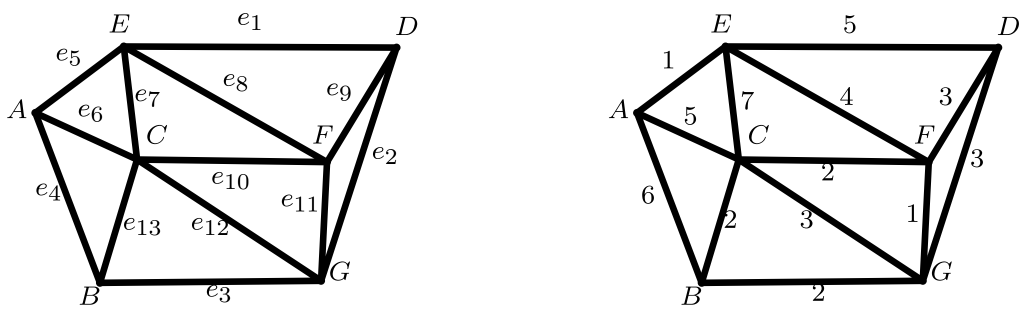
2.2.5 Minimum Spanning Forest

Another option for the class of spanning subgraph, which minimally represents the dependence structure, is the forest class. In contrast to trees, forests are not restricted to the class of connected graphs. Due to this advantage, we will show that extremal forest models are able to represent the complete independence of extreme variables, which is not possible with extremal tree models. Therefore, we would like to find a spanning forest of a weighted graph that has the minimal overall sum of the weights. To find the minimum spanning forests, denoted by $\mathcal{F}_{msf}$, we develop modified versions of Kruskal's algorithm corresponding to each constraint. These algorithms are greedy where at each step of the algorithms, a minimal choice is made from the remaining available data.



(a) Graph $\mathcal{G}$ (b) Weighted graph $\mathcal{G}$

Figure 2.6: An example of a weighted graph.

Let graph $\mathcal{G} = (V, E)$ be a connected and weighted graph where $V = \{1, ..., d\}$, and each edge $e \in E$ is associated with a positive weight w_e. To find the minimum spanning forest denoted by $\mathcal{F}_{msf} = (V, E_{\mathcal{F}})$, we should **minimize** the objective function, i.e., minimize the overall sum of weights on the selected subgraph $\mathcal{F}$ of $\mathcal{G}$ with respect to the **constraint** that $\mathcal{F}$ is an acyclic spanning subgraph– a spanning forest.

Additional constraints are required for a meaningful minimum spanning forest

Since the spanning forest without any edges can be the solution of the described optimization problem, we need other restrictions to have a meaningful definition for the minimum spanning forest. We define a minimum spanning forest $\mathcal{F} = (V, E_{\mathcal{F}})$ of $\mathcal{G} = (V, E)$, where $V = \{1, ..., d\}$, with one of the following additional constraints.

1. **Size constraint:** $\mathcal{F} = (V, E_{\mathcal{F}})$ is a spanning forest of $\mathcal{G} = (V, E)$ with the additional constraint that $|E_{\mathcal{F}}| = q$ where q can be any number in $\{0, 1, ..., d-1\}$.

2. **Weight constraint:** $\mathcal{F} = (V, E_{\mathcal{F}})$ is a spanning forest of $\mathcal{G} = (V, E)$ with the additional constraints that for all $e \in E_{\mathcal{F}}$ and a known $\tau > 0$, the weight w_e must be $w_e < \tau$; and $|E_{\mathcal{F}}| = d - c$, where c is the number of the connected components of $\mathcal{S}$ which is the spanning subgraph of $\mathcal{G}$ where for all $e' \in \mathcal{G}$, such that $w_{e'} < \tau$, $e' \in \mathcal{S}$.

Remark 2.2.9. The weight constraint is equivalent to saying that $\mathcal{F} = (V, E_{\mathcal{F}})$ is a spanning forest of $\mathcal{G} = (V, E)$ with the additional constraints that for all $e \in E_{\mathcal{F}}$ and a known $\tau > 0$, the weight w_e must be $w_e < \tau$; and $|E_{\mathcal{F}}|$ is maximal. These two are equivalent, since, $\mathcal{F}$ is an acyclic spanning subgraph of $\mathcal{S}$, and the most edges that an acyclic spanning subgraph of $\mathcal{S}$, can contain is $d-c$ edges. Containing more edges is in contradiction with either the acyclic property or the number of connected components in $\mathcal{S}$ (since we know that a spanning subgraph of any graph, does not contain less number of connected components than that graph.)

We suggest two modified versions of Kruskal's algorithm in this contribution to finding the minimum spanning forest, where each version corresponds to one of the additional constraints.

2.2.6 Modified Kruskal's algorithms for minimum spanning forest

In this section, we introduce the modified greedy algorithms that efficiently find the minimum spanning forest defined by one of the additional constraints. The algorithms are provided in Algorithm 2 and Algorithm 3. Examples of applying the algorithms on the weighted graph in Figure (2.6b) are illustrated in Figures (2.7) and (2.8).

Algorithm 2 Modified Kruskal's algorithm with respect to the size constraint

```
Input: A connected and non-negatively weighted graph $\mathcal{G} = (V, E)$, where $V = \{1,...,d\}_{d\geq 1}$,
    $E = \{e_1,...,e_n\}$, and $q \in \{0,...,d-1\}$.
Output: The minimum spanning forest of $\mathcal{G}$ that is denoted by $\mathcal{F}_s = (V, E_{\mathcal{F}_s})$ where $|E_{\mathcal{F}_s}| = q$.
1: Sort the edges of $E$ in the ascending order of their weights. We let $L$ denote an ordered
    sequence of the edges, $e^{(1)},...,e^{(n)}$, i.e., $L = e^{(1)},...,e^{(n)}$, where $e^{(i)}$ is the $i^{th}$ lightest edge of
    $E$.
2: Initiate with a graph $\mathcal{F}_s = (V, E_{\mathcal{F}_s})$, where $E_{\mathcal{F}_s} = \emptyset$.
3: If $|E_{\mathcal{F}_s}| = q$, stop the algorithm and return $\mathcal{F}_s = (V, E_{\mathcal{F}_s})$. Else, select the edge $e^{(1)}$ from
    the sequence $L$. If $e^{(1)}$ does not create a cycle with the edges in the $E_{\mathcal{F}_s}$, add it to $E_{\mathcal{F}_s}$.
4: Remove the edge $e^{(1)}$ from the sequence $L$.
5: Repeat lines 3 and 4 until the algorithm stops in line 3.
```

Algorithm 3 Modified Kruskal's algorithm with respect to the weight constraint

```
Input: A connected and non-negatively weighted graph $\mathcal{G} = (V, E)$, where $V = \{1,...,d\}_{d\geq 1}$,
    $E = \{e_1,...,e_n\}$, and a known positive threshold $\tau$.
Output: The minimum spanning forest of $\mathcal{G}$ that is denoted by $\mathcal{F}_w = (V, E_{\mathcal{F}_w})$ where for all
    $e \in E_{\mathcal{F}_w}$ we have $w_e < \tau$, and $|E_{\mathcal{F}_w}|$ is maximal.
1: Sort the edges of $E$ in the ascending order of their weights. We let $L$ denote an ordered
    sequence of the edges, $e^{(1)},...,e^{(n)}$, i.e., $L = e^{(1)},...,e^{(n)}$, where $e^{(i)}$ is the $i^{th}$ lightest edge of
    $E$.
2: Initiate with a graph $\mathcal{F}_w = (V, E_{\mathcal{F}_w})$, where $E_{\mathcal{F}_w} = \emptyset$.
3: If $w_{e^{(1)}} \geq \tau$ or $|E_{\mathcal{F}_w}| = d-1$, stop the algorithm and return $\mathcal{F}_w = (V, E_{\mathcal{F}_w})$. Else, Select the
    edge $e^{(1)}$ from the sequence $L$. If $e^{(1)}$ does not create a cycle with the edges in the $E_{\mathcal{F}_w}$,
    add it to $E_{\mathcal{F}_w}$.
4: Remove the edge $e^{(1)}$ from the sequence $L$.
5: Repeat lines 3 and 4 until the algorithm stops in line 3.
```

The resulting subgraphs of Algorithms 2 and 3 have these properties:

1. It is **acyclic**.
2. The resulting subgraph, $\mathcal{F}$, is a **spanning subgraph** of the initial graph $\mathcal{G}$.
3. The resulting subgraph, $\mathcal{F}$, satisfies one of the two restrictions, the size constraint or the weight constraint.

Properties 1 to 3 show that the resulting subgraph $\mathcal{F}$ of $\mathcal{G}$ is an acyclic spanning subgraph of $\mathcal{G}$, satisfying one of the additional constraints. Consequently, $\mathcal{F}$ is a spanning forest of $\mathcal{G}$ satisfying one of the additional constraints.

Optimality of modified Kruskal's algorithms

In order to guarantee that the modified Kruskal's algorithms provide the minimum spanning forest of the given weighted graph, we must show that the resulting spanning forest is minimal.

Proposition 2.2.10. *Let $\mathcal{G} = (V,E)$ be a connected and non-negatively weighted graph, where $V = \{1,...,d\}$. Any size-restricted spanning forest $\mathcal{F}_s = (V,E_{\mathcal{F}_s})$ for $\mathcal{G}$, i.e., $|E_{\mathcal{F}_s}| = q$ where $q \in \{0,...,d-1\}$, obtained by the size-restricted version of Kruskal's algorithm, (Algorithm 2) is minimal.*

We adjust the proof of Theorem (2.2.1), provided for the minimality of Kruskal's algorithm in Grimaldi, 1993, page 640, to prove the minimality of the resulting spanning forest of Algorithm 2.

Proof. Let $\mathcal{F}_s$ be the spanning forest of $\mathcal{G} = (V,E)$ obtained by the size-restricted Kruskal's algorithm, i.e., Algorithm 2. The $\mathcal{F}_s$ has q edges, and its edges are labeled $e_1,e_2,...,e_q$, where e_i is i^{th} edge which is added to the graph by the algorithm, and the corresponding weights are labeled $w_1,w_2,...,w_q$, respectively. Suppose that $\mathcal{F}'$ is a minimal spanning forest of $\mathcal{G}$. Define $c(\mathcal{F}') = k$ if the k is the smallest positive integer for which both $\mathcal{F}'$ and $\mathcal{F}_s$ contain $e_1,e_2,...,e_{k-1}$ but $e_k \notin \mathcal{F}'$. Let $\mathcal{F}_1$ be a minimal spanning forest for which $c(\mathcal{F}_1) = r$ is **maximal**. If $r = q+1$, then $\mathcal{F}_s = \mathcal{F}_1$, and so the forest $\mathcal{F}_s$ obtained by Kruskal's algorithm is minimal.

Otherwise, if $r \leq q$, adding e_r of $\mathcal{F}_s$ to $\mathcal{F}_1$ produces two cases:

1. It does not create any cycle with edges in $\mathcal{F}_1$, where there exists another edge e_{r_1} of $\mathcal{F}_1$, which is not in $\mathcal{F}_s$.
2. It creates a cycle C with edges in $\mathcal{F}_1$, where there exists another edge e_{r_1} of $\mathcal{F}_1$, which is not in $\mathcal{F}_s$.

Start with forest $\mathcal{F}_1$. Add e_r to $\mathcal{F}_1$ and delete e_{r_1}. In both the above cases, the obtained graph denoted by $\mathcal{F}_2$ is an acyclic graph with d vertices and q edges. Hence, $\mathcal{F}_2$ is a spanning forest of size $|E_{\mathcal{F}_2}| = q$ for which,

$$\sum_{i \in \mathcal{F}_2} w_i = \sum_{j \in \mathcal{F}_1} w_j + w_{e_r} - w_{e_{r_1}}.$$

Following the selection of $e_1,e_2,...,e_{r-1}$, the algorithm selects e_r and adds it to the $e_1,e_2,...,e_{r-1}$. Therefore, e_r should be minimal and does not create any cycle with edges $e_1,e_2,...,e_{r-1}$. On the other hand, e_{r_1} also produces no cycle when it is added to the subgraph $e_1,e_2,...,e_{r-1}$. This is so because $\mathcal{F}_1$ is acyclic and $e_{r_1},e_1,e_2,...,e_{r-1} \in E_{\mathcal{F}_1}$. The minimality of w_{e_r} implies that $w_{e_{r_1}} \geq w_{e_r}$. Therefore, $w_{e_r} - w_{e_{r_1}} \leq 0$ and $\sum_{i \in \mathcal{F}_2} w_i \leq \sum_{j \in \mathcal{F}_1} w_j$. Since, we know that $\mathcal{F}_1$ is minimal, an immediate consequence is that, $\sum_{i \in \mathcal{F}_2} w_i$ must be equal to $\sum_{j \in \mathcal{F}_1} w_j$. Therefore, $\mathcal{F}_2$ is also minimal, and it has edges $e_1,...,e_{r-1},e_r$ in common with $\mathcal{F}_s$. So then, $c(\mathcal{F}_2) = r+1 > r = c(\mathcal{F}_1)$. This is in contradiction with the selection of $\mathcal{F}_1$ (since $\mathcal{F}_1$ has the highest number of common edges with $\mathcal{F}_s$ among all minimal forests). Therefore, $r = d$ and $\mathcal{F}_1 = \mathcal{F}_s$ so that the forest $\mathcal{F}_s$ obtained by the size-restricted Kruskal's algorithm is minimal. □

Lemma 2.2.11. *Let $\mathcal{G} = (V,E)$ be a connected and non-negatively weighted graph, where $V = \{1,...,d\}$ and for a positive τ, $\mathcal{S}_\tau$ is a spanning subgraph of $\mathcal{G}$ which contain the entire edges in $\mathcal{G}$ that are strictly lighter than τ and the number of connected components $|\mathfrak{C}(\mathcal{S}_\tau)| = c$. The resulting spanning forest of Algorithm 2 using $q = d-c$, and the resulting spanning forest of Algorithm 3 using the τ as the threshold, are the same.*

Proof. Let us denote the resulting spanning forest of Algorithm 2, using $q = d-c$, and the resulting spanning forest of Algorithm 3, using τ as the threshold, by $\mathcal{F}_s$ and $\mathcal{F}_w$, respectively. The edges in $\mathcal{F}_s$ are labeled $e_{s,1},...,e_{s,q}$ and the edges in $\mathcal{F}_w$ are labeled $e_{w,1},...,e_{w,d-c}$. Algorithm

3 creates an acyclic subgraph of $\mathcal{S}$ with the maximum number of edges, for which all edges are lighter than τ. An acyclic subgraph of $\mathcal{S}$ with the maximum number of edges has $d-c$ edges, i.e., $|E_{\mathcal{F}_w}| = d-c$, because a $|E_{\mathcal{F}_w}| > d-c$ violates either acyclic property or weight constraint, and a $|E_{\mathcal{F}_w}| < d-c$ is against having the possible maximum number of edges. Hence, the stopping criteria of Algorithm 3 using τ implies the stopping criteria of Algorithm 2 using $q = d-c$. Moreover, Algorithm 2 creates an acyclic subgraph of $\mathcal{G}$ with $q = d-c$ edges as follows. Suppose that $|E_{\mathcal{F}_s} \cap \mathcal{S}| = r$. If $r = d-c$, then every edge in $|E_{\mathcal{F}_s}|$ must be lighter than τ. Otherwise, if $r < d-c$, following the selection of light edges $e_{s,1},...,e_{s,r}$ from $\mathcal{S}$, Algorithm 2 selects an edge $e_{s,r+1} \in E \setminus E_{\mathcal{S}}$. This is against the criteria of the minimal selection of edges since there exists $d-c-r > 0$ number of edges that could be selected by the algorithm, whereas the algorithm selected heavier edges. Therefore, $r = d-c$, and every edge in $|E_{\mathcal{F}_s}|$ is lighter than τ. Hence, the stopping criteria of Algorithm 2 imply the stopping criteria of Algorithm 3. Consequently, both stopping criteria imply each other. Moreover, except for the stopping conditions, both algorithms are the same. Therefore, both algorithms return the same resulting spanning forests. □

Lemma 2.2.12. *Let $\mathcal{G} = (V, E)$ be a connected and non-negatively weighted graph, where $V = \{1,...,d\}$ and for a positive τ, $\mathcal{S}_\tau$ is a spanning subgraph of $\mathcal{G}$ which contain the entire edges in $\mathcal{G}$ that are strictly lighter than τ and the number of connected components $|\mathfrak{C}(\mathcal{S}_\tau)| = c$. Every minimal spanning forest under the size constraint $q = d-c$ is a minimal spanning forest restricted to edges lighter than τ and vice versa.*

Proof. ($\Rightarrow$) Let $\mathcal{F}_s$ be a minimal spanning forest restricted to have $q = d-c$ edges and its edges are denoted by $e_{\mathcal{F},1},...,e_{\mathcal{F},d-c}$. By contradiction, suppose that $\mathcal{F}_s$ has $r \geq 1$ edges which are heavier than τ. Therefore, $\mathcal{F}_s$ has $d-c-r$ edges in $\mathcal{S}$, denoted by $e_{\mathcal{F},1},...,e_{\mathcal{F},d-c-r}$. Because $\mathcal{F}_s$ is acyclic, $e_{\mathcal{F},1},...,e_{\mathcal{F},d-c-r}$ edges do not create a cycle. Due to the fact that $\mathcal{S}$ has c connected components, there are exactly r edges in $\mathcal{S}$ denoted by $e_{s,1},...,e_{s,r}$ that do not create a cycle with $e_{\mathcal{F},1},...,e_{\mathcal{F},d-c-r}$. If we replace any edges in $e_{\mathcal{F},d-c-r+1},...,e_{\mathcal{F},d-c}$ by $e_{s,1},...,e_{s,r}$, the resulting graph has lower overall weight since every $e_{s,1},...,e_{s,r}$ is lighter than τ, so it is lighter than $e_{\mathcal{F},d-c-r+1},...,e_{\mathcal{F},d-c}$. The created graph is acyclic; it has $q = d-c$ edges, and its overall weight is lower than the overall weight of $\mathcal{F}_s$. This is in contradiction with the assumption that $\mathcal{F}_s$ is a minimal spanning forest under the size constraint. Therefore, $\mathcal{F}_s$ is a minimal spanning forest under the weight constraint that $w_{e_{\mathcal{F},1}},...,w_{e_{\mathcal{F},d-c}}$ are lighter than τ.

($\Leftarrow$) Let $\mathcal{F}_w$ be a minimal spanning forest that for all $e \in E_{\mathcal{F}_w}$, and a known $\tau > 0$, the weight w_e must be $w_e < \tau$; additionally $|E_{\mathcal{F}}| = d-c$, where c is the number of the connected components of $\mathcal{S}$ (the subset of $\mathcal{G}$ such that all $e' \in \mathcal{S}$ that are $w_{e'} < \tau$. The immediate consequence is that $\mathcal{F}_w$ satisfies the size constraint $|E_{\mathcal{F}_w}| = d-c$. Therefore, $\mathcal{F}_w$ is a minimal spanning forest under the size constraint that $|E_{\mathcal{F}_w}| = d-c$. □

Proposition 2.2.13. *Let $\mathcal{G} = (V, E)$ be a connected and non-negatively weighted graph, where $V = \{1,...,d\}$. Any weight-restricted spanning forest $\mathcal{F}_w = (V, E_{\mathcal{F}_w})$ for $\mathcal{G}$, i.e., $w_e < \tau$ for all $e \in E_{\mathcal{F}_w}$ and the positive threshold τ, obtained by the weight-restricted version of Kruskal's algorithm, Algorithm 3, is minimal.*

Proof. The statement is an immediate consequence of Lemma (2.2.11), Proposition (2.2.10), and Lemma (2.2.12). This is because, by Lemma (2.2.11) the resulting forest of two algorithms, $\mathcal{F}_w$ and $\mathcal{F}_s$ are identical. By Proposition (2.2.10), $\mathcal{F}_s$ with $q = d-c$ is minimal. Eventually, by Lemma (2.2.12) any minimum spanning forest made by constraint $q = d-c$ is a minimum spanning forest created by the edges that are strictly lighter than τ. Therefore, $\mathcal{F}_w$ is a minimum spanning forest. □

Uniqueness of the modified Kruskal's algorithms

We will exploit the modified Kruskal's algorithms to learn the underlying forest, which represents the dependence structure between the components of multivariate extreme variables. We require a unique dependence structure. Therefore, the solutions by each of these algorithms should be unique. We must then provide the necessary and sufficient conditions required for the uniqueness of the solution of the modified Kruskal's algorithms. Due to the equivalence of the resulting forest of two algorithms, it is enough to provide conditions under which one of the algorithms provided by Algorithm 2 and Algorithm 3 is unique. In the following, we introduce a sufficient condition that guarantees the uniqueness of the solution of Algorithm 3.

Proposition 2.2.14. *Let $\mathcal{G} = (V, E)$ be a connected and non-negatively weighted graph. Then the sufficient condition for the uniqueness of the solution of Algorithm 3 is that $\mathcal{G}$ only includes edges with distinct weights.*

Proof. Kruskal, 1956 shows that the sufficient condition for the uniqueness of Kruskal's algorithm is to have uniquely weighted edges. Following this condition, Kruskal's algorithm uniquely selects the lightest edge in each step. The only difference between Kruskal's algorithm in Algorithm 1 and in Algorithm 3, is their stopping criteria. Therefore, adhering to the condition results in the weight-restricted Kruskal's algorithm uniquely selecting the lightest edge in each step. Consequently, the solution of Algorithm 3 for $\mathcal{G}$ under the condition that the edges in $\mathcal{G}$ are uniquely weighted edges, is unique. $\square$

A sufficient and necessary condition is stated in the following [see Conjecture (2.2.15)]. However, providing the proof for this argument is out of the scope of this thesis.

Conjecture 2.2.15. *Let $\mathcal{G} = (V, E)$ be a connected and non-negatively weighted graph. These properties are equivalent:*

- ***Uniqueness of the Minimum Spanning Forest defined by the weight constraint:*** *There is a unique minimum spanning forest $\mathcal{F} = (V, E_{\mathcal{F}})$ of $\mathcal{G} = (V, E)$ under the additionally constraints that for all $e \in E_{\mathcal{F}}$ and a known $\tau > 0$, the weight w_e must be $w_e < \tau$; and $|E_{\mathcal{F}}| = d - c$, where c is the number of the connected components of $\mathcal{S}$ (the subset of $\mathcal{G}$ such that for all $e' \in \mathcal{S}$, $w_{e'} < \tau$.)*

- ***The weight-restricted well-behaved cycle edge,*** *Every edge in $\mathcal{G}$ is either "unique-cycle-heaviest or is not lighter than τ" or "non-cycle-heaviest and lighter than τ".*

Intuitively, any edge that is "unique-cycle-heaviest or is not lighter than τ" is not in the weight constrained minimum spanning forest. Otherwise, the presence of the edge in $\mathcal{F}$ violates either the minimality or the weight constraint. And any edge that is "non-cycle-heaviest and lighter than τ" is in the weight constrained minimum spanning forest. Otherwise, if the edge is not in a cycle, the absence of the edge in $\mathcal{F}$ violates $|E_{\mathcal{F}}| = d - c$, and if the edge is in a cycle, the absence of the edge in $\mathcal{F}$ violates the minimality of $|E_{\mathcal{F}}|$ since it can be replaced by the heaviest edge in that cycle and results in a lighter forest.