \mathbb{P}\Big(\Big\{ \bigvee_{p\in P_{ij}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \varepsilon_k \le \frac{b_{ij}}{d_{ij}(p)}\Big\} \bigcap \Big\{ \bigvee_{p\in P_{ml}\setminus\{p_1\}} \prod_{k\in S_{p\setminus p_1+p_2}} \varepsilon_k \le \frac{b_{ml}}{d_{ml}(p)}\Big\} \bigcap \bigcap_{n\in \mathrm{An}(i)} \Big\{ \bigcap_{p\in P_{in}\setminus P_{ijn}} \Big\{ d_{in}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \varepsilon_k Z_n \le b_{ij} \bigvee_{n\in \mathrm{An}(j)} \bigvee_{\tilde p\in P_{jn}} d_{jn}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\} \bigcap \bigcap_{n\in \mathrm{An}(m)} \Big\{ \bigcap_{p\in P_{mn}\setminus P_{mln}} \Big\{ d_{mn}(p) \prod_{k\in S_{p+n\setminus p_1+p_2}} \varepsilon_k Z_n \le b_{ml} \bigvee_{n\in \mathrm{An}(l)} \bigvee_{\tilde p\in P_{ln}} d_{ln}(\tilde p) \prod_{k\in S_{\tilde p+n\setminus p_1+p_2}} \varepsilon_k Z_n \Big\}\Big\}\Big).$$

Since $i \neq m$ we can use the same arguments as for c_1 in the proof of Theorem 2.3.12 to show that $c > 0$. Therefore, we only need to show that $I_i(x_1,x_2) = o(I_1(x_1,x_2))$ for $i \in \{2,3,4\}$. It is obvious that $I_2(x_1,x_2) = o(I_1(x_1,x_2))$ implies the other two cases. Using the same arguments from the proof of Theorem 2.3.12 regarding $I_2(x) = o(I_1(x))$ and $I_{12}(x) = o(I_{11})(x)$, the result follows. □

2.7.2 Proofs of Section 2.4

Proof of Proposition 2.4.3 We first consider the convergence of the simple minimum ratio $\bigwedge_{k=1}^n (U_i^k/U_j^k)$. For $j = i$ the result is obvious. Moreover, for $j \in \mathrm{an}(i)$ we have with Lemma 2.3.8 d), for $x > 0$,

$$\lim_{n\to\infty} \mathbb{P}\Big(\Big| \bigwedge_{k=1}^n \frac{U_i^k}{U_j^k} - b_{ij}\Big| > x\Big) = \lim_{n\to\infty} \Big(\mathbb{P}\Big(\frac{U_i}{U_j} - b_{ij} > x\Big)\Big)^n = 0,$$

showing that the minimum ratio converges for $n \to \infty$ in probability to b_{ij}. Since $\bigwedge_{k=1}^n U_i^k/U_j^k$ is non-increasing, it converges almost surely. For $j \notin \mathrm{an}(i)$ we have $b_{ij} = 0$ and the same result holds. Therefore, the estimators (2.27), (2.30) and (2.31) converge almost surely. Considering the inequality (2.29) for the estimator (2.28), this estimator as well converges almost surely. Finally using (2.33) and (2.34), the same also holds for the estimator (2.36). □

Proof of Lemma 2.4.4 Assume first that the output $\hat{\boldsymbol{B}}$ is not idempotent. Then there exists an entry $\hat{b}_{ij}$ in $\hat{\boldsymbol{B}}$ such that $\hat{b}_{ij} < \hat{b}_{kj}\hat{b}_{ik}$. Therefore, since the input matrix $\check{\boldsymbol{B}}$ is idempotent, we have $\hat{b}_{ij} = 0$ while $\hat{b}_{kj} > 0$ and $\hat{b}_{ik} > 0$. This is a contradiction to the if-condition on line 9 in the algorithm.

Now assume that there is an idempotent matrix $\boldsymbol{B}'$ that preserves all values that are larger than δ_1 but contains more zero entries. Then there is an entry b'_{ij} such that $b'_{ij} = 0$ while $\hat{b}_{ij} > 0$. Since $\hat{b}_{ij} > 0$ there must be some $k \in \{j+1, \dots, i-1\}$ such that $\hat{b}_{kj} \notin S$ and $\hat{b}_{ik} \notin S$, otherwise we would have set $\hat{b}_{ij}$ equal to zero. Because we sort pairs (j,i) by distance and (j,k) and (k,i) both have smaller distance it must also hold that both, $\hat{b}_{kj}$ and $\hat{b}_{ik}$ are strictly greater than zero. In comparison, since $\boldsymbol{B}'$ is idempotent, either b'_{kj} or b'_{ik} is equal to zero.

Therefore, we have $b'_{kj} = 0$ while $\hat{b}_{kj} > 0$ or $b'_{ik} = 0$ while $\hat{b}_{ik} > 0$. In both cases, the distance compared to the pair (j,i) is decreased. Repeating this argument we can assume that $(j-i) = 1$. This, however, leads to a contradiction since $\hat{b}_{ij}$ is set to zero for all pairs (j,i) of distance one if $\check{b}_{ij} < \delta_1$. □

2.7.3 Proofs of Section 2.5

Proof of Lemma 2.5.1 By Theorem 2.3.12 we get for $x \downarrow 1$,

$$\begin{aligned}\lim_{t\downarrow 0} \frac{\mathbb{P}(\ln(U_i/U_j) - \ln(b_{ij}) \le tx)}{\mathbb{P}(\ln(U_i/U_j) - \ln(b_{ij}) \le t)} &= \lim_{t\downarrow 0} \frac{\mathbb{P}(U_i/U_j \le b_{ij}\exp(tx))}{\mathbb{P}(U_i/U_j \le b_{ij}\exp(t))} \\ = \lim_{t\downarrow 0} \frac{c\,\mathbb{P}\Big(\prod_{k\in S_p}\varepsilon_k \le \exp(tx)\Big)}{c\,\mathbb{P}\Big(\prod_{k\in S_p}\varepsilon_k \le \exp(t)\Big)} &= \lim_{t\downarrow 0} \frac{\mathbb{P}\Big(\sum_{k\in S_p}\ln(\varepsilon_k) \le tx\Big)}{\mathbb{P}\Big(\sum_{k\in S_p}\ln(\varepsilon_k) \le t\Big)} = x^{\zeta(p)\alpha}\end{aligned}$$

for $\zeta(p) = |S_p|$ by Corollary 2.2.6 a) and the fact that $\ln(\varepsilon_k) \in RV^0_\alpha$. □

For the proof of Theorem 2.5.2 we need the following distribution family.

Definition 2.7.1. A positive random variable Y is *Fréchet distributed with shape* $\alpha > 0$ *and scale* $s > 0$ and we write $Y \sim$ Fréchet(α, s) if the distribution function of Y is given by

$$\Phi_{\alpha,s}(x) = \exp\left(-\left(\frac{x}{s}\right)^{-\alpha}\right), \quad x > 0.$$

The proof of Theorem 2.5.2 is divided into a proof of the one-dimensional marginal limit distributions, followed by the proof of the multidimensional result. We start with the one-dimensional limits.

Proposition 2.7.2. *Let $\boldsymbol{U}$ be a recursive ML vector with propagating noise on a DAG $\mathcal{D}$ as defined in* (2.17) *and assume that the path $p := [j \to \cdots \to i]$ from j*

to i is generic. Assume further that $\tilde{\varepsilon} = \ln(\varepsilon) \in RV_\alpha^0$. For the node set S_p choose $a_n \sim F^{\leftarrow}_{\sum_{k\in S_p}\tilde{\varepsilon}_k}(1/n)$ as $n \to \infty$. Let $\boldsymbol{U}^1, \dots, \boldsymbol{U}^n$ be an i.i.d. sample from $\boldsymbol{U}$. Then

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{1}{a_n b_{ij}}\Big(\bigwedge_{k=1}^{n} U_i^k/U_j^k - b_{ij}\Big) \le x\right) = \Psi_{(\zeta(p)\alpha, c^{1/(\zeta(p)\alpha)})}(x), \quad x > 0,$$

for the same constant c as in Theorem 2.3.12, and $\Psi_{\alpha,s}$ denotes the Weibull distribution from Definition 2.2.2 with $x_L = 0$.

Proof. Define $X := \ln(U_i/U_j) - \ln(b_{ij})$ with distribution function F_X. Then by Lemma 2.5.1, $X \in RV^0_{\zeta(p)\alpha}$, which implies that $1/X \in RV^\infty_{\zeta(p)\alpha}$. Using e.g. Theorem 3.3.7 of Embrechts et al. [1997], for

$$a_{1/X}(n) \sim F^{\leftarrow}_{1/X}(1 - 1/n) \sim 1/F^{\leftarrow}_X(1/n) \to \infty, \quad n \to \infty, \tag{2.67}$$

we get

$$\begin{aligned}\lim_{n\to\infty}\mathbb{P}\Big(\bigvee_{k=1}^{n}\Big(\frac{1}{X}\Big)^k \le a_{1/X}(n)x\Big) \qquad &= \lim_{n\to\infty}\mathbb{P}\Big(\bigwedge_{k=1}^{n} X^k \ge \frac{1}{a_{1/X}(n)x}\Big)\\ = \Phi_{\zeta(p)\alpha,1}(x), \quad x > 0,\end{aligned}$$

which implies by the continuity of X,

$$\lim_{n\to\infty}\mathbb{P}\Big(\bigwedge_{k=1}^{n} X^k \le \frac{x}{a_{1/X}(n)}\Big) = 1 - \Phi_{\zeta(p)\alpha,1}(1/x), \quad x > 0. \tag{2.68}$$

Choose now

$$\tilde{a}_n \sim F^{\leftarrow}_X(1/n) \sim 1/a_{1/X}(n) \downarrow 0, \quad n \to \infty. \tag{2.69}$$

Hence, we have with (2.68) and (2.69) by Lemma 2.5.1,

$$\lim_{n\to\infty}\mathbb{P}\Big(\frac{1}{\tilde{a}_n}\Big(\bigwedge_{k=1}^{n}\ln(U_i^k/U_j^k) - \ln(b_{ij})\Big) \le x\Big) = 1 - \Phi_{\zeta(p)\alpha,1}(1/x), \quad x > 0. \tag{2.70}$$

Recall from Theorem 2.3.12 and the regular variation of $\tilde{\varepsilon}$, that for the same c as defined in Theorem 2.3.12 we have

$$\begin{aligned}F_X(x) &\sim c\,\mathbb{P}\Big(\sum_{k\in S_p}\tilde{\varepsilon}_k \le x\Big) \sim \mathbb{P}\Big(\sum_{k\in S_p}\tilde{\varepsilon}_k \le xc^{1/(\zeta(p)\alpha)}\Big)\\ &= F_{\sum_{k\in S_p}\tilde{\varepsilon}_k}(xc^{1/(\zeta(p)\alpha)}), \quad x \downarrow 0,\end{aligned}$$

which implies that $1/n \sim F_X(\tilde{a}_n) \sim F_{\sum_{k\in S_p}\tilde{\varepsilon}_k}(\tilde{a}_n c^{1/(\zeta(p)\alpha)})$. For the generalized inverses this implies that

$$\tilde{a}_n \sim F^{\leftarrow}_X(1/n) \sim c^{-1/(\zeta(p)\alpha)} F^{\leftarrow}_{\sum_{k\in S_p}\tilde{\varepsilon}_k}(1/n) \sim c^{-1/(\zeta(p)\alpha)} a_n, \quad n \to \infty.$$

From this we find

$$\mathbb{P}\Big(\frac{1}{\tilde{a}_n}\Big(\bigwedge_{k=1}^{n}\ln(U_i^k/U_j^k) - \ln(b_{ij})\Big) \le x\Big) = \mathbb{P}\Big(\bigwedge_{k=1}^{n} U_i^k/U_j^k \le \exp(\tilde{a}_n x) b_{ij}\Big), x > 0.$$

A Taylor expansion around 0 yields $\exp(\tilde{a}_n x) = 1 + \tilde{a}_n x(1 + o(1))$ as $n \to \infty$, because $\tilde{a}_n \downarrow 0$. Since for $x > 0$,

$$\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{\tilde{a}_n b_{ij}}\Big(\bigwedge_{k=1}^{n} U_i^k/U_j^k - b_{ij}\Big) \le x\Big) = \lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n b_{ij}}\Big(\bigwedge_{k=1}^{n} U_i^k/U_j^k - b_{ij}\Big)$$
$$\le c^{1/(\zeta(p)\alpha)}x\Big),$$

we obtain with (2.70)

$$\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n b_{ij}}\Big(\bigwedge_{k=1}^{n} U_i^k/U_j^k - b_{ij}\Big) \le x\Big) = 1 - \Phi_{\zeta(p)\alpha, c^{-1/(\zeta(p)\alpha)}}(1/x)$$
$$= \Psi_{\zeta(p)\alpha, c^{1/(\zeta(p)\alpha)}}(x),$$

which proves the assertion. □

Now we can prove Theorem 2.5.2.

Proof of Theorem 2.5.2 As we shall find asymptotic independence of estimates between different node pairs, it suffices to prove the bivariate result.

We first simplify notation as follows. Assume pairs of nodes $(j,i) \neq (l,m)$ and denote the generic paths $p_1 = p_{ij}$ with node set S_{p_1} and $p_2 = p_{ml}$ with node set S_{p_2}, respectively. Further denote $a_n^1 = a_n^{(ij)}$ and $a_n^2 = a_n^{(ml)}$.

By Proposition 2.7.2 we have

$$\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\Big(\bigwedge_{k=1}^{n} U_i^k/U_j^k - b_{ij}\Big) \ge x_1\Big) = \Phi_{\zeta(p_1)\alpha, c_1^{-1/(\zeta(p_1)\alpha)}}(1/x_1)$$
$$= \exp\Big(-\frac{x_1^{\zeta(p_1)\alpha}}{c_1}\Big),$$

for $x_1 > 0$. Since $\lim_{n\to\infty}(1 - \frac{a}{n})^n = \exp(-a)$ for $a \in \mathbb{R}$, we get by independence of the ratios U_i^k/U_j^k for $k = 1, \ldots, n$ as $n \to \infty$,

$$\mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\Big(U_i/U_j - b_{ij}\Big) \le x_1\Big) = \frac{x_1^{\zeta(p_1)\alpha}}{c_1 n}(1 + o(1)), \quad x_1 > 0. \tag{2.71}$$

With the same argument, we have

$$\mathbb{P}\Big(\frac{1}{a_n^2 b_{ml}}\Big(U_m/U_l - b_{ml}\Big) \le x_2\Big) = \frac{x_2^{\zeta(p_2)\alpha}}{c_2 n}(1 + o(1)), \quad x_2 > 0. \tag{2.72}$$

Therefore, we have on the one hand

$$\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\bigwedge_{k=1}^{n}\Big(U_i^k/U_j^k - b_{ij}\Big) \ge x_1\Big)\mathbb{P}\Big(\frac{1}{a_n^2 b_{ml}}\bigwedge_{k=1}^{n}\Big(U_m^k/U_l^k - b_{ml}\Big) \ge x_2\Big)$$
$$= \exp\Big(-\frac{x_1^{\zeta(p_1)\alpha}}{c_1}\Big)\exp\Big(-\frac{x_2^{\zeta(p_2)\alpha}}{c_2}\Big) = \exp\Big(-\frac{x_1^{\zeta(p_1)\alpha}}{c_1} - \frac{x_2^{\zeta(p_2)\alpha}}{c_2}\Big),\ x_1, x_2 > 0, \tag{2.73}$$

whereas, on the other hand, we have by independence of the bivariate ratios $(U_i^k/U_j^k, U_m^k/U_l^k)$ for $k = 1, \ldots, n$,

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}} \bigwedge_{k=1}^n \Big(U_i^k/U_j^k - b_{ij}\Big) \geq x_1, \frac{1}{a_n^2 b_{ml}} \bigwedge_{k=1}^n \Big(U_m^k/U_l^k - b_{ml}\Big) \geq x_2\Big) \\
&= \lim_{n\to\infty} \Big\{\mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\Big(U_i^k/U_j^k - b_{ij}\Big) \geq x_1, \frac{1}{a_n^2 b_{ml}}\Big(U_m^k/U_l^k - b_{ml}\Big) \geq x_2\Big)\Big\}^n \\
&= \lim_{n\to\infty} \Big\{1 - \mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\Big(U_i/U_j - b_{ij}\Big) \leq x_1\Big) - \mathbb{P}\Big(\frac{1}{a_n^2 b_{ml}}\Big(U_m/U_l - b_{ml}\Big) \leq x_2\Big) \\
&+ \mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\Big(U_i/U_j - b_{ij}\Big) \leq x_1, \frac{1}{a_n^2 b_{ml}}\Big(U_m/U_l - b_{ml}\Big) \leq x_2\Big)\Big\}^n. \qquad (2.74)
\end{aligned}$$

By (2.64) we have $U_i \geq b_{ij} \prod_{k\in S_{p_1}} \varepsilon_k U_j$ and $U_m \geq b_{ml} \prod_{k\in S_{p_2}} \varepsilon_k U_l$, which implies

$$\begin{aligned}
&\mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}}\Big(U_i/U_j - b_{ij}\Big) \leq x_1, \frac{1}{a_n^2 b_{ml}}\Big(U_m/U_l - b_{ml}\Big) \leq x_2\Big) \qquad (2.75) \\
&\leq \mathbb{P}\Big(\frac{1}{a_n^1}\Big(\prod_{k\in S_{p_1}} \varepsilon_k - 1\Big) \leq x_1, \frac{1}{a_n^2}\Big(\prod_{k\in S_{p_2}} \varepsilon_k - 1\Big) \leq x_2\Big).
\end{aligned}$$

Since $(j,i) \neq (l,m)$, either $S_{p_1} \setminus S_{p_2} \neq \emptyset$ or $S_{p_2} \setminus S_{p_1} \neq \emptyset$ and without loss of generality, we assume $S_{p_2} \setminus S_{p_1} \neq \emptyset$. Since $\varepsilon \geq 1$ and all ε_k are independent, we get

$$\begin{aligned}
(2.75) &\leq \mathbb{P}\Big(\frac{1}{a_n^1}\Big(\prod_{k\in S_{p_1}} \varepsilon_k - 1\Big) \leq x_1, \frac{1}{a_n^2}\Big(\prod_{k\in S_{p_2}\setminus S_{p_1}} \varepsilon_k - 1\Big) \leq x_2\Big) \\
&= \mathbb{P}\Big(\frac{1}{a_n^1}\Big(\prod_{k\in S_{p_1}} \varepsilon_k - 1\Big) \leq x_1\Big)\mathbb{P}\Big(\frac{1}{a_n^2}\Big(\prod_{k\in S_{p_2}\setminus S_{p_1}} \varepsilon_k - 1\Big) \leq x_2\Big). \qquad (2.76)
\end{aligned}$$

By Corollary 2.2.6 b) we know that $(\prod_{k\in S_{p_1}} \varepsilon_k - 1) \in RV^0_{\zeta(p_1)\alpha}$. Moreover, observe that by a Taylor expansion we have $a_n^1 \sim F^{\leftarrow}_{\sum_{k\in S_p} \tilde{\varepsilon}_k}(1/n) \sim F^{\leftarrow}_{\prod_{k\in S_p} \varepsilon_k - 1}(1/n)$ as $n \to \infty$. Therefore, by Theorem 3.3.7 of Embrechts et al. [1997] as in the proof of Proposition 2.7.2, similarly to (2.70), it holds that

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n^1}\Big(\bigwedge_{t=1}^n \prod_{k\in S_{p_1}} \varepsilon_k^t - 1\Big) \geq x_1\Big) = \Phi_{\zeta(p_1)\alpha,1}(1/x_1) \\
&= \exp\Big(-x_1^{\zeta(p_1)\alpha}\Big), \quad x_1 > 0.
\end{aligned}$$

We proceed as in (2.71) and (2.72) to obtain as $n \to \infty$

$$\mathbb{P}\Big(\frac{1}{a_n^1}\Big(\prod_{k\in S_{p_1}} \varepsilon_k - 1\Big) \leq x_1\Big) = \Big(\frac{x_1^{\zeta(p_1)\alpha}}{n}\Big)(1 + o(1)), \quad x_1 > 0. \qquad (2.77)$$

Moreover, since $S_{p_2} \setminus S_{p_1} \neq \emptyset$, ε is atom-free and $a_n^2 \to 0$ as $n \to \infty$, we have

$$\mathbb{P}\Big(\frac{1}{a_n^2}\Big(\prod_{k\in S_{p_2}\setminus S_{p_1}} \varepsilon_k - 1\Big) \leq x_2\Big) = o(1), \quad x_2 > 0. \qquad (2.78)$$

Therefore, we have by (2.76), (2.77) and (2.78)

$$(2.75) = \Big(\frac{x_1^{\zeta(p_1)\alpha}}{n}\Big)(1+o(1))o(1), \quad x_1, x_2 > 0.$$

Comparing this with (2.71) and (2.72), we find that the last term in (2.74) is negligible. Hence, we obtain

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}\Big(\frac{1}{a_n^1 b_{ij}} \bigwedge_{k=1}^{n} \Big(U_i^k/U_j^k - b_{ij}\Big) \geq x_1, \frac{1}{a_n^2 b_{ml}} \bigwedge_{k=1}^{n} \Big(U_m^k/U_l^k - b_{ml}\Big) \geq x_2\Big) \\
&= \lim_{n\to\infty} \Big(1 - \frac{x_1^{\zeta(p_1)\alpha}}{c_1 n}(1+o(1)) - \frac{x_2^{\zeta(p_2)\alpha}}{c_2 n}(1+o(1))\Big)^n \\
&= \exp\Big(- \frac{x_1^{\zeta(p_1)\alpha}}{c_1} - \frac{x_2^{\zeta(p_2)\alpha}}{c_2}\Big).
\end{aligned}$$

Comparing this to (2.73) yields the result.

□

Chapter 3

Estimating a Directed Tree for Extremes

3.1 Introduction

Causal inference from extremes aims to discover cause and effect relations between large observed values of random variables. To understand causality of high risk variables is much needed as rare events like environmental or financial risks are often cascading through a network. Pollutants can propagate through an unseen underground waterway, causing extreme measurements at multiple locations [Leigh et al., 2019]. Credit markets might fail due to some endogenous systemic risk propagation [Rochet and Tirole, 1996]. However, it is not immediately obvious how to extend the past decades of work on causal inference [Bollen, 1989, Drton and Maathuis, 2017, Lauritzen, 1996, Maathuis et al., 2019, Pearl, 2009, Spirtes et al., 2000] for Gaussian and discrete distributions to extreme values. Since the focus is on maxima rather than averages, correlations or other bivariate measures of dependence in the center of the distribution are replaced by *extreme dependence measures* [Coles et al., 1999, Engelke and Volgushev, 2020, Larsson and Resnick, 2012, Sibuya, 1960], which are often difficult to estimate from limited data.

These extreme dependence measures are derived through asymptotic theory from generalized extreme value distributions—modeling sample extremes—or generalized Pareto distributions—modeling excesses over high thresholds. Textbook treatments can be found in [Beirlant et al., 2004, Coles et al., 2001, De Haan and Ferreira, 2007, Resnick, 1987, 2007]. A very readable review paper is Davison and Huser [2015]. The development of graphical models for extremes follows these two approaches. Max-linear causal graphical models or max-linear Bayesian networks are motivated by extreme value distributions, which are max-stable (closed with respect to taking maxima). Introduced in Gissibl and Klüppelberg [2018], they are defined via a max-linear recursively defined structural equation models on a directed acyclic graph (see Pearl et al. [2009]). Each node represents a positive random variable defined as a weighted maximum of its parent variables and an independent innovation. The multivariate distribution of a max-linear vector is not restricted to a Fréchet distribution; indeed the independent innovations can have arbitrary continuous distributions. The emphasis of the model is on its structure given by a directed graph. Nonparametric statistical inference aims at identifying the directed graphical structure regardless of the node distributions.

Engelke and Hitz [2020], on the other hand, define a new extreme conditional dependence concept for multivariate Pareto distributions and use this concept to define extreme graphical models similarly as the classical concept for densities. The multivariate distribution determines the model and has to be specified for statistical inference. Here the multivariate Hüsler-Reiss distribution plays a prominent role; see Asenova et al. [2021], Asenova and Segers [2022], Engelke and Volgushev [2020], Engelke et al. [2021], Hu et al. [2022], Rötter et al. [2022].

Motivated by extreme value theory, it is not surprising that max-linear Bayesian networks have been mostly investigated for heavy-tailed innovations: [Einmahl et al., 2017, Section 3.3] consider tail dependence functions for i.i.d. Fréchet innovations. Gissibl et al. [2018] investigate tail dependence for i.i.d. regularly varying innovations, and Klüppelberg and Krali [2021] the scaling properties of the same model. Asenova et al. [2021] and Segers [2019] investigate regularly varying Markov trees, and more recently, Asenova and Segers [2022] investigate a new max-linear graphical model on trees of transitive tournaments.

3.1.1 The Extremal River Problem

The relevance of extremal graphical models for multivariate distributions has been validated on several data sets, prominently on the Upper Danube river network. The goal is to recover a river network from *only extreme flow* measured at a set V of stations, *without* any information on the stations' location. We refer to it as the *Extremal River Problem*. Here, the true river network serves as the 'gold standard', allowing one to verify the performance of the proposed estimator. Success in solving the Extreme River Problem can translate to new solutions to the *contaminant tracing* challenge in hydrology [Leigh et al., 2019, McGrane, 2016, Rodriguez-Perez et al., 2020, Ver Hoef and Peterson, 2010, Ver Hoef et al., 2006, Wolf et al., 2012]. There, one needs an inexpensive method to trace pollutants or chemical constituents transported by a complex and unknown underground waterway that is prohibitive to model or survey with traditional fluid mechanics methods [Anderson et al., 2015]. Recent advances point towards an imminent data explosion [Bartos et al., 2018, Mao et al., 2019], where pollutants exceeding certain thresholds can be detected via a sensor network. Thus, contaminant tracing with sensors data is a version of the Extremal River Problem without the gold standard, where the network is truly unknown.

The Extremal River Problem with test data given by river discharges of the Upper Danube river network has proven to be challenging and very stimulating for extreme value theory, with each paper taking a different technique. The data have been preprocessed in Asadi et al. [2015] and are available in the `R` package `graphicalExtremes` [Engelke et al., 2019].

The preprocessed data have been analyzed in a number of publications with focus on modeling extreme dependence: flow- and spatial dependence [Asadi et al., 2015] and undirected graphical models for extremes [Engelke and Hitz, 2020, Engelke et al., 2021, Hu et al., 2022, Gong et al., 2022, Rötter et al., 2022]. In a first paper, Engelke and Hitz [2020] returned a highly accurate but *undirected* graph, followed by publications using new models and applying different methods for reconstructing the undirected graph.

Our focus is on causality in extremes, modelled by a directed tree; hence, a large value at node j causes a large value at node i, whenever there is an edge from j to i. Similar problems have also been considered modeling causal extreme dependence by expected quantile scores in [Mhalla et al., 2020] and by causal dependence coefficients in [Gnecco et al., 2021]. Gnecco et al. [2021] correctly recovered the causal order of 12 nodes out of 31, but did not learn the entire river network, while Mhalla et al. [2020] focused on flow-connections and did well at detecting nodes connected by a directed path; see Figure 7 in Mhalla et al. [2020]. These two publications have slightly different notions of causality; we discuss this in Section 4.

3.1.2 Main contributions and structure of the chapter

The novelty of this chapter lies in several directions.

1. We suggest a new algorithm QTree to recover causality in extremes, where causality is modelled by a directed tree. The advantage of the proposed max-linear Bayesian tree is a structural model, which does not require to specify multivariate distributions. Moreover, no normalization of the data to standard Fréchet or Gumbel is needed.
2. The QTree algorithm estimates a pair-wise score matrix W, and applies Chu–Liu/-Edmonds' algorithm to output a root-directed spanning tree of optimum score. The algorithm is simple, using properties of a max-linear Bayesian tree, and has a stabilizing subsampling procedure that is based on bootstrap aggregation.
3. We prove strong consistency of the estimated trees in an extreme value setting with possible noise as the sample size tends to infinity. This proof is based on a new variational argument to account for noise in the data.
4. We analyze three new data sets from the Lower Colorado river network in Texas. We also show by a simulation study that QTree is robust with respect to different dependence structures (given by edge-weights) and different node distributions entailed from different innovation distributions.

QTree performs extremely well on real-world data sets. Algorithm 5 achieves almost perfect recovery of the Upper Danube network. In addition to the Upper Danube, we further test QTree on three sectors of the Lower Colorado river network in Texas. These are much more challenging data sets. The Colorado river network suffers from severe drought, extreme flooding, and sensors failure, with up to 36.9% missing data (the Upper Danube has none). These challenges make recovering the Lower Colorado network much closer to the trace contaminant challenge. Remarkably, on all three sectors of the Lower Colorado, QTree Algorithm 5 also achieves almost perfect recovery (cf. Section 3.4).

Beyond hydrology, QTree can be applied to cause and effect detection in every high risk problem assuming that the network is a root-directed tree. At a high level, QTree aims to fit a *max-linear Bayesian tree* to the data. Max-linear Bayesian networks have recently emerged as a suitable directed graphical model for causality in extremes [Améndola et al., 2022, Buck and Klüppelberg, 2021, Gissibl, 2018, Gissibl and Klüppelberg, 2018], however, existing methods for learning them aim to learn the model parameters and thus are highly sensitive to model misspecifications; see Buck and Klüppelberg [2021], Gissibl [2018], Gissibl et al. [2021]; Gissibl et al. [2018], Klüppelberg and Krali [2021], Klüppelberg and Lauritzen [2020]. In particular, they do not perform well on the Upper Danube data set. In contrast, QTree relies on *qualitative* aspects of the max-linear Bayesian network model to score each potential edge independently, and then applies Chu–Liu/Edmonds' algorithm to return an optimal root-directed spanning tree; see e.g. Gabow et al. [1986], Section 3. We detail the algorithm and the intuition behind it in Section 3.2. Note that QTree heavily relies on the assumption that there are sufficiently many extreme observations, and that the signal has heavier tail than the noise. Assuming that the data come from a noisy max-linear Bayesian network with appropriate signal-to-noise ratio, we prove that the tree output by QTree is strongly consistent (cf. Theorem 3.2.4).

QTree is very flexible, has only two tuning parameters, and is very efficient. It runs in time $O(n|V|^2)$, where n is the number of observations and $|V|$ is the number of nodes. QTree maximizes the information available from missing data since at each step it only utilizes the data projected onto two coordinates. QTree is implemented as a plug-and-play package in `Python` [Tran, 2021] at

`https://github.com/princengoc/qtree`

which includes all data and codes to produce the results and figures in this chapter.

This chapter is organized as follows. We introduce QTree (Algorithm 4) and `auto-tuned` QTree (Algorithm 5) in Section 3.2 and discuss its intuition supported by preliminary simulation results. In Section 3.3, we present the data sets, discuss their specific challenges and describe the data preprocessing steps. In Section 3.4, we present the estimation results of QTree and analyze the performance of the automated parameter selection. Here we also compare different algorithms in the literature with ours. In Section 3.5, we test the limits of QTree by a small simulation study. Section 3.6 concludes with a summary. The section of Supplementary Material includes the proof of the Consistency Theorem (Theorem 3.2.4) as well as supplemental figures to Sections 3.4 and Section 3.5.

Notations. Estimators are compared based on standard metrics in causal inference [Zheng et al., 2018]: normalized structural Hamming distance (nSHD), false discovery rate (FDR), false positive rate (FPR), and true positive rate (TPR). All of these metrics lie between 0 and 1. We recall their definitions here. Let $\mathcal{G}$ be the true graph and $\hat{\mathcal{G}}$ an estimated graph. The *structural Hamming distance* $\mathrm{SHD}(\mathcal{G}, \hat{\mathcal{G}})$ between $\mathcal{G}$ and $\hat{\mathcal{G}}$ is the minimum number of edge additions, deletions and reversals to obtain $\mathcal{G}$ from $\hat{\mathcal{G}}$. Denote $E(\mathcal{G})$ and $E(\hat{\mathcal{G}})$ the set of edges in $\mathcal{G}$ and $\hat{\mathcal{G}}$, respectively. Note that $|E(\hat{\mathcal{G}}) \setminus E(\mathcal{G})|$ is the number of edges in $\hat{\mathcal{G}}$ that are not in $\mathcal{G}$, while $|E(\hat{\mathcal{G}}) \cap E(\mathcal{G})|$ is the number of correctly estimated edges. We then have

$$\begin{aligned}
\mathrm{nSHD}(\hat{\mathcal{G}}, \mathcal{G}) &:= \frac{\mathrm{SHD}(\hat{\mathcal{G}}, \mathcal{G})}{|E(\hat{\mathcal{G}})| + |E(\mathcal{G})|}, & \mathrm{FDR}(\hat{\mathcal{G}}, \mathcal{G}) &:= \frac{|E(\hat{\mathcal{G}}) \setminus E(\mathcal{G})|}{|E(\hat{\mathcal{G}})|}, \qquad (3.1) \\
\mathrm{FPR}(\hat{\mathcal{G}}, \mathcal{G}) &:= \frac{|E(\hat{\mathcal{G}}) \setminus E(\mathcal{G})|}{|V| \times (|V| - 1) - |E(\mathcal{G})|}, & \mathrm{TPR}(\hat{\mathcal{G}}, \mathcal{G}) &:= \frac{|E(\hat{\mathcal{G}}) \cap E(\mathcal{G})|}{|E(\mathcal{G})|}.
\end{aligned}$$

The performance of an algorithm is better the smaller the first three metrics are and the larger TPR is. We shall use this throughout Section 3.4.

3.2 The algorithm

3.2.1 The data generation model

Throughout we assume data on a root-directed spanning tree $\mathcal{T}$ on V nodes. That is, each node $i \in V$ except the root r has exactly one child, the root r has none, and there is a path from every node $i \neq r$ to r. In the Extremal River Problem, our goal is to recover the unknown $\mathcal{T}$ from extreme discharges X_i at nodes $i \in V$. Our starting point is the max-linear Bayesian network [Gissibl and Klüppelberg, 2018], a model for risk propagation in a directed acyclic graph. When the graph is a tree $\mathcal{T}$, then the model is defined as

$$X_i = \bigvee_{j:j\to i\in\mathcal{T}} c_{ij} X_j \vee Z_i, \quad c_{ij}, Z_i > 0, \quad i \in V. \tag{3.2}$$

Here the Z_i, called innovations, are independent with support $\mathbb{R}_+$ and have atom-free distributions. The model says that each edge $j \to i$ in $\mathcal{T}$ has a weight $c_{ij} > 0$, interpreted as some measure of the flow rate from j to i, and an extreme discharge at i is either the result of an unknown external input Z_i (e.g. heavy rainfall), or it is the maximum of weighted discharges Z_j from an ancestral node j of i.

Here, the root-directed tree $\mathcal{T}$ describes the causal structure in the data. The Extremal River Problem aims at finding this tree from extreme observations only. Indeed, in Tran [2022], Section 2.1 it is shown that for the Upper Danube data also a naive algorithm based on the pairwise correlation matrix as score matrix performs well, whereas for the Lower Colorado data it returns a less precise tree. So this is an example, where the extremes contain more causal information than the average observations.

QTree can, however, also solve a slightly more general problem. Assume that the tree structure is only in the extremes, whereas "average" data follow a different model. It can also happen that only data from certain nodes follow a heavy-tailed distribution (able to model extreme events, while other nodes are negligible from an extreme value point of view; see e.g. Embrechts et al. [1997], De Haan and Ferreira [2007], Resnick [1987, 2007]). Then it may well be possible that the causality in the extremes can be modeled by a tree on a subset of nodes.

For numerical stability, we prefer to work with the logarithm of the extreme data. To avoid new symbols, we keep the same notation, so the max-linear Bayesian tree becomes

$$X_i = \bigvee_{j:j\to i\in\mathcal{T}} (c_{ij} + X_j) \vee Z_i, \quad c_{ij}, Z_i \in \mathbb{R}, \quad i \in V. \tag{3.3}$$

We further assume that data is corrupted with independent noise in each coordinate. The Extremal River Problem thus becomes the following.

> **The Extremal River Problem**. Given n observations $\mathcal{X} = \{x^1 + \varepsilon^1, \ldots, x^n + \varepsilon^n\}$ in $\mathbb{R}^V$, where the x_i are generated via (3.3), and the ε_i are independent noise variables in $\mathbb{R}^V$, find $\mathcal{T}$.

We stress that the root-directed tree assumption is *different* from the usual tree in Bayesian networks, where each *child* has at most one parent. Learning the single-parent tree can be done with the message passing algorithm, which recursively identifies the parent of a node through likelihood calculations [Wainwright and Jordan, 2008]. This strategy does not work for the root-directed tree, since each child can have multiple parents.

3.2.2 Intuition of QTree

In general, learning Bayesian networks with more than one parent is NP-hard, see [Chickering, 1996]. However, learning the max-linear Bayesian network from i.i.d. *noise-free observations* is solvable in time $O(|V|^2 n)$ with $O(|V|(\log(|V|))^2)$ observations (cf. Lemma 3.7.1 in Section3.7). Here is the intuition.

Fix an edge $j \to i$ and consider the noise-free model (3.3). If for an observation $x \in \mathbb{R}^V$ the value at j causes that at i, then $x_i = c_{ij} + x_j$. If j does not cause i, then $x_i > c_{ij} + x_j$. Over n independent observations, if the value at j causes the value at i at least twice, then the distribution of $x_i - x_j$ has an *atom* at its left-end point. Repeating this argument shows that if j causes k and k causes i, then one also has $x_i - x_j = c_{ik} + c_{kj}$. That is, if the sample $\mathcal{X}$ is noise-free, the empirical distribution of

$$\mathcal{X}_{ij} := \{x_i - x_j : x \in \mathcal{X}\} \tag{3.4}$$

has for sufficiently many observations multiple values *at* the minimum of its support *if and only if* $j \rightsquigarrow i$. Thus, with enough observations, one can recover the directed path $j \rightsquigarrow i$, from which $\mathcal{T}$ can be uniquely constructed as it is a root-directed tree.

QTree exploits the above intuition and makes it work under the presence of noise. Consider an ordered pair of nodes $(j, i) \in V$. If the noise at i is small relative to the signal at j, one can expect a concentration *near* the minimum of $\mathcal{X}_{ij}$ if and only if $j \rightsquigarrow i$. This is the intuition of QTree. While we have no control over the noise, one way to obtain 'strong signals x_j' is to replace (3.4) by the set

$$\mathcal{X}_{ij}(\alpha) := \{x_i - x_j : x \in \mathcal{X}, x_j > Q_{\mathcal{X}_j}(\alpha)\}, \tag{3.5}$$

where $Q_{\mathcal{X}_j}(\alpha)$ is the α-th quantile of the empirical distribution of $\mathcal{X}$ in the j-th coordinate. For $\alpha > 0$, this amounts to a transformation of $\mathcal{X}_{ij}$ that amplifies its concentration near the minimum, at the cost of keeping only a fraction of the available observations (cf. Figure 3.1).

3.2.3 The QTree Algorithm

The QTree Algorithm 4 computes independently for each potential edge $j \to i$ a score w_{ij}, seen as a measure of concentration of $\mathcal{X}_{ij}(\alpha)$ near its minimum, then outputs a minimum directed spanning tree of the graph $\mathcal{G}$ with scores $W = (w_{ij})$. The idea is that at each node j, data would show the highest concentration at the true edge among all edges from some parent of j to j. Theorem 3.2.4 proves this for the Gumbel-Gaussian noise model; see around (3.9). The default concentration measure for QTree is the empirical *quantile-to-mean gap*

$$w_{ij}(\underline{r}) := \frac{1}{n_{ij}} \Big(\mathbb{E}(\mathcal{X}_{ij}(\alpha)) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r})\Big)^2, \tag{3.6}$$

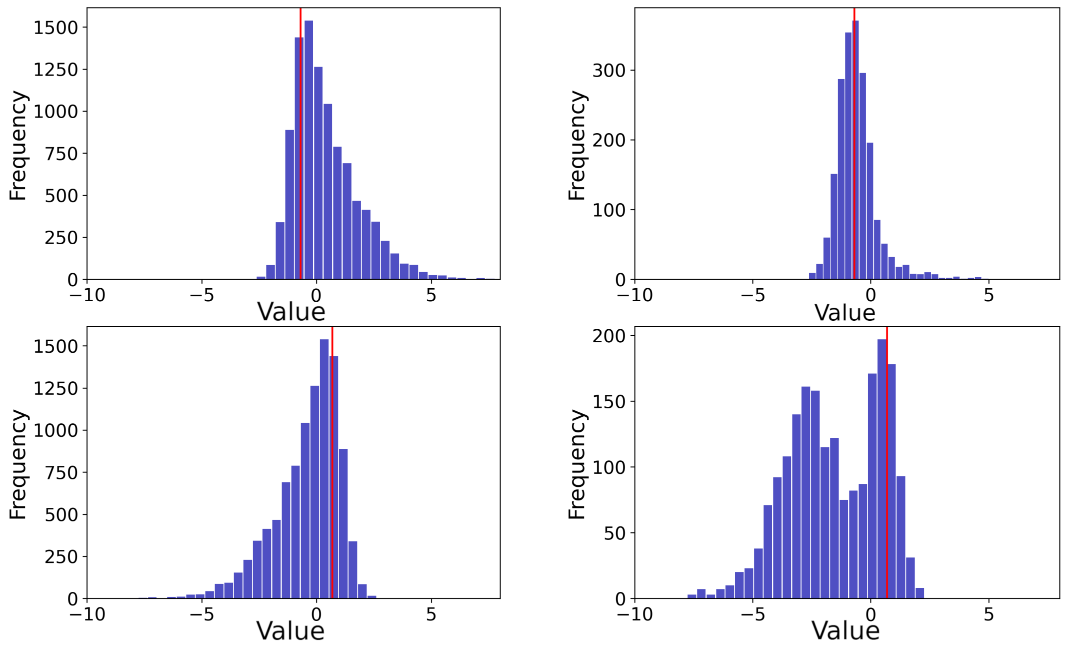


Figure 3.1: For the simple graph $1 \to 2$ with $c_{21} = \log(0.5) = -0.69$ and normal centered noise with standard deviation 0.5: *First row:* Histograms of the observations $\mathcal{X}_{21}$ as in (3.4) (left) and truncated observations $\mathcal{X}_{21}(0.8)$ as in (3.5). In most cases, for $\mathcal{X}_{21}(0.8)$ depicted in the right-hand figure, a large value for x_2 is realised from a large value of x_1, hence the increasing symmetry around c_{21}. *Second row:* Histograms of the observations $\mathcal{X}_{12}$ and $\mathcal{X}_{12}(0.8)$ corresponding to the reversely directed edge. For $\mathcal{X}_{12}(0.8)$ depicted in the right-hand figure, a large value x_2 is realised either from large x_1 or from a large innovation Z_2, giving the bimodal distribution. The vertical lines show the position of the atom in the noise-free distribution, i.e. $\log(0.5)$ in the first row and $-\log(0.5)$ in the second row.

where $\mathbb{E}$ is the empirical mean, Q is the empirical quantile, $\underline{r} \in (0,1)$ is a small quantile level and $n_{ij} = |\mathcal{X}_{ij}(\alpha)|$ is the number of observations in the set $\mathcal{X}_{ij}(\alpha)$ defined in (3.5). The normalization factor n_{ij} only matters when missing values are unevenly distributed across pairs, such as for the Lower Colorado network (cf. Section 3.3.2). Then, pairs with fewer observations get a relative penalty in the concentration estimate to account for larger variability in sample quantile estimates due to a small sample size. If no missing values are present, as is the case with the Upper Danube network, then $n_{ij} = n \cdot (1-\alpha)$ for all pairs (i,j) and the algorithm would return the exact same tree $\hat{\mathcal{T}}$ as if the concentration measure was defined without dividing by n_{ij}.

We note that there are other choices for a concentration measure, such as the empirical *lower quantile gap*,

$$w_{ij}(\underline{r}, \overline{r}) := \frac{1}{n_{ij}} \left(Q_{\mathcal{X}_{ij}(\alpha)}(\overline{r}) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r}) \right)^2, \tag{3.7}$$

where $0 < \underline{r} < \overline{r} < 1$ is a fixed pair of quantile levels. If $\overline{r}$ is small, then $w_{ij}(\underline{r}, \overline{r})$ is a local measure of concentration in the lower tail of $\mathcal{X}_{ij}(\alpha)$. Note that, if the number of observations is small, then $\overline{r}$ cannot be too small, so the two empirical concentration measures are in fact rather similar on a real data set. In practice, the lower quantile gap has one more parameter to tune, and thus we choose the quantile-to-mean gap as our default.

Remark 3.2.1. *Observe that both concentration measures,* (3.6) *and* (3.7)*, are translation invariant. Consequently, considering log data, both measures are invariant to scaling.*

Algorithm 4 QTree for fixed parameters

Parameters: $\underline{r} \in (0,1)$, $\alpha \in [0,1)$.
Input: data $\mathcal{X} = \{x^1, \dots, x^n\} \subset \mathbb{R}^V$.
Output: a root-directed spanning tree $\hat{\mathcal{T}}$ on V.

```
1: for $j \to i$, $j, i \in V, j \neq i$ do
2:     Compute $w_{ij}(\underline{r})$ by (3.6).
3: end for
4: Compute $\hat{\mathcal{T}}$ := minimum root-directed spanning tree on the directed graph
   $(V, \mathcal{G})$ with score matrix $W = (w_{ij}(\underline{r})) \in \mathbb{R}^{V \times V}$ with Chu–Liu/Edmonds'
   algorithm with variable root.
5: Return $\hat{\mathcal{T}}$
```

Remark 3.2.2. *Given a score matrix W (equivalently a bidirected graph) and a unique root (the initial node), Chu–Liu/Edmonds' algorithm (see Gabow et al. [1986] and Grötschel et al. [1988], Sections 7.2 and 8.4 for more background) finds a minimum directed spanning tree; i.e., a network of minimum score with $\sum_{j:j\to i\in\hat{\mathcal{T}}} w_{ij}(\underline{r})$ as small as possible. As we want a minimum* root-directed *spanning tree, we simply reverse edge directions. Moreover, we run the algorithm for every possible node as root, and take a tree with minimum score. Finally, provided that all scores are different, the algorithm finds a unique minimum root-directed spanning tree.*

3.2.3.1 Theoretical properties of QTree

We prove consistency of Algorithm 1 under natural conditions on the distribution of the innovations. We focus on the structural tree model of a max-linear Bayesian network as introduced in Gissibl et al. [2018] taking i.i.d. innovations with Frechét distribution function $P(Z_i \le x) = e^{-x^{-\alpha}}, x > 0$, for $\alpha > 0$. Then, using the solution of (3.2) given in Theorem 2.2 of Gissibl and Klüppelberg [2018], by max-stability (e.g. Embrechts et al. [1997], Section 3.2), X is multivariate Fréchet distributed with marginals as in Proposition A.2 of Gissibl et al. [2018]):

$$P(X_i \le x) = \exp\{-(x_i\mu_i)^{-\alpha}\}, \quad x > 0,$$

for $\mu_i = \left(\sum_{j:j\rightsquigarrow i, j=i} c_{ji}{}^{\alpha}\right)^{-1/\alpha}$. Taking logarithms of the X_i is equivalent to taking logarithms of the innovations Z_i with $P(\log(Z_i) \le x) = \exp\{-e^{-x/\beta}\}, x \in \mathbb{R}$, for $\beta := 1/\alpha > 0$. This results in a Gumbel model

$$P(X_i \le x) = \exp\{-e^{-(x-\mu_i)/\beta}\}, \quad x \in \mathbb{R}.$$

Therefore, for log data, the X_i are Gumbel(β, μ_i) distributed with scale $\beta := 1/\alpha$ and location μ_i.

Instead of taking the logarithmic analog of a *Generalised* Fréchet model as often done in the literature, we prefer instead to add a small independent noise to the max-linear Bayesian tree model (3.3) with Gumbel($\beta, 0$) innovations. This motivates the *noise model*

$$X_i = \Big(\bigvee_{j:j\to i\in\mathcal{T}} (c_{ij} + X_j) \vee Z_i \Big) + \varepsilon_i, \quad c_{ij}, Z_i, \varepsilon_i \in \mathbb{R}, \quad i \in V. \tag{3.8}$$

with the following innovation-noise distributions:

> **Gumbel-Gaussian noise model.** For $i \in V$, the innovations Z_i are i.i.d. Gumbel($\beta, 0$), the noise variables ε_i are i.i.d. with symmetric, light-tailed density f_ε satisfying
>
> $$f_\varepsilon(x) \sim e^{-Kx^p} \text{ as } x \to \infty, \tag{3.9}$$
>
> for some $p > 1$ and $\gamma, K > 0$ and the derivative of f_ε exists in the tail region. Throughout, for two functions a, b, positive in their right tails, we write $a(x) \sim b(x)$ as $x \to \infty$ for $\lim_{x\to\infty} a(x)/b(x) = c$, where $c > 0$ is some arbitrary constant.

Remark 3.2.3. *The density f_ε in* (3.9) *belongs to a special class of light-tailed densities whose convolution tail can be derived asymptotically [Balkema et al., 1993]. The family includes the Gaussian ($p = 2$), and though it is strictly more general than the Gaussian, we follow Balkema et al. [1993], and call our noise model Gumbel-Gaussian for ease of reference. Condition* (3.9) *guarantees that the upper tail of $\varepsilon_i - \varepsilon_j$ is* lighter *than that of $Z_i - Z_j$ (cf. Lemma 3.7.6 in Section 3.7).*

Theorem (3.2.4) below, proved in Section 3.7.2, says that under the Gumbel-Gaussian noise model, both quantile-to-mean and lower quantile gap produce together with Chu-Liu/Edmonds' algorithm strongly consistent estimators for the true root-directed spanning tree $\mathcal{T}$ for appropriate choice of parameters. Simulation results (cf. Figure 3.2) indicate that the error scales as $O(1/n)$ for *any* fixed graph size $|V| = d$. In particular, for a large graph with $d = 100$, QTree only needs $n = 200$ observations to bring the metrics nSHD to less than 5% and TPR to more than 95%; see definitions in (3.1).

We are now ready to state our main theorem. Observe that while α as in (3.5) is an important tuning parameter, asymptotically it does not matter as the consistency holds for $\alpha = 0$; i.e., by taking the full set of observations.

Theorem 3.2.4 (Consistency Theorem)**.** *Assume the Gumbel-Gaussian noise model* (3.8) *with distributions specified above.*
(a) There exists an $r^* > 0$ such that for any pair $0 < \underline{r} < \overline{r} < r^*$, the QTree algorithm with score matrix $W = (w_{ij})$ defined by the lower quantile gap $w_{ij}(\underline{r}, \overline{r})$ in (3.7) *returns a strongly consistent estimator for the tree $\mathcal{T}$ as the sample size $n \to \infty$.*
(b) There exists an $r^* > 0$ such that for any $0 < \underline{r} < r^*$, the QTree algorithm with score matrix $W = (w_{ij})$ defined by the quantile-to-mean gap $w_{ij}(\underline{r})$ in (3.6) *returns a strongly consistent estimator for the tree $\mathcal{T}$ as the sample size $n \to \infty$.*

Remark 3.2.5. *[When QTree may fail] To understand why a condition like* (3.9) *is necessary, suppose that $V = \{1, 2\}$ and that the true graph is $1 \to 2$. Let F_{21} be the distribution function of $(\varepsilon_2 - \varepsilon_1) + (Z_2 - Z_1) \vee c_{21}$. The lower tail of F_{21} essentially*

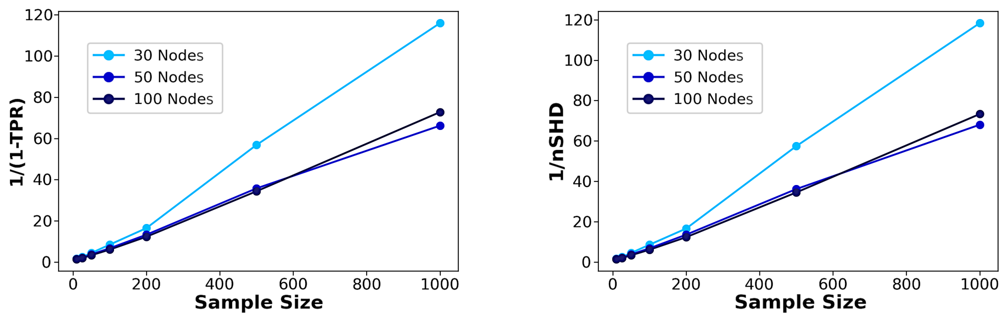


Figure 3.2: 1/(mean errors) vs. number of observations n for different graph sizes $d = 30, 50, 100$. We simulated 100 root-directed spanning trees as described in Section 3.5, where we use the Gumbel-Gaussian setting (1). Then we applied QTree Algorithm 1 with quantile-to-mean gap (3.7) with $\underline{r} = 0.05$ and $\alpha = 0$ to estimate the true (simulated) tree, and computed the average error as measured by 1-TPR (left) and nSHD (right) given in (3.1).

is the lower tail of $(\varepsilon_2 - \varepsilon_1)$*, while the upper tail is essentially the upper tail of the convolution* $(\varepsilon_2 - \varepsilon_1) + (Z_2 - Z_1)$*, which is dominated by the signal* $(Z_2 - Z_1)$ *if it is the heavier tail, and otherwise it is dominated by the noise* $(\varepsilon_2 - \varepsilon_1)$*. Since* $(\varepsilon_2 - \varepsilon_1)$ *has symmetric distribution, if the noise term dominates the distribution,* $w_{12} \approx w_{21}$ *and it would be impossible to distinguish the edge* $1 \to 2$ *from the edge* $2 \to 1$*. If the signal dominates, the asymmetry between the lower and upper tails of* F_{12} *lends us the crucial inequality to distinct between the two graphs as illustrated in Figure 3.1.*

The argument extends to $d > 2$ *for a graph with only one directed path. Then it is not possible to distinguish the direction from a symmetric score matrix. However, for a realistic matrix with real-valued entries, Chu–Liu/Edmonds' algorithm outputs an approximately correct root-directed tree. Intuitively, reversing every edge direction gives the same score but is generally not a root-directed tree.* □

3.2.4 Parameter tuning by bootstrap aggegation

Algorithm 4 has two parameters: the quantile $\underline{r} \in (0,1)$ and the cut-off quantile $\alpha \in (0,1)$. If data comes from a noise-free max-linear Bayesian tree, then we should select $\underline{r} = 0$ as small as possible, $1 - \alpha = 1$, and fit QTree on all of the available data $\mathcal{X}$. However, due to the presence of noise, setting $\underline{r}$ too small and $1 - \alpha$ too large would make the estimator volatile to large values of the noise variables.

In this section, we propose in a first step a subsampling procedure to stabilize the tree estimator of Algorithm 4 and, in a second step, automatically choose $\underline{r}$ and α in QTree. This results in Algorithm 5, which we also refer to as `auto-tuned` QTree.

The basic idea is to run an algorithm on multiple subsets of the data, and then average the resulting estimator. This subsampling approach is also called bootstrap aggregation or bagging; see [James et al., 2013, Section 8.2.1] and [Politis et al., 1999] for a variety of subsampling procedures. Since QTree outputs a directed tree as its estimator, which is a combinatorial object, one cannot simply take the average of their adjacency matrices, as that would not produce a tree. Instead, we see the set of

Algorithm 5 `Auto-tuned QTree`

Parameters: subsampling fraction $f \in [0, 1]$, number of subsamples $m \in \mathbb{N}$, a set of parameters $\Theta = \{(\underline{r}, \alpha)\} \subset [0, 1)^2$ to search over.
Input: data $\mathcal{X} = \{x^1, \dots, x^n\} \subset \mathbb{R}^V$.
Output: the optimal parameter $(\underline{r}^*, \alpha^*) \in \Theta$ and the corresponding root-directed spanning tree $\hat{\mathcal{T}}_{\max}$ on V.

```
1: for $(\underline{r}, \alpha) \in \Theta$ do
2:     for $\ell = 1, \dots, m$ do
3:         Sample without replacement a random subset $\mathcal{X}^\ell$ of $n \cdot f$ observations
           from $\mathcal{X}$.
4:         Let $\mathcal{T}^\ell$ be the output of QTree $(\underline{r}, \alpha)$ fitted on $\mathcal{X}^\ell$.
5:     end for
6:     Let $\mathsf{T}(\underline{r}, \alpha) = \{\mathcal{T}^\ell : \ell = 1, \dots, m\}$
7:     Compute $S(\mathsf{T}(\underline{r}, \alpha))$ by (3.12).
8:     Compute $E(\mathsf{T}(\underline{r}, \alpha))$ as the maximum root-directed spanning tree of
       $S(\mathsf{T}(\underline{r}, \alpha))$ per Lemma 3.2.7.
9:     Compute $\mathrm{Var}(\mathsf{T}(\underline{r}, \alpha))$ by (3.11)
10: end for
11: Define $(\underline{r}^*, \alpha^*) := \arg\min\{\mathrm{Var}(\mathsf{T}(\underline{r}, \alpha)) : (\underline{r}, \alpha) \in \Theta\}$.
12: Return the optimal pair $(\underline{r}^*, \alpha^*)$ and $\hat{\mathcal{T}}_{\max} := E(\mathsf{T}(\underline{r}^*, \alpha^*))$.
```

output trees as a distribution over trees. Then, we solve a second problem, namely, to find the centroid tree E(T) of this distribution, defined as that tree which minimizes the expected Hamming distance to a typical tree (cf. Definition 3.2.6). Lemma 3.2.7 below proves that the centroid can be computed with another application of Chu–Liu/Edmonds' algorithm. This ensures that the estimator produced by auto-tuned QTree can be computed quickly (cf. Lemma 3.2.8).

Our key indicator for model performance is variability in the estimated tree, that is, whether the tree $\hat{\mathcal{T}}$ and its reachability graph $\hat{\mathcal{R}}$ output by QTree would change significantly if we fit it to different subsamples of the data. Here, we denote the reachability graph $\hat{\mathcal{R}}$ of $\hat{\mathcal{T}}$ as the graph that results from drawing an edge between a pair (j, i) whenever there is path from j to i in $\hat{\mathcal{T}}$. We propose the following definition of variability for a distribution of root-directed spanning trees.

Definition 3.2.6. Let V be a set of nodes and $\mathsf{T} = \{\mathcal{T}^1, \dots, \mathcal{T}^m\}$ a collection of root-directed spanning trees on V, and let $\mathsf{R} = \{\mathcal{R}^1, \dots, \mathcal{R}^m\}$ be their corresponding reachability graphs. The *centroid* of T, denoted $E(\mathsf{T})$, is the root-directed spanning tree on V that minimizes the sum of normalized structural Hamming distances d_H defined in (3.1) as follows:

$$E(\mathsf{T}) := \arg\min_{\mathcal{T} \in \Psi} \sum_{i=1}^{m} d_H(\mathcal{T}, \mathcal{T}^i), \tag{3.10}$$

where Ψ is the space of root-directed spanning trees on V.

Let $E(\mathsf{R})$ denote the reachability graph of $E(\mathsf{T})$. Let e_{T} be the number of edges of $E(\mathsf{T})$, and e_{R} be the number of edges of $E(\mathsf{R})$, respectively. We define the *variability* of T, denoted $\mathrm{Var}(\mathsf{T})$, as

$$\mathrm{Var}(\mathsf{T}) := \frac{1}{e_{\mathsf{T}}}\frac{1}{m}\sum_{i=1}^{m} d_H(\mathcal{T}^i, E(\mathsf{T})) + \frac{1}{e_{\mathsf{R}}}\frac{1}{m}\sum_{i=1}^{m} d_H(\mathcal{R}^i, E(\mathsf{R})). \tag{3.11}$$

Involving the Hamming distance of the reachability graphs in (3.11) penalizes the situation where $\mathcal{T}^i$ and $E(\mathsf{T})$ differ in a few edges low down in the tree, for example, if they have different roots. Such a difference would lead to a small structural Hamming distance between the two trees, but a large structural Hamming distance between their reachability graphs, and in particular, very different river networks.

The following lemma says that $E(\mathsf{T})$ is a maximum root-directed spanning tree of a particular graph with score matrix $S(\mathsf{T})$ that measures the stability among the trees in T. In particular, $E(\mathsf{T})$ can be computed using Chu–Liu/Edmonds' algorithm (choosing the root realizing the minimum score), and thus $\mathrm{Var}(\mathsf{T})$ can be computed in polynomial time.

Lemma 3.2.7. *Let V be a set of nodes and $\mathsf{T} = \{\mathcal{T}^1, \dots, \mathcal{T}^m\}$ a collection of root-directed spanning trees on V. Define the* stability score matrix $S := S(\mathsf{T}) \in \mathbb{R}^{d\times d}_{\geq 0}$ *by*

$$s_{ij} := S(\mathsf{T})_{ij} := \#\{\mathcal{T} \in \mathsf{T} : j \to i \in \mathcal{T}\}. \tag{3.12}$$

Suppose that the maximum root-directed spanning tree $\mathcal{T}_{\max}$ of the graph on V with score matrix $S(\mathsf{T})$ is unique. Then $E(\mathsf{T}) = \mathcal{T}_{\max}$.

Proof. Identify a root-directed tree $\mathcal{T}$ with the vector $T = (T_{uv}) \in \{0,1\}^{|V|^2-|V|}$. Write $\mathbf{1} = (\mathbf{1}_{uv})$ for the all-one vector of the same dimension. Let $\mathcal{T}' \in \Psi$ be any root-directed spanning tree on V. Our goal is to show that

$$\sum_{i=1}^{m} d_H(\mathcal{T}', \mathcal{T}^i) \geq \sum_{i=1}^{m} d_H(\mathcal{T}_{\max}, \mathcal{T}^i),$$

which would establish that $\mathcal{T}_{\max} = E(\mathsf{T})$ by (3.10). Indeed,

$$\begin{aligned}
\sum_{i=1}^{m} d_H(\mathcal{T}', \mathcal{T}^i) &= \sum_{i=1}^{m} \sum_{u,v\in V:u\neq v} \mathbf{1}\{T'_{uv} \neq T^i_{uv}\} = \sum_{u,v\in V:u\neq v} \sum_{i=1}^{m} \mathbf{1}\{T'_{uv} \neq T^i_{uv}\} \\
&= \sum_{u,v\in V:u\neq v} (s_{uv}\mathbf{1}_{u\to v\notin \mathcal{T}'} + (m - s_{uv})\mathbf{1}_{u\to v\in\mathcal{T}'}) \\
&= -2\langle S, T'\rangle + \langle S, \mathbf{1}\rangle + \langle m\mathbf{1}, T'\rangle \quad \text{where } \langle\cdot,\cdot\rangle \text{ denotes the Frobenius inner product} \\
&= -2\langle S, T'\rangle + \langle S, \mathbf{1}\rangle + m(d-1) \quad \text{since } \mathcal{T}' \text{ as root-directed spanning tree has } d-1 \text{ edges} \\
&\geq -2\langle S, T_{\max}\rangle + \langle S, \mathbf{1}\rangle + m(d-1)\rangle \quad \text{by definition of } \mathcal{T}_{\max} \\
&= -2\langle S, T_{\max}\rangle + \langle S, \mathbf{1}\rangle + \langle m\mathbf{1}, T_{\max}\rangle\rangle \quad \text{since } \mathcal{T}_{\max} \text{ is a spanning tree on } V \\
&= \sum_{i=1}^{m} d_H(\mathcal{T}_{\max}, \mathcal{T}^i).
\end{aligned}$$

□

As shown in Lemma 3.7.2 of Section 3.7, Algorithm 4 runs in time $O(|V|^2 n)$. The quadratic dependence on $|V|$ and linear dependence on n is optimal, since it takes $O(|V|^2 n)$ just to compute pairwise statistics such as the concentration measures in (3.6) or (3.7) for every pair of nodes. Similarly, the runtime of Algorithm 5 (auto-tuned `QTree`) also has optimal runtime, which scales linearly with the number of repetitions m and the size of the parameter grid $|\Theta|$.

Lemma 3.2.8. *The `auto-tuned` QTree Algorithm 5 has complexity* $O(|V|^2 nm|\Theta|)$.

Proof. For each pair $(\underline{r}, \alpha) \in \Theta$, step 3 takes $O(mn)$ and step 4 takes $O(|V|^2 nm)$ by Lemma 3.7.2 in Section 3.7. Step 6 takes $O(m|V|^2)$, and step 7 takes $O(|V|^2)$ by Chu–Liu/Edmonds' Algorithm. Computing the reachability graph for a root-directed tree on $|V|$ nodes takes $O(|V|)$, so step 8 takes $O(m|V|^2)$, since for each of the m trees in T we need to compute its structural Hamming distance from the estimated tree $E(\mathsf{T})$. So, for each pair $(\underline{r}, \alpha) \in \Theta$, steps 3 to 8 take $O(|V|^2 nm)$ time. Thus overall, the algorithmic complexity is $O(|V|^2 nm|\Theta|)$. $\square$

3.3 Data description

We focus on river discharge data in two river networks, the Upper Danube network with data from Bavaria, Germany, and the Lower Colorado network in Texas, USA. Large flood events are classical examples for high risk analysis. The Danube data as well as the data of all three sectors of the Colorado are available in the `Python` package `QTree` (Tran [2021]). The Danube data are available in the `R` package `graphicalExtremes` (Engelke et al. [2019]).

In general, river discharges across a set of stations is recorded multiple times per hour and some preprocessing is needed to turn the raw data into independent extreme discharge data. This was detailed in Asadi et al. [2015] for the Danube data; cf. Figure 3.3. We follow their procedure (descibed in Section 3.3.1) with slight modifications for the Colorado data (descibed Section 3.3.2).

3.3.1 The Upper Danube network

The Danube network data consist of measurements collected at $d = 31$ gauging stations over 50 years from 1960 to 2009 by the Bavarian Environmental Agency (`http:www.gkd.bayern.de`). Preprocessing the data, Asadi et al. [2015] first take daily mean values in each time series. The idea is then to find non-overlapping time windows of p days, centered around the observation of maximal rank across all series. For the Danube, the authors choose $p = 9$ days (± 4 days around the observation of maximal rank). For each time series, they then take the maximum within the given time window, delete the data of this window, and proceed until no window of p consecutive days remains. In order to reduce temporal non-stationarities and the effect of snow melt, only the months June, July and August are considered. This results in $n = 428$ observations from a $d = 31$-dimensional random vector whose i-th entry corresponds to the maximum water discharge at the i-th station, observed within a 9-day window where at least one station witnessed a large discharge value; these observations are assumed to be independent.

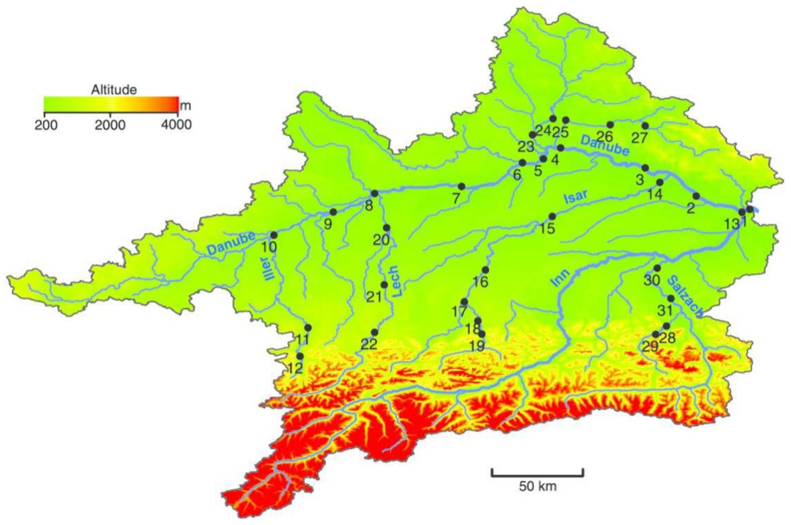


Figure 3.3: Topographic map of the Upper Danube Basin, showing sites of the 31 gauging stations along the Danube and its tributaries.

3.3.2 The Lower Colorado network in Texas

This section describes the new data set of the Lower Colorado river network in Texas collected by the Lower Colorado River Authority (LCRA, `https://www.lcra.org/`) and details the preprocessing.

The Lower Colorado is one of the major rivers in Texas. Flowing through major population centers such as Austin, the state capital of Texas, flood and drought mitigation in the Lower Colorado Basin is of prominent interest. A particularly challenging feature of the Lower Colorado is prolonged drought (discharge of 0) followed by flash flooding which can damage sensors, resulting in loss of data over multiple days (cf. Figure 3.4). This makes the Lower Colorado data much more challenging than the Danube data.

The river discharges at the Lower Colorado network, measured in cubic feet per second (cfs), are collected multiple times per day at a total of 104 stations around the Colorado River and its tributaries in Texas from the 1st of December 1991 to the 14th of April 2020 (10,363 days); see Figure 3.5. We do not take into account 5 nodes of the Blanco River and San Bernand River, which are not flow-connected to the Lower Colorado River, and also 21 nodes with zero observations. Moreover, we exclude the nodes 5476, 5634, 5635, 6397, and 6533 as they are located close to hydropower plants. This gives a total of 73 nodes.

Another problem occurs, because in the Lower Colorado Basin, multiple dams cut off the river into disjoint sectors [Lower Colorado River Authority (LCRA), Accessed August 2020]. Thus, we split the river network such that in each section, we get the largest set of nodes where (i) no node is within 10km of a major dam, (ii) all nodes are connected, and (iii) for each pair (j, i) of this subset, there are at least 1000 pairwise daily observations, which is 9.6% of the total amount of 10,363 days available. Criterion (iii) ensures that among the given pairs of nodes, any possible causal relation can be discovered, and not be affected by the lack of concurrent data.

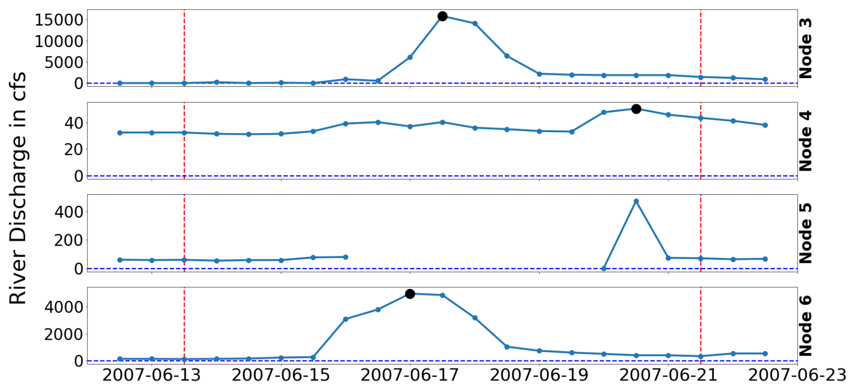


Figure 3.4: Typical discharge at various nodes around one flood event on the Lower Colorado. The vertical lines mark a time window of $p = 17 \times 12h$ or $p = 8.5$ days. Dots denote the 12-hour maximum water discharges. In nodes 3,4 and 6, the thick dot denotes the peak discharge during this window. For node 5, the sensor did not function during this entire time period, so the peak for node 5 is recorded as missing. Node 3 has a median discharge of only 15 cfs and is hence mostly drained over the entire period (1991 to 2020, 10,363 days), but the water flow regularly aggregates to over 16,000 cfs within a very short period of time.

This results in 42 nodes divided into three sectors, which we call the *Top, Middle and Bottom* sectors of the Lower Colorado with 9, 12 and 21 nodes, respectively. From here on we treat these three sectors as three separated, unrelated data sets.

In contrast to the Danube network with snow melt and seasonal periodicity, we can and do take all data of the 12 months into account. We observe further that, by the special weather conditions, flood events can last as little as a few hours. Therefore, in contrast to Asadi et al. [2015], who take daily time slots, we take 12-hour time slots. As a first step, we take maxima of each 12-hour time slot to retain the knowledge about large possible peaks and such periods where no data are collected. In a second step we then take ± 8 12-hour time slot around the time-slots with the observation of maximal rank resulting in a time window of p=8.5 days; see Figure 3.4. We take the most conservative approach to missing data, namely, if node i has any missing data during the considered time window, then its maximum discharge over this window is labeled as missing (cf. Figure 3.4). This is because a sensor can break before the river reaches peak discharge and for practical reasons can only be replaced after the flood event is over [Lower Colorado River Authority (LCRA), 2020], and thus the sensor potentially did not measure the largest possible water discharge that occurred at node i. This results in the Top sector having 9 nodes, 975 observations, 18% missing data; the Middle sector has 12 nodes, 972 observations, 27% missing data. The Bottom sector is most challenging, for it has the most nodes (21 nodes), 961 observations, the highest amount of missing data (37 %), and many nodes around the city of Austin with only a few miles apart from each other. In the Bottom sector there are many nodes with a very small number of observations due to the many missing observations, and we create a new data set by excluding all nodes with less than 150 observations and refer to them as *Bottom150*.

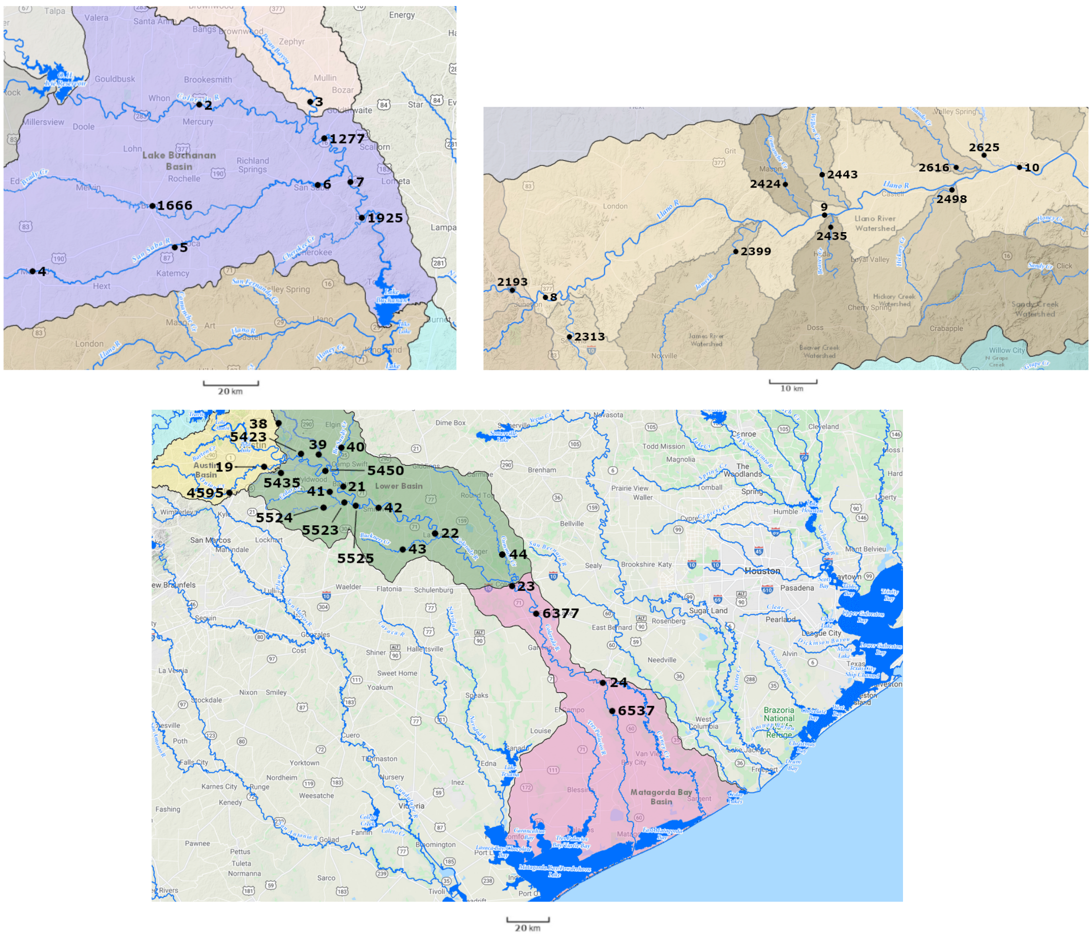


Figure 3.5: Topographic maps of the Top, Middle and Bottom sectors (arranged clockwise) of the Lower Colorado network, showing sites of the gauging stations along the Colorado River and its tributaries. We treat them as three unrelated data sets.

The close proximity of nodes induces strong spatial dependence even among nodes that are not flow-connected, making it potentially more challenging to recover the true network. A summary of the abailable data for each data set is given in Table 3.1.

3.4 Results

3.4.1 Results of `auto-tuned` QTree for all river networks

For each of the four river networks (Danube, Top, Middle and Bottom sectors of the Colorado), we ran `auto-tuned` QTree (Algorithm 5) with fixed $\underline{r} = 0.05$, subsampling rate $f = 0.75$, and number of repetitions $m = 1000$ to choose the tuning parameter α automatically from $\{0.7, 0.725, 0.75, , \ldots, 0.9\}$. The optimal parameters α^* selected by QTree for these networks are shown in Table 3.2.

Figures 3.6—3.8 show the estimated trees of the Danube, Top, Middle and Bottom sectors of the Colorado, respectively. We do two estimated-vs-true comparisons:

Table 3.1: Number of nodes d, number of observations n and percentage of missing data used for the algorithmic reconstruction of the river network.

	Danube	Top	Middle	Bottom	Bottom150
d	31	9	12	21	16
n	428	975	972	961	961
%	0%	18%	27%	37%	22%

Table 3.2: Optimal parameters α^* selected by QTree using grid search with subsampling.

	Danube	Top	Middle	Bottom	Bottom150
α^*	0.775	0.825	0.75	0.85	0.725

one for the tree, and one for its reachability graph. The four metrics we use are normalized Structural Hamming Distance (nSHD), False Discovery Rate (FDR) and False Positive Rate (FPR) and True Positive Rate (TPR), defined in (3.1). Table 3.3 gives all performance metrics over all data sets. We recall that the performance of an algorithm is better the smaller the first three metrics are and the larger TPR is. QTree performs very well across all data sets, with nSHD and FDR ranging from 10-20%, FPR close to 0, and TPR around 80-90%. For the reachability graph, the statistics are even better: nSHD, FPR and FDR are below 9% and TPR is over 87%. In other words, a wrongly estimated edge directs rather from an ancestor (which is not a parent) to a child (flow-connection is preserved), than a spurious edge (an edge which contradicts flow-connection). The number of missing edges are determined by the fact that a tree has exactly $d - 1$ edges.

Figure 3.8(top) visualizes the estimation of the Bottom sector of the Colorado. As expected, this data set is the most challenging due to large portions of missing data and the clustering of nodes around the city of Austin. Nevertheless, even for this data set the estimated tree has only two spurious edges, between 6537 and 24 and between 42 and 5525. Both of these node pairs are physically close. All the remaining wrongly estimated edges are flow-connected.

We note that the majority of errors made by QTree involves nodes with less than 150 observations (which are the nodes 5525, 5450, 5423, 5435 and 5524). This is not at all surprising. The model was fitted to only 75% of the data, and the optimally chosen α^* is 0.85, which means that for each edge involving one of the above nodes, the number of observations available to QTree is at most $150 \times 0.75 \times 0.15 = 14$. To check the hypothesis that this threshold is too small for QTree to perform reliably, we excluded all nodes with less than 150 observations and refitted QTree on the remaining 16 nodes (Bottom150). The result depicted in Figure 3.8(bottom) shows significant improvements. This manifests another desirable feature of QTree, namely, that it relies on local (pairwise) estimation, and thus changes to the node set in one part of the tree do not affect the estimated network elsewhere.

As expected from a statistical estimation procedure, the statistical choice of the parameter selection by QTree does not always output the best result on every data set. However, it fails only by very few edges to the graph estimated with the best choice

Table 3.3: Metrics nSHD, FPR, FDR and TPR for QTree. Numbers display the respective metric for the pair $(\mathcal{T}, \hat{\mathcal{T}})$ and numbers in brackets for the pair $(\mathcal{R}, \hat{\mathcal{R}})$ of their respective reachability graphs.

	Danube	Colorado			
		Top	Middle	Bottom	Bottom150
nSHD	0.18(0.09)	0.13(0.02)	0.09(0.06)	0.45(0.15)	0.10(0.12)
FPR	0.01(0.02)	0.04(0.00)	0.02(0.04)	0.05(0.03)	0.02(0.02)
FDR	0.20(0.05)	0.13(0.00)	0.09(0.10)	0.50(0.02)	0.13(0.02)
TPR	0.80(0.87)	0.88(0.95)	0.90(1.00)	0.50(0.74)	0.87(0.78)

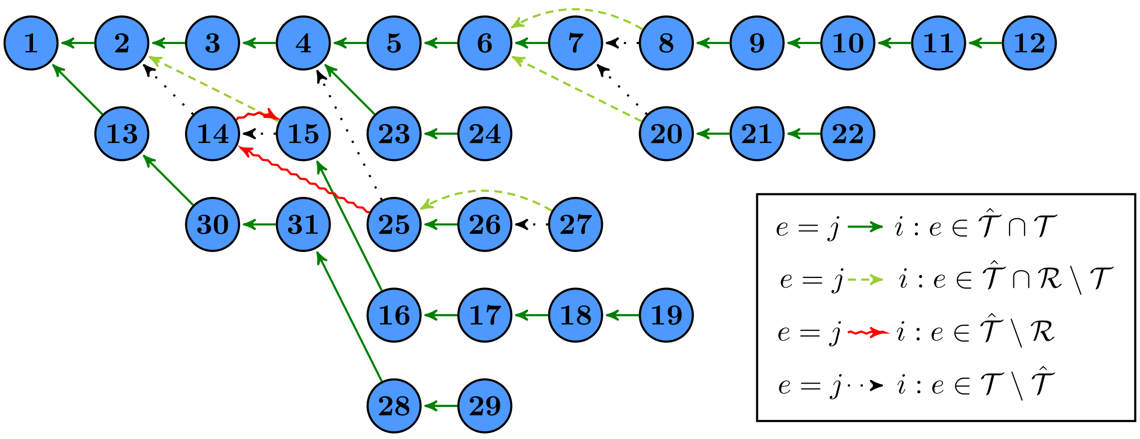

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
23
24
20
21
22
30
31
25
26
27
16
17
18
19
28
29
$e = j \longrightarrow i : e \in \hat{\mathcal{T}} \cap \mathcal{T}$
$e = j \dashrightarrow i : e \in \hat{\mathcal{T}} \cap \mathcal{R} \setminus \mathcal{T}$
$e = j \rightsquigarrow i : e \in \hat{\mathcal{T}} \setminus \mathcal{R}$
$e = j \cdots\to i : e \in \mathcal{T} \setminus \hat{\mathcal{T}}$

Figure 3.6: Danube river network, estimated by QTree vs. true. Solid (green) edges are correct. Dashed (green) edges are not in the tree but in the reachability graph, that is, the causal direction or flow-connection is correct. Squiggly (red) edges are spurious (neither in the tree nor in the reachability graph). Dotted (black) edges are in the true tree, but not in the estimated tree. QTree outputs a tree with only six wrongly estimated edges, four of them flow-connected and one path skipping a single node (the edge $8 \to 6$ skips node 7). Two edges are spurious.

of parameters. For example, for the Danube the parameter $\alpha = 0.75$ (instead of the optimal $\alpha^* = 0.775$) would have lead to a better result (cf. Figure 3.9). Also for the Top Colorado, $\alpha = 0.9$ would have given perfect recovery of the true network; this is clear as all four metrics become optimal (Figure 3.18 of of Section 3.7).

In summary, on all four data sets considered, QTree performed well for nodes with a sufficient number of observations as becomes obvious from Figure 3.8(top) and (bottom). The estimated optimal parameter α^* is either the best one (i.e., the corresponding estimated tree is best across all α) (Top and Bottom sectors of the Colorado), or such that it is within one to two wrong edges of the best one (Danube and Top sector of the Colorado). The method can handle data with missing observations and close spatial proximity between nodes.

3.4.2 Comparison to other scores in the literature

We compare QTree and `auto-tuned` QTree with existing algorithms for extremal causal estimation in the literature. To this end, we first define the scores and sum-

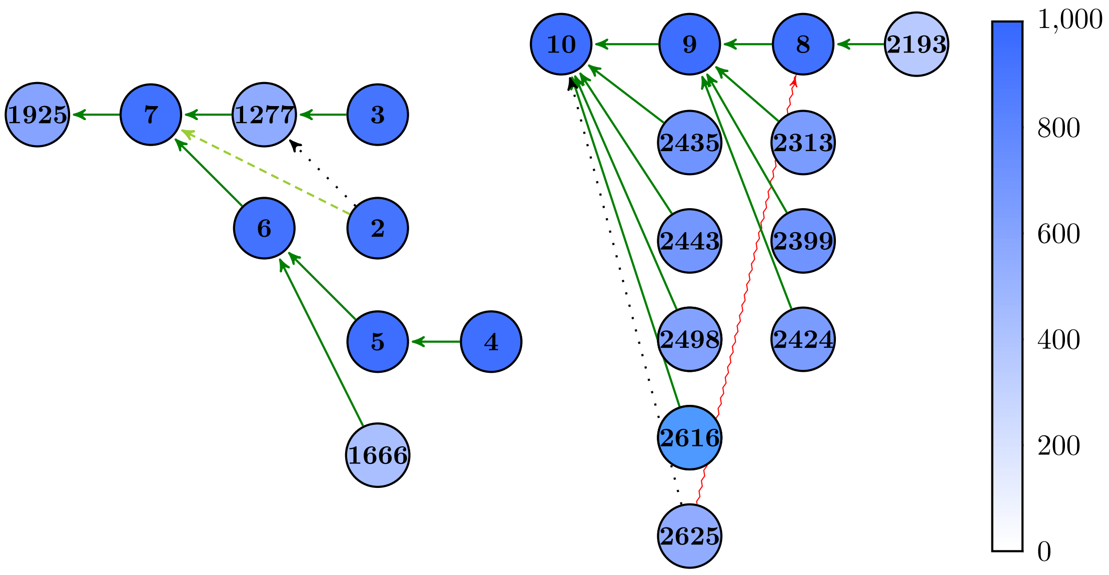


Figure 3.7: Top (left) and Middle (right) sectors of the Colorado network, estimated by QTree vs. true. Node colors represent the amount of available data after taking care of missing data. Arrows are as described in Figure 3.6. Both estimated networks only contain one single wrongly estimated edge. Top: one edge wrong but flow-connected; Middle: one edge spurious.

marize them in score matrices. We utilize the empirical versions of the following extreme dependence measures, where we also include the quantile-to-mean gap for comparison:

1. The empirical *quantile-to-mean gap* as in (3.6) for fixed $(\underline{r}, \alpha) = (0.05, 0.9)$. We fix $\underline{r} = 0.05$ as we have used this throughout, and $\alpha = 0.9$ as an arbitrary parameter.

2. The *causal tail coefficient* $\Gamma_{ij} = \lim_{u \to 1} \mathbb{E}[F_i(X_i) \mid F_j(X_j) > u]$ [Gnecco et al., 2021, eq. (3)]. Observe that in Gnecco et al. [2021], the algorithm EASE outputs from the estimated score matrix Γ an estimated causal order of the set of nodes, not a directed graph.

3. The *causal score* S^{ext}_{ij} is based on expected quantile scores [Mhalla et al., 2020, eq. (15)].[1] The goal of the authors is to discover causality in a directed graph modelled by flow-connection. The scores S^{ext}_{ij} satisfy $S^{\text{ext}}_{ji} + S^{\text{ext}}_{ij} = 1$ and an extreme observation at node j causes an extreme observation at node i, whenever $S^{\text{ext}}_{ij} > 0.5$. The authors propose a bootstrap method to generate 95% confidence bounds to guarantee that the score is larger than 0.5. All these scores are interpreted as directed edges between nodes. Also for the tree example of the Danube data treated in Section 5 of the paper (cf. Figure 7). The algorithm CausEV outputs flow-connections induced by all scores larger than 0.5. Thus, their causal edges rather resemble the edges in the reachability graph of the tree.

[1]We want to thank Linda Mhalla for helping us to set up the CausEv implementation.

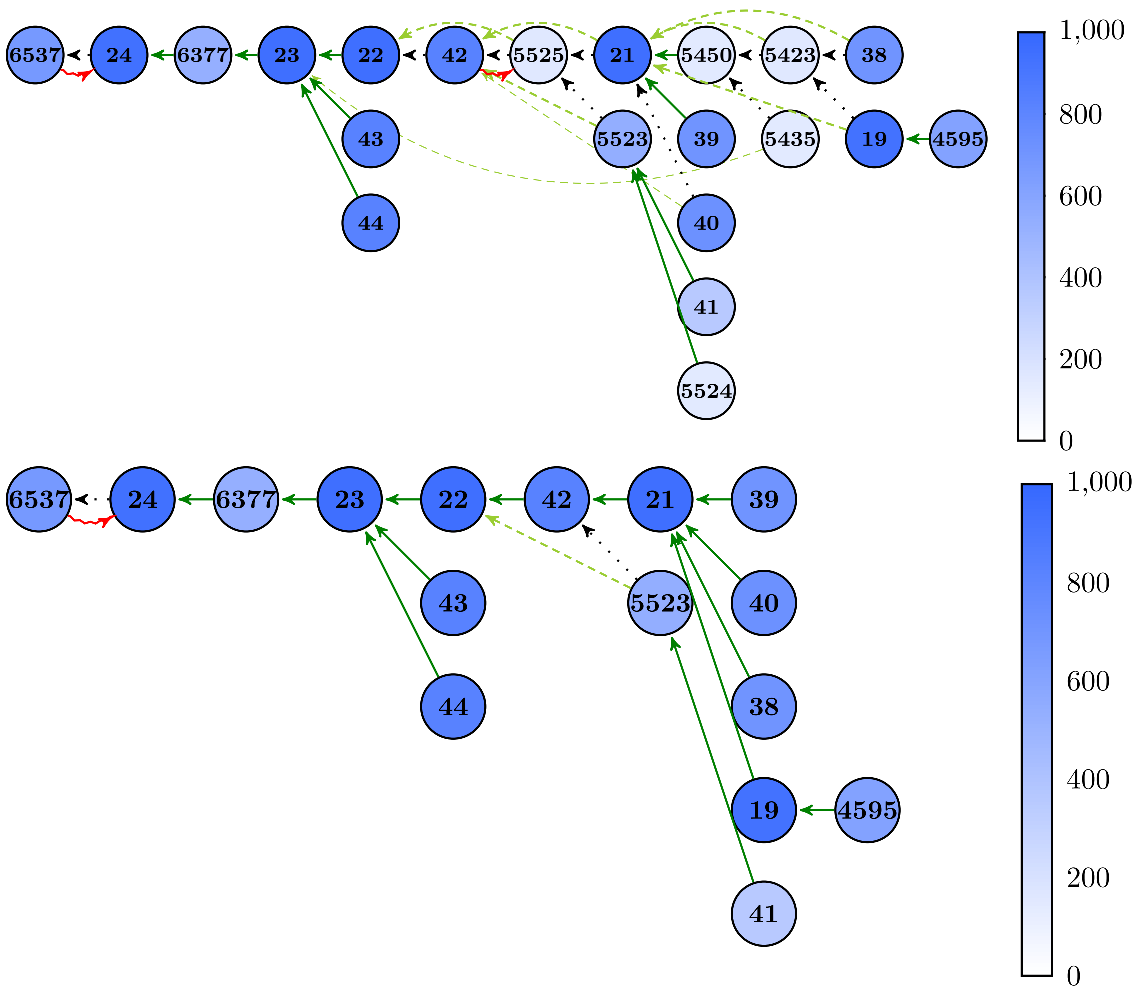
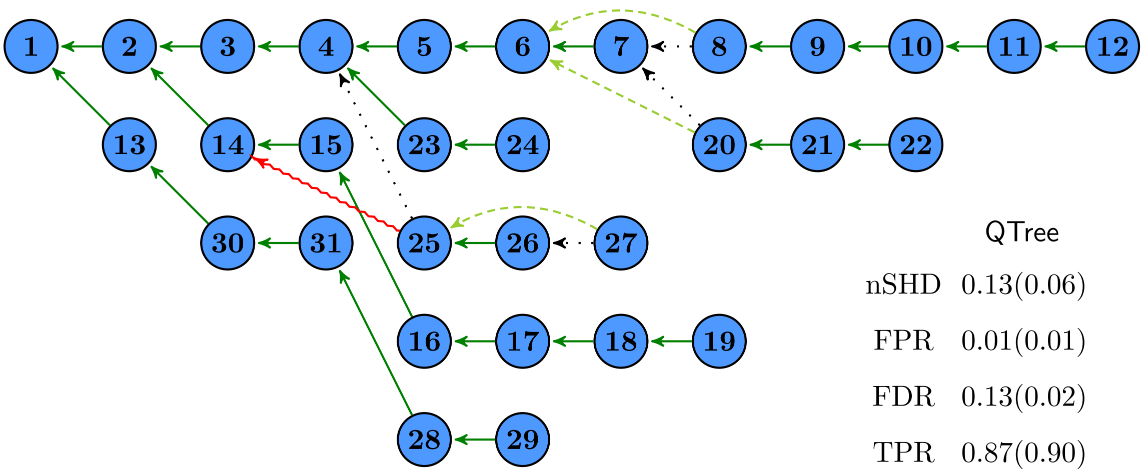


Figure 3.8: Bottom sector of the Colorado network (Bottom and Bottom150), estimated by QTree vs. true. Top Figure: Bottom, based on all 21 nodes, QTree outputs a tree with ten wrongly estimated edges, eight of them flow-connected, two spurious edges pointing in the wrong direction. Bottom Figure: Bottom150, based on 16 nodes, after removing nodes with less than 150 observations. There are only two wrongly estimated edges, one flow-connected, one spurious edge pointing in the wrong direction. Compared to the Bottom sector, this is a significant improvement. Node colors represent the amount of available data after taking care of missing data. Arrows are as described in Figure 3.6

.



	QTree
nSHD	0.13(0.06)
FPR	0.01(0.01)
FDR	0.13(0.02)
TPR	0.87(0.90)

Figure 3.9: Danube river network, estimated by QTree vs. true for $\alpha = 0.75$. Compared to Figure 3.6, the edges $15 \to 14$ and $14 \to 2$ are here correctly estimated. Also the performance measures at the right bottom of the figure compare favourably to those in the first row of Table 3.3.

4. The *tail dependence coefficient* $\chi_{ij} = \lim_{u\to 1} P(F_i(X_i) > u \mid F_j(X_j) > u)$, also called *extremal correlation*. It goes back to Sibuya [1960]; see also [Coles et al., 1999]. We add this dependence measure in our comparison as it is *the* classic one, having been used for more than 60 years in multivariate extreme value statistics. Theoretical properties of the tail dependence coefficient in a max-linear Bayesian network have been investigated in Gissibl et al. [2018]. It has values in $[0, 1]$ and a large value of χ_{ij} indicate strong extreme dependence. Its empirical estimator takes u large, but finite, and our estimator is based on 10% of the data (corresponding to $\alpha = 0.9$ in (a)). The paper Engelke and Volgushev [2020] uses χ and two related measures on p. 14, the extremal correlation, the extremal variogram and the combined extremal variogram for estimating undirected trees with Prim's algorithm Prim [1957].

The scores discussed (b)-(d) have not been used to estimate a directed tree, but as they are all pair-wise scores, their respective estimated matrices can serve as input for Chu–Liu/Edmonds' algorithm. This gives a fair comparison, as we use then for all scores the knowledge that the true graph is a root-directed spanning tree. A few words on the estimation of the matrices Γ and S^{ext} are necessary to fully understand the estimation procedure. For better comparability, we estimate the score matrix Γ also for the declustered Danube data. For the causal score S^{ext}_{ij} we recall that a spanning tree with d nodes has exactly $d-1$ edges, which are taken by Chu–Liu/Edmonds' algorithm as large as possible under the restriction that the outcome is a spanning tree. The estimated root-directed spanning tree has smallest score of 0.5183, thus, it is above 0.5, so all directions are causal in the sense of Mhalla et al. [2020]. While the matrix χ is symmetric, causal inference is possible with Chu–Liu/Edmonds' algorithm. This is because reversing every edge direction gives the same score but is generally not a root-directed spanning tree; see Remark 3.2.5. Finally, we keep all tuning parameters used for Γ_{ij} and S^{ext}_{ij} the same as in the respective papers.

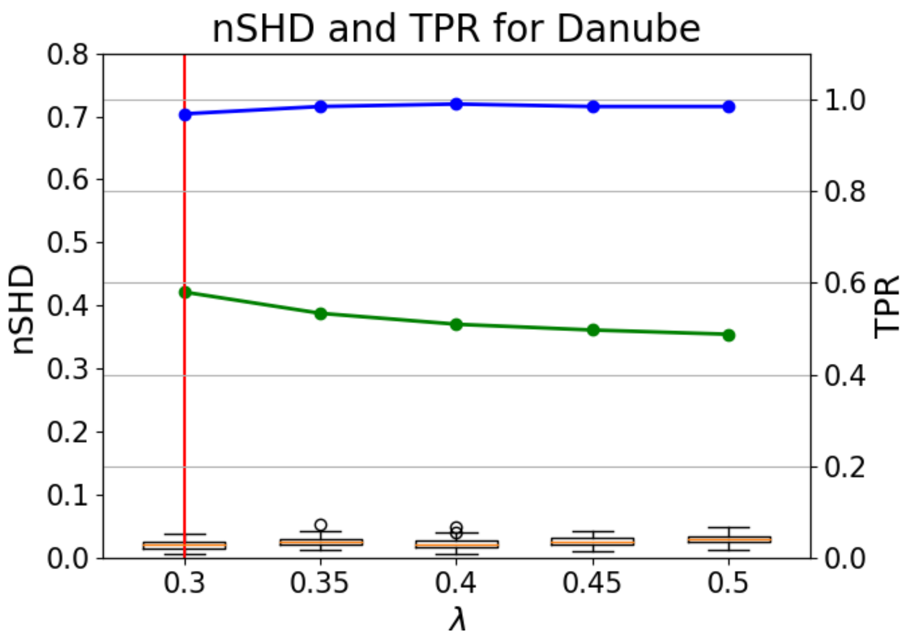


Figure 3.10: Metrics nSHD, TPR, FDR and FPR for the output $\hat{\mathcal{T}}_\alpha$ of the steps of `auto-tuned` QTree for varying parameter α for the Danube network. We subsample 75% of the data 1000 times. For each α we fit QTree on these subsamples to obtain 1000 estimated trees $\mathsf{T}_\alpha := \{\hat{\mathcal{T}}_\alpha^1, \ldots, \hat{\mathcal{T}}_\alpha^{1000}\}$ (Step 5 of Algorithm 2). The metrics of $(\hat{\mathcal{T}}_\alpha^\ell, \mathcal{T})$ and $(\hat{\mathcal{R}}_\alpha^\ell, \mathcal{R})$ are represented in boxplots, one for the tree (blue) and one for the reachability graph (green). The blue and green dots present the four metrics for the centroid $E(\mathsf{T}_\alpha)$ (Step 7 of Algorithm 2) and its reachability graph. The lines are interpolations for better visibility, solid blue for the tree and dashed green for the reachability graph. The chosen α^* is the parameter with the least variablity in T_α (Step 9 of Algorithm 2), indicated by a red vertical line.

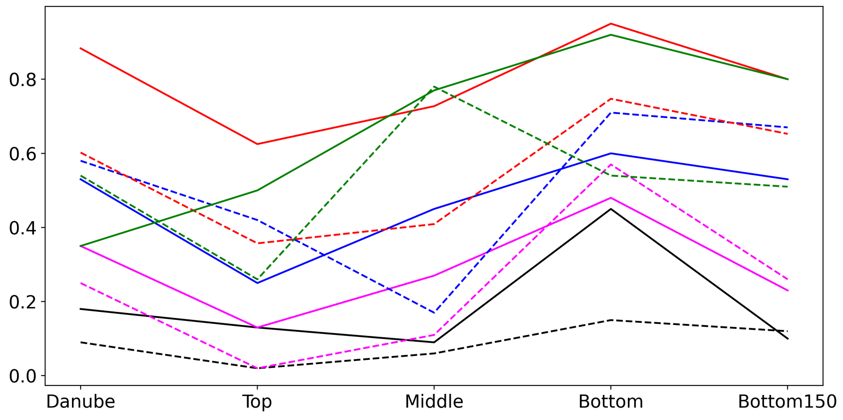


Figure 3.11: Performance metric nSHD for all five data sets and scores vs. QTree: `auto-tuned` QTree, QTree, S^{ext} as in Mhalla et al. [2020], Γ as in Gnecco et al. [2021] and χ as in Sibuya [1960]. Solid lines display the respective metrics for the pair $(\mathcal{T}, \hat{\mathcal{T}})$ and dashed lines for the pair $(\mathcal{R}, \hat{\mathcal{R}})$ of their respective reachability graphs. Black lines are used for `auto-tuned` QTree, magenta lines for QTree, red lines for S^{ext}, blue lines for Γ and green lines for χ.

Table 3.4: Metrics nSHD, FPR, FDR and TPR for the simple QTree Algorithm 4 with $\alpha = 0.9$. Numbers display the metrics for the pair $(\mathcal{T}, \hat{\mathcal{T}})$ and numbers in brackets for the pair $(\mathcal{R}, \hat{\mathcal{R}})$ of their respective reachability graphs.

	Danube	Colorado			
		Top	Middle	Bottom	Bottom150
nSHD	0.35(0.25)	0.13(0.02)	0.27(0.11)	0.48(0.57)	0.23(0.26)
FPR	0.01(0.02)	0.02(0.00)	0.02(0.03)	0.03(0.15)	0.02(0.01)
FDR	0.40(0.14)	0.13(0.00)	0.27(0.19)	0.60(0.58)	0.27(0.02)
TPR	0.60(0.64)	0.88(0.95)	0.72(1.00)	0.40(0.23)	0.73(0.59)

Table 3.5: Metrics nSHD, FPR, FDR and TPR presented as in Table 3.3 for the maximum root-directed spanning tree estimated by Chu–Liu/Edmonds' algorithm with score matrix Γ as in Gnecco et al. [2021], eq. (8).

	Danube	Colorado			
		Top	Middle	Bottom	Bottom150
nSHD	0.53(0.58)	0.25(0.42)	0.45(0.17)	0.60(0.71)	0.53(0.67)
FPR	0.02(0.18)	0.05(0.12)	0.04(0.03)	0.04(0.17)	0.04(0.27)
FDR	0.73(0.71)	0.38(0.40)	0.45(0.21)	0.70(0.78)	0.67(0.83)
TPR	0.27(0.37)	0.63(0.43)	0.55(0.88)	0.30(0.10)	0.33(0.12)

Hence, our comparison is two-fold.

1. We compare the new score of a quantile-to-mean gap with other scores from the literature.
2. We compare `QTree` with `auto-tuned QTree` assessing the benefit of the stabilizing subsampling procedure.

Tables 3.5, 3.6 and 3.7 present the metrics nSHD, FPR, FDR and TPR based on the three scores (b)—(d) and for each data set previously considered. Comparing the metrics with those of `QTree` in Table 3.4, we find for the Danube network a comparably weak performance for all three alternative scores. Among the three scores, χ performs best, followed by Γ. We visualize our findings from Table 3.3 and Tables 3.4—3.7 for nSHD in Figure 3.11.

Solid lines present nSHD for the estimated tree vs. true tree. We find that `auto-tuned QTree` (black line) is uniformly best over all data sets, followed by `QTree`. We conclude that the stabilizing subsampling procedure of `auto-tuned QTree` improves the estimated tree. Even for the Top Colorado network, where $\alpha = 0.9$ is optimal, the nSHD is still positive for `QTree` whereas for `auto-tuned QTree`, nSHD is equal to zero and the estimated tree is equal to the true one. For all Colorado sectors, Γ follows next, χ is surprisingly successful for the Danube, but not for any of the Colorado sectors. S^{ext} does not perform well in any tree recovery, but this was also not the goal of Mhalla et al. [2020]

Dashed lines present nSHD for the reachability matrix of the estimated trees vs. true. Here `auto-tuned QTree` and `QTree` give the same answers as for the trees above. The blue dashes line representing Γ is only moderately worse than the magenta line for

Table 3.6: Metrics nSHD, FPR, FDR and TPR presented as in Table 3.3 for the maximum root-directed spanning tree estimated by Chu–Liu/Edmonds' algorithm with score matrix S^{ext} as in Mhalla et al. [2020], eq. (15).

	Danube	Colorado			
		Top	Middle	Bottom	Bottom150
nSHD	0.88(0.60)	0.63(0.36)	0.73(0.41)	0.95(0.75)	0.80(0.65)
FPR	0.03(0.10)	0.08(0.18)	0.07(0.12)	0.05(0.12)	0.05(0.17)
FDR	0.90(0.60)	0.63(0.43)	0.73(0.52)	0.95(0.68)	0.80(0.68)
TPR	0.10(0.33)	0.38(0.57)	0.27(0.76)	0.05(0.12)	0.20(0.17)

Table 3.7: Metrics nSHD, FPR, FDR and TPR presented as in Table 3.3 for the maximum root-directed spanning tree estimated by Chu–Liu/Edmonds' algorithm with score matrix χ as in Sibuya [1960] or Coles et al. [1999].

	Danube	Colorado			
		Top	Middle	Bottom	Bottom150
nSHD	0.35(0.54)	0.50(0.26)	0.77(0.78)	0.92(0.54)	0.80(0.51)
FPR	0.02(0.21)	0.06(0.25)	0.07(0.33)	0.05(0.27)	0.06(0.30)
FDR	0.47(0.69)	0.50(0.45)	0.82(0.93)	0.95(0.69)	0.87(0.69)
TPR	0.53(0.48)	0.50(0.76)	0.18(0.18)	0.05(0.26)	0.13(0.28)

QTree for the Middle and Bottom sector of the Colorado. χ is worst for the Middle Colorado, for the Danube and the other sectors of the Colorado it performs better than Γ and S^{ext}.

We conclude that for the Danube as well as for the various sectors of the Colorado, `auto-tuned` QTree outperforms uniformly all algorithms without the stabilizing subsampling procedure; see Figure 3.11. Moreover, the QTree score outperforms the other scores when applying Chu-Liu/Edmonds' algorithm, therefore we conclude that the quantile-to-mean gap (3.6) is superior to the other scores on all data sets considered.

3.5 A small simulation study

Our main result, Theorem 1 ensures strong consistency of the output trees of QTree when the sample size n tends to infinity. In this section we show the quality of QTree through the two metrics nSHD and TPR for varying n and d by a small simulation study.

We generate data $\mathcal{X}$ from a max-linear Bayesian tree as defined in equation (3.3) with $|V| = d$ nodes. For each node i, we calculate the sample standard deviation $\hat{\sigma}_{X_i}$ of $(X_i^1, \dots, X_i^n)$ and take the sample median $\hat{\sigma}$ over all nodes $1, \dots, d$. We then generate i.i.d. normally distributed noise variables ε_i^t with mean zero and standard deviation $k \cdot \hat{\sigma}$ for $i \in V$ and $t = 1, \dots, n$. For the *noise-to-signal ratio* k, we choose $k = 30\%$.

We generate root-directed spanning trees as follows. We first generate a random undirected spanning of size d using the graph generators module `networkX` (Release

2.8.8) in `Python` [Hagberg et al., 2008]. We then choose the root node uniformly at random which uniquely determines the root-directed spanning tree. Finally, we assign edge weights c_{ij} independently.

For the distributions of the innovations $Z_1, \dots, Z_d$ and the edge weights c_{ij}, we consider the following three settings:

(1) Innovations $Z_1, \dots, Z_d$ are independent Gumbel$(1, 0)$ distributed and for every edge, we draw an edge weight c_{ij} from the interval $[\log(0.1), \log(1)]$ uniformly. We refer to this as the *standard Gumbel setting.*

(2) Innovations $Z_1, \dots, Z_d$ are independent Gumbel$(1, 0)$ distributed and for every edge, we draw an edge weight c_{ij} from the interval $[\log(0.1), \log(0.3)]$ uniformly. We refer to this as the *weak dependence setting.*

(3) 50% of the innovations $Z_1, \dots, Z_d$ are Gumbel$(1, 0)$ and 50% are $\mathcal{N}(0, 1)$. For every edge, we also draw an edge weight c_{ij} from the interval $[\log(0.1), \log(1)]$ uniformly. We refer to this as the *mixed distribution setting.*

For the score, we take the quantile-to-mean gap as in (3.6) (normalization by n_{ij} is not needed in a simulation setting) given by

$$w_{ij}(\underline{r}) := \Big(\mathbb{E}(\mathcal{X}_{ij}(\alpha)) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r})\Big)^2$$

and apply the QTree Algorithm 4 with parameters $\underline{r} = 0.05$ and $\alpha = 0$; we remark that a sensitivity analysis has shown that altering $\underline{r}$ influences the results only insignificantly.

We use graph sizes $d = 10, 30, 50, 100$ and 100 repetitions. For each repetition, we calculate the normalized Structural Hamming Distance (nSHD) and the True Positive Rate (TPR), and then take the mean over all 100 repetitions. For definitions of these metrics, see equation (3.1). Observe that both metrics are normalized to lie in the interval $[0, 1]$ and for nSHD smaller values are better, whereas for TPR larger values are better.

As QTree performs so well not only on simple data like those from the Danube network, but also on all sectors of the Colorado network, we guess that it is fairly robust towards the strength of dependence, given by the c_{ij} and even different node distributions. The weak dependence setting (2) should manifest whether QTree is also able to recover the underlying network if the dependence given by the weights c_{ij} is much much smaller. We want to quantify robustness towards node distributions with the mixed distribution setting (3).

Figures 3.12 and 3.13 depict the mean nSHD and TPR standard Gumbel setting (1) and all four graph sizes. Both metrics quickly tend to zero, respectively one, as the sample sizes n increase. Moreover, comparing the four subfigures for a fixed sample size n, the metrics perform only slightly worse for increasing graph size d.

Figures 3.22 and 3.23 in Section 3.7.3 depict the mean nSHD and TPR for the weak dependence setting (2) and all four graph sizes. Again, both metrics quickly tends to zero, respectively one. In comparison to the previous setting (1), it performs slightly worse.

Figures 3.24 and 3.25 in Section 3.7.3 depict the mean nSHD and TPR for the mixed distribution setting (3) and all four graph sizes. Despite the different distributions of the innovations, both metrics quickly tend to zero, respectively one. The performance compared to settings (1) and (2) is expectedly worse, however, less than perhaps could be expected.

The decrease in performance for increasing graph size d is presented in Figure 3.14, where we plot the minimum amount of data n needed to reach a mean nSHD of 10%. The first observation is that larger networks need a larger sample size to reach a nSHD below 10%. Since larger networks have more opportunities for a wrongly estimated causal influence, this is in line with what we expect. Moreover, the standard Gumbel setting (1) converges much faster than both other settings. Although the weights c_{ij} of setting (2) are in general much smaller than the weights of setting (1), only for a very large graph, substantially more data are needed to reach the lower bound for the nSHD. This implies that the smaller dependence impacts the estimation for a small graph only moderately, but for a larger graph more data are required. The mixed distribution setting (3), however, requires for increasing sample size substantially more data. This is also in line with our expectation as this makes the discrimination between signal and noise more difficult.

To summarize the results of our simulation study, QTree is sensitive to weaker dependence, but much more sensitive to the tail behavior of innovations/noise distributions.

We conclude that QTree works for sufficiently many observations very well even for limited data, across different dependence structures in the tree, and different distributions at the nodes.

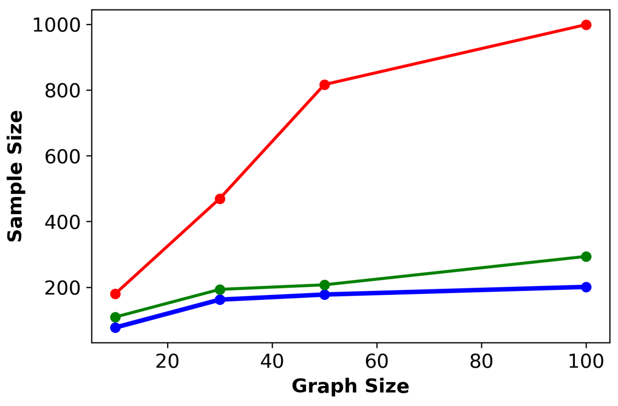


Figure 3.14: Minimum amount of observations needed to reach a mean nSHD of 10%. Blue line for the standard Gumbel setting (1), green line for the weak dependence setting (2), and red line for the mixed distribution setting (3). For the standard Gumbel setting (1) and its weak dependence version (2), increasing the graph size d, the amount of data needed only increases moderately to reach the same level of performance. The mixed distribution setting (2) requires for increasing sample size substantially more data.

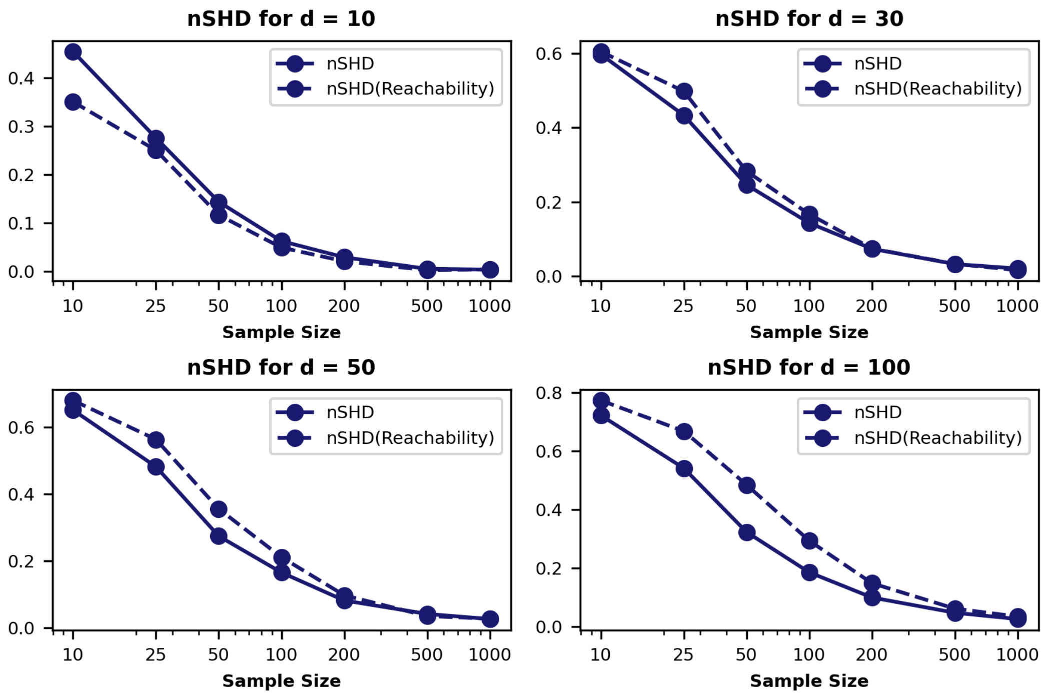


Figure 3.12: Mean nSHD for the standard Gumbel setting (1) and graph size $d = 10$ (top left), $d = 30$ (top right), $d = 50$ (bottom left), $d = 100$ (bottom right) and noise-to-signal ratio $k = 30\%$. Solid lines denote the metric for the pair $(\mathcal{T}, \hat{\mathcal{T}})$ while dashed lines denote the metric for its reachability pair $(\mathcal{R}, \hat{\mathcal{R}})$. For all graph sizes, the nSHD quickly converges to 0 as n increases. Increasing the graph size only moderately decreases the performance.

3.6 Summary

In this chapter, we proposed `auto-tuned QTree`, a new algorithmic solution to the Extremal River Problem—a benchmark problem for causal inference in extremes—combining the benefits of a new score matrix as input to Chu-Liu/Edmonds' algorithm with a stabilizing subsampling procedure. We also presented three new data sets of the Lower Colorado network for the Extremal River Problem, which are more challenging than the by now classic Upper Danube network data due to a large fraction of missing data and close spatial proximity between nodes. Across all four data sets, `auto-tuned QTree` performed very well, indeed better than previous state-of-the-art results. Our plug-and-play `Python` implementation in Tran [2021] can fit `QTree` on ten thousand observations in the range of 10 to 30 nodes on a personal laptop within half an hour. We proved that for a max-linear Bayesian network with Gumbel-Gaussian distributions for innovations and noise, the tree outputs of `QTree` are strongly consistent as the number of observations tends to infinity. Open research directions include (i) generalizations to learning directed acyclic graphs, (ii) better subsampling procedures with theoretical guarantees, and (iii) have the algorithm output a distribution over possible root-directed trees instead of a single best tree.

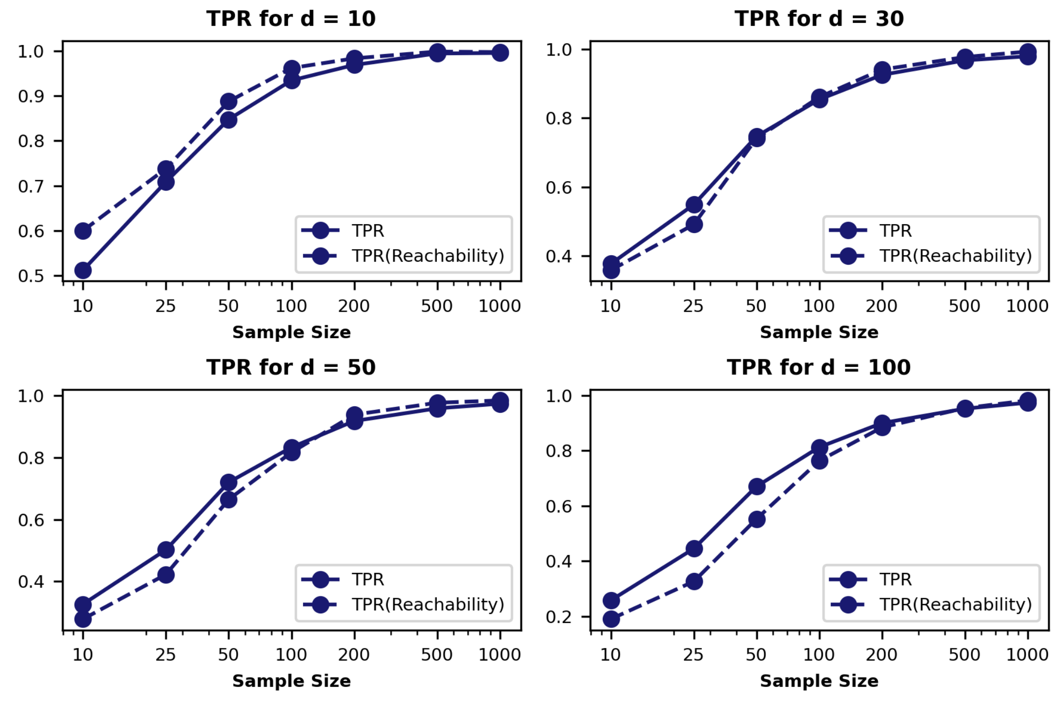


Figure 3.13: Mean TPR for the standard Gumbel setting (1) and graph size $d = 10$ (top left), $d = 30$ (top right), $d = 50$ (bottom left), $d = 100$ (bottom right) and noise-to-signal ratio $k = 30\%$. Solid lines denote the metric for the pair $(\mathcal{T}, \hat{\mathcal{T}})$ while dashed lines denote the metric for its reachability pair $(\mathcal{R}, \hat{\mathcal{R}})$. For all graph sizes, the TPR quickly converges to 1 as n increases. Increasing the graph size only moderately decreases the performance.

3.7 Supplementary Material

3.7.1 Proof of the complexity of QTree

We work with the solution of the max-linear Bayesian network on a tree $\mathcal{T} = (V, E)$ defined in eq. (3.3) [Baccelli et al., 1992, §3]; [Gissibl and Klüppelberg, 2018, Theorem 2.2]. Let $C^* = (c^*_{ij})$ be the matrix of longest paths, also known as the *Kleene star of* $C = (c_{ij})$. Then

$$X_i = \bigvee_{j:j \rightsquigarrow i \in \mathcal{T}} (c^*_{ij} + Z_j), \quad c^*_{ij}, Z_{ij} \in \mathbb{R}, \quad i \in V. \tag{3.13}$$

If there is no path $j \rightsquigarrow i$, then, by definition, $c^*_{ij} := -\infty$.

Lemma 3.7.1 concerns the noise-free case, and Lemma 3.7.2 concerns the noisy case.

Lemma 3.7.1. *Let $\mathcal{X} = \{x^1, \dots, x^n\}$ be i.i.d. observations from the max-linear model given by* (3.13), not *corrupted with noise. Assume that the Z_i are independent and have continuous distributions. Define*

$$\hat{c}_{ij} = \min_{x \in \mathcal{X}} (x_i - x_j). \tag{3.14}$$

Suppose that for each edge $j \to i$ such that $c_{ij} > -\infty$, there exist at least two *observations $x \in \mathcal{X}$ where j causes i. Then C^* can be uniquely recovered from $\hat{C}$*

since

$$\hat{c}_{ij} = c^*_{ij} \iff \text{min in (3.14) is achieved at least twice.} \tag{3.15}$$

In particular, C^* can be computed in time $O(|V|^2 n)$. If for every node, each of the parents is independently and equally likely to be the one that achieves the maximum, then the matrix C^* is recovered exactly for $n = O(|V|(\log(|V|))^2)$.

Proof. Since a root-directed spanning tree has at most one path between any pair of nodes (j, i) we have for an edge $j \to i$ that $c^*_{ij} = c_{ij}$. Furthermore, as indicated at the beginning of Section 3.2.2, if for an observation x the value at j causes that at i, then $x_i = c^*_{ij} + x_j$. If j does not cause i, then $x_i > c^*_{ij} + x_j$. Rearranging motivates the estimator (3.14) with properties as stated in [Gissibl et al., 2021, Proposition 1]. Equation (3.15) follows from [Gissibl et al., 2021, Lemma 1].

Now we prove the complexity claim. Since there are $O(|V|^2)$ many edges, and for each edge we need $O(n)$ operations to compute the minimum in (3.14), the complexity is $O(|V|^2 n)$. The number of observations needed, so that each edge is seen at least twice, is a variant of coupon-collecting [Boneh and Hofri, 1997, Boneh and Papanicolaou, 1996], where each node must collect two coupons (parents) among its set of parents. Since the nodes are collecting the coupons simultaneously, by the union bound, the number of observations needed is at most $\log(|V|)$ times the number of observations needed for the node with highest degree to collect all of its coupons, which in turn is $O(|V| \log(|V|))$. □

Lemma 3.7.2 (Complexity of QTree)**.** *QTree Algorithm 1 runs in time $O(|V|^2 n)$.*

Proof. For each pair $i, j \in V, i \neq j$, to estimate w_{ij}, one needs to compute the α-th quantile of $\mathcal{X}_j$, the $\underline{r}$-th quantile and the mean of $\mathcal{X}_{ij}(\alpha)$. Since α and $\underline{r}$ are fixed in advance, the empirical quantiles can be computed in time $O(n)$, see Musser [1997]. As there are $O(|V|^2)$ pairs, computing $W = (w_{ij})$ takes $O(|V|^2 n)$. Chu–Liu/Edmonds' algorithm runs on the complete bidirected graph supported by W, and thus takes $O(|V|^2)$, see Gabow et al. [1986]. So the complexity of QTree is $O(|V|^2 n + |V|^2) = O(|V|^2 n)$. □

3.7.2 Proof of the Consistency Theorem

In this section, we prove Theorem 3.2.4, which we recall here for ease of reference.

> **Gumbel-Gaussian noise model.** For $i \in V$, the innovations Z_i are i.i.d. Gumbel$(\beta, 0)$ (location 0 and scale β), the independent noise variables ε_i are i.i.d. with symmetric, light-tailed density f_ε satisfying
>
> $$f_\varepsilon(x) \sim e^{-Kx^p} \text{ as } x \to \infty, \tag{3.16}$$
>
> for some $p > 1$ and $\gamma, K > 0$ and the derivative of f_ε exists in the tail region. Throughout, for two functions a, b, positive in their right tails, we write $a(x) \sim b(x)$ as $x \to \infty$ for $\lim_{x\to\infty} a(x)/b(x) = c$, where $c > 0$ is some arbitrary constant.

Theorem 3.7.3 (Theorem 3.2.4). *Assume the Gumbel-Gaussian noise model.*
(a) There exists an $r^* > 0$ such that for any pair $0 < \underline{r} < \overline{r} < r^*$, the QTree *algorithm with score matrix $W = (w_{ij})$ defined as the lower quantile gap*

$$w_{ij}(\underline{r}, \overline{r}) := \frac{1}{n_{ij}} \left(Q_{\mathcal{X}_{ij}(\alpha)}(\overline{r}) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r}) \right)^2 \tag{3.17}$$

returns a strongly consistent estimator for the tree $\mathcal{T}$ as the sample size $n \to \infty$.
(b) There exists an $r^* > 0$ such that for any $0 < \underline{r} < r^*$, the QTree *algorithm with score matrix $W = (w_{ij})$ defined as the quantile-to-mean gap*

$$w_{ij}(\underline{r}) := \frac{1}{n_{ij}} \left(\mathbb{E}(\mathcal{X}_{ij}(\alpha)) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r}) \right)^2 \tag{3.18}$$

returns a strongly consistent estimator for the tree $\mathcal{T}$ as the sample size $n \to \infty$.

The proof of this theorem comes in a series of steps. Moreover, for simplicity, we omit the normalization by n_{ij} and the squaring in (3.17) and (3.18) as this leaves the proof unchanged. Also we set in the proof $\alpha = 0$.

As a preliminary result, Lemma 3.7.4 identifies a set of 'good' deterministic input matrices $W = (w_{ij})$, where if we apply the QTree algorithm to such an input, then it returns the true tree $\mathcal{T}$ *exactly.* The proof then reduces to the problem of proving that as $n \to \infty$, the matrices W_n derived from data converge a.s. to a 'good' W. Intuitively, W is 'good' if for each node j, the weight w_{ij} is smallest when i is the child of j. For the root we have a special explicit condition. For each fixed j, we split the set of node pairs $\{(j, i) : j, i \in V, i \neq j\}$ into three scenarios:

- $j \rightsquigarrow i$, that is, i is a descendant j in the true tree,
- $i \rightsquigarrow j$, that is, i is an ancestor of j in the true tree, and
- $i \not\rightsquigarrow j$, that is, i is neither of the above.

We first consider the case where $W = (w_{ij})$ is the matrix of lower quantile gaps (3.17) of the *true* distribution. Note that this W is no longer random. The goal is to show that if the true quantiles are known, then one can choose the parameters $(\underline{r}, \overline{r})$ such that W is good.

Next Proposition 3.7.5 gives an explicit representation for w_{ij} in each of the three scenarios above as the lower quantile gap of a certain family of distributions $(F^b : b \in \mathbb{R})$, parametrized by a *single parameter* b, one value for each edge $j \to i$. Then, we use a calculus of variation argument to detail how w_{ij} changes as b varies. This allows us to show (cf. Corollary 3.7.8 and Lemma 3.7.9) that among the three scenarios above, there exist some choices of quantile levels $(\underline{r}, \overline{r})$ such that for *any* fixed j, w_{ij} is smallest when i is the child of j in the true tree. A separate argument is made for the root. Thus, this proves that if the true quantiles are known, then the resulting W is good.

Finally, we invoke the fact that the empirical quantiles converge a.s. to the true quantiles as $n \to \infty$, and thus the empirical w_{ij} are a.s. close to the true ones. A union bound over the d nodes of the graph thus says that, the empirical W_n is a.s. 'good' as $n \to \infty$, and thus proves the Consistency Theorem for the lower quantile gap.

The proof for the quantile-to-mean gap is similar, with Proposition 3.7.11 playing the role of Proposition 3.7.5.

Lemma 3.7.4 (A criterion for 'good' inputs W)**.** *Let $W = (w_{ij})$ be a score matrix such that each true edge $j \to i \in \mathcal{T}$ satisfies*

$$w_{ij} < w_{i'j} \text{ for all } i' \in V, i' \neq i, j, \tag{3.19}$$

and in addition, the true root i^* satisfies

$$\min_{i'} w_{i'i^*} > \max_{i,j:j\to i} w_{ij}. \tag{3.20}$$

Then the QTree *algorithm applied to input W returns the true tree $\mathcal{T}$.*

Proof. QTree applies Chu–Liu/Edmonds' algorithm to find a minimum directed spanning tree from the complete graph with score matrix W, and returns that tree. We shall prove that under the conditions (3.19) and (3.20) on W, Chu–Liu/Edmonds' algorithm would converge after one iteration and returns the true tree $\mathcal{T}$. Indeed, let $\mathcal{G}$ denote the graph that consists of the smallest outgoing edge at each node. By (3.19), $\mathcal{G} = \mathcal{T} \cup i^* \to i'$ for some node $i' \in V$. By Chu–Liu/Edmonds' algorithm, the minimum spanning tree $\mathcal{T}_w$ is a subset of $\mathcal{G}$. In particular, $\mathcal{T}_w$ is a minimum spanning tree of $\mathcal{G}$. By (3.20), edge $i^* \to i'$ is the maximal edge. Since it belongs to the unique cycle in $\mathcal{G}$, deleting this edge would yield the minimum directed spanning tree of $\mathcal{G}$. Therefore $\mathcal{T}_w = \mathcal{T}$. ☐

3.7.2.1 Proof of Theorem 3.2.4 for the lower quantile gap

For known quantiles, W is 'good' for appropriate choices of $(\underline{r}, \overline{r})$

In this subsection we work with the lower quantile gap matrix $W = (w_{ij})$ derived from the *true* quantiles of the distributions of $X_i - X_j$ under the Gumbel-Gaussian model, for some quantile levels $(\underline{r}, \overline{r})$. The goal is to show that there exist some appropriate choices of $(\underline{r}, \overline{r})$ such that the resulting W is 'good', that is, it satisfies Lemma 3.7.4.

The first main result is Proposition 3.7.5, which gives an explicit representation for w_{ij} in the three scenarios. We start with the necessary definitions to state it.

Recall the definition of C^* from the beginning of Section 1. Since the true graph is a tree, if $j \rightsquigarrow i$, there is a unique directed path from j to i. Let $\bar{c}_{ij}$ denote the sum of all the edges along this unique path. Path uniqueness implies that $\bar{c}_{ij} = c^*_{ij}$ and C^* is transitive, i.e. $c^*_{ij} = c^*_{ik} + c^*_{kj}$ if $j \rightsquigarrow k \rightsquigarrow i$. Thus, by the Helmholtz decomposition on graphs [Lim, 2015, equation 2.6], c^*_{ij} is an edge flow. That is, there exists a unique $t^* \in \mathbb{R}^d$ with $t^*_1 = 0$ such that for all $j \to i \in \mathcal{G}$,

$$c^*_{ij} = t^*_i - t^*_j. \tag{3.21}$$

For each $i \in V$, define the constant

$$\theta_i := \sum_{k \rightsquigarrow i} \exp(-t^*_k/\beta). \tag{3.22}$$

For $b \in \mathbb{R} \cup \{-\infty\}$, define the random variable

$$\xi_b := (\varepsilon_i - \varepsilon_j) + ((Z_i - Z_j) \vee b) \tag{3.23}$$

with the convention that $\xi_{-\infty} := (\varepsilon_i - \varepsilon_j) + (Z_i - Z_j)$. Let F^b denote the distribution function of ξ_b and $q_r(F^b)$ the r-th quantile of F^b for $r \in (0,1)$. These quantities are deterministic and do not depend on i, j since by assumption, $\varepsilon_i, \varepsilon_j$ are i.i.d. and Z_i, Z_j are i.i.d.

Proposition 3.7.5. *Assume the Gumbel-Gaussian model. Fix $0 < \underline{r} < \overline{r} < 1$. Let $w_{ij} = w_{ij}(\underline{r}, \overline{r})$ be the lower quantile gap* (3.17)*. Fix $j \in V$. For $i \in V, i \neq j$, we have three cases.*

(1) If $j \rightsquigarrow i$, then $w_{ij} = q_{\overline{r}}(F^b) - q_{\underline{r}}(F^b)$ for $b = \beta(\log\theta_j - \log(\theta_i - \theta_j))$.

(2) If $j \not\rightsquigarrow i$, then $w_{ij} = q_{\overline{r}}(F^b) - q_{\underline{r}}(F^b)$ for $b = -\infty$.

(3) If $i \rightsquigarrow j$, then $w_{ij} = q_{1-\underline{r}}(F^b) - q_{1-\overline{r}}(F^b)$ for $b = \beta(\log\theta_i - \log(\theta_j - \theta_i))$.

Proof. We first consider the noise-free case. Observe that by (3.23) ξ_b simplifies to $(Z_i - Z_j) \vee b$. Therefore, it is sufficient to prove that w_{ij} equals the lower quantile gap of $(Z_i - Z_j) \vee b$. For $i \in V$, let $\bar{X}_i := X_i - t_i^*$. Then $\bar{X}_i - \bar{X}_j$ is a constant translation of $X_i - X_j$, so the lower quantile gap of the two corresponding distributions are the same. In other words, it is sufficient to prove the Proposition for $\bar{X}$ instead of X. Let $\bar{Z}_i := Z_i - t_i^*$. Then

$$\begin{aligned}
\bar{X}_i = X_i - t_i^* &= \bigvee_{j : j \rightsquigarrow i} (c_{ij}^* + Z_j) - t_i^* && \text{by (3.13)} \\
&= \bigvee_{j : j \rightsquigarrow i} (t_i^* - t_j^* + Z_j) - t_i^* && \text{by (3.21)} \\
&= \bigvee_{j : j \rightsquigarrow i} \bar{Z}_j. &&
\end{aligned} \tag{3.24}$$

For each ordered pair (i, j), define

$$S_i = \bar{Z}_i \vee \bigvee_{i' \neq i, i' \rightsquigarrow i, i' \not\rightsquigarrow j} \bar{Z}_{i'} \qquad\qquad S_j = \bar{Z}_j \vee \bigvee_{j' \neq j, j' \rightsquigarrow j} \bar{Z}_{i'} \tag{3.25}$$

In Figure 3.15 we illustrate the two index sets of the random variables S_i and S_j.

By definition, S_i and S_j are independent. Since $\bar{Z}_i$'s are translated independent Gumbel(β) by assumption, standard properties of the Gumbel(β) distribution yield that S_i and S_j are also translated independent Gumbel(β). The exact constants of translation depend on the relation between i and j, as this dictates the definition of S_i and S_j. Now we consider the three cases. In the first case, $j \rightsquigarrow i$. Then, (3.24) implies $\bar{X}_i = S_i \vee S_j$ and $\bar{X}_j = S_j$.

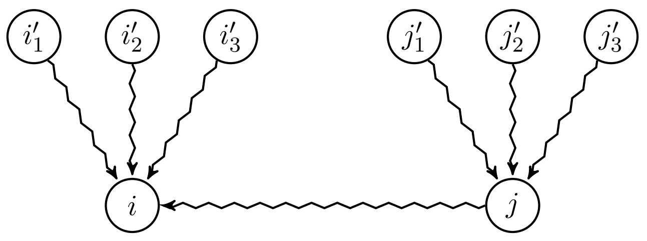


Figure 3.15: Illustration of the index sets of S_i and S_j for an ordered pair (i, j). The index set for S_i includes besides i also $i'_1, \dots, i'_3$ and all nodes on the paths $j \rightsquigarrow i$ (excluding j), $i'_k \rightsquigarrow i$ for $k = 1, \dots, 3$, while the index set for S_j includes $j, j'_1, \dots, j'_3$ and all nodes on the paths $j'_k \rightsquigarrow j$ for $k = 1, \dots, 3$.

A short computation yields $S_i \overset{d}{=} Z_i + \beta \log(\theta_i - \theta_j)$, $S_j \overset{d}{=} Z_j + \beta \log \theta_j$. Therefore, denoting $\overset{\mathrm{d}}{=}$ equality in distribution,

$$\begin{aligned}
\bar{X}_i - \bar{X}_j &= (S_i \vee S_j) - S_j = (S_i - S_j) \vee 0 \\
&\overset{d}{=} (Z_i - Z_j - \beta(\log \theta_j - \log(\theta_i - \theta_j))) \vee 0 \\
&= ((Z_i - Z_j) \vee \beta(\log \theta_j - \log(\theta_i - \theta_j))) - \beta(\log \theta_j - \log(\theta_i - \theta_j)) \\
&= ((Z_i - Z_j) \vee b) - \beta(\log \theta_j - \log(\theta_i - \theta_j)),
\end{aligned}$$

where $b = \beta(\log \theta_j - \log(\theta_i - \theta_j))$. Since $\beta(\log \theta_j - \log(\theta_i - \theta_j))$ is a translation constant, the quantile gap of $\bar{X}_i - \bar{X}_j$ is equal to the quantile gap of $(Z_i - Z_j) \vee b$. This concludes the case $j \rightsquigarrow i$. Computations for the third case, $i \rightsquigarrow j$, is similar, with the role of i and j reversed, $\underline{r}$ is replaced by $1 - \overline{r}$, and $\overline{r}$ is replaced by $1 - \underline{r}$. For the second case, $i \not\rightsquigarrow j$, then $\bar{X}_i = S_i, \bar{X}_j = S_j$, where $S_j \overset{d}{=} \beta \log \theta_j + Z_j$ and $S_i \overset{d}{=} \beta \log \theta_i + Z_i$. Then

$$\bar{X}_i - \bar{X}_j = S_i - S_j \overset{d}{=} Z_i - Z_j + \beta(\log \theta_i - \log \theta_j).$$

Since $\beta(\log \theta_i - \log \theta_j)$ is a translation constant, the quantile gap of $\bar{X}_i - \bar{X}_j$ is equal to the quantile gap of $Z_i - Z_j$, as claimed. □

How the lower quantile gap w_{ij} varies with b

Now, we aim to show through a variational argument that under the Gumbel-Gaussian assumption, among the three scenarios of Proposition 3.7.5, w_{ij} is smallest when it falls in a subset of case (1), namely, $j \to i$. We first give an overview. By Proposition 3.7.5, the lower quantile gaps w_{ij} in cases (1) and (2) are all of the form $q(b, \overline{r}) - q(b, \underline{r})$ for some constant $b = b(i, j)$. In particular, for fixed j, $b(i, j)$ is largest when $j \to i$. Lemma 3.7.7 says that one can choose the quantile levels $(\underline{r}, \overline{r})$ such that $q(b, \overline{r}) - q(b, \underline{r})$ is monotone *increasing* as a function of b on a large interval. Corollary 3.7.8 then shows that a good choice can be made so that for each fixed j, the quantile gap is smallest for the edge from j to its child $ch(j)$. Case (3) of Proposition 3.7.5, where i is an ancestor of j, is handled by Lemma 3.7.9. The Gumbel-Gaussian assumption comes in through Lemma 3.7.6, which is a technical result that gives an explicit form for the density of the noise differences $\eta := \varepsilon_i - \varepsilon_j$. Intuitively, it shows that under the Gumbel-Gaussian model, the tail of η is lighter than the tail of the signal differences $Z_i - Z_j$. This is a key observation exploited in the proofs.

Lemma 3.7.6. *Under the Gumbel-Gaussian model, for any pair of nodes $i, j \in V, i \neq j$, $\xi := Z_i - Z_j$ has density*

$$f_\xi(x) = \frac{e^{x/\beta}}{\beta(1+e^{x/\beta})^2} \sim \frac{1}{\beta} e^{-x/\beta} \text{ as } x \to \infty, \tag{3.26}$$

and $\eta := \varepsilon_i - \varepsilon_j$ has density

$$f_\eta(x) \sim x^{1-p/2}\, e^{-Kx^p} \text{ as } x \to \infty. \tag{3.27}$$

Proof. Computing the convolution integral yields

$$\mathbb{P}(Z_i - Z_j > x) = \frac{1}{1+e^{x/\beta}}, \quad x \in \mathbb{R}, \tag{3.28}$$

and taking the derivative gives the first statement. For the second statement, the density f_ε is a density with Gaussian tail in the sense of Balkema et al. [1993]:

$$f(x) \sim \gamma(x) e^{-\psi(x)} \text{ as } x \to \infty,$$

for constants γ and $\psi(x) = Kx^p$. The asymptotic form of f_η follows by Laplace's integration principle as shown in [Balkema et al., 1993, page 2]. □

Since f_ε is differentiable in the tail, f_η is also differentiable in the tail, and differentiation of (3.27) yields the following formula for the derivative:

$$f'_\eta(x) \sim f_\eta(x)\Big(- Kpx^{p-1} + (1-p/2)x^{-1}\Big) \tag{3.29}$$

For functions with two arguments, let ∂_1 denotes the derivative in the first argument, ∂_2 denotes the derivative in the second argument, $\partial^2_{12} := \partial_1\partial_2$ denote the mixed second derivatives and so forth. Define the functions $H : \mathbb{R} \times \mathbb{R} \to [0,1]$, $q : \mathbb{R} \times [0,1] \to \mathbb{R}$ by

$$H(b,a) = P(\xi_b \leq a), \quad q(b,r) = r\text{-th quantile of } \xi_b.$$

Lemma 3.7.7. *Under the Gumbel-Gaussian model, for each finite constant B, there exists some $r^* = r^*(B) \in (0,1)$ such that*

$$\partial^2_{12} q(b,r) < 0 \quad \text{for all } r \in (0, r^*), b \leq B.$$

Equivalently, for any pair $(\underline{r}, \overline{r})$ such that $0 < \underline{r} < \overline{r} < r^*$ and any pair (b', b) such that $b' < b \leq B$,

$$q(b, \overline{r}) - q(b, \underline{r}) < q(b', \overline{r}) - q(b', \underline{r}). \tag{3.30}$$

Proof. By definition,

$$H(b, q(b,r)) = r. \tag{3.31}$$

We take derivatives of both sides, first with respect to r, then to b. Note that functions and derivatives of H are always evaluated at $(b, q(b,r))$ while those of q are evaluated at (b,r), so we suppress them in the notations. Differentiate both sides sof (3.31) with respect to r gives

$$\partial_2 H \cdot \partial_2 q = 1. \tag{3.32}$$

Now, differentiating both sides of (3.31) with respect to b, we get

$$\frac{\partial}{\partial b} H_1(b, q(b,r)) = \partial_1 H + \partial_2 H \cdot \partial_1 q = 0,$$

therefore,

$$\partial_1 q = \frac{-\partial_1 H}{\partial_2 H}. \tag{3.33}$$

Differentiate (3.32) with respect to b using implicit differentiation and chain rules, we get

$$\begin{aligned} 0 = \frac{\partial}{\partial b}(\partial_2 H \cdot \partial_2 q) &= \frac{\partial}{\partial b}(\partial_2 H(b, q(b,r)) \cdot \partial_2 q + \partial_2 H \cdot \partial^2_{12} q \\ &= (\partial^2_{12} H + \partial^2_{22} H \cdot \partial_1 q) \cdot \partial_2 q + \partial_2 H \cdot \partial^2_{12} q \\ &= \frac{\partial^2_{12} H - \partial^2_{22} H \cdot \frac{\partial_1 H}{\partial_2 H}}{\partial_2 H} + \partial_2 H \cdot \partial^2_{12} q \quad \text{by (3.32) and (3.33).} \end{aligned} \tag{3.34}$$

Rearranging the last equation gives

$$\partial^2_{12} q = \frac{\partial^2_{22} H \cdot \partial_1 H - \partial^2_{12} H \cdot \partial_2 H}{(\partial_2 H)^3}. \tag{3.35}$$

For fixed b, by definition of H, $\partial_2 H$ is the density of ξ_b, so $\partial_2 H > 0$. So $\partial^2_{12} q(b,r) < 0$ if and only if

$$(\partial^2_{22} H \partial_1 H - \partial^2_{12} H \partial_2 H)(b, q(b,r)) < 0. \tag{3.36}$$

Now we compute each of the terms $\partial_2 H, \partial_1 H, \partial^2_{12} H$ and $\partial^2_{22} H$ in the LHS of (3.36) explicitly in terms of the density f_η of the noise difference $\eta = \varepsilon_i - \varepsilon_j$. Note that $\xi_b = \eta + (\xi \vee b)$ where $\xi := Z_i - Z_j$. Then we have for $\varepsilon > 0$ (see Fig. (a))

$$H(b+\varepsilon, a) - H(b,a) = \mathbb{P}(\eta + \xi \vee (b+\varepsilon) \le a) - \mathbb{P}(\eta + \xi \vee b \le a)$$

$$= \begin{cases} 0 & \text{if } \xi > b + \varepsilon, \\ \mathbb{P}(\eta + b + \varepsilon \le a) - \mathbb{P}(\eta + b \le a) & \text{if } \xi \le b \quad \text{(light shaded)}, \\ \mathbb{P}(\eta + b + \varepsilon \le a) - \mathbb{P}(\eta + \xi \le a) & \text{if } b \le \xi \le b + \varepsilon \quad \text{(dark shaded)}. \end{cases}$$

Since η and ξ are independent, this implies

$$H(b+\varepsilon, a) - H(b,a) = -\mathbb{P}(\xi \le b)\mathbb{P}(a - b - \varepsilon \le \eta \le a - b) + O(\varepsilon^2). \tag{3.37}$$

Hence,

$$\partial_1 H(b,a) = \lim_{\varepsilon \downarrow 0} \frac{H(b+\varepsilon, a) - H(b,a)}{\varepsilon} = -\mathbb{P}(\xi \le b) f_\eta(a-b). \tag{3.38}$$

A similar calculation gives (see Fig. (b))

$$\begin{aligned} \partial_2 H(b,a) = \lim_{\varepsilon \downarrow 0} \frac{H(b, a+\varepsilon) - H(b,a)}{\varepsilon} &= \mathbb{P}(\xi \le b) f_\eta(a-b) + \int_b^\infty f_\eta(a-x) f_\xi(x)\, dx \\ =: (-\partial_1 H + A)(b,a). \end{aligned} \tag{3.39}$$

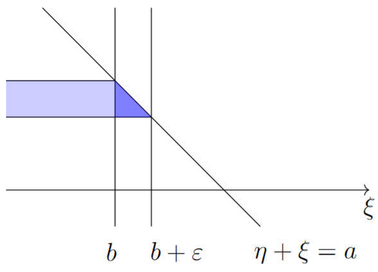


Figure 3.16: $H(b+\varepsilon,a)-H(b,a)$ is the probability that (ξ,η) lies in the shaded regions (light + dark shaded)

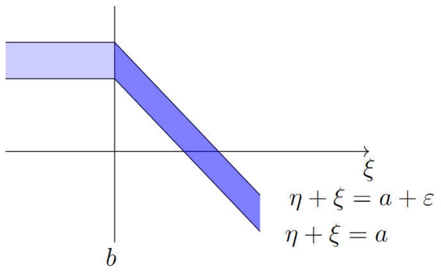


Figure 3.17: $H(b,a+\varepsilon)-H(b,a)$ is the probability that (ξ,η) lies in the shaded region (light + dark shaded)

Thus,

$$\begin{aligned}\partial^2_{12}H(b,a) &= -\mathbb{P}(\xi\le b)f'_\eta(a-b),\\ \partial^2_{22}H(b,a) &= \mathbb{P}(\xi\le b)f'_\eta(a-b)+\int_b^\infty f'_\eta(a-x)f_\xi(x)\,dx\\ &= -\partial^2_{12}H(b,a)+\int_b^\infty f'_\eta(a-x)f_\xi(x)\,dx.\\ &= (-\partial^2_{12}H+A_a)(b,a).\end{aligned}$$

Now we have

$$\begin{aligned}&(\partial^2_{22}H\partial_1H-\partial^2_{12}H\partial_2H)(b,a)=(A_a\partial_1H-A\partial^2_{12}H)(b,a)\\ =&\mathbb{P}(\xi\le b)\left(-f_\eta(a-b)\int_b^\infty f'_\eta(a-x)f_\xi(x)\,dx+f'_\eta(a-b)\int_b^\infty f_\eta(a-x)f_\xi(x)\,dx\right).\end{aligned}$$

Thus, (3.36) holds if and only if

$$-f_\eta(q(b,r)-b)\int_b^\infty f'_\eta(q(b,r)-x)f_\xi(x)\,dx+f'_\eta(q(b,r)-b)\int_b^\infty f_\eta(q(b,r)-x)f_\xi(x)\,dx<0. \tag{3.40}$$

Now we need to show that for each constant B, there exists an $r^*(B)>0$ such that for all $r<r^*$ and $b\le B$, (3.40) holds. Fix the constant B. Since the noise ε has no upper bound, η and hence ξ_b have unbounded support below. For each fixed b, $q(b,r)\to-\infty$ as $r\downarrow 0$. Therefore, there exists some sufficiently small $r^*>0$ such that for all $r<r^*$, $q(b,r)-b$ is a large negative number. Fix such an r^*. Therefore, for all $x>b$, $q(b,r)-x$ is a large negative number. This allows us to use Lemma 3.7.6 to make the LHS of (3.40) explicit. In particular, by (3.27), as $t\to\infty$,

$$f_\eta(t)=K_1t^{1-p/2}e^{-Kt^p}(1+o(1)),\quad f'_\eta(t)=f_\eta(t)\Big(-Kpt^{p-1}+(1-p/2)t^{-1}\Big)(1+o(1)). \tag{3.41}$$

Setting $t = |x - q(b,r)|$ and use the fact that f_η is symmetric, $f_\eta(q(b,r) - x) = f_\eta(|x - q(b,r)|)$, we have

$$\begin{aligned}
&- f_\eta(q(b,r) - b) \int_b^\infty f'_\eta(q(b,r) - x) f_\xi(x)\, dx + f'_\eta(q(b,r) - b) \int_b^\infty f_\eta(q(b,r) - x) f_\xi(x)\, dx \\
=& f_\eta(q(b,r) - b) \int_b^\infty [-Kp(x - q(b,r))^{p-1} + (1 - p/2)(x - q(b,r))^{-1}] f_\eta(q(b,r) - x) f_\xi(x)\, dx \\
&+ [Kp(b - q(b,r))^{p-1} - (1 - p/2)(b - q(b,r))^{-1}] f_\eta(q(b,r) - b) \int_b^\infty f_\eta(q(b,r) - x) f_\xi(x)\, dx \\
=& f_\eta(q(b,r) - b) \int_b^\infty A(x) f_\eta(q(b,r) - x) f_\xi(x)\, dx,
\end{aligned}$$

where

$$A(x) = \Big(Kp[(b-q(b,r))^{p-1} - (x-q(b,r))^{p-1}] - (1-p/2)[(b-q(b,r))^{-1} - (x-q(b,r))^{-1}]\Big).$$

If $p \in (1,2]$, since $x > b$, the first term $(b - q(b,r))^{p-1} - (x - q(b,r))^{p-1}$ and the second term $-(1 - p/2)[(b - q(b,r))^{-1} - (x - q(b,r))^{-1}]$ are both negative. If $p > 2$, since $x > b \gg q(b,r)$ the first term $(b - q(b,r))^{p-1} - (x - q(b,r))^{p-1}$ is a large negative number. Since $q(b,r)$ is a large negative number, $b - q(b,r)$ is a large positive number, so $(b - q(b,r))^{-1}, (x - q(b,r))^{p-1} < 1$. Thus $|(1-p/2)[(b-q(b,r))^{-1} - (x-q(b,r))^{-1}]| \leq |1 - p/2|$. For this reason, $A(x) < 0$ for all $p > 1$, $x > b$, while $f_\eta, f_\xi > 0$ everywhere as they are densities. Thus the integral is negative, that is, (3.40) holds for all $r \in (0, r^*)$ and $b \leq B$, as needed. □

Below we denote $ch(j)$ the child of node j.

Corollary 3.7.8. *Under the Gumbel-Gaussian model, there exists an $r_1^* > 0$ such that: for all $0 < \underline{r} < \overline{r} < r_1^*$, for all $j \in V$ and for all $i' \in V, i' \neq j, ch(j)$ and either $j \rightsquigarrow i'$ or $j \not\rightsquigarrow i'$, then*

$$w_{ch(j)\,j} < w_{i'j}.$$

Proof. It is sufficient to show that the above holds with some constant $r^*(j)$ for each fixed j, then set $r_1^* = \min_j r^*(j)$. Fix j and i' as stated. Let $b^* := \beta(\log\theta_j - \log(\theta_{ch(j)} - \theta_j))$, and let $r^*(j)$ be the constant r^* that works for $B = b^*$ in Lemma 3.7.7. By Proposition 3.7.5,

$$w_{ch(j)\,j} = q(b^*, \overline{r}) - q(b^*, \underline{r}).$$

Now we consider two cases.
Case 1: i' is a descendant of j, that is, $j \rightsquigarrow i'$. Then by Proposition 3.7.5,

$$w_{ji'} = q(b, \overline{r}) - q(b, \underline{r})$$

where $b = \beta(\log\theta_j - \log(\theta_{i'} - \theta_j))$. But since $i' \neq ch(j)$, i' must be a descendant of i as well. By definition of θ's in (3.22), $i \rightsquigarrow i'$ implies $\theta_{i'} > \theta_i$. Therefore, $b < b^*$, so by (3.30), $w_{ij} < w_{i'j}$. This concludes case 1.
Case 2: $j \not\rightsquigarrow i'$. Then by Proposition 3.7.5,

$$w_{i'j} = q(-\infty, \overline{r}) - q(-\infty, \underline{r}).$$

Since $-\infty < b^*$, so by (3.30), $w_{ij} < w_{i'j}$. This concludes case 2. □

Lemma 3.7.9. *There exists some $r_2^* > 0$ such that for all $0 < \underline{r} < \overline{r} < r_2^*$, for all $j \in V$, $i' \rightsquigarrow j$ implies*

$$q(b, \overline{r}) - q(b, \underline{r}) < q(b', 1 - \underline{r}) - q(b', 1 - \overline{r}), \tag{3.42}$$

where $b = \beta(\log \theta_j - \log(\theta_{ch(j)} - \theta_j))$ and $b' = \beta(\log \theta_{i'} - \log(\theta_j - \theta_{i'}))$. In particular, if $i' \rightsquigarrow j$, then for all quantile levels $\underline{r}, \overline{r}$ such that $0 < \underline{r} < \overline{r} < r_2^*$,

$$w_{ch(j)\,j} < w_{i'j}. \tag{3.43}$$

Proof. It is sufficient to prove that (3.42) holds for each fixed j with some constant $r_2^*(j)$, and then set $r_2^* = \min_j r_2^*(j)$. Fix j. First, we do some manipulations on (3.42) to relate it to the partial derivatives of H. Define

$$\mathcal{B} := \{\beta(\log \theta_j - \log(\theta_i - \theta_j)) : i, j \in V, j \rightsquigarrow i\}. \tag{3.44}$$

Note that (3.42) is equivalent to

$$\partial_2 q(b, r) < \partial_2 q(b', 1 - r) \quad \text{for all } r \in (0, r_2^*) \text{ and for all } b, b' \in \mathcal{B}. \tag{3.45}$$

By (3.32), we have

$$\partial_2 q(b, r) - \partial_2 q(b', 1 - r) = \frac{1}{\partial_2 H(b, q(b, r))} - \frac{1}{\partial_2 H(b', q(b', 1 - r))}.$$

By (3.39), $\partial_2 H > 0$ point-wise, thus our goal now is to show that for sufficiently small r,

$$\partial_2 H(b', q(b', 1 - r)) - \partial_2 H(b, q(b, r)) < 0 \tag{3.46}$$

for all $b, b' \in \mathcal{B}$, that is, some finite set of constants.We shall do this by writing $\partial_2 H$ in terms of the tail densities f_η and f_ξ using (3.39), then apply Lemma 3.7.6. Indeed, by (3.39),

$$\partial_2 H(b', a) = \mathbb{P}(\xi \le b') f_\eta(a - b') + \int_{b'}^{\infty} f_\eta(a - x) f_\xi(x)\, dx$$

By Lemma 3.7.6, f_ξ has heavier tail than f_η, so for $a \to \infty$, the main contribution from $\int_{b'}^{\infty} f_\eta(a - x) f_\xi(x)\, dx$ comes from $f_\xi(a)$. That is, for large a, there exists some constant $b_1 > 0$ such that

$$\partial_2 H(b', a) > b_1 f_\xi(a). \tag{3.47}$$

Now we consider $\partial_2 H(b, -a)$. From (3.39),

$$\partial_2 H(b, -a) = \mathbb{P}(\xi \le b) f_\eta(-a - b) + \int_{b}^{\infty} f_\eta(-a - x) f_\xi(x)\, dx.$$

Again, for large a

$$f_\eta(-a - x) < f_\eta(-a - b) \text{ for all } x > b.$$

Therefore, we can bound the second term above as

$$\int_b^{\infty} f_\eta(-a - x) f_\xi(x)\, dx < f_\eta(-a - b) \int_b^{\infty} f_\xi(x)\, dx = f_\eta(-a - b) \mathbb{P}(\xi > b).$$

Adding in the first term, we get that for large a,

$$\partial_2 H(b, -a) < f_\eta(-a - b)$$

Combining this with (3.47) and noting that $\partial_2 H(b, a)$ is just the density $f_{\xi_b}(a)$ of ξ_b, we get

$$f_{\xi_b}(-a) = O(f_{\xi_{b'}}(a)) \tag{3.48}$$

for all $b, b' \in \mathcal{B}$ and a large. Now $\partial_2 H(b, q(b, r))$ is just the slope of the cdf of f_{ξ_b} at its r-th quantile. Therefore, for r small, by (3.48), $\partial_2 H(b', q(b', 1 - r)) < \partial_2 H(b, q(b, r))$ which proves (3.46) and thus completes the proof of (3.42). The last statement follows from Proposition 3.7.5, case (3). □

Corollary 3.7.10. *If the true quantiles are known, then there exist some choices of* $(\underline{r}, \overline{r})$ *such that the lower quantile gap matrix* W *satisfies the conditions of Lemma 3.7.4, that is,* (3.19) *and* (3.20)*.*

Proof. Set $r^* = \min(r_1^*, r_2^*)$ where r_1^* comes from Corollary 3.7.8, and r_2^* comes from Lemma 3.7.9. Let $(\underline{r}, \overline{r})$ be any pair such that $0 < \underline{r} < \overline{r} < r^*$, and let W be the corresponding lower quantile gap matrix with the true quantiles. Then (3.20) holds because of (3.43) and the fact that for the root r of the root-directed spanning tree, $i' \rightsquigarrow r$ holds for every $i' \neq r$. Corollary 3.7.8 and Lemma 3.7.9 together guarantee that (3.19) is satisfied for W. □

Proof of Theorem 3.2.4 for the lower quantile gap

Fix $(\underline{r}, \overline{r})$ such that Corollary 3.7.10 holds, and let W be the corresponding lower quantile gap matrix derived from the true quantiles. Let W_n be the lower quantile gap matrix derived from an empirical distribution with sample size n. Note that the set of 'good' matrices, that is, those that satisfy Lemma 3.7.4, is an open polyhedral cone in the space of matrices $\mathbb{R}^{d \times d}$, since the conditions of 'goodness' is a set of linear inequalities. By Corollary 3.7.10, W is a point inside this cone. Recall that empirical quantiles converge a.s. as $n \to \infty$ to the true ones for continuous limit distributions, hence, also the empirically-derived lower quantile gap converges a.s.. By a union bound over the $d^2 - d$ possible edge pairs (i, j), for any metric D (e.g. induced by a matrix norm), we thus have $D(W_n, W) \to 0$ a.s. The Consistency Theorem then follows from Lemma 3.7.4. □

3.7.2.1 Proof of Theorem 3.2.4 for the quantile-to-mean gap

Our proof follows the same structure as the previous proof, but the calculations in all steps are a bit simpler, since there is only one quantile parameter to deal with. First, expectation is linear, so we work with empirical means $\bar{X}_i$ for $i \in V$ and mention in passing that they converge a.s. to the true mean as $n \to \infty$. The analog of Proposition 3.7.5 is the following.

Proposition 3.7.11. *Fix* $\underline{r} \in [0, 1)$*, and let* w_{ij} *be the quantile-to-mean gap* (3.18)*. Assume the Gumbel-Gaussian model. Then*

(1) If $j \rightsquigarrow i$, then $w_{ij} = -q_{\underline{r}}(\xi^b)$ where $b = \beta(\log\theta_j - \log(\theta_i - \theta_j))$.

(2) If $j \not\rightsquigarrow i$, then $w_{ij} = -q_{\underline{r}}(\xi^b)$ where $b = -\infty$.

(3) If $i \rightsquigarrow j$, then $w_{ij} = q_{1-\underline{r}}(\xi^b)$ where $b = \beta(\log\theta_i - \log(\theta_j - \theta_i))$.

Instead of a lengthy proof of the analog of Proposition 3.7.5 by duplicating arguments, we provide some informal reasoning. We check that our quantile-to-mean gaps w_{ij} satisfy the inequalities of Corollary 3.7.8 and Lemma 3.7.9 by first checking the noise-free case, where $\varepsilon_i \equiv \varepsilon_j \equiv 0$. We consider the three cases of Proposition 3.7.11.

1. If $j \rightsquigarrow i$. Then ξ^b has a left-most atom at $b = \beta(\log\theta_j - \log(\theta_i - \theta_j))$, so for sufficiently small r, $w_{ij} = -b$. This is minimal when i is a direct descendant of j. So Corollary 3.7.8 for the case $j \rightsquigarrow i$ holds in the noise-free case.

2. If $j \not\rightsquigarrow i$. Then ξ^b has no left-most atom, so as $\underline{r} \downarrow 0$, $q_{\underline{r}}(\xi^b) \to -\infty$, so $w_{ij} \to \infty$. So Corollary 3.7.8 also holds in the noise-free case for the remaining case, $j \not\rightsquigarrow i$.

3. If $i \rightsquigarrow j$. Then ξ^b has a left-most atom, but no right-most atom. Again, as $\underline{r} \downarrow 0$, $q_{1-\underline{r}}(\xi^b) \to \infty$, so $w_{ij} \to \infty$. Thus, Lemma 3.7.9 holds in the noise-free case.

Now we consider the effect of noise. We send $\underline{r} \downarrow 0$. As long as $\eta := \varepsilon_i - \varepsilon_j$ has lighter tail than $Z_i - Z_j$, as guaranteed by Lemma 3.7.6, then we have the following.

- In case (1), $q_{\underline{r}}(\xi^b)$ is dominated by the lower tail of η.
- In case (2), $q_{\underline{r}}(\xi^b)$ is dominated by the lower tail of $Z_i - Z_j$ and, in particular, is going to $-\infty$ at a faster rate than case (1).
- In case (3), $q_{1-\underline{r}}(\xi^b)$ is dominated by the upper tail of $Z_i - Z_j$, and in particular, is going to ∞ at a faster rate than case (1).

This domination calculation is the same calculation done in the proof of Lemma 3.7.9. The above says that for fixed j, for small enough $\underline{r}$, the minimum of $\{w_{ij} : i \neq j, i \in V\}$ lies in case (1). Within case (1), we want to make sure that, if w_{ij} is smallest, then i is the child of j. Indeed, write

$$\xi^b = \varepsilon_i - \varepsilon_j + \xi'_{ij}$$

where $\xi'_{ij} = (Z_i - Z_j) \vee (\beta(\log\theta_j - \log(\theta_i - \theta_j)))$. For fixed j, $(\xi'_{ij} : j \rightsquigarrow i)$ is a particular family of distribution indexed by i. By a decoupling argument, it is sufficient to show that $q_{\underline{r}}(\xi'_{ij})$ is smallest when i is the child of j. But this reduces to the noise-free case, which we already proved above. This finishes the proof of Theorem 3.2.4 for the quantile-to-mean gap. □

3.7.3 Supplementary Figures

Below we present the figures analogous to Figure 3.10 for the different data sets of the Colorado river network.

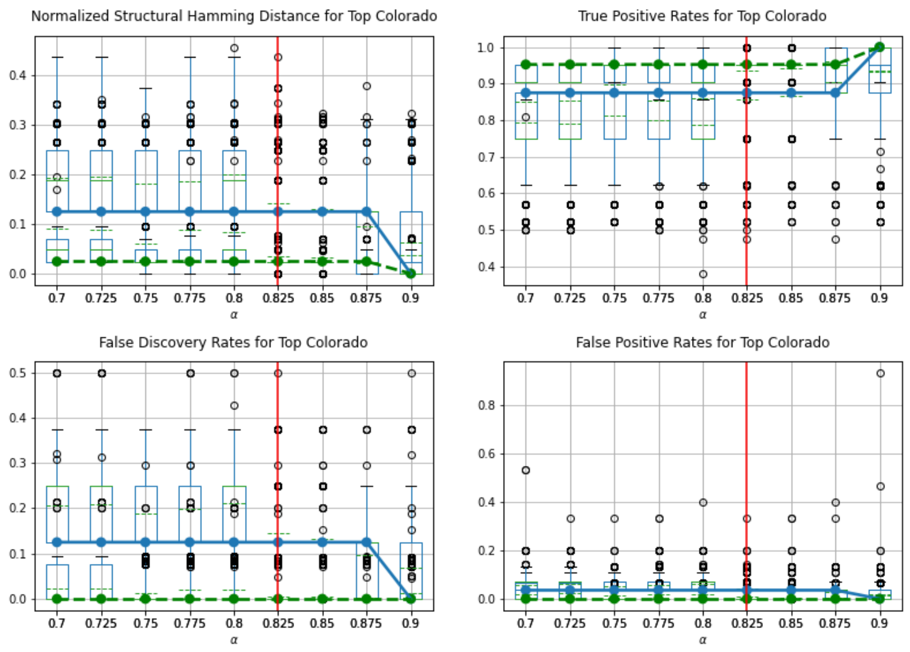


Figure 3.18: Metrics nSHD, TPR, FDR and FPR for the Top sector of the Colorado network and varying parameters α. For detailed explanations see Figure 3.10.

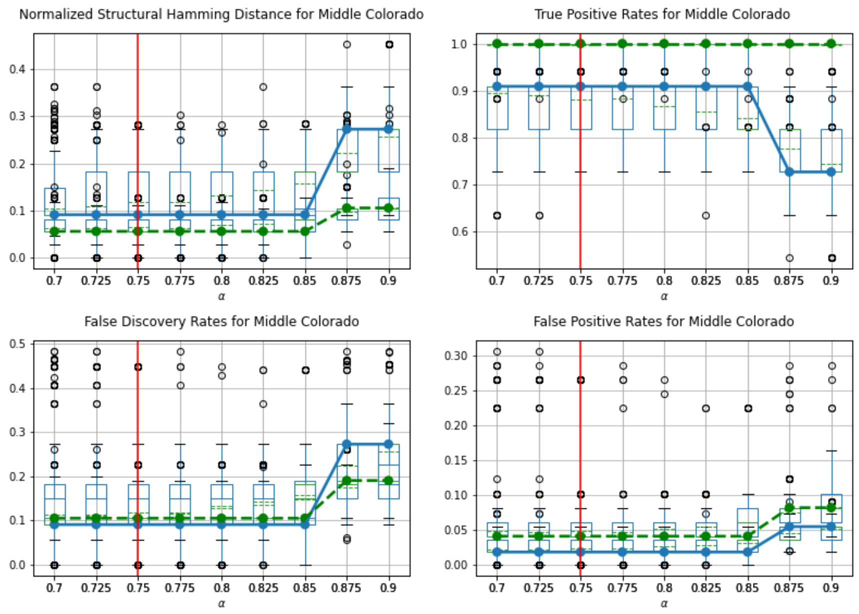


Figure 3.19: Metrics nSHD, TPR, FDR and FPR for the Middle sector of the Colorado network and varying parameters α. For detailed explanations see Figure 3.10.

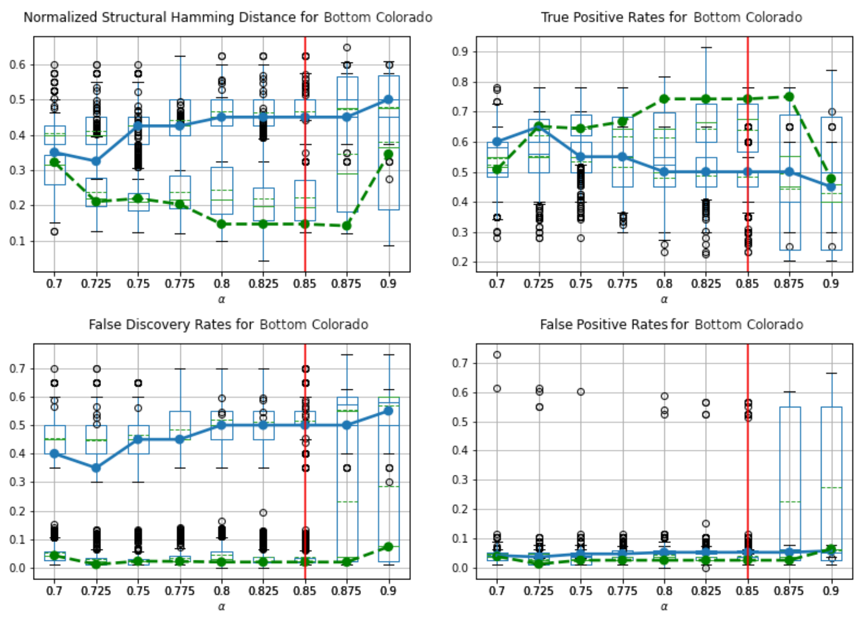


Figure 3.20: Metrics nSHD, TPR, FDR and FPR for the Bottom sector of the Colorado network and varying parameters α. For detailed explanations see Figure 3.10.

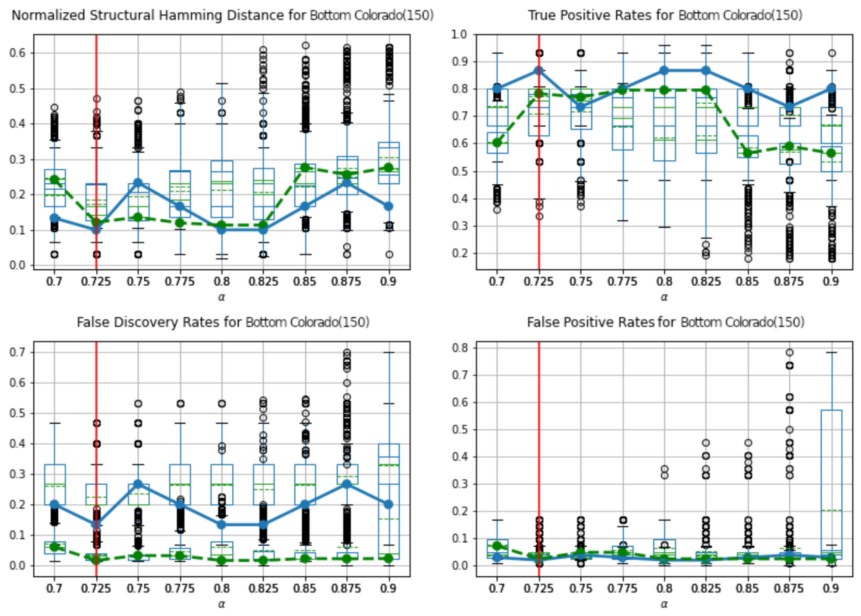


Figure 3.21: Metrics nSHD, TPR, FDR and FPR for Bottom150 of the Colorado network and varying parameters α. For detailed explanations see Figure 3.10.

Below we visualize the performance measures nSHD and TPR as defined in (3.1) from simulations of settings (2) and (3) of Section 3.5.

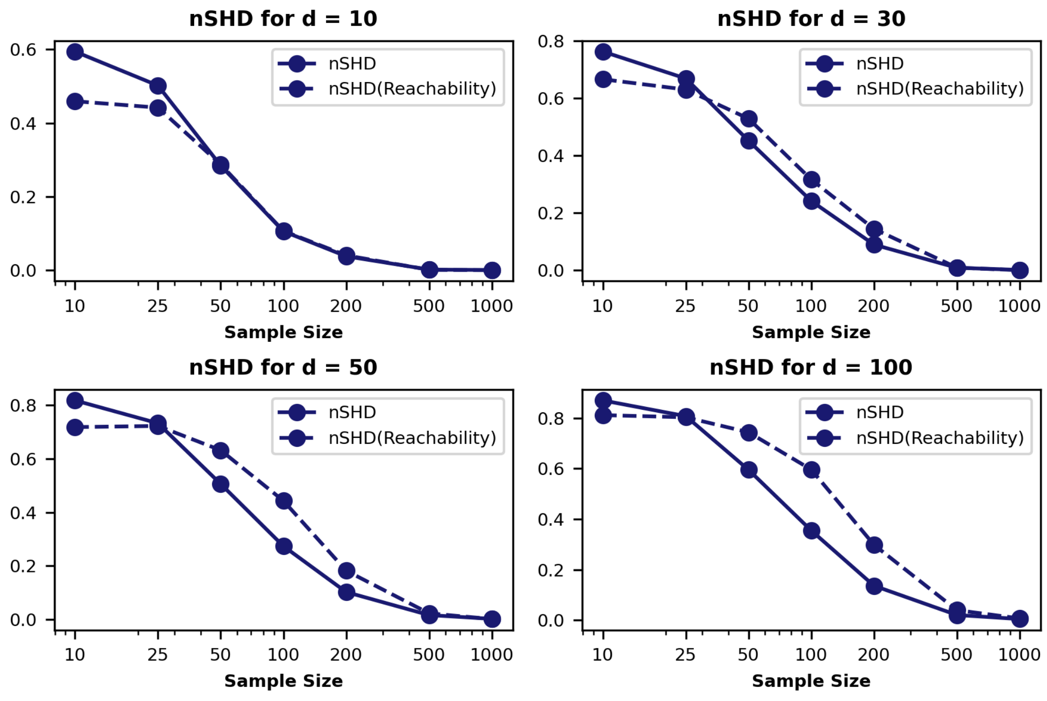


Figure 3.22: Mean nSHD for the weak dependence setting (2) and different graph sizes. For detailed explanations, see Figure 3.12.

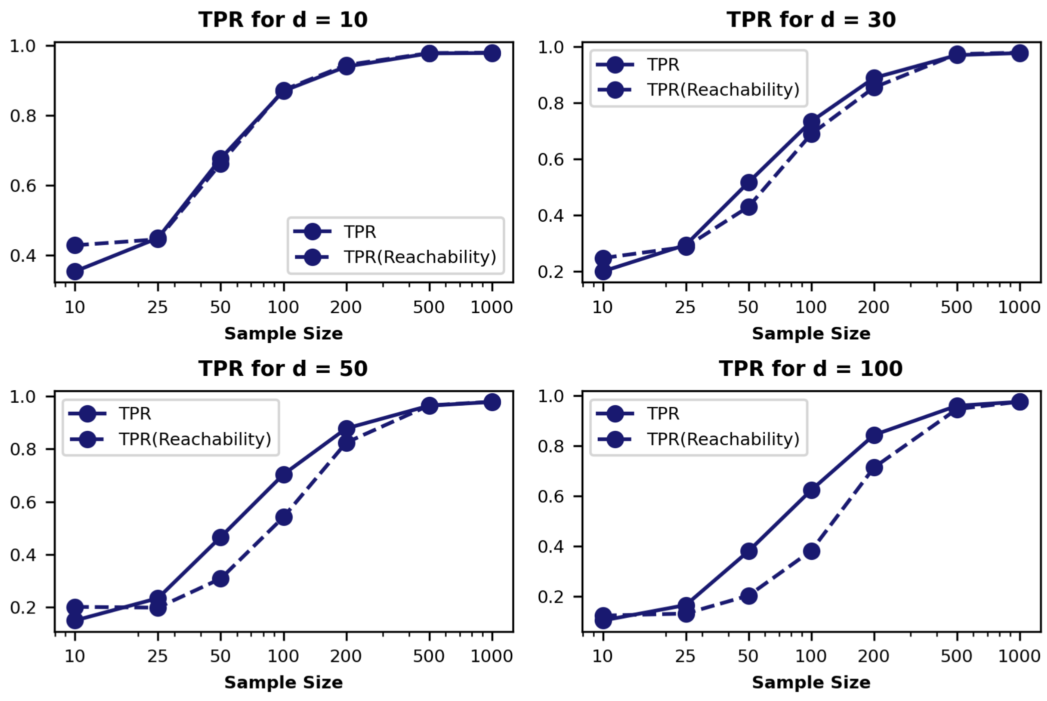


Figure 3.23: Mean TPR for the weak dependence (2) and different graph sizes. For detailed explanations, see Figure 3.13.

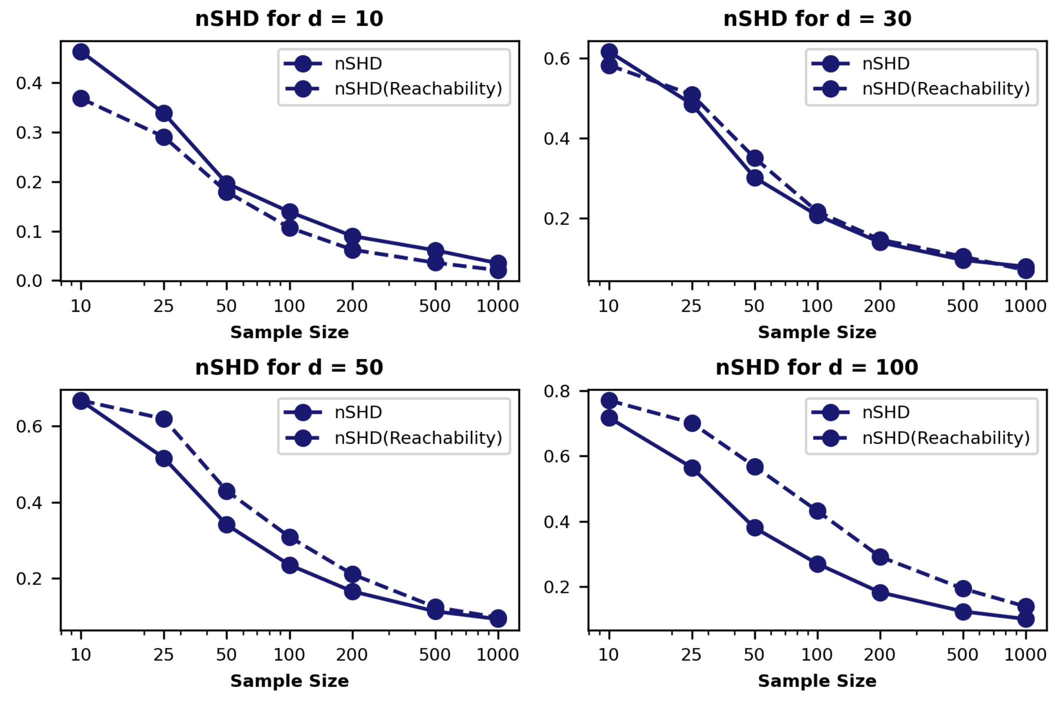


Figure 3.24: Mean nSHD for the mixed distribution setting (3) and different graph sizes. For detailed explanations, see Figure 3.12.

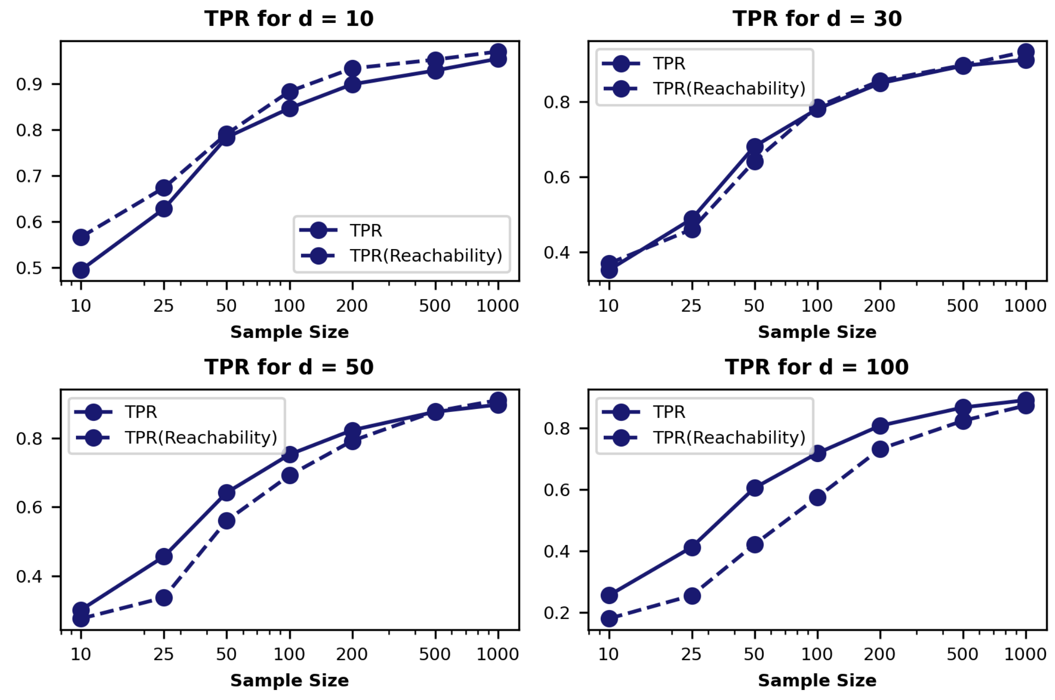


Figure 3.25: Mean TPR for the mixed distribution setting (3) and different graph sizes. For detailed explanations, see Figure 3.13.

Chapter 4

Learning Bayesian Networks from Extreme Data

4.1 Introduction

Graphical models have gained a wide interest in modeling causal dependence in a multivariate random vector. Bayesian networks, also known as directed acyclic graphs (DAGs) have been of particular interest in research as it allows for the factorization of the joint probability of all random variables into a product of conditional distributions, see e.g. Pearl [2009].

However, most models have been limited to discrete or Gaussian distributions(see e.g. Koller and Friedman [2009] and Lauritzen [1996]). Such models are unsuitable for extreme value analysis and often lead to a severe underestimation of risk. To overcome this problem, extreme dependence measures have been proposed that focus on maxima rather than the center of the distribution (Beirlant et al. [2004], De Haan and Ferreira [2007], Resnick [2007]). Still, the combinatorial explosion in the joint likelihood makes them often intractable to fit on larger data sets. In recent years, max-linear models have gained a wider interest in the literature for modeling extremal dependence (Buck and Klüppelberg [2021], Gissibl and Klüppelberg [2018], Gissibl et al. [2018, 2021], Klüppelberg and Krali [2021], Tran et al. [2021]). However, methods for learning them remain scarce. They often assume noise-free observations (Gissibl et al. [2021], Klüppelberg and Krali [2021]) or one-sided noise (Buck and Klüppelberg [2021]). This makes them highly sensitive to model misspecifications and, thus, often cannot be applied to real-world data sets. Recently, Tran et al. [2021] have proposed a new estimator for learning max-linear Bayesian networks. The method relies on a score between ordered pairs of vertices and uses a measure of concentration to estimate Bayesian networks. The authors apply the score to the *Extremal River Problem*, which seeks to recover the underlying river network based on extreme discharges along a set of stations only (Gnecco et al. [2021], Mhalla et al. [2020], Tran et al. [2021]). The algorithm in Tran et al. [2021] strongly outperforms all state-of-the-art methods. However, the method has serious drawbacks: it is limited to recovering root-directed spanning trees, a special class of Bayesian networks, and cannot be applied to general Bayesian networks. Moreover, the scores are only a measure of concentration and lack interpretation.

This work improves upon this by introducing a novel estimator based on the optimization framework of Zheng et al. [2018]. Although learning arbitrary Bayesian networks is NP-complete as it scales super exponentially with the number of nodes (cf. Chickering [1996]), the authors achieved a novel characterization of acyclicity in terms of the smooth function $\operatorname{tr}(\exp(C \circ C))$, where $\circ$ is the Hadamard product, $\operatorname{tr}(A)$ denotes the trace and $\exp(A)$ is the matrix exponential of A.

A matrix $C \in \mathbb{R}^{d\times d}$ is supported on a Bayesian network if and only if the upper expression equals d. In this chapter, we are going to adapt max-linear models to the upper optimization framework. We are going to show that max-linear models are a special case of max-out networks (Goodfellow et al. [2013]) that can be learned in terms of a neural network. For a noisy sample Y, we seek to optimize

$$\hat{C}^* := \underset{\substack{A\in\mathbb{R}^{d\times d}\\ A \text{ Kleene star matrix}}}{\arg\min} \quad \|(A \oplus \tilde{Y}) \vee Y - Y\|_1 - \lambda f(A), \tag{4.1}$$

where a *Kleene star matrix* is the transitive closure of the underlying edge weight matrix C, $\tilde{Y}$ is a preprocessed version of Y, $\oplus$ denotes the tropical matrix multiplication, $\lambda > 0$ is a tuning parameter and f is a regularization term. All the terms will be defined later. Throughout the chapter, for a matrix $A := (a_{ij})_{d\times d}$, we use the regularization term

$$f(A) = \log\left(\sum_{i,j=1,\dots,d} \exp(a_{ji})\right)$$

even though other regularization terms work equally.

Neural networks are a standard tool for learning graphical models (Johnson et al. [2016], Jordan [1999], Yoon et al. [2019]). However, with respect to causal inference in extremes, literature is more scarce. The algorithm learns the model based only on the upper tail of the distribution in order to recover the extremal dependence structure. It is very flexible and has only two tuning parameters. We implemented it as a plug-and-play package in Python at `https://github.com/johanneseebuck/mldag`, which includes all data and codes to produce the results and figures in this chapter. Moreover, we suggest an automated parameter-tuning procedure based on resampling. We get good results for simulated, noisy data (cf. Table 4.1 and 4.2) as well as two real-world data sets.

The novelty of this chapter lies in several directions.

1. We suggest a new algorithm to recover causality in extremes. The algorithm is the first to estimate noisy max-linear Bayesian networks without very specific assumptions on the Bayesian network [Tran et al., 2021] or the noise distribution [Buck and Klüppelberg, 2021]. The algorithm implements the max-linear network as a max-out network and uses gradient descent and backpropagation to iteratively determine the optimal solution.

2. We prove the consistency of the estimated DAG in an extreme value setting with possible noise as the sample size tends to infinity.

3. We show by a simulation study that the algorithm is robust with respect to different dependence structures and model misspecifications.

The chapter is organized as follows. In Section 4.2, we recall max-linear models and introduce Algorithm 6, respectively Algorithm 7 with automated parameter selection.

In Section 4.3, we give estimation results for the algorithm for noisy simulations and two real-world data sets. Finally, Section 4.4 concludes the chapter with a proof of consistency (cf. Theorem 4.4.1).

Throughout we use the following notation. For a directed, acyclic graph $\mathcal{D} = (V, E)$, also known as a *Bayesian network*, the node sets $\mathrm{pa}(i)$, $\mathrm{an}(i)$ and $\mathrm{de}(i)$ denote the parents, ancestors, and the descendants of node i, respectively, and we abbreviate $\mathrm{An}(i) := \mathrm{an}(i) \cup \{i\}$.

Unless stated otherwise, d stands for the number of nodes in the graph while n stands for the number of observations in a sample. A matrix $C \in \bar{\mathbb{R}}^{d\times d}$ defines a *weighted directed graph*, where $j \to i \in \mathcal{D}$ if and only if its *edge weight* c_{ij} is positive (respectively greater than negative infinity when taking the logarithm).

For random variables, we use uppercase variables and for observations we use lowercase variables. We use $\mathcal{X}$ to refer to a sample $\mathcal{X} := \{x^1, \ldots, x^n\} \in \mathbb{R}^d$. If necessary for matrix multiplication, we also collect $\mathcal{X}$ in a matrix $X \in \mathbb{R}^{d\times n}$. For a matrix $A = (a_{ij})_{d\times n}$ we denote

$$\|A\|_1 = \sum_{\substack{i=1,\ldots,d\\ j=1,\ldots,n}} |a_{ij}|/n. \tag{4.2}$$

Finally, we introduce the following standard metrics for causal inference in graphs: *normalized Structural Hamming Distance(nSHD)* and *True Positive Rate(TPR)* as follows. Let $\mathcal{D}$ be the true graph and $\hat{\mathcal{D}}$ be the estimated graph. The *structural Hamming distance* $\mathrm{SHD}(\mathcal{D}, \hat{\mathcal{D}})$ between $\mathcal{D}$ and $\hat{\mathcal{D}}$ is the minimum number of edge additions, deletions, and reversals to obtain $\mathcal{D}$ from $\hat{\mathcal{D}}$. Denote $E(\mathcal{D})$ and $E(\hat{\mathcal{D}})$ the set of edges in $\mathcal{D}$ and $\hat{\mathcal{D}}$, respectively. We then have

$$\mathrm{nSHD}(\hat{\mathcal{D}}, \mathcal{D}) := \frac{\mathrm{SHD}(\hat{\mathcal{D}}, \mathcal{D})}{|E(\hat{\mathcal{D}})| + |E(\mathcal{D})|}, \qquad \mathrm{TPR}(\hat{\mathcal{D}}, \mathcal{D}) := \frac{|E(\hat{\mathcal{D}}) \cap E(\mathcal{D})|}{|E(\mathcal{D})|}. \tag{4.3}$$

4.2 Recursive Max-Linear Models

The recursive max-linear model was first defined by Gissibl and Klüppelberg [2018] and studied in Améndola et al. [2021, 2022], Asenova and Segers [2022], Buck and Klüppelberg [2021], Gissibl and Klüppelberg [2018], Gissibl et al. [2018, 2021], Tran et al. [2021] as a model for extreme value analysis.

For numerical stability, in practice, one often works with the logarithm of the extremes. We, therefore, work with the equivalent log-version of this model given as

$$X_i = \bigvee_{j\in\mathrm{pa}(i)} (c_{ij} + X_j) \vee Z_i, \quad i \in V, \tag{4.4}$$

with edge weights $c_{ik} \in \mathbb{R}$ for $i \in V$ and $k \in \mathrm{pa}(i)$, and i.i.d. random variables $Z_1, \ldots, Z_d$ with support $\mathbb{R}$ and atom-free distributions. For this chapter, we assume that C is supported on a Bayesian network, i.e. the graph $\mathcal{D} = (V, E)$ that results from drawing an edge $j \to i$ whenever $c_{ij} > -\infty$ contains no cycles. For a path $p = [j = k_0 \to k_1 \to \ldots \to k_n = i]$ from j to i we define the *path weight*

$$d_{ij}(p) := \sum_{l=0}^{n-1} c_{k_{l+1}k_l}. \tag{4.5}$$

Denoting the set of all paths from j to i by P_{ij}, we define the Kleene star matrix $C^* = (c^*_{ij})_{d\times d}$ of X with entries

$$c^*_{ij} := \bigvee_{p\in P_{ij}} d_{ij}(p) \quad \text{for } j \in \operatorname{an}(i), \quad c^*_{ii} = 0, \quad \text{and} \quad c^*_{ij} = -\infty \quad \text{for } j \in V \setminus \operatorname{An}(i). \tag{4.6}$$

It is possible to write the model in terms of the Kleene star matrix $C^* := (c^*_{ij})_{d\times d}$ as

$$X_i = \bigvee_{j\in\operatorname{An}(i)} (c^*_{ij} + Z_j), \quad i = 1, \ldots, d. \tag{4.7}$$

Observe that an alternative representation is given by

$$X_i = \bigvee_{j\in\operatorname{An}(i)} (c^*_{ij} + X_j), \quad i = 1, \ldots, d, \tag{4.8}$$

see Buck and Klüppelberg [2021], Corollary 3.4. From (4.8), it follows immediately that the distribution of $X_i - X_j$ is lower bounded by $c^*_{ij} > -\infty$ if and only if there is a path from j to i. For a sample $\mathcal{X} := \{x^1, \ldots, x^n\} \in \mathbb{R}^d$, this motivates the *minimum-difference estimator* given by

$$\hat{c}^*_{ij} := \bigwedge_{t=1}^{n} (x^t_i - x^t_j), \quad (i, j) \in d \times d. \tag{4.9}$$

If data is generated by a max-linear model without noise, a version of this estimator converges almost surely and is the generalized maximum likelihood estimator in the sense of Kiefer and Wolfowitz [1956], see Gissibl et al. [2021].

However, literature on estimating max-linear models under observational noise remains scarce. For the equivalent non-logarithmic model, Buck and Klüppelberg [2021] showed that for one-sided noise and some regular variation condition, the minimum-difference estimator $\hat{C}^* = (\hat{c}^*_{ij})_{d\times d}$ converges to a matrix of independent Weibull entries. Nevertheless, if X_i is polluted by some (two-sided) additive noise $\varepsilon_i \in \mathbb{R}$ for $i = 1, \ldots, d$, then the minimum-difference estimator might diverge even if $j \rightsquigarrow i$, i.e. there is a path from j to i. Another problem is that despite asymptotic consistency, for a finite sample, it might not be directly obvious which edges to include when using a minimum-difference estimator as in (4.9). To overcome this problem, Tran et al. [2021] recently suggested an extension of the minimum-difference estimator. This approach relies, similarly to the minimum-difference estimator, on pairwise observations. In the first step, for a pair (j, i) only these observations, where x_j exceeds a certain quantile threshold α are used. Therefore, Tran et al. [2021] define the pairwise sample $\mathcal{X}_{ij}$ as

$$\mathcal{X}_{ij}(\alpha) := \{x_i - x_j : x \in \mathcal{X}, x_j > Q_{\mathcal{X}_j}(\alpha)\}, \tag{4.10}$$

where $Q_{\mathcal{X}_j}(\alpha)$ is the α-th quantile of the empirical distribution of $\mathcal{X}$ in the j-th coordinate. The aim is to filter the impact of noise and get more meaningful observations by only taking samples where X_j is large. For such observations, the signal is strong and, therefore, the noise is small relative to the signal.

Given pairwise samples $\mathcal{X}_{ij}(\alpha)$, the authors propose the following score

$$w_{ij}(\underline{r}) := \Big(\mu(\mathcal{X}_{ij}(\alpha)) - Q_{\mathcal{X}_{ij}(\alpha)}(\underline{r})\Big)^2, \tag{4.11}$$

where $\underline{r} \in (0,1)$ is a fixed, small empirical quantile and $\mu(\mathcal{X}_{ij}(\alpha))$ the empirical mean. Now observe that for $\underline{r} = 0$, this is equal to $(\mu(\mathcal{X}_{ij}(\alpha)) - \min(\mathcal{X}_{ij}(\alpha)))^2 = (-\min(\bar{\mathcal{X}}_{ij}(\alpha)))^2$, where $\bar{\mathcal{X}}_{ij}$ is $\mathcal{X}_{ij}$ normalized by its mean. For this reason, the scores can be seen as a generalization of the minimum-difference estimator.

Applied to the *Extremal River Problem*, i.e. the task to recover a river network from only extreme flow measured at a set V of stations, the authors apply Chu–Liu/Edmonds' algorithm to find the minimum root-directed spanning tree. This yields highly significant results for various river networks with True Positive Rate at around 80% on average, outperforming existing methods, see Tran et al. [2021]. However, if the underlying network is not a root-directed spanning tree, Chu–Liu/Edmonds' algorithm cannot be applied and, therefore, the decision on what edges to include is far less obvious. Moreover, only looking at bivariate data does not give the possibility to learn the model as a whole but only information about the pairwise connection of variables. This bears the risk that wrong conclusions are drawn as strong causality from j to i might be caused only by a confounder k. Finally, the scores lack an interpretation and cannot be interpreted as estimates of the Kleene star matrix C^*.

4.2.1 Learning general max-linear Bayesian Networks

We introduce a novel way to learn max-linear Bayesian networks under observational noise. The idea is based on multivariate linear regression [Montgomery et al., 2021]. However, tropical matrix multiplication replaces the usual matrix multiplication. Furthermore, we use a machine-learning approach and Zheng et al. [2018], a technique to turn the DAG constraint into a continuous optimization problem.

When estimating the underlying Bayesian network, we have the possibility to either estimate the edge weight matrix C as in (4.4) or the Kleene star matrix C^* as in equation (4.7), which acts as a weighted reachability matrix. However, max-linear models have the property that the edge weight matrix C is generally not identifiable, see Example 3.2. in Gissibl et al. [2021]. Though, the max-linear model is uniquely determined by C^*, see equation (4.7). For this reason, in the subsequent chapter, we focus on recovering the Kleene star matrix C^*.

Let $X := (X_1, \dots, X_d)$ follow a recursive max-linear model as in (4.7). We define the *noisy max-linear model* as

$$Y_i := \bigvee_{j \in \mathrm{An}(i)} (c^*_{ij} + Z_j) + \varepsilon_i = X_i + \varepsilon_i, \quad i = 1, \dots, d, \tag{4.12}$$

with $\varepsilon_i \in \mathbb{R}$ and i.i.d. for $i = 1, \dots, d$. Then, by (4.8), since the noise is *not* propagating and $\varepsilon_1, \dots, \varepsilon_d$ are i.i.d., $Y_i - \bigvee_{j \in \mathrm{An}(i)}(c^*_{ij} + Y_j) \overset{d}{\approx} (\varepsilon_i - \varepsilon_j)$. Therefore, the noise becomes small relative to the signal, whenever we filter for large observations. Thus, for a given sample $\mathcal{Y} := \{y^1, \dots, y^n\} \in \mathbb{R}^d$ from a noisy max-linear model and any node $i \in \{1, \dots, d\}$, we want to keep those observations in $\{y_i^1, \dots, y_i^n\}$ as explanatory variables that exceed a certain α-quantile. If y_i^t is below the threshold, the observation is dominated by noise and, hence, unsuitable to serve as an explanatory variable. However, it might still be useful as a response variable. More precisely, if y_j^t is large but y_i^t is small, it is less likely that j is an ancestor of i.

For this reason, the need arises for generating two different data sets, one as explanatory and one as response variables. For the explanatory variables, we look at the univariate quantiles and set those below the quantile equal to negative infinity. This will ensure that these observations will not be used in the estimation later. More precisely, we define $\tilde{\mathcal{Y}} := \{\tilde{y}^1, \dots, \tilde{y}^n\} \in \mathbb{R}^d$ with

$$\tilde{y}_j^t := y_j^t \quad \text{if } y_j^t > Q_{\mathcal{Y}_j}(\alpha), \quad \text{and} \quad \tilde{y}_j^t = -\infty \quad \text{otherwise.} \tag{4.13}$$

as the data set of *explanatory variables*. The original data set $\mathcal{Y}$ then serves as the data set of *response variables*. Observe that this can result in observations $\tilde{y}^t = (-\infty, \dots, -\infty)$. This means that at time t, no suitable, extreme measurement that exceeded the chosen quantile was recorded for any given node. Therefore, we lack a suitable explanatory variable and we delete this observation in $\tilde{\mathcal{Y}}$ and $\mathcal{Y}$.

4.2.2 Setting up the optimization problem

We first introduce tropical matrix multiplication. To do so, recall the noise-free max-linear model from equation (4.8). We can write it in terms of a matrix product $\oplus$. For two non-negative matrices F and G, where the number of columns in F is equal to the number of rows in G, we define the matrix product $\oplus : \overline{\mathbb{R}}^{m\times n} \times \overline{\mathbb{R}}^{n\times p} \to \overline{\mathbb{R}}^{m\times p}$ by

$$(F = (f_{ij})_{m\times n}, G = (g_{ij})_{n\times p}) \mapsto F \oplus G := \Big(\bigvee_{k=1}^{n} f_{kj} + g_{ik}\Big)_{m\times p}. \tag{4.14}$$

The triple $(\overline{\mathbb{R}}, \vee, +)$, is an idempotent semiring with $-\infty$ as 0-element and 0 as 1-element and the operation $\oplus$ is therefore a matrix product over this semiring; see for example Butkovič [2010]. We can rewrite (4.8) to obtain

$$X = C^* \oplus X.$$

For this reason, for a given sample $\mathcal{Y} := \{y^1, \dots, y^n\} \in \mathbb{R}^d$ from a noisy max-linear model, denoting by $Y \in \mathbb{R}^{d\times n}$ the matrix of observations, we use

$$\hat{C}^* := \underset{\substack{A \in \mathbb{R}^{d\times d} \\ A \text{ Kleene star matrix}}}{\arg\min} \|(A \oplus \tilde{Y}) \vee Y - Y\|_p. \tag{4.15}$$

We want to quickly discuss the upper expression. As explained in (4.13), we separate Y into a matrix of explanatory variables $\tilde{Y}$ and a set of response variables Y. Moreover, for the noise-free case, it holds that $X = C^* \oplus X$, so it makes sense to minimize $\|\hat{C}^* \oplus \tilde{Y} - Y\|$. This, however, penalizes also the upper tail. Considering equation

(4.4), a large value of X_i can be driven by a parent of i but also by an exogenous innovation Z_i. For this reason, we only penalize the left tail which results in (4.15).

For simplicity, we take the norm defined in (4.2) with $p = 1$. Other matrix norms are possible and lead to similar results. Observe that max-linear models as defined in (4.8) are a special case of max-out networks, see Goodfellow et al. [2013]. Given an input $X \in \mathbb{R}^d$ a max-out layer implements the function

$$h_i(x) = \bigvee_{j=1}^{d} X^T \cdot W_{ij} + b_{ij},$$

where $W_{i,j} \in \mathbb{R}^d$ and $b_{ij} \in \mathbb{R}$. Setting $W_{ij} = (0, \dots, 0, \underset{\text{jth entry}}{1}, 0, \dots, 0)$, renaming $h_i(x)$ to X_i and $b_{ij} = c^*_{ij}$, this leads to

$$X_i = \bigvee_{j=1}^{d} (X_j + c^*_{ij}),$$

which matches with (4.8) since $c^*_{ij} = -\infty$ if $j \notin \mathrm{An}(i)$. However, one needs to note that, unlike max-linear models, max-out networks are used as an activation function for a hidden layer in a neural network. Therefore, max-out networks are used to learn some output Y from input X. The exact weights, however, lack interpretation. In our case, however, the estimate $\hat{C}^*$ is the estimated Kleene star matrix and gives an interpretation of how risk propagates through a Bayesian network. Also the optimization term is different. Still, the connection helps us to solve the problem efficiently by constructing a neural network and using gradient descent. However, observe that $\hat{c}^*_{ii} = 0$ for $i = 1, \dots, d$ and $\hat{c}^*_{ij} = -\infty$ for $i \neq j$ satisfies the definition of a Kleene star matrix by (4.6). Thus, it leads to the optimal score of zero.

For this reason, we need to add a regularization term. While most models rather suffer from overfitting than underfitting, the inherent problem is the same. However, instead of adding the regularization term, we simply subtract the regularization term. This encourages more complex solutions and prevents us from choosing the trivial solution of a graph with only isolated nodes.

For the regularization f, we would like to use a simple ℓ_1 regularization. However, observe that $\sum_{i,j=1,\dots,d} a_{ij} = -\infty$, whenever there exists at least one $a_{ij} = -\infty$. This, however, is true whenever A is supported on a DAG. Therefore, we need to slightly modify the penalty such that we take the following regularization function

$$f(A) = \log \left(\sum_{i,j=1,\dots,d} \exp(a_{ij}) \right). \tag{4.16}$$

While other regularization terms are generally possible, the chosen f is easy to use and has the following desirable features.

(a) $f : A \to \mathbb{R}$ is a function that assigns a real value to any $d \times d$ matrix

(b) f is increasing in every argument, i.e. if $a_{kl} = \tilde{a}_{kl}$ for any ordered pair $(k, l) \neq (j, i)$ and $a_{ji} < \tilde{a}_{ji}$, then $f(A) < f(\tilde{A})$.

(c) Any permutation of values gives the same value, i.e. if $\tilde{A}$ is a permutation of A, then $f(A) = f(\tilde{A})$.

(d) $f(A) = -\infty$ if and only if all values in A equal $-\infty$

(e) The derivative of f exists and its marginals $\frac{\partial}{\partial a_{ij}} f(A)$ are upper-bounded.

For the noise variables $\varepsilon_1, \ldots, \varepsilon_d$, it is common to assume continuous, symmetric, light-tailed density f_ε given by

$$f_\varepsilon(x) \sim e^{-Kx^p} \text{ as } x \to \infty, \tag{4.17}$$

for some $p > 1$ and $\gamma, K > 0$. Then we have additionally

(f)

$$\lim_{a_{ij} \to -\infty} \mathbb{P}(\varepsilon_j - \varepsilon_i \geq -a_{ij} + c) / \frac{\partial}{\partial a_{ij}} f(A) \to 0,$$

for any fixed constant c.

We can see that (f) holds by applying L'Hôpital's rule and

$$\lim_{a_{ij} \to -\infty} \frac{\partial}{\partial a_{ij}} f(A) = \begin{cases} \frac{1}{b} \exp(x), & \text{if } a_{kl} > -\infty \text{ for some ordered pair } (k,l) \neq (j,i) \\ 1, & \text{else.} \end{cases}$$

for $b := \sum_{\substack{(k,l) \in (d \times d) \\ (k,l) \neq (j,i)}} \exp(a_{lk})$ and

$$f_{\varepsilon_i - \varepsilon_j}(x) \sim x^{1-p/2} \, e^{-Kx^p} \text{ as } x \to \infty. \tag{4.18}$$

Therefore, adding the regularization term to (4.15), we optimization problem becomes:

$$\hat{C}^* := \underset{\substack{A \in \mathbb{R}^{d \times d} \\ A \text{ Kleene star matrix}}}{\arg\min} \quad \|(A \oplus \tilde{Y}) \vee Y - Y\|_1 - \lambda f(A), \tag{4.19}$$

where the parameter $\lambda > 0$ acts as a tuning parameter. However, it seems not obvious how to minimize the upper expression over the class of Kleene star matrices. In order for A to be a Kleene star matrix, it needs to be i) supported on a DAG, ii) $a_{ii} = 0$ for $i = 1, \ldots, d$, and iii) A is idempotent, i.e. $a_{ij} \leq a_{ik} + a_{kj}$ for any triple of nodes (j, k, i), where properties ii) and iii) follows by definition in (4.6).

While learning general Bayesian networks is NP-complete [Chickering, 1996] as the space of DAGs is exponential in the number of nodes, we can write the constraint in terms of a continuous optimization problem, as given in the subsequent lemma.

Lemma 4.2.1 (Theorem 1, Zheng et al. [2018]). *A matrix $A \in \mathbb{R}^{d \times d}$ is supported on a Bayesian network if and only if*

$$h(A) := tr(\exp(A \circ A)) - d = 0$$

where $\circ$ is the Hadamard product, $tr(A)$ denotes the trace and $\exp(A)$ is the matrix exponential of A.

Regarding the idempotency as given in iii), observe that by Theorem 2.4. respectively Corollary 2.5. in Gissibl and Klüppelberg [2018] it holds that

$$C^* := \log(I) \vee C^{\oplus(d-1)},$$

where I is the $d \times d$ identity matrix, $A^{\oplus i} := \underbrace{A \oplus \ldots \oplus A}_{i \text{ times}}$ and C is defined as in (4.4). For this reason, defining $\tilde{C} := (\tilde{c}_{ij})_{d\times d}$ with

$$\tilde{c}_{ij} := c_{ij} \quad \text{for } j \neq i, \quad \tilde{c}_{ii} = 0,$$

we have

$$C^* := \tilde{C}^{\oplus(d-1)},$$

We can use this equation in our forward function to ensure the idempotency of C^*. More precisely, we fix $a_{ii} = 0$ for $i = 1, \ldots, d$ such that our estimation problem (4.19) becomes

$$\hat{C}^* := \underset{\substack{A \in \mathbb{R}^{d\times d} \\ A \text{ supported on DAG}}}{\arg\min} \|(A^{\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1 - \lambda f(A), \tag{4.20}$$

Algorithm 6 MLDAG , fixed parameters

Parameters: Intensity $\lambda > 0$, quantile level $\alpha \in [0, 1)$, and penalty function f.
Input: Data $\mathcal{Y} = \{y_1, \ldots, y_n\} \in \mathbb{R}^d$ stored as a matrix $Y \in \mathbb{R}^{d\times n}$.
Output: A Kleene star matrix $\hat{C}^*$.

```
1: Calculate $\tilde{Y}$ as in (4.13).
2: Assign $\hat{c}^*_{ii} := 0$ for $i = 1, \ldots, d$ and assign equal starting values $\hat{c}^*_{ij}$ for $i \neq j$.
3: Solve $\hat{C}^* := \underset{\substack{A \in \mathbb{R}^{d\times d} \\ A \text{ supported on DAG}}}{\arg\min} \|(A^{\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1 - \lambda f(A)$.
4: Use 2-means clustering to divide all values $\hat{c}^*_{ij} > -\infty$ into two groups
5: Set $\hat{c}^*_{ij} = -\infty$ whenever $\hat{c}_{ij}$ belongs to the group of smaller means.
6: **Return** the Kleene star matrix of $\hat{C}^*$.
```

These ideas are summarized in Algorithm 6. We address the algorithm:

1. In order to solve the optimization problem in line 4, we use the *torch* package in *python*. We define a max-linear model as a torch neural network, where for inputs $\tilde{Y}$ and Y, the forward function calculates $(A^{\oplus(d-1)} \oplus \tilde{Y}) \vee Y$. Then, the loss function implements $\|(A^{\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1$ and we add the regularization term. To ensure that A is supported on a DAG, we use Lemma 4.2.1. This is done by augmented Lagrangian, see e.g. Chapter 2.1. in Bertsekas [2014]. We then use backpropagation [Goodfellow et al., 2016, Chapter 2, Section 6.5] to automatically calculate the gradient, where we use the torch optimizer *SGD*.

2. Line 5 and 6 are optional. However, if we do not do this, we will obtain a complete DAG, i.e. a DAG with $d(d-1)/2$ edges. In general, we expect the values of $\hat{C}^*$ to separate into a group of strong signals which are characterized by large values $\hat{c}^*_{ij}$ and a group of spurious edges that are very small. To successfully

identify and separate these two groups, we apply 2-means clustering to these values and set all values equal to zero that belong to the second group. k-means clustering is a classification algorithm that is standard in machine learning. It aims at creating a partition into k sets such that the mean squared deviation of each value to its mean is minimized. For more information, see MacQueen [1967]. Since this might violate the idempotency, we return the Kleene star matrix of $\hat{C}^*$ in line 7.

Before discussing the choice of parameters, we want to illustrate the optimization problem with an example.

Example 4.2.2. *In this example we want to illustrate three things:*

1. $A^{\oplus(d-1)}$ *as in Algorithm 6 is a Kleene star matrix as defined in* (4.6).
2. *If* $\hat{C}^*$ *is the solution of line 3 in Algorithm 6, then* $\hat{C}^*$ *is already a Kleene star matrix.*
3. $\hat{C}^*$ *gives an estimate about the weights* C^**. Particularly, if* c^*_{ij} *increases, then* $\hat{c}^*_{ij}$ *increases as well.*

To do so, consider the DAG $\mathcal{D}$*:*

$$\begin{tikzcd} 1 \arrow[r] \arrow[rr, bend left] & 2 \arrow[r] & 3 \end{tikzcd}$$

We assume that $c_{21} = c_{32} = \log(0.5) \approx -0.69$*. For the edge* c_{31} *we consider two scenarios: i)* $c_{31} = log(0.2) \approx -1.61$ *and ii)* $c_{31} = \log(0.5) \approx -0.69$*. Then by* (4.6)*, we have*

$$C_1^* = \begin{pmatrix} 0 & -\infty & -\infty \\ \log(0.5) & 0 & -\infty \\ \log(0.5)+\log(0.5) & \log(0.5) & 0 \end{pmatrix} \qquad C_2^* = \begin{pmatrix} 0 & -\infty & -\infty \\ \log(0.5) & 0 & -\infty \\ \log(0.5) & \log(0.5) & 0 \end{pmatrix}$$

Now let us assume that $Y \in \mathbb{R}^d$ *is an observation from a noisy recursive max-linear model as in* (4.12)*, i.e.* $Y = X + \varepsilon = (X_1 + \varepsilon_1, X_2 + \varepsilon_2, X_3 + \varepsilon_3)$ *with the underlying DAG* $\mathcal{D}$ *as described above. We then have*

$$A^{\oplus(2)} = \begin{bmatrix} 0 & a_{12} \vee (a_{32} + a_{13}) & a_{13} \vee (a_{23} + a_{12}) \\ a_{21} \vee (a_{31} + a_{23}) & 0 & a_{23} \vee (a_{13} + a_{21}) \\ a_{31} \vee (a_{21} + a_{32}) & a_{32} \vee (a_{12} + a_{31}) & 0 \end{bmatrix}$$

Now looking at the entries of $A^{\oplus(2)}$*, we can see that each entry is the maximum path weight of the Kleene star matrix, see definition* (4.6)*. For example, a path from* 1 *to* 2 *can be either the direct edge* a_{12} *or the directed path* $a_{31} + a_{23}$*. Therefore, we have* $A^{\oplus(2)}_{21} = a_{21} \vee (a_{31} + a_{23})$*. Hence, we can see that* $A^{\oplus(2)}$ *is indeed idempotent. Now let* $\hat{C}^*$ *solve the problem as in line 4 of Algorithm 6. Then it also holds that* $\hat{c}^*_{21} \geq \hat{c}^*_{31} + \hat{c}^*_{23}$*. This is because if* $\hat{c}^*_{21} < \hat{c}^*_{31} + \hat{c}^*_{23}$*, then setting* $\hat{c}^*_{21} = \hat{c}^*_{31} + \hat{c}^*_{23}$ *gives the same result for* $\hat{C}^{*\oplus(2)}$*. However, by feature (b) of the regularization term, this increases the regularization and, since we take the negative, would lead to a smaller score. Note that* $\hat{c}^*_{21}$ *is also present in other entries but since* $\hat{C}^*$ *is supported on a DAG, we can recursively use this argument to validate that* $\hat{C}^*$ *is indeed idempotent.*

Finally, we want to return to the two scenarios introduced before. We wish to give an intuition that for scenario ii), the estimated value $\hat{c}^*_{31}$ is larger compared to scenario i) since c^*_{31} is larger as well. For illustration, we need to make a few simplifying assumptions. Let us first assume that $\alpha = 0$. Then, $(\hat{C}^{*\oplus(d-1)} \oplus \tilde{Y}) \vee Y = \hat{C}^{*\oplus(d-1)} \oplus Y$ and we have

$$\begin{aligned}
&\|(\hat{C}^{*\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1 \\
=&Y_1 \vee (\hat{c}^*_{12} \vee (\hat{c}^*_{32} + \hat{c}^*_{13}) + Y_2) \vee (\hat{c}^*_{13} \vee (\hat{c}^*_{23} + \hat{c}^*_{12}) + Y_3 - Y_1 \\
+&(\hat{c}^*_{21} \vee (\hat{c}^*_{31} + \hat{c}^*_{23}) + Y_1) \vee Y_2 \vee (\hat{c}^*_{23} \vee (\hat{c}^*_{13} + \hat{c}^*_{21}) + Y_3) - Y_2 \\
+&(\hat{c}^*_{31} \vee (\hat{c}^*_{21} + \hat{c}^*_{32}) + Y_1) \vee (\hat{c}^*_{32} \vee (\hat{c}^*_{12} + \hat{c}^*_{31}) + Y_2) \vee Y_3 - Y_3
\end{aligned}$$

Now since $c^*_{23} = -\infty$, let us assume that $\hat{c}^*_{23} = -\infty$. Then, the upper sum differs between the two scenarios only in the last term

$$(\hat{c}^*_{31} \vee (\hat{c}^*_{21} + \hat{c}^*_{32}) + Y_1) \vee (\hat{c}^*_{32} \vee (\hat{c}^*_{12} + \hat{c}^*_{31}) + Y_2) \vee Y_3 - Y_3$$

Now if $X_3 = X_1 + c^*_{31}$, then this simplifies to

$$(\hat{c}^*_{31} \vee (\hat{c}^*_{21} + \hat{c}^*_{32}) + X_1 + \varepsilon_1) \vee (\hat{c}^*_{32} \vee (\hat{c}^*_{12} + \hat{c}^*_{31}) + Y_2) \vee (X_1 + c^*_{31} + \varepsilon_3) - (X_1 + c^*_{31} + \varepsilon_3)$$

Now depending on c^*_{31}, the probability

$$\mathbb{P}\left((\hat{c}^*_{31} \vee (\hat{c}^*_{21} + \hat{c}^*_{32}) + \varepsilon_1) \vee (\hat{c}^*_{32} \vee (\hat{c}^*_{12} + \hat{c}^*_{31}) + Y_2 - X_1) \geq (c^*_{31} + \varepsilon_3)\right)$$

gets smaller if c^*_{31} increases. Therefore, asymptotically, $\|(\hat{C}^{*\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1$ becomes smaller for scenario ii) over scenario i). Since we increase $\hat{c}^*_{31}$ as long as the regularization term decreases the score more than $\|(\hat{C}^{*\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1$ increases it, $\hat{c}^*_{31}$ will be larger in scenario ii). We want to quickly illustrate this with a simulation. From the DAG $\mathcal{D}$, we generate a sample from a noisy recursive max-linear model as in (4.12) where $Z_1, \dots, Z_d$ are i.i.d. Gumbel(1) and the noise is normally distributed with noise-to-signal ratio 30% (i.e. the noise has 30% of the standard deviation of the noise-free sample X). We generate $n = 100$ observations and run Algorithm 6 with standard parameters $\lambda = 0.3$ and $\alpha = 0.8$ to calculate the estimated Kleene star matrix $\hat{C}^*$. We repeat this 100 times and build the mean value in each entry across all 100 repetitions. The result is given as

$$\hat{C}^*_1 = \begin{pmatrix} 0 & -\infty & -\infty \\ -1.04 & 0 & -\infty \\ -1.48 & -1.08 & 0 \end{pmatrix} \qquad \hat{C}^*_2 = \begin{pmatrix} 0 & -\infty & -\infty \\ -1.15 & 0 & -\infty \\ -0.93 & -0.95 & 0 \end{pmatrix}$$

Observe that $-1.48 < -0.93$, so in scenario ii) we correctly estimate $\hat{c}^*_{31}$ significantly larger. Overall, the result resembles the underlying DAG $\mathcal{D}$ well, see C^*_1 and C^*_2.

4.2.2.1 On parameters and missing values

We now want to briefly discuss the choice of potential parameters. For the parameters α, only values in $[0, 1]$ are possible. If the data comes from a max-linear Bayesian network without noise, then we should choose $\alpha = 0$ as it will give us the maximum number of observations. On the other hand, the larger α is chosen, the smaller the influence of the light-tailed noise variables relative to the signal. We, therefore, have a trade-off between the number of observations and the significance of each observation. Typically, we want values for $\alpha \geq 0.75$.

For λ, we have

$$\begin{aligned}&\mathbb{E}\left(\frac{\partial}{\partial \hat{a}_{ij}} \|(A^{\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1\right)\\ &\geq \mathbb{P}\left(a_{ij} + \tilde{Y}_j > \bigvee_{k \in V \setminus \{j\}} a_{ik} + \tilde{Y}_k \vee Y_i\right) \in [0,1].\end{aligned}$$

Contrarily, $\frac{\partial}{\partial a_{ij}} f(A) \leq 1$ For this reason, only values $\lambda \in [0,1)$ are useful with typical values ranging at $\lambda \leq 0.5$.

Another challenge is missing values in the sample. It is possible that some stations do not have measurements for the entire period of the sample. This can be because sensors break or because not every node has the time data range of recordings. In case we do not observe a value for node i at point t, we can not use them either as an explanatory or as a response variable. For this reason, we first calculate $\tilde{Y}$ as defined in (4.13), ignoring all missing values. Next, we replace missing values in $\tilde{Y}$ by negative infinity and missing values in Y by a very large positive number. This way, missing values are not used and do not interfere with the estimation of parameters.

4.2.3 Tuning the parameters

To automatically tune the parameters, we use a subsampling procedure, which is standard in the literature, see e.g. James et al. [2013]. We search on a grid of possible values for α and λ. For each combination of parameters, estimate the DAG. Then resample the data n_s times with replacement. For each of the n_s resulting samples, estimate the resulting DAG for each combination of parameters and choose the combination with the lowest normalized structural Hamming distance(nSHD) to the DAG estimated with the original sample. We formally note down this idea in Algorithm 7.

Remark 4.2.3. *Utilizing the idempotency of C^* comes with a computational price. If we minimize*

$$\|(A^{\oplus(d-1)} \oplus \tilde{S}) \vee S - S\|_1 \tag{4.21}$$

as in line 8 of Algorithm 7, the backpropagation becomes computationally significantly more expensive compared to the much simpler task of minimizing

$$\|(A \oplus \tilde{S}) \vee S - S\|_1. \tag{4.22}$$

Therefore, it is often advisable in Algorithm 7 to use (4.21) *for the original data set Y, while for the resampled data $Y_1, \dots, Y_{n_s}$, we use* (4.22) *to save computation time. This will lead to very similar results and significantly reduces computation time.*

Algorithm 7 MLDAG with automated parameter selection

Parameters: Intensities $\{\lambda_1, \dots, \lambda_{n_\lambda}\} > 0$, quantile levels $\{\alpha_1, \dots, \alpha_{n_\alpha}\} \in [0, 1)$, penalty function f and resample rate n_s.
Input: Data $\mathcal{Y} = \{y_1, \dots, y_n\} \in \mathbb{R}^d$ stored as a matrix $Y \in \mathbb{R}^{d \times n}$.
Output: A Kleene star matrix $\hat{C}^*$.

```
1: Generate resampled data $Y_1, \dots, Y_{n_s} \in \mathbb{R}^{d\times n}$ with replacement.
2: for $S$ in $(Y, Y_1, \dots, Y_{n_s})$ do
3:     for $\alpha$ in $\alpha_1, \dots, \alpha_{n_\alpha}$ do
4:         Calculate $\tilde{S}$ as in (4.13).
5:         for $\lambda$ in $\lambda_1, \dots, \lambda_{n_\lambda}$ do
6:             Assign $\hat{c}^*_{ii} := -\infty$ for $i = 1, \dots, d$ and assign equal starting values $\hat{c}^*_{ij}$
   for $i \neq j$.
7:             Solve $\hat{C}^* = \underset{\substack{A\in\mathbb{R}^{d\times d} \\ A \text{ supported on DAG}}}{\arg\min} \|(A^{\oplus(d-1)} \oplus \tilde{S}) \vee S - S\|_1 - \lambda f(A)$.
8:             Use 2-means clustering to divide all values $\hat{c}^*_{ij} > -\infty$ of $\hat{C}^*$ into two
   groups
9:             Set $\hat{c}^*_{ij} = -\infty$ whenever $\hat{c}_{ij}$ belongs to the group of smaller means and
   update $\hat{C}^*$ by its Kleene star matrix using (4.6).
10:        end for
11:    end for
12: end for
13: For each pair of parameters $(\lambda, \alpha)$, calculate the mean normalized Structural
    Hamming distance between $\hat{C}^*$ and $\hat{C}^*_1, \dots, \hat{C}^*_{n_s}$, where $\hat{C}^*$ is the estimate from
    data $(Y, \tilde{Y})$ and $\hat{C}^*_k$ the estimate from data $(Y_k, \tilde{Y}_k)$ for $k = 1, \dots, n_s$
14: Choose the set of parameters $(\lambda_{\text{opt}}, \alpha_{\text{opt}})$ with the lowest mean normalized
    Structural Hamming distance
15: Choose $\hat{C}^*$ to be the result for sample $(Y, \tilde{Y})$ and parameters $(\lambda_{\text{opt}}, \alpha_{\text{opt}})$.
16: Return $\hat{C}^*$ and parameters $(\lambda_{\text{opt}}, \alpha_{\text{opt}})$.
```

4.3 Data Application

We want to investigate the performance of the estimator by conducting a simulation study and apply the theory to real-world data sets.

4.3.1 Application I: Simulation Study

We conduct a simulation study to test the quality of the estimator. Particularly, we are interested in the overall performance and breaking points of the estimator. To do so, we consider Bayesian networks with number of nodes $d \in \{10, 20, 30\}$. For each d, we generate a directed Erdős–Rényi-graph $\mathcal{G}$ with probability $p = 30\%$ of drawing an edge and then delete all values of the lower triangle resulting in a DAG $\mathcal{D}$. For each edge in $\mathcal{D}$, we assign a random edge weight drawn from a uniform distribution $\mathcal{U}[0.1, 1]$ which results in an edge weight matrix $C \in \mathbb{R}^{d\times d}$ upon which we calculate the Kleene star matrix C^*.

We then define X as in (4.7) by

$$X_i = \bigvee_{j\in \mathrm{An}(i)} c^*_{ij} + Z_j, \quad i = 1, \dots, d. \tag{4.23}$$

We also consider an additive model, i.e.

$$X_i = \sum_{j\in \mathrm{An}(i)} c^*_{ij} + Z_j, \quad i = 1, \dots, d, \tag{4.24}$$

to test for robustness in the presence of misspecifications in the model. We consider sample sizes $n \in \{50, 100, 250, 500\}$. For each n, we generate vectors $z_1, \dots, z_d \in \mathbb{R}^n$ of innovations drawn from a Fréchet distribution with shape 1 and scale uniformly drawn from $\mathcal{U}[0.1, 1]$, i.e. $z_1, \dots, z_d$ differ in distribution by a different and randomly chosen scale parameter. This ensures that nodes with many ancestors do not necessarily have larger observations. Finally, as we work with the logarithmized version of the max-linear model, we take the log of the Kleene star matrix C^* and $z_1, \dots, z_d$ and calculate $\mathcal{X} = \{x_1, \dots, x_n\}$ as in (4.23), respectively (4.24) which we each collect in a matrix $X \in \mathbb{R}^{d\times n}$.

For both models, we calculate the empirical standard deviation of $X_1, \dots, X_d \in \mathbb{R}^n$, calculate the mean σ_X over all standard deviations, and then set $Y := X + \mathcal{E}$, where $\mathcal{E} = (\varepsilon_{ij})_{n\times d}$ with mutually independent entries $\varepsilon_{ij} \sim \mathcal{N}(0, r\sigma_X)$. Here, r denotes the noise-to-signal ratio and we take values $r \in \{0.1, 0.3, 0.5\}$.

For each setting, we use Algorithm 6 with standard parameters $\alpha = 0.8$ and $\lambda = 0.3$ to estimate the Kleene star matrix C^*. Changing the setting does not alter the results very much. Moreover, we use (4.22) to save computation time, see Remark 4.2.3. We repeat this step $n_s = 100$ times and for each setting, we calculate the nSHD and TPR to the true DAG $\mathcal{D}$ as defined in (4.3) and then take the mean across all n_s estimations. The results can be found in Table 4.1 and Table 4.2.

In Table 4.1, we see the results for data generated by a max-linear model as in (4.23) polluted with noise. The estimator performs reliably with estimates quickly converging to the true underlying network. For the smallest noise-to-signal ratio of

r		10%				30%				50%			
n		50	100	250	500	50	100	250	500	50	100	250	500
$d = 10$	nSHD	0.21	0.11	0.05	0.03	0.15	0.09	0.05	0.03	0.21	0.15	0.11	0.07
	TPR	0.85	0.93	0.97	0.98	0.94	0.97	0.98	0.99	0.88	0.92	0.93	0.95
$d = 20$	nSHD	0.19	0.13	0.04	0.02	0.14	0.09	0.06	0.03	0.18	0.14	0.10	0.08
	TPR	0.89	0.93	0.99	0.99	0.92	0.96	0.99	0.99	0.88	0.91	0.95	0.96
$d = 30$	nSHD	0.17	0.13	0.07	0.04	0.14	0.11	0.07	0.04	0.16	0.13	0.09	0.07
	TPR	0.89	0.95	0.97	0.97	0.92	0.96	0.98	0.99	0.92	0.94	0.95	0.97

Table 4.1: Normalized Structural Hamming Distance (first row) and True Positive Rate (second row) as defined in (4.3) for various number of nodes d, sample sizes n and noise-to-signal ratios r and X generated as in (4.23). Colors illustrate the estimation quality of results (from yellow to green). The estimator performs reliably even for small data sets and converges to the underlying network.

r		10%				30%				50%			
n		50	100	250	500	50	100	250	500	50	100	250	500
$d = 10$	nSHD	0.16	0.09	0.04	0.03	0.15	0.10	0.05	0.04	0.20	0.15	0.11	0.08
	TPR	0.90	0.97	0.99	1.00	0.94	0.97	0.98	0.99	0.88	0.92	0.92	0.94
$d = 20$	nSHD	0.15	0.10	0.06	0.03	0.15	0.11	0.08	0.05	0.18	0.16	0.12	0.10
	TPR	0.88	0.95	0.99	1.00	0.88	0.94	0.98	0.99	0.84	0.86	0.92	0.93
$d = 30$	nSHD	0.14	0.10	0.06	0.04	0.14	0.11	0.08	0.06	0.16	0.13	0.11	0.09
	TPR	0.89	0.96	0.98	0.99	0.92	0.94	0.97	0.99	0.88	0.90	0.91	0.93

Table 4.2: Normalized Structural Hamming Distance (first row) and True Positive Rate(second row) as defined in (4.3) for various number of nodes d, sample sizes n and noise-to-signal ratios r and X generated as in (4.24). Colors illustrate the estimation quality of results (from yellow to green). Despite the misspecification, the estimator performs reliably even for small data sets and converges to the underlying network. However, the performance is slightly worse than the model that is not misspecified, see 4.1.

10%, results are even good for very small sample sizes. For example, for $n = 50$, we still reach a nSHD and TPR between 0.17 and 0.21, respectively 0.85 and 0.89. As the noise gets heavier, accuracy expectedly decreases. However, even for a noise-to-signal ratio of 50% and $n = 50$ observations, we reach a similar performance if the number of observations is small. However, the convergence generally is slower. An interesting observation is that an increase in the number of nodes leads to a slightly better estimation. This can be explained by the graph generation. For a pair of nodes (j, i), we have a 30% chance of drawing an edge between the two nodes. However,

Code	Foreign Exchange Rate (into GBP)	Code	Foreign Exchange Rate (into GBP)
AUS	Australian Dollar	MYS	Malaysian Ringgit
CAN	Canadian Dollar	NOR	Norwegian Krone
CHE	Swiss Franc	NZL	New Zealand Dollar
CHN	Chinese Yuan	POL	Polish Zloty
CZE	Czech Koruna	RUS	Russian Ruble
DNK	Danish Crown	SAU	Saudi Riyal
EUR	Euro	SGP	Singapore Dollar
HKG	Hong Kong Dollar	SWE	Swedish Krona
HUN	Hungarian Forint	THA	Thai Baht
IND	Indian Rupee	TUR	Turkish Lira
ISR	Israeli Shekel	TWN	Taiwan Dollar
JPN	Japanese Yen	USA	US Dollar
KOR	South Korean Won	ZAF	South African Rand

Table 4.3: Currency codes and the respective currency, measured as foreign exchange rate into British Pound sterling.

when calculating the Kleene star matrix C^*, we obtain a denser matrix C^* if the number of nodes increases.

As for the model defined in (4.24), the estimator surprisingly performs very well, on par with data from the max-linear model. For smaller sample sizes, the estimator often performs better, however, the convergence is often slower, especially for noise-to-signal ratio 50%. Overall, the good results illustrate the robustness in terms of misspecifications. We, therefore, conclude that the estimator performs well even under large misspecifications and the presence of significant noise.

4.3.2 Application II: Foreign Exchange Rates

We have seen that the estimator performs reliably even under strong noise and misspecified models. Motivated by this success, we also want to illustrate the performance of the proposed methodology on foreign exchange rates of $d = 26$ currencies expressed in terms of the British Pound sterling. Although the assumption that foreign exchange rates follow a DAG might be unrealistic, the example allows for an easy and intuitive interpretation of the estimation result. The data has been first considered in Engelke and Volgushev [2020] and further studied in Engelke et al. [2021].

The data ranges from the 1st of October 2005 to the 30th of September 2020, resulting in $n = 3790$ observations. To eliminate the temporal dependence, Engelke and Volgushev [2020] first preprocess the data. They take log returns and then fit each univariate series to a ARMA-GARCH process, taking absolute values of the standardized filtered returns. For more information, see Engelke and Volgushev [2020]. Observe that taking absolute values leads to considering both tails. We run Algorithm 7 with $\lambda \in \{0.3, 0.35, 0.4, 0.45, 0.5\}$, $\alpha = 0.8$ and resample rate $n_s = 100$. Moreover, we follow Remark 4.2.3.

In order to illustrate the estimation results, we do not plot the entire DAG $\hat{\mathcal{D}}$ that results from Algorithm 7. Instead, we draw only the 50 largest weights $\hat{c}^*_{ij}$ of $\hat{C}^*$. This leads to a better interpretation as otherwise, the overload of ancestral relations would make the graphic interpretation of the result very difficult. Moreover, the thickness of an edge $j \to i$ illustrates the weight $\hat{c}^*_{ij}$, where thicker edges mean larger weight $\hat{c}^*_{ij}$.

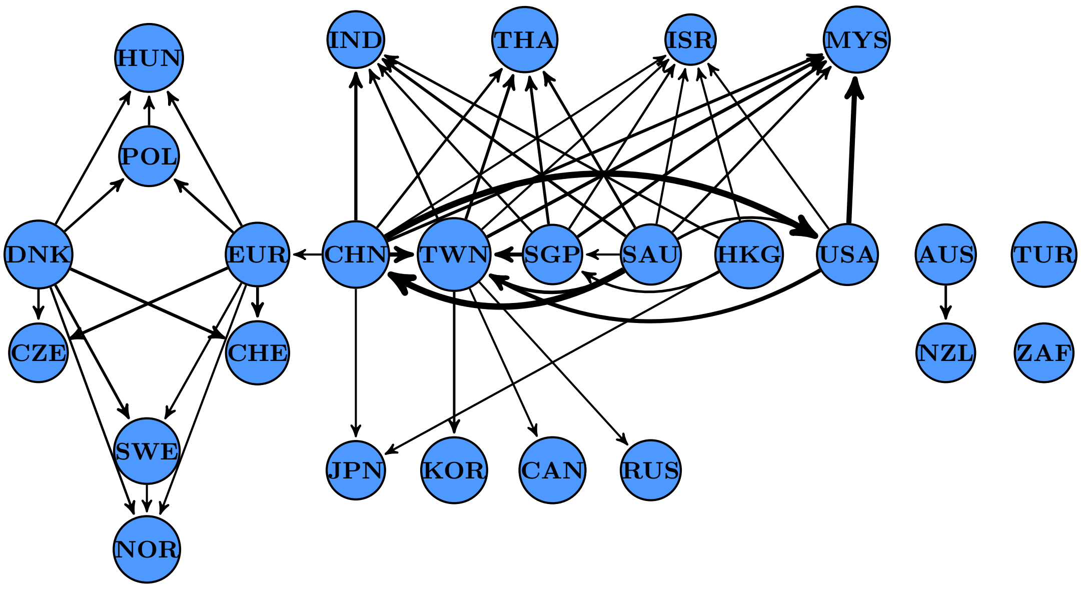


Figure 4.1: 50 largest weights of $\hat{C}^*$ based on Algorithm 7 with $\lambda \in \{0.3, 0.35, 0.4, 0.45, 0.5\}$, $\alpha = 0.8$ and $n_s = 100$. The algorithm chooses $\lambda = 0.45$ as the optimal weight. The thickness of an edge illustrates its respective estimated weight, i.e. the larger the estimated weight, the thicker the edge. The 26 nodes separate into three clusters. The first cluster is strictly European and connected with the second cluster by only a single edge. The third cluster is the smallest one, containing only New Zealand and Australia. There are 2 nodes that are isolated, i.e. none of the 50 largest weights in $\hat{C}^*$ connect with any of the two nodes.

The result can be found in Figure 4.1. A table of the currency codes can be found in Table 4.3.

We can see that the 50 largest weights of $\hat{C}^*$ divide the set of currencies into three clusters. The first cluster is a purely European cluster, consisting of the currencies of Hungary, Poland, the Euro area, Denmark, the Czech Republic, Switzerland, Norway, and Sweden. Out of the mentioned currencies, the Euro is by far the most traded currency. Therefore, one expects that the Euro also rather impacts other European currencies than being impacted by them. This exact phenomenon can be seen in the estimation result. While EUR has only outgoing edges within the cluster, other smaller currencies like HUN, NOR, CZE, and CHE have only incoming edges. It also seems reasonable that Poland and Hungary and Sweden and Norway share a connection as they share close ties by spatial proximity. A little surprising, however, is the strong role of the Danish currency in this cluster.

Moving to the second cluster, we can see that the only connection is the one between the currency of China and the Euro area. This makes sense, considering that China is one of Europe's largest suppliers of manufactured goods. Within the second cluster, only the currencies of Singapore, the United States, Saudi Arabia, China, Hong Kong, and Taiwan have outgoing edges. Moreover, the only two currencies with only outgoing edges are the currencies of Saudi Arabia and Hong Kong. This makes a lot of sense. For Saudi Arabia, its economy is extremely dependent on sales of fossil fuels. Therefore, the currency serves as a good indicator for the world's economic sentiment.

Similarly, also Singapore and Hong Kong are good indicators for the world's economic sentiment as they are both highly developed free-market economies with almost free port trade and very strong financial markets. It is also reasonable to assume that the US and Chinese currencies impact other currencies. The United States is the world's largest economy and its currency is by far the most traded in the world. China, on the other hand, became the 2nd largest economy in the second quarter of 2010 and world's largest producer of manufactured goods in 2011. Finally, with Taiwan being the world market leader in the semiconductor industry, one can also argue that its currency is an indicator for the economic sentiment of the world. Therefore, it is reasonable, that these currencies impact the currencies of Thailand, Israel and Malaysia, though it is a bit surprising for India. As for Japan, there are two incoming edges from China And Hong Kong which can be explained well by spatial proximity and economic ties. The edge from Taiwan to Korea, Canada, and Russia, however, is surprising.

The third cluster is the smallest containing only the currencies of Australia and New Zealand. Nevertheless, for obvious reasons, it seems very logical that the currency of Australia impacts the one from New Zealand. The only two isolated edges are the currencies of Turkey and South Africa. Considering that both currencies depreciated massively in the considered time span, this is not surprising.

Finally, the three thickest edges are the one between the currencies of Saudi Arabia and China, China and USA, and USA and Myanmar. We interpret it in the sense that China as a country of manufacturing are very sensitive to the economic sentiment of the world. Also, the United States as the biggest importer of Chinese goods naturally share a strong connection. The strong connection between the USA and Malaysia, however, is surprising. Comparing the overall result with Engelke and Volgushev [2020], we can see that many connections coincide. However, one needs to note that their result is inherently different as the authors estimate a minimum spanning tree.

4.3.3 Application III: Upper Danube Basin

The river network of the *Upper Danube* consists of 31 measuring points along the Danube basin and its tributaries. This data set was considered first in Asadi et al. [2015] and subsequently considered in research [Engelke and Hitz, 2020, Gnecco et al., 2021, Mhalla et al., 2020, Tran et al., 2021]. The data set ranges from 1960–2009 and Asadi et al. [2015] suggest a declustering approach to generate a data set of 428 supposedly independent observations. For more information on the Upper Danube data set, see Asadi et al. [2015].

We use MLDAG with automated parameter selection as described in Algorithm 7 with $\lambda \in \{0.3, 0.35, 0.4, 0.45, 0.5\}$ and $\alpha = 0.8$. An analysis has shown that the result does not vary on α very much. We use $n_s = 100$ resamples $Y_1, \dots, Y_{100}$. Moreover, we follow Remark 4.2.3.

Since the true DAG $\mathcal{D}$ is known, we output the results in terms of the nSHD and TPR as in the simulation study. For a definition, see (4.3). We also compare the estimator with automated QTree which uses the pair-wise score matrix as defined in (4.11) and applies Chu-Liu/Edmonds' algorithm. For more information, see Tran et al. [2021].

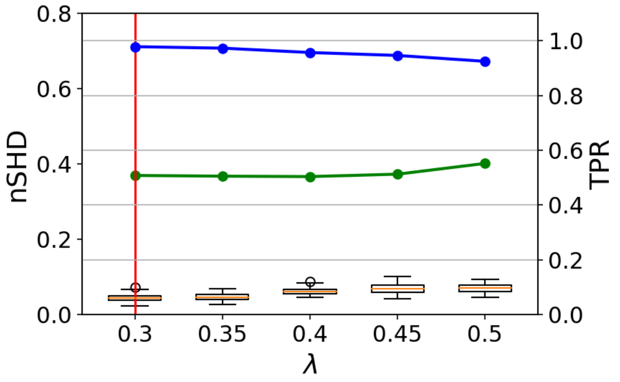


Figure 4.2: Metrics nSHD and TPR as defined in (4.3) for the outputs of MLDAG as in Algorithm 7 for varying parameters λ and the Danube network. The figure has two x-axes. The green line together with the left axis shows the nSHD for $(\hat{\mathcal{D}}_\lambda, \mathcal{D})$ while the blue line together with the right axis shows the TPR for $(\hat{\mathcal{D}}_\lambda, \mathcal{D})$.
To choose λ, we sample with replacement and repeated this 100 times, creating 100 data sets. For each λ, we fit MLDAG on these data sets to obtain 100 estimated DAGs and compute the nSHD between $(\hat{\mathcal{D}}_\lambda$ and the 100 DAGs (presented as boxplots, using the left axis). We choose the parameter λ where the mean nSHD between $(\hat{\mathcal{D}}_\alpha$ and the 100 DAGs is minimized(red, vertical line).

Since QTree estimates the graph and not its reachability, we compare MLDAG to the reachability graph of QTree . The results can be found in Table 4.4. We can see that while QTree has a nSHD of just 9%, MLDAG is significantly higher at 37%. Since both algorithms are developed around max-linear models, we conclude that the DAG assumption significantly helps to recover the network. Interestingly, however, the TPR for MLDAG is almost perfect at 98%, significantly higher than the one of QTree . This shows that MLDAG is very successful in recovering ancestral relations but often fails in separating the set of ancestral relations from the set of spurious edges.

Finally, we want to assess the parameter selection. This is done in Figure 4.2. Observe that the green line together with the left axis shows the nSHD for $(\hat{\mathcal{D}}_\lambda, \mathcal{D})$ while the blue line together with the right axis shows the TPR for $(\hat{\mathcal{D}}_\lambda, \mathcal{D})$. We can see that for the chosen parameter $\lambda_{\text{opt}} = 0.3$, the nSHD is lowest while the TPR is highest. For this reason, on the considered data set, the parameter selection chooses the optimal parameter.

	nSHD	TPR
MLDAG	0.37	0.98
QTree	0.09	0.87

Table 4.4: Normalized Structural Hamming Distance(nSHD) and True Positive Rate(TPR) as defined in (4.3) for the Danube and the estimator as in Algorithm 7 with $alpha = 0.8$ and $\lambda \in \{0.3, 0.35, 0.4, 0.45, 0.5\}$. The third column displays the λ chosen in the parameter selection. The TPR is close to 1, suggesting that we recover almost every edge of the underlying network. However, with nSHD at 0.42, we can see that the number of ancestral relations in $\mathcal{D}$ exceeds the number of ancestral relations in $\mathcal{D}$.

4.4 Asymptotics

We are now ready to state the main theorem which ensures asymptotic consistency. Observe that restricting to large values as in (4.13) ensures that the data approximately follows a recursive ML model. As for asymptotics, we can just prove the case of $\alpha = 0$.

Theorem 4.4.1. *Let Y be a noisy max-linear model from a DAG $\mathcal{D}$ as in* (4.12) *with a given i.i.d. sample $\mathcal{Y} := \{y_1, \ldots, y_n\} \in \mathbb{R}^d$. Let the regularization term f satisfy properties (a)-(f) and let both, the innovations $Z_1, \ldots, Z_d$ and $\varepsilon_1, \ldots, \varepsilon_d$ be i.i.d. with continuous densities and*

$$\lim_{x\to\infty} f_{\varepsilon_i - \varepsilon_j}(x)/f_{Z_i}(x+c) \to 0, \tag{4.25}$$

where $c > 0$ is an arbitrary constant.

Moreover, assume that $\mathcal{D}$ is neither the empty nor the complete DAG, i.e. it has more than 0 and less than $d(d-1)/2$ ancestral relations. Then, there exists some λ^* such that for any $0 < \lambda \le \lambda^*$ and $\alpha = 0$, Algorithm 6 returns a consistent estimator, i.e. $\hat{c}^*_{ij} > -\infty$ if and only if $c^*_{ij} > -\infty$ as the sample size $n \to \infty$.

Proof. We start by considering the empirical terms. Let us define $g : A \to \|A \oplus Y - Y\|_1$. We first consider the term

$$\|A \oplus Y - Y\|_1 - \lambda f(A) = g(A) - \lambda f(A). \tag{4.26}$$

Recall that by (4.2), we have

$$g(A) = \|A \oplus Y - Y\|_1 = \frac{1}{n}\sum_{t=1}^{n}\sum_{i=1}^{d}\Big(\bigvee_{k=1}^{d}(a_{ik} + Y_k^t) - Y_i^t\Big).$$

We are interested in the partial derivative with respect to a_{ij} of the upper expression. Observe that for an observation $t \in \{1, \ldots, n\}$, it holds that

$$\frac{\partial}{\partial a_{ij}} \bigvee_{k=1}^{d}(a_{ik} + y_k^t) - y_i^t = \begin{cases} 1, & \text{if } \; a_{ij} + y_j^t > \bigvee\limits_{\substack{k=1\\k\neq j}}^{d}(a_{ik} + Y_k^t) \\ 0, & \text{if } \; a_{ij} + y_j^t < \bigvee\limits_{\substack{k=1\\k\neq j}}^{d}(a_{ik} + y_k^t). \end{cases} \tag{4.27}$$

However, if $a_{ij} + Y_j^t = \bigvee_{\substack{k=1,\ldots,d\\k\neq j}}(a_{ik} + Y_k^t)$, then the partial derivative with respect to a_{ij} does not exist since the left and the right partial derivative do not match. However, we will just work with the right partial derivative and shortly refer to it as the partial derivative.

We also consider the partial derivative of f with respect to a_{ij}. Since the partial derivative of f exists and is upper bounded by condition (e), for any $\varepsilon > 0$ we have a $\lambda^* > 0$ such that for all $0 < \lambda \le \lambda^*$

$$\frac{\partial}{\partial a_{ij}}\lambda\, f(A) \le \varepsilon. \tag{4.28}$$

Now observe that when increasing a_{ij}, the value of (4.26) gets smaller if and only if

$$\frac{\partial}{\partial a_{ij}} g(A) < \frac{\partial}{\partial a_{ij}} \lambda f(A).$$

For this reason, using (4.28), for any arbitrarily chosen threshold a_{ij}^T, for a sufficiently large sample size n, we can find a $\lambda^* > 0$ sufficiently small, such that

$$\frac{\partial}{\partial a_{ij}} \lambda f(A) < \frac{\partial}{\partial a_{ij}} g(A), \tag{4.29}$$

whenever $a_{ij} \geq a_{ij}^T$ and for all $0 < \lambda \leq \lambda^*$. For this reason, for the matrix A that minimizes (4.26), by choosing λ^* sufficiently small, we can assume $a_{ij} < a_{ij}^T$ and since a_{ij}^T is arbitrary, without loss of generality we can assume a_{ij} to be arbitrarily small. Repeating this argument, we can assume the same for any ordered pair of nodes (j, i).

Now let us assume that there is a path from j to i in $\mathcal{D}$. Then, observing (4.27) and (4.12) and

$$\bigvee_{\substack{k=1,\dots,d \\ k\neq j}} (a_{ik} + y_k^t) = \bigvee_{\substack{k=1,\dots,d \\ k\neq j,i}} (a_{ik} + y_k^t) \vee y_i^t,$$

we have

$$\begin{aligned}
\frac{\partial}{\partial a_{ij}} \|A \oplus Y - Y\|_1 &= \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + y_j^t \geq \bigvee_{\substack{k=1,\dots,d \\ k\neq j,i}} (a_{ik} + y_k^t) \vee y_i^t \right) \leq \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + y_j^t \geq y_i^t \right) \\
&\leq \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + x_j^t + \varepsilon_j^t \geq x_j^t + c_{ij}^* + \varepsilon_i^t \right) = \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(\varepsilon_j^t - \varepsilon_i^t \geq c_{ij}^* - a_{ij} \right) \\
&\to \mathbb{P}(\varepsilon_j - \varepsilon_i \geq c_{ij}^* - a_{ij}) \quad \text{as } n \to \infty,
\end{aligned} \tag{4.30}$$

where we used the law of large numbers in the last line.

Contrarily, if j is not an ancestor of i, then $j \in \mathrm{An}(j) \setminus \mathrm{An}(i)$. Moreover, recall that by (4.29) and the subsequent text, we can assume a_{ik} to be arbitrarily small for any ordered pair (k, i). For this reason, by (4.12) we get

$$\begin{aligned}
\frac{\partial}{\partial a_{ij}} \|A \oplus Y - Y\|_1 &= \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + y_j^t \geq \bigvee_{\substack{k=1,\dots,d \\ k\neq j,i}} (a_{ik} + y_k^t) \vee y_i^t \right) \geq \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + y_j^t \geq y_i^t + \varepsilon \right) \\
&= \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + \bigvee_{k \in \mathrm{An}(j)} c_{jk}^* + z_k^t + \varepsilon_j^t \geq \bigvee_{k \in \mathrm{An}(i)} c_{ik}^* + z_k^t + \varepsilon_i^t + \varepsilon \right) \\
&\geq \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}\left(a_{ij} + z_j^t + \varepsilon_j^t \geq \bigvee_{k \in \mathrm{An}(i)} c_{ik}^* + z_k^t + \varepsilon_i^t + \varepsilon \right) \\
&\to \mathbb{P}\left(Z_j - \left(\bigvee_{k \in \mathrm{An}(i)} c_{ik}^* + Z_k + \varepsilon_i^t \right) + \varepsilon_j^t \geq -a_{ij} + \varepsilon \right)
\end{aligned}$$

for some constant ε that is independent of a_{ij}. Now observe that for any independent random variables X and Y with $\mathbb{P}(Y \geq 0) > 0$, it holds that for some $c > 0$

$$\begin{aligned}\mathbb{P}(X+Y \geq x) &= \mathbb{P}(X+Y \geq x \ \cap \ Y \geq 0) + \mathbb{P}(X+Y \geq x \ \cap \ Y < 0) \\ &\geq \mathbb{P}(X+Y \geq x \ \cap \ Y \geq 0) \geq P(X \geq x \ \cap \ Y \geq 0) = c\, P(X \geq x).\end{aligned}$$

Therefore, using this argument twice, first for $Y := (\bigvee_{k \in \mathrm{An}(i)} c^*_{ik} + Z_k + \varepsilon^t_i)$ and then repeating the argument for $Y := \varepsilon_i$, we have

$$\begin{aligned}&\mathbb{P}\big(Z_j - (\bigvee_{k \in \mathrm{An}(i)} c^*_{ik} + Z_k + \varepsilon^t_i) + \varepsilon^t_j \geq -a_{ij} + \varepsilon\big) \\ &\geq c\, \mathbb{P}\left(Z_j \geq -a_{ij} + \varepsilon\right) \quad \text{as } n \to \infty, \qquad (4.31)\end{aligned}$$

where $c > 0$ is some constant independent of a_{ij} and the choice of λ. Now let (j,i) and (k,l) be arbitrary pair of nodes with $j \neq i$, $k \neq l$ and $j \in \mathrm{An}(i)$ and $k \notin \mathrm{An}(l)$. Then, comparing (4.30) and (4.31) and considering (4.25), we can see that

$$\frac{\partial}{\partial a_{lk}} \|A \oplus Y - Y\|_1 / \frac{\partial}{\partial a_{ij}} \|A \oplus Y - Y\|_1 \to 0 \quad \text{for } n \to \infty,$$

for $a_{lk} = a^*_{ij} + c$ and $a_{ij} \to -\infty$ with $c > 0$ an arbitrary, fixed constant. At the same time, by condition (f) of the regularization term and (4.30), we have

$$\frac{\partial}{\partial a_{ij}} \|A \oplus Y - Y\|_1 / \frac{\partial}{\partial a_{ij}} f(C^*) \to 0 \quad \text{for } n \to \infty,$$

and $a_{ij} \to -\infty$. Since by (4.29) and the subsequent text, we can assume a_{ij} to be arbitrarily small. For every $\delta > 0$, we can find λ^* such that (4.26) has minimum solution A with

$$\min_{\substack{(j,i) \in d \times d \\ j \in \mathrm{An}(i)}} a_{ij} > \max_{\substack{(j,i) \in d \times d \\ j \notin \mathrm{An}(i)}} a_{ij} + \delta \qquad (4.32)$$

for any fixed λ with $0 < \lambda \leq \lambda^*$ and $n \to \infty$. Since $\alpha = 0$, $\tilde{Y} = Y$, where $\tilde{Y}$ is defined as in (4.13). Therefore, moving to the original optimization problem

$$\underset{A \in \mathbb{R}^{d \times d}}{\arg\min} \|(A^{\oplus(d-1)} \oplus \tilde{Y}) \vee Y - Y\|_1 - \lambda f(A)$$

as given in line 4 of 6, iteratively applying the previous arguments, as $n \to \infty$, (4.32) holds equally for the upper expression.

Finally, we apply a 2-mean clustering approach in line 5 of Algorithm 6, i.e. we seek to find sets $S := \{S_1, S_2\}$ with respective means μ_1 and $\mu_2 \in \mathbb{R}$ such that

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{2} \sum_{\mathbf{x} \in S_i} (\mathbf{x} - \boldsymbol{\mu}_i)^2. \qquad (4.33)$$

Since the DAG $\mathcal{D}$ is neither complete nor isolated, the two sets are not empty and $\bigcup_{\substack{(j,i) \in d \times d \\ j \in \mathrm{An}(i)}} a_{ij}$ and $\bigcup_{\substack{(j,i) \in d \times d \\ j \notin \mathrm{An}(i)}} a_{ij}$ diverge and, hence, for λ^* sufficiently small and all $0 < \lambda \leq \lambda^*$, as $n \to \infty$, (4.33) splits the two groups correctly. Therefore, as $n \to \infty$, the estimator is consistent. $\square$

Corollary 4.4.2. *If the innovations are Gumbel, the noise is as in* (4.17) *and f is chosen as in* (4.16)*, then the condition for the theorem is satisfied.*

Proof. We already discussed that for f as in (4.16) and the noise as in (4.17), conditions (a)-(f) are satisfied, see (4.16) and the subsequent text. Therefore, we only need to show (4.25). However, by (4.18) and comparing it with the density of the Gumbel distribution, we see immediately that the noise has lighter tail and, therefore, (4.25) holds. $\square$

Bibliography

C. Améndola, B. Hollering, S. Sullivant, and N. Tran. Markov equivalence of max-linear bayesian networks. In *Uncertainty in Artificial Intelligence*, pages 1746–1755. PMLR, 2021.

C. Améndola, C. Klüppelberg, S. Lauritzen, and N. M. Tran. Conditional independence in max-linear Bayesian networks. *The Annals of Applied Probability*, 32(1): 1 – 45, 2022.

M. P. Anderson, W. W. Woessner, and R. J. Hunt. *Applied Groundwater Modeling: Simulation of Flow and Advective Transport.* Academic Press, 2015.

P. Asadi, A. C. Davison, and S. Engelke. Extremes on river networks. *The Annals of Applied Statistics*, 9(4):2023–2050, 2015.

S. Asenova and J. Segers. Max-linear graphical models with heavy-tailed factors on trees of transitive tournaments. arXiv preprint arXiv: 2209.14938, 2022.

S. Asenova, G. Mazo, and J. Segers. Inference on extremal dependence in the domain of attraction of a structured hüsler–reiss distribution motivated by a markov tree with latent variables. *Extremes*, 24:461 – 500, 2021.

F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat. *Synchronization and linearity: an algebra for discrete event systems.* John Wiley & Sons Ltd, 1992.

A. Balkema, C. Klüppelberg, and S. Resnick. Densities with Gaussian tails. *Proceedings of the London Mathematical Society*, 66(3):568–588, 1993.

M. Bartos, B. Wong, and B. Kerkez. Open storm: a complete framework for sensing and control of urban watersheds. *Environmental Science: Water Research & Technology*, 4(3):346–358, 2018.

J. Beirlant, Y. Goegebeur, J. Segers, and J. Teugels. *Statistics of Extremes: Theory and Applications.* Wiley, Chichester, 2004.

D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods.* Academic press, 2014.

N. Bingham, C. Goldie, and J. Teugels. *Regular Variation.* Cambridge University Press, Cambridge, 1987.

K. A. Bollen. *Structural Equations with Latent Variables.* Wiley, New York, 1989.

A. Boneh and M. Hofri. The coupon-collector problem revisited—a survey of engineering problems and computational methods. *Stochastic Models*, 13(1):39–66, 1997.

S. Boneh and V. G. Papanicolaou. General asymptotic estimates for the coupon collector problem. *Journal of Computational and Applied Mathematics*, 67(2):277–289, 1996.

J. Buck and C. Klüppelberg. Recursive max-linear models with propagating noise. *Electronic Journal of Statistics*, 15(2):4770 – 4822, 2021.

P. Butkovič. *Max-linear Systems: Theory and Algorithms.* Springer, London, 2010.

W. Chen, M. Drton, and Y. Wang. On causal discovery with equal variance assumption. *Biometrika*, 106(4):973–980, 2019.

D. M. Chickering. Learning Bayesian networks is np-complete. In *Learning from data*, pages 121–130. Springer, 1996.

S. Coles, J. Heffernan, and J. Tawn. Dependence measures for extreme value analyses. *Extremes*, 2(4):339–365, 1999.

S. Coles, J. Bawa, L. Trenner, and P. Dorazio. *An Introduction to Statistical Modeling of Extreme Values.* Springer, 2001.

R. Davis and W. McCormick. Estimation for first-order autoregressive processes with positive or bounded innovations. *Stoch. Proc. Appl.*, 31:237–250, 1989.

R. Davis and S. Resnick. Basic properties and prediction of max-arma processes. *Advances in Applied Probability*, 21(4):781–803, 1989.

A. C. Davison and R. Huser. Statistics of extremes. *Annual Review of Statistics and its Application*, 2:203–235, 2015.

L. De Haan and A. Ferreira. *Extreme value theory: an introduction.* Springer Science & Business Media, 2007.

M. T. Drton and M. H. Maathuis. Structure learning in graphical modeling. *Annual Review of Statistics and Its Application*, 4(3):365–393, 2017.

J. H. J. Einmahl, A. Kiriliouk, and J. Segers. A continuous updating weighted least squares estimator of tail dependence in high dimensions. *Extremes*, 21:205 – 233, 2017.

P. Embrechts, C. Klüppelberg, and T. Mikosch. *Modelling Extremal Events for Insurance and Finance.* Springer, Berlin, 1997.

S. Engelke and A. Hitz. Graphical models for extremes. *JRSS B*, 82:871–932, 2020.

S. Engelke and S. Volgushev. Structure learning for extremal tree models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84:2055 – 2087, 2020.

S. Engelke, A. S. Hitz, and N. Gnecco. *graphicalExtremes: Statistical Methodology for Graphical Extreme Value Models*, 2019. URL `https://CRAN.R-project.org/package=graphicalExtremes`. R package version 0.1.0.

S. Engelke, M. Lalancette, and S. Volgushev. Learning extremal graphical structures in high dimensions. *arXiv preprint arXiv: 2111.00840*, 2021.

H. N. Gabow, Z. Galil, T. Spencer, and R. E. Tarjan. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica*, 6(2): 109–122, 1986.

N. Gissibl. *Graphical Modeling of Extremes: Max-linear Models on Directed Acyclic Graphs.* Ph.D. thesis, Technical University of Munich, 2018.

N. Gissibl and C. Klüppelberg. Max-linear models on directed acyclic graphs. *Bernoulli*, 24(4A):2693–2720, 2018.

N. Gissibl, C. Klüppelberg, and M. Otto. Tail dependence of recursive max-linear models with regularly varying noise variables. *Econometrics and Statistics*, 6:149 – 167, 2018.

N. Gissibl, C. Klüppelberg, and S. Lauritzen. Identifiability and estimation of recursive max-linear models. *Scandinavian Journal of Statistics*, 48(1):188–211, 2021. doi: 10.1111/sjos.12446.

N. Gnecco, N. Meinshausen, J. Peters, and S. Engelke. Causal discovery in heavy-tailed models. Annals of Statistics, to appear, 2021.

Y. Gong, P. Zhong, T. Opitz, and R. Huser. Partial tail-correlation coefficient applied to extremal-network learning. arXiv preprint arXiv: 2210.07351, 2022.

I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *30th International Conference on Machine Learning, ICML 2013*, 1302, 02 2013.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning.* MIT press, 2016.

M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer, Heidelberg, 1988.

A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using networkX. In G. Varoquaux, T. Vaught, and J. Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

P. Hall and I. Van Keilegom. Nonparametric “regression” when errors are positioned at end-points. *Bernoulli*, 15(3):614–633, 2009.

P. Hall, L. Peng, and Q. Yao. Moving-maximum models for extrema of time series. *Journal of Statistical Planning and Inference*, 103:51–63, 04 2002.

B. Hollering and S. Sullivant. Discrete max-linear bayesian networks. *Algebraic Statistics*, 12:213–225, 12 2021. doi: 10.2140/astat.2021.12.213.

S. Hu, Z. Peng, and J. Segers. Modelling multivariate extreme value distriubtions via markov trees. arXiv preprint arXiv: 2208.02627, 2022.

G. Italiano. Amortized efficiency of a path retrieval data structure. *Theoretical Computer Science*, 48(3):273–281, 1986.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R.* Springer, 2013.

A. Janßen and P. Wan. k-means clustering of extremes. *Electronic Journal of Statistics*, 14(1):1211 – 1233, 2020.

M. Jirak, A. Meister, and M. Reiss. Adaptive function estimation in nonparametric regression with one-sided errors. *Annals of Statistics*, 42(5):1970–2002, 2014.

M. J. Johnson, D. K. Duvenaud, A. Wiltschko, R. P. Adams, and S. R. Datta. Composing graphical models with neural networks for structured representations and fast inference. *Advances in neural information processing systems*, 29, 2016.

M. I. Jordan. *Learning in graphical models.* MIT press, 1999.

M. Joswig. *Essentials of Tropical Combinatorics.* Springer-Verlag, Heidelberg, New York, 2020.

J. Kiefer and J. Wolfowitz. Consistency of the maximum likelihood estimator in the presence of infinitely many incidental parameters. *Annals of Mathematical Statistics*, 27(4):887–906, 1956.

C. Klüppelberg and M. Krali. Estimating an extreme Bayesian network via scalings. *Journal of Multivariate Analysis*, 181:104672, 2021. doi.org/10.1016/j.jmva.2020.104672.

C. Klüppelberg and S. Lauritzen. Bayesian networks for max-linear models. In F. Biagini, G. Kauermann, and T. Meyer-Brandis, editors, *Network Science - An Aerial View from Different Perspectives.* Springer, 2020.

C. Klüppelberg and E. Sönmez. Max-linear models in random environment. *Journal of Multivariate Analysis*, 190:104999, 03 2022. doi: 10.1016/j.jmva.2022.104999.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, Cambridge, Massachusetts, 2009.

J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.

M. Larsson and S. I. Resnick. Extremal dependence measure and extremogram: the regularly varying case. *Extremes*, 15:231–256, 2012.

S. Lauritzen. *Graphical Models.* Clarendon Press, Oxford, United Kingdom, 1996.

S. Lauritzen. Causal inference from graphical models. In *Complex Stochastic Systems*, pages 63–107. Chapman and Hall/CRC Press, London/Boca Raton, 2001.

S. Lauritzen, A. P. Dawid, B. N. Larsen, and H.-G. Leimer. Independence properties of directed Markov fields. *Networks*, 20(5):491–505, 1990.

C. Leigh, O. Alsibai, R. J. Hyndman, S. Kandanaarachchi, O. C. King, J. M. McGree, J. S. Catherine Neelamraju, P. D. Talagala, R. D. Turner, K. Mengersen, and E. E. Peterson. A framework for automated anomaly detection in high frequency water-quality data from in situ sensors. *Science of The Total Environment*, 664(5): 885–898, 2019.

L.-H. Lim. Hodge Laplacians on graphs. *SIAM Review*, 62(3):685–715, 2015.

Lower Colorado River Authority (LCRA). Private correspondence, 2020.

Lower Colorado River Authority (LCRA). Highland lakes and dams, Accessed August 2020. `https://www.lcra.org/water/dams-and-lakes`.

M. Maathuis, M. Drton, S. Lauritzen, and M. Wainwright, editors. *Handbook of Graphical Models.* Chapman & Hall/CRC, 2019.

D. Maclagan and B. Sturmfels. *Introduction to Tropical Geometry.* Graduate Studies in Mathematics, Vol. 161. American Mathematical Society, Providence, Rhode Island, 2015.

J. MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297, 1967.

F. Mao, K. Khamis, S. Krause, J. Clark, and D. M. Hannah. Low-cost environmental sensor networks: recent advances and future directions. *Frontiers in Earth Science*, 7:221, 2019.

S. J. McGrane. Impacts of urbanisation on hydrological and water quality dynamics, and urban water management: a review. *Hydrological Sciences Journal*, 61(13): 2295–2311, 2016.

L. Mhalla, V. Chavez-Demoulin, and D. J. Dupuis. Causal mechanism of extreme river discharges in the upper danube basin network. *Journal of the Royal Statistical Society: Series C*, 69(4):741–764, 2020.

D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to linear regression analysis.* John Wiley & Sons, 2021.

D. R. Musser. Introspective sorting and selection algorithms. *Software: Practice and Experience*, 27(8):983–993, 1997.

J. Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, Cambridge, 2nd edition, 2009.

J. Pearl et al. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009.

J. Peters, J. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15:2009–2053, 2014.

D. Politis, J. Romano, and M. Wolf. *Subsampling.* Springer, New York, 1999.

R. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 35:1389–1401, 1957.

S. I. Resnick. *Extreme Values, Regular Variation, and Point Processes.* Springer, New York, 1987.

S. I. Resnick. *Heavy-Tail Phenomena: Probabilistic and Statistical Modeling.* Springer, New York, 2007.

J.-C. Rochet and J. Tirole. Interbank lending and systemic risk. *Journal of Money, Credit and Banking*, 28(4):733–762, 1996.

J. Rodriguez-Perez, C. Leigh, B. Liquet, C. Kermorvant, E. Peterson, D. Sous, and K. Mengersen. Detecting technical anomalies in high-frequency water-quality data using artificial neural networks. *Environmental Science & Technology*, 54(21): 13719–13730, 2020.

F. Rötter, S. Engelke, and P. Zwiernik. Total positivity in multivariate extremes. arXiv preprint arXiv: 2112.14727, 2022.

J. Segers. One- versus multi-component regular variation and extremes of Markov trees. *Advances in Applied Probability*, 52:855 – 878, 2019.

M. Sibuya. Bivariate extreme statistics. *Annals of the Institute of Statistical Mathematics*, 11(2):195–210, 1960.

R. Smith. Maximum likelihood estimation in a class of nonregular cases. *Biometrika*, 72(1):67–90, 1985.

R. Smith. Nonregular regression. *Biometrika*, 81(1):173–183, 1994.

P. Spirtes, C. N. Glymour, R. Scheines, and D. Heckerman. *Causation, Prediction, and Search.* MIT press, 2000.

N. Tran. The tropical geometry of causal inference for extremes. arXiv preprint arXiv: 2207.10227, 2022.

N. Tran and J. Yu. Product-mix auctions and tropical geometry. *Mathematics of Operations Research*, 44(4):1396–1411, 2019. doi: 10.1287/moor.2018.0975.

N. M. Tran. QTree Python Implementation. `https://github.com/princengoc/qtree`, 2021.

N. M. Tran, J. Buck, and C. Klüppelberg. Estimating a directed tree for extremes. arXiv preprint: 2102.06197, 2021.

J. M. Ver Hoef and E. Peterson. A moving average approach for spatial statistical models of stream networks. *Journal of the American Statistical Association*, 105 (489):6–18, 2010.

J. M. Ver Hoef, E. Peterson, and D. Theobald. Spatial statistical models that use flow and stream distance. *Environmental and Ecological Statistics*, 13(4):449–464, 2006.

M. J. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference.* Now Publishers Inc, 2008.

L. Wolf, C. Zwiener, and M. Zemann. Tracking artificial sweeteners and pharmaceuticals introduced into urban groundwater by leaking sewer networks. *Science of The Total Environment*, 430:8–19, 2012. doi: https://doi.org/10.1016/j.scitotenv.2012.04.059.

K. Yoon, R. Liao, Y. Xiong, L. Zhang, E. Fetaya, R. Urtasun, R. Zemel, and X. Pitkow. Inference in probabilistic graphical models by graph neural networks. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 868–875. IEEE, 2019.

X. Zheng, B. Aragam, P. K. Ravikumar, and E. P. Xing. DAGs with NO TEARS: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, volume 31, pages 9472–9483. Curran Associates, Inc., 2018.