The Complexity of k-Means Clustering when Little is Known

Robert Ganian [*1] **Thekla Hamm** [*1] **Viktoriia Korchemna** [*1] **Karolina Okrasa** [*2] **Kirill Simonov** [*1]

Abstract

In the area of data analysis and arguably even in machine learning as a whole, few approaches have been as impactful as the classical k-means clustering. Here, we study the complexity of k-means clustering in settings where most of the data is not known or simply irrelevant. To obtain a more fine-grained understanding of the tractability of this clustering problem, we apply the parameterized complexity paradigm and obtain three new algorithms for k-means clustering of incomplete data: one for the clustering of bounded-domain (i.e., integer) data, and two incomparable algorithms that target real-valued data. Our approach is based on exploiting structural properties of a graphical encoding of the missing entries, and we show that tractability can be achieved using significantly less restrictive parameterizations than in the complementary case of few missing entries.

1. Introduction

k-means clustering is a fundamental task in machine learning and data analysis, one that has been the focus of extensive empirical as well as theoretical research. In general, the aim in k-means clustering is to partition the rows in an input matrix A into k clusters and compute one center per cluster so as to minimize the within-cluster sum of squares—or, when viewed as a decision problem, achieve a within-cluster sum of squares of at most a specified target value ℓ. Depending on the setting, A could be real-valued (Lloyd, 1982; Moshkovitz et al., 2020; Fomin et al., 2021) or contain integers from a bounded domain; for instance, Fomin et al. (Fomin et al., 2018) studied the case where A was binary, and Ganian et al. (Ganian et al., 2018) the bounded-domain case where $\ell = 0$.

It is well known that k-means clustering is NP-complete, even when restricted to $k = 2$ clusters (Drineas et al., 2004; Aloise et al., 2009). On the theoretical side, a prominent approach used to circumvent such lower bounds is the use of the *parameterized complexity*[2] paradigm, which provides a more fine-grained look into the complexity of problems. Since its inception, parameterized complexity has been successfully applied to a multitude of problems relevant to machine learning and artificial intelligence such as, e.g., Bayesian network learning (Ordyniak & Szeider, 2013; Grüttemeier et al., 2021; Ganian & Korchemna, 2021), integer linear programming (Ganian & Ordyniak, 2018; 2019; Dvorák et al., 2021), principal component analysis (Simonov et al., 2019; Dahiya et al., 2021), and of course clustering (Cohen-Addad et al., 2018; Fomin et al., 2018).

Apart from classical clustering problems, one direction that has been gaining traction in recent years is the study of clustering in settings where some of the data is not known or simply irrelevant—a task that essentially combines data completion with clustering (Wang et al., 2019). Indeed, several authors have studied the parameterized complexity of various clustering problems for data with missing entries (Ganian et al., 2018; Koana et al., 2020; 2021; Eiben et al., 2021b), and Eiben et al. (Eiben et al., 2021a) obtained a fixed-parameter approximation algorithm specifically for (k-)MEANS CLUSTERING WITH MISSING ENTRIES (MCME).

While their approaches and techniques differ, all of these works target the case where nearly all entries are known, i.e., where the unknown entries only occur in a precisely defined "sparse" way. On the other hand, very little is known about the complexity of MCME when the number of unknown entries is large; such situations are of practical relevance in, e.g., recommender systems and predictive analytics. For example, in the classical Netflix Prize challenge where the task was to predict user ratings for movies based on previous ratings, only about 1% of the user-movie pairs were originally supplied with a rating.[3] The central mission of this

[*]Equal contribution [1]Algorithms and Complexity Group, TU Wien, Austria [2]Institute of Informatics, University of Warsaw, Poland. Correspondence to: Robert Ganian <rganian@gmail.com>, Thekla Hamm <thamm@ac.tuwien.ac.at>, Viktoriia Korchemna <vkorchemna@ac.tuwien.ac.at>, Kirill Simonov <ksimonov@ac.tuwien.ac.at>, Karolina Okrasa <k.okrasa@mini.pw.edu.pl>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[2]See Preliminaries for a brief introduction.

[3]The dataset is available at https://www.kaggle.com/netflix-inc/netflix-prize-data.

article is to push the boundaries of tractability for MCME to instances where *known* entries are sparse.

Contribution. A natural approach for handling instances of MCME with many missing entries would be to invert the parameterizations that have been developed for tackling instances where almost all entries are known. For instance, Eiben et al. (Eiben et al., 2021b) and Ganian et al. (Ganian et al., 2018) obtained several fixed-parameter clustering algorithms by using a parameter called the *covering number*, which is the minimum number of rows and columns needed to cover all the unknown entries. Equivalently, if we use an auxiliary binary matrix W (the *mask*) to specify which entries of A are relevant/known, the covering number is simply the minimum number of rows and columns needed to cover all the 0's in W. It would be tempting to instead parameterize by the number of rows and columns needed to cover all the 1's in W; such an "inverse covering number" would lead to fixed-parameter algorithms targeting instances where all known entries occur in a few columns and rows.

However, we show that in our setting it is in fact possible to do much better than that. Our approach is based on using the *incidence graph* representation of W, which is the graph containing one vertex for each column and for each row of W and where an edge connects row a with column b if and only if $W[a, b] = 1$. Observe that the inverse covering number considered in the previous paragraph would simply be the size of a minimum vertex cover in W, while the covering number is the size of a minimum vertex cover in the complement of W. It is worth noting that in graph-theoretic settings, the size of a minimum vertex cover is considered a highly restrictive parameterization, one that is used primarily when more desirable parameterizations fail. On the other hand, the by far most prominent parameter used for graphs is *treewidth* (Robertson & Seymour, 1984), which has found ubiquitous applications throughout computer science and which is smaller (i.e., less restrictive as a parameter) than the size of a minimum vertex cover. As our first result, we use the treewidth of the incidence graph of the mask W (the *incidence treewidth*) to prove:

Result 1: Bounded-domain MCME is fixed-parameter tractable when parameterized by the incidence treewidth of the mask.

Result 1 is noteworthy not only due to the use of treewidth instead of the inverse covering number, but also because incidence treewidth is the *only* parameter required to achieve tractability. In particular, unlike previous algorithms for clustering incomplete (bounded-domain or real-valued) data by Eiben et al. (Eiben et al., 2021a;b), Ganian et al. (Ganian et al., 2018), and Koana et al. (Koana et al., 2020; 2021), Result 1 can also be applied to instances where the number k of clusters is large. As we will see later, we will be able to retain this benefit in all our algorithms.

Unfortunately, an extension of Result 1 towards real-valued instances seems difficult at this point. Indeed, a fixed-parameter algorithm for real-valued MCME parameterized by the incidence treewidth of the mask would imply, as a special case, fixed-parameter tractability parameterized by the number d of columns. However, the existence of a fixed-parameter algorithm even for k-MEANS (corresponding to the restriction of MCME where all entries are known) when parameterized by d is a long-standing open problem dating back to Inaba et al.'s celebrated XP algorithm parameterized by $k + d$ (Inaba et al., 1994).

Instead, we show that one can obtain fixed-parameter algorithms for real-valued MCME by using the treewidth of a different graph representation of W. In particular, we use the *primal graph*—a well-established counterpart to the incidence graph representation in settings such as Boolean satisfiability (Samer & Szeider, 2009), constraint satisfaction (Samer & Szeider, 2010) or integer programming (Ganian & Ordyniak, 2019). In our context, the primal graph of W contains a vertex for each row of W, with edges connecting pairs of rows that share at least one coordinate which is known/relevant according to the mask, and the primal treewidth is simply the treewidth of this graph.

Result 2: Real-valued MCME is fixed-parameter tractable when parameterized by the primal treewidth of the mask.

Note that, due to the nature of the primal graph representation, the primal treewidth is a more restrictive parameter than the incidence treewidth (in the sense of being bounded on fewer classes of instances). In fact, the primal treewidth of a mask may in certain cases be unboundedly large even when its incidence graph is just a tree. It would hence be useful to also have an algorithm for real-valued MCME that can exploit properties of the incidence graph, even for properties that are more restrictive than incidence treewidth.

However, the same obstacle towards extending Result 1 to the real-valued setting also applies to all traditional structural parameters that are, intuitively, based on small vertex separators in the graph—including *pathwidth* (Robertson & Seymour, 1983), *treedepth* (Nesetril & de Mendez, 2012), the *feedback vertex number* e.g., (Mertzios et al., 2020) and even the *vertex cover number* e.g., (Gaspers & Najeebullah, 2019; Fellows et al., 2018), which is the most restrictive vertex separator based parameter. In spite of this, as our third and final result we identify a parameter of the incidence graph that yields fixed-parameter tractability of real-valued MCME: the recently introduced *local feedback edge number* (lfen) (Ganian & Korchemna, 2021). On a high level, lfen can be seen as an edge-based restriction of treewidth, and as a less restrictive parameter than the edge deletion distance to acyclicity (i.e., the *feedback edge number*).

Result 3: Real-valued MCME is fixed-parameter tractable when parameterized by the local feedback edge number of the incidence graph of the mask.

We note that the parameters used in Results 2 and 3 are orthogonal and the gap between them may be arbitrarily large. Because of this, both results are incomparable and give rise to different tractable classes for the problem.

Statements where proofs or details are omitted due to space constraints are marked with ⋆. A full version of the paper containing all proofs and details is provided in the appendix.

2. Preliminaries

For an integer i, we let $[i] = \{1, 2, \ldots, i\}$ and $[i]_0 = [i] \cup \{0\}$. On the other hand, if p is an equivalence relation on domain X, we denote by $[p]$ the set of equivalence classes of p and by $|[p]|$ the cardinality of $[p]$, i.e., the number of equivalence classes of p. For every $x \in X$, $[x]_p \in [p]$ is the equivalence class containing x.

Notation and Problem Formulation. Let A be a matrix with n rows and d columns. We use V_A and C_A to denote the set of row and column indices of A, i.e., $V_A = [n]$ and $C_A = [d]$. Let $M_A = \max_{v \in V_A} \max_{c \in C_A} |A[v, c]|$. For two matrices A, B, we denote by $A - B$ the entry-wise subtraction of the two matrices, i.e., $(A - B)[i, j] = A[i, j] - B[i, j]$, and similarly by $A \circ B$ the entry-wise product of the two matrices, i.e., $(A \circ B)[i, j] = A[i, j] \cdot B[i, j]$. For a matrix $A \in \mathbb{R}^{n \times d}$ we denote its squared Frobenius norm by $||A||_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{d} A[i, j]^2$. We can now formalize our problems of interest (abbreviated as MCME):

MEANS CLUSTERING WITH MISSING ENTRIES

Input: Matrix $A \in D^{n \times d}$ over a domain $D \subseteq \mathbb{R}$, binary matrix W (the *mask*), integers k and ℓ.

Task: Determine if there exists a matrix B over D containing at most k distinct rows such that $\|W \circ (A - B)\|_F^2 \leq \ell$.

We distinguish between BOUNDED-DOMAIN MCME and REAL-VALUED MCME depending on whether $D = [z]_0$ for some fixed integer z, or D is the set of reals. In particular, for $z = 1$ this coincides with BINARY k-MEANS CLUSTERING (Fomin et al., 2018). In this case, minimizing the Frobenius norm also minimizes the sum of Hamming distances for each row to its cluster center. Using the Frobenius norm also for $z > 1$ is not only consistent with the continuous case, but also allows us to capture further applications, e.g., when the data matrix to cluster is composed of user ratings each taking a small numeric value. While we state MCME as a decision problem for complexity-theoretic reasons, every algorithm presented here is constructive and can output a solution matrix B if one exists.

In this formulation, the distinct rows of B are the centers of the sought-after clusters while ℓ is the target upper-bound on the sum of squares. For $V' \subseteq V_A$, we call a mapping $\varphi : V' \to [k]$ a *cluster-assignment* w.r.t. B if for every $v_1, v_2 \in V'$ such that $\varphi(v_1) = \varphi(v_2)$, it holds that $B[v_1] = B[v_2]$. Notice that the existence of a cluster assignment w.r.t. a matrix B' implies that B' has at most k distinct rows. The *clusters* associated with φ are then sets of the form $\{v \in V \mid \varphi(v) = i\}$ for some $i \in [k]$.

In our algorithms, we often construct the solution matrix dynamically by "merging" smaller matrices; we formalize this procedure below. Let $\{B_i \mid i \in [m]\}$ be a set of matrices, where each B_i has row labels V_i, column labels C_i and is associated with a cluster-assignment $\varphi_i : V_i \to [k]$ and the following two *consistency conditions* hold:

- $\varphi_i(v) = \varphi_j(v)$ for every $v \in V_i \cap V_j$, $i, j \in [m]$,
- if $\varphi_i(v_i) = \varphi_j(v_j)$ for some $v_i \in V_i$ and $v_j \in V_j$, then $B_i[v_i, c] = B_j[v_j, c]$ for every $c \in C_i \cap C_j$.

Let $\varphi = \bigcup_{i \in [m]} \varphi_i$. We define the *composition* of (B_i, φ_i), $i \in [m]$, to be the matrix B^* with row labels $V = \bigcup_{i \in [m]} V_i$, column labels $C = \bigcup_{i \in [m]} C_i$ and entries as follows. For every $v \in V$ and $c \in C$, pick any $i \in [m]$ such that $c \in C_i$ and there exists $v_i \in V_i$ with $\varphi(v_i) = \varphi(v)$. We set $B^*[v, c_i] = B_i[v_i, c_i]$. If there is no such $i \in [m]$, we simply set $B^*[v, c]$ to some arbitrary but uniform default value—in our case, we will always use 0. Observe that if both consistency conditions hold, then both B^* and φ are well-defined and that φ is a cluster-assignment w.r.t. B^*.

Parameterized Complexity. In parameterized algorithmics (Downey & Fellows, 2013; Cygan et al., 2015), the running-time of an algorithm is studied with respect to a parameter $k \in \mathbb{N}_0$ and input size n. The basic idea is to find a parameter that describes the structure of the instance such that the combinatorial explosion can be confined to this parameter. In this respect, the most favorable complexity class is FPT (*fixed-parameter tractable*), which contains all problems that can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$, where f is a computable function. Algorithms with this running-time are called *fixed-parameter algorithms*.

Sparse Matrices. It follows from definition of MCME that entries of the input matrix A and the target matrix B are only relevant when the respective entry of the mask W is set to "1". We thus assume that only those entries are stored explicitly; in particular, the size of the input is therefore linear in the number of non-zero entries of W. All the structural parameters in our considerations automatically imply that the mask W is sparse, in the sense that it has at most $\mathcal{O}_k(n + d)$ non-zero entries, where the constant under $\mathcal{O}_k(\cdot)$ depends on the respective parameter k. All algorithms in this paper except one are linear-time in the number of

non-zero entries of the mask, and the above allows us to state their running times in the form $(n+d)\cdot f(k)$, where f is a certain function of the respective parameter k.

Treewidth. A *nice tree-decomposition* $\mathcal{T}$ of a graph $G=(V,E)$ is a pair (T,χ), where T is a tree (whose vertices we call *nodes*) rooted at a node r and χ is a function that assigns each node t a set $\chi(t)\subseteq V$ such that:

- For every $uv\in E$ there is a node t where $u,v\in\chi(t)$.
- For every vertex $v\in V$, the set of nodes t satisfying $v\in\chi(t)$ forms a subtree of T.
- $|\chi(\ell)|=1$ for every leaf ℓ of T and $|\chi(r)|=0$.
- There are only three kinds of non-leaf nodes in T:
  - **Introduce node:** a node t with exactly one child t' such that $\chi(t)=\chi(t')\cup\{v\}$ for some vertex $v\notin\chi(t')$.
  - **Forget node:** a node t with exactly one child t' such that $\chi(t)=\chi(t')\setminus\{v\}$ for some vertex $v\in\chi(t')$.
  - **Join node:** a node t with two children t_1,t_2 such that $\chi(t)=\chi(t_1)=\chi(t_2)$.

The *width* of a nice tree-decomposition (T,χ) is $\max_{t\in V(T)}(|\chi(t)|-1)$, and the *treewidth* of the graph G, denoted $\mathrm{tw}(G)$, is the minimum width of a nice tree-decomposition of G. We let T_t denote the subtree of T rooted at a node t, and we use $\chi_t^{\downarrow}$ to denote the set $\bigcup_{t'\in V(T_t)}\chi(t')$. The sets $\chi(t)$ are commonly called *bags*.

Fixed-parameter algorithms are known for computing nice tree-decompositions of optimal width and linearly many nodes (Bodlaender, 1996; Kloks, 1994), albeit more efficient fixed-parameter approximation algorithms are often used to achieve better running times (Bodlaender et al., 2016).

Local Feedback Edge Number. The *local feedback edge number* was recently introduced by Ganian and Korchemna (2021) as an edge-cut based restriction of treewidth that acts as a "local" measure of the size of a feedback edge set; they used the parameter to obtain fixed-parameter algorithms for learning Bayesian networks and polytrees.

For a connected graph $G=(V,E)$ and a spanning tree T of G, the *local feedback edge set* at $x\in V$ is defined as $E^T_{\mathrm{loc}}(x)=\{yz\in E\setminus E(T)\mid$ the unique path between y and z in T contains $x\}$. The *local feedback edge number of* (G,T) (denoted $\mathrm{lfen}(G,T)$) is then equal to $\max_{x\in V}|E^T_{\mathrm{loc}}(x)|$, and the *local feedback edge number* of G is simply the smallest local feedback edge number among all possible spanning trees of G, i.e., $\mathrm{lfen}(G)=\min_{T\text{ is a spanning tree of }G}\mathrm{lfen}(G,T)$. An example of a graph G with $\mathrm{lfen}(G)=2$ is provided in Figure 1.

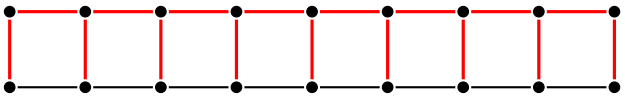

Figure 1. Example of a graph G with the spanning tree T (marked in thick red) such that $\mathrm{lfen}(G)=\mathrm{lfen}(G,T)=2$. The feedback edge number of G, i.e., its edge deletion distance to acyclicity, is exactly the number of black edges and can be made arbitrarily large in this fashion while preserving $\mathrm{lfen}(G)=2$.

3. Solving Bounded-Domain MCME by Exploiting the Mask

Since BOUNDED-DOMAIN MCME is NP-complete (Drineas et al., 2004; Aloise et al., 2009), it is natural to attempt and circumvent this lower bound by exploiting structural measures of the inputs to obtain fixed-parameter tractability. Eiben et al. (Eiben et al., 2021a) recently obtained a fixed-parameter approximation algorithm for BOUNDED-DOMAIN requiring the simultaneous parameterization by three measures: the number of clusters, the desired approximation ratio, and a technical measure that captures the sparsity of the missing data (i.e., the "0" entries in W).

In this section, we will present a fixed-parameter algorithm for BOUNDED-DOMAIN MCME which is aimed at the complementary case where the relevant data is sparse. To formalize this, we consider the *incidence graph* representation G_I of W which is defined as follows: $V(G_I)=V_W\cup C_W$, and $E(G_I)=\{ab\mid a\in V_W, b\in C_W, W[a,b]=1\}$. Figure 2 later on can also be used as an example of the representation.

Intuitively, each "1" entry in W will correspond to an edge in G_I, and hence "structurally sparse" incidence graphs correspond to settings where most data is unknown. There is a well-studied hierarchy of structural graph parameters which measure, in a certain sense, the sparsity of graphs (see, e.g., Figure 1 in (Bodlaender et al., 2013)). Treewidth will be our parameter of choice here, as the best known and also most general parameter in this hierarchy. However, it is worth noting that the use of the incidence graph may provide a new perspective on previous algorithms for clustering problems—for instance, the so-called *covering number* parameter (Ganian et al., 2018; Eiben et al., 2021b) is simply the vertex cover number of the complement of G_I.

The main goal of this section is to establish the fixed-parameter tractability of MCME parameterized by $\mathrm{tw}(G_I)$. As a first step towards this goal, we obtain a dynamic programming algorithm that handles the simpler case where k is also part of the parameterization.

Theorem 1. BOUNDED-DOMAIN MCME *is fixed-parameter tractable when parameterized by* $k+\mathrm{tw}(G_I)$.

Proof. We begin by applying the well-established 5-approximation algorithm for treewidth (Bodlaender et al., 2016) to compute a nice tree-decomposition of G_I of width $q \leq 5\,\mathrm{tw}(G_I)$ in time $2^{\mathcal{O}(\mathrm{tw}(G_I))}(n+d)$. Let r be the root of T. Given a node t of T, let V_t and C_t be the sets of vectors and coordinates in $\chi(t)$, respectively. Moreover, let $C_t^{\downarrow}$ and $V_t^{\downarrow}$ be the restrictions of $\chi_t^{\downarrow}$ to sets of coordinates and vectors correspondingly.

To prove the theorem, we will design a leaf-to-root dynamic programming algorithm which will compute and store a set of records at each node t of T, whereas once we ascertain the records for r we will have the information required to output a correct answer. Intuitively, the records will store the cluster centers restricted to the coordinates that appear in the bag, the partition of the vectors in the bag into clusters and the sum of minimum distances from vectors in $V_t^{\downarrow}$ to the cluster centers along the coordinates in $C_t^{\downarrow}$.

Formally, the records will have the following structure. We call a pair $(\mathrm{cent}, \mathrm{part})$ a *snapshot* in t if the following holds:

- $\mathrm{cent} : [k] \times C_t \to D$,
- $\mathrm{part} : V_t \to [k]$.

Let $S(t)$ be the set of all snapshots of t. The record $\mathcal{R}_t$ of t is then a mapping from $S(t)$ to the set $\mathbb{R}^+$ of non-negative reals. Observe that $|S(t)| \leq |D|^{k(q+1)}k^{q+1}$.

To introduce the semantics of our records, let $\mathcal{B}_t$ be the set of all matrices with row labels $V_t^{\downarrow}$, column labels $C_t^{\downarrow}$ and entries of domain D. Let B_t be a matrix in $\mathcal{B}_t$. We define the *partial weighted distance* from B_t to A in t as follows:

$$\mathrm{pwd}(B_t, t) = \sum_{v \in V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[v,c] \cdot (A[v,c] - B_t[v,c])^2.$$

For $B_t \in \mathcal{B}_t$, we say that $(\mathrm{cent}, \mathrm{part}) \in S(t)$ is the *snapshot of* B_t in t if there is a cluster-assignment φ w.r.t. B_t such that the following conditions hold:

- $\mathrm{part} = \varphi|_{V_t}$,
- for every $c \in C_t$ and $v \in V_t^{\downarrow}$, $B_t[v,c] = \mathrm{cent}[\varphi(v), c]$.

Recall that the existence of a cluster-assignment implies, in particular, that B_t has at most k distinct rows. We are now ready to define the record $\mathcal{R}_t$. For each snapshot $(\mathrm{cent}, \mathrm{part}) \in S(t)$, we set $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$ if there exists $B_t \in \mathcal{B}_t$ such that:

- $(\mathrm{cent}, \mathrm{part})$ is the snapshot of B_t in t,
- $\mathrm{pwd}(B_t, t) = \tau$, and
- $\forall B_t' \in \mathcal{B}_t$ such that $(\mathrm{cent}, \mathrm{part})$ is the snapshot of B_t', $\mathrm{pwd}(B_t', t) \geq \tau$.

In this case we say that B_t *witnesses* $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$.

Recall that for the root r, we assume $\chi(r) = \emptyset$. Hence $S(r)$ contains only one element $(\emptyset, \emptyset)$, and $\mathcal{R}_r(\emptyset, \emptyset)$ is equal to the minimum value of $\|W \circ (A - B)\|_F^2$ that can be achieved by any matrix B with entries of domain D containing at most k distinct rows. In other words, the instance is a YES-instance if and only if $\mathcal{R}_r(\emptyset, \emptyset) \leq \ell$. To prove the theorem, it now suffices to show that the records can be computed in a leaf-to-root fashion by proceeding along the nodes of T.

We distinguish the following cases:

t is a leaf node. Let $\chi(t) = \{v\}$ where v is a vector. By definition, $S(t) = \{(\emptyset, \mathrm{part}) | \, \mathrm{part} : [1] \to [k]\}$ and $\mathcal{R}_t(\emptyset, \mathrm{part}) = 0$ for every $(\emptyset, \mathrm{part}) \in S(t)$ as there are no coordinates to sum over. Let $\chi(t) = \{c\}$ for some coordinate c. Then $S(t) = \{(\mathrm{cent}, \emptyset) | \, \mathrm{cent} : [k] \times [1] \to D\}$ and $\mathcal{R}_t(\mathrm{cent}, \emptyset) = 0$ for each $(\mathrm{cent}, \emptyset) \in S(t)$.

t is a forget node. Let t' be the child of t in T and $\chi(t) = \chi(t') \setminus \{v\}$ for some vector v. We set $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) := \min_{i \in [k]} \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i))$ for each $(\mathrm{cent}, \mathrm{part}) \in S(t)$. For correctness, it will be useful to observe that $\mathcal{B}_t = \mathcal{B}_{t'}$. If $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$, then there exists a matrix B_t which witnesses this. But then B_t also admits a snapshot $(\mathrm{cent}, \mathrm{part} \cup (v,i))$ at t' for some $i \in [k]$ and witnesses $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i)) \leq \tau$. So in our algorithm $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) \leq \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i)) \leq \tau$. If on the other hand $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \tau$, then there exists a snapshot $(\mathrm{cent}, \mathrm{part} \cup (v,i))$ at t' for some $i \in [k]$ such that $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i)) = \tau$. $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$ now follows from the existence of a matrix witnessing the value of $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i))$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}_t^0$.
If $\chi(t) = \chi(t') \setminus \{c\}$ for some coordinate c, we set $\mathcal{R}_t(\mathrm{cent}, \mathrm{part})$ equal to $\min\{\mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) | \, \mathrm{cent} \text{ is a restriction of } \mathrm{cent}' \text{ to all coordinates except } c\}$. Correctness can be argued similarly to the case of a forgotten vector.

t is an introduce node (introducing a vector). Let t' be the child of t in T and let $\chi(t) = \chi(t') \cup \{v_0\}$ for some vector v_0. Fix a snapshot $(\mathrm{cent}, \mathrm{part})$ in $S(t)$, let $i = \mathrm{part}(v_0)$ and $\mathrm{part}' = \mathrm{part} \setminus (v_0, i)$. We will denote by Δ_0 the sum of distances from v_0 to the i-th cluster center along the coordinates in C_t, i.e., $\Delta_0 = \sum_{c \in C_t} W[v_0, c] \cdot (A[v_0, c] - \mathrm{cent}[i, c])^2$. We set $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0$. For correctness, assume that $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \tau = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0$. Construct a matrix B_t from the witness $B_{t'}$ of $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}')$ with a cluster-assignment φ' by adding a new row with a label v_0 as follows. If $i = \varphi'(v)$ for some $v \in V_{t'}^{\downarrow}$, we set $B_t[v_0] := B_t[v]$. Otherwise we define $B_t[v_0, c] := \mathrm{cent}[i, c]$ for $c \in C_t$ and $B_t[v_0, c] := 0$ in the rest of coordinates c. Note that in both cases $B_t[v_0, c] = \mathrm{cent}[i, c]$ for every $c \in C_t$: in the second case it follows from the definition, while in the first one we have $B_t[v_0, c] := B_t[v, c] = B_{t'}[v, c] = \mathrm{cent}[\varphi'(v), c] = \mathrm{cent}[i, c]$. The matrix B_t with a cluster-

assignment $\varphi = \varphi' \cup (v_0, i)$ has a snapshot $(\mathrm{cent}, \mathrm{part})$ in t and $\mathrm{pwd}(B_t, t) = \sum_{v \in V_t^\downarrow} \sum_{c \in C_t^\downarrow} W[v,c] \cdot (A[v,c] - B_t[v,c])^2 = \mathrm{pwd}(B_{t'}, t') + \sum_{c \in C_t^\downarrow} W[v_0,c] \cdot (A[v_0,c] - B_t[v_0,c])^2$. As $\mathrm{pwd}(B_{t'}, t') = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}')$, $W[v_0,c] = 0$ for every forgotten coordinate c and $B_t[v_0,c] := \mathrm{cent}[i,c]$ for every $c \in C_t$, we have $\mathrm{pwd}(B_t, t) = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0 = \tau$. So B_t witnesses that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$.

On the other hand, assume that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$. Then there exists a matrix B_t in $\mathcal{B}_t$ admitting the snapshot $(\mathrm{cent}, \mathrm{part})$ in t such that $\mathrm{pwd}(B_t, t) = \tau$. Let φ be the corresponding cluster-assignment w.r.t. B_t. We construct $B_{t'}$ from B_t by deletion of the row v_0. Then $B_{t'}$ with cluster-assignment $\varphi' = \varphi \setminus \{(v_0, i)\}$ has a snapshot $(\mathrm{cent}, \mathrm{part}')$ in t and witnesses that $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') \leq \mathrm{pwd}(B_{t'}, t') = \tau - \Delta_0$. Therefore in our algorithm $\mathcal{R}_t^0 \leq \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0 \leq \tau$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}_t^0$.

t is an introduce node (introducing a coordinate). Let t' be the child of t in T and let $\chi(t) = \chi(t') \cup \{c_0\}$ for some coordinate c_0. Fix a snapshot $(\mathrm{cent}, \mathrm{part})$ in $S(t)$, we wil denote by Δ_0 the sum of distances from $v \in V_t^\downarrow$ to the corresponding cluster center along the coordinate c_0, i.e., $\Delta_0 = \sum_{v \in V_t} W[v,c_0] \cdot (A[v,c_0] - \mathrm{cent}[\mathrm{part}(v), c_0])^2$. Let cent' be the restriction of cent to all the coordinates except c_0. We set $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0$. For correctness, assume that $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \tau = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0$. We construct a matrix B_t from a witness $B_{t'}$ of $\mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part})$ with cluster-assignment φ' by adding a column c_0. For every $v \in V_t^\downarrow$, we set $B_t[v,c] = B_{t'}[v,c]$ for all $c \in C_{t'}^\downarrow$ and $B_t[v,c_0] = \mathrm{cent}[\varphi'(v), c_0]$. Then B_t with the same cluster-assignment φ' has a snapshot $(\mathrm{cent}, \mathrm{part})$ in t and $\mathrm{pwd}(B_t, t) = \mathrm{pwd}(B_{t'}, t') + \Delta_0 = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0 = \tau$. Therefore B_t witnesses that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$. On the other hand, assume that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$. Then there exists a matrix B_t in $\mathcal{B}_t$ with the snapshot $(\mathrm{cent}, \mathrm{part})$ in t such that $\mathrm{pwd}(B_t, t) = \tau$. Let $B_{t'}$ be obtained from B_t by deletion of the column with the label c_0. Then $B_{t'}$ witnesses $\mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) \leq \tau - \Delta_0$, so in our algorithm $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0 \leq \tau$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}_t^0$.

t is a join node. Let t_1, t_2 be the two children of t in T, recall that $\chi(t_1) = \chi(t_2) = \chi(t)$ and $\chi_{t_1}^\downarrow \cap \chi_{t_2}^\downarrow = \chi(t)$. For every $(\mathrm{cent}, \mathrm{part})$ in $S(t)$ we set $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t_1}(\mathrm{cent}, \mathrm{part}) + \mathcal{R}_{t_2}(\mathrm{cent}, \mathrm{part}) - \mathrm{doublecount}$, where $\mathrm{doublecount} = \sum_{v \in V_t} \sum_{c \in C_t} W[v,c] \cdot (A[v,c] - \mathrm{cent}[\mathrm{part}(v), c])^2$.

For correctness, assume that $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \tau = \tau_1 + \tau_2 - \mathrm{doublecount}$, where $\tau_1 = \mathcal{R}_{t_1}(\mathrm{cent}, \mathrm{part})$, $\tau_2 = \mathcal{R}_{t_2}(\mathrm{cent}, \mathrm{part})$. Let B_i with the cluster-assignment φ_i be the witness of $\mathcal{R}_{t_i}(\mathrm{cent}, \mathrm{part}) = \tau_i$, $i = 1, 2$. We obtain a matrix B_t with cluster-assignment $\varphi = \varphi_1 \cup \varphi_2$ as a composition of (B_1, φ_1) and (B_2, φ_2). To check the consistency conditions, observe that the sets of common row and column labels of B_1 and B_2 are V_t and C_t correspondingly. Recall that $\varphi_1|_{V_t} = \mathrm{part} = \varphi_2|_{V_t}$. Moreover, if $v_1 \in V_1$ and $v_2 \in V_2$ are such that $\varphi_1(v_1) = \varphi_2(v_2)$, then we have $B_1[v_1,c] = \mathrm{cent}[\varphi_1(v_1), c] = \mathrm{cent}[\varphi_2(v_2), c] = B_2[v_2,c]$ for every $c \in C_t$.

Note that for every $v \in V_t$, $\varphi(v) = \varphi_1(v) = \mathrm{part}$. Pick $c \in C_t$ and $v \in V_t^\downarrow$, then $v \in V_{t_i}^\downarrow$ for some $i \in \{1, 2\}$ and so $B_t[v,c] = B_i[v,c] = \mathrm{cent}[\varphi_i(v), c] = \mathrm{cent}[\varphi(v), c]$. Therefore $(\mathrm{cent}, \mathrm{part})$ is a snapshot of B_t in t. Recall that $\mathrm{pwd}(B_t, t) = \sum_{v \in V_t^\downarrow} \sum_{c \in C_t^\downarrow} W[v,c] \cdot (A[v,c] - B_t[v,c])^2$. Here $W[v,c] = 0$ for every $v \in V_{t_i}^\downarrow$ and $c \notin C_{t_i}^\downarrow$, $i = 1, 2$. So $\mathrm{pwd}(B_t, t) = \sum_{v \in V_{t_1}^\downarrow} \sum_{c \in C_{t_1}^\downarrow} W[v,c] \cdot (A[v,c] - B_t[v,c])^2 + \sum_{v \in V_{t_2}^\downarrow} \sum_{c \in C_{t_2}^\downarrow} W[v,c] \cdot (A[v,c] - B_t[v,c])^2 - \sum_{v \in V_t} \sum_{c \in C_t} W[v,c] \cdot (A[v,c] - B_t[v,c])^2 = \mathrm{pwd}(B_1, t_1) + \mathrm{pwd}(B_2, t_2) - \mathrm{doublecount} = \tau_1 + \tau_2 - \mathrm{doublecount} = \tau$. Hence B_t witnesses $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$.

For the converse, assume that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$ and B_t is a matrix witnessing this. Let B_i be the restriction of B_t to rows $V_{t_i}^\downarrow$ and columns $C_{t_i}^\downarrow$, $i = 1, 2$. Then B_1 and B_2 have a snapshot $(\mathrm{cent}, \mathrm{part})$ and in our algorithm $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t_1}(\mathrm{cent}, \mathrm{part}) + \mathcal{R}_{t_2}(\mathrm{cent}, \mathrm{part}) - \mathrm{doublecount} \leq \mathrm{pwd}(B_1) + \mathrm{pwd}(B_2) - \mathrm{doublecount} = \mathrm{pwd}(B) = \tau$. Hence the resulting record $\mathcal{R}_t = \mathcal{R}_t^0$ is correct, which concludes the correctness proof of the algorithm.

To bound the runtime of the algorithm, observe that at each node t we compute the record $\mathcal{R}_t$ for $|S(t)| \leq |D|^{kq} k^q = |D|^{\mathcal{O}(k \cdot \mathrm{tw}(G_I))}$ entries, where each entry is computed in time at most quadratic in $\mathrm{tw}(G_I)$. Since the tree-decomposition is nice and has at most $\mathcal{O}(n + d)$ nodes, the runtime of the algorithm is upper-bounded by $(n + d) \cdot |D|^{\mathcal{O}(k \cdot \mathrm{tw}(G_I))}$. □

Towards proving fixed-parameter tractability without involving k, we consider the case where k is significantly larger than $\mathrm{tw}(G_I)$. We prove that this case always admits a "perfect" solution, which can also be computed efficiently.

Lemma 2 ($\star$). *If $k \geq (\mathrm{tw}(G_I) + 1) \cdot |D|^{\mathrm{tw}(G_I)+1}$ then there exists a matrix B over domain D containing at most k distinct rows such that $\|W \circ (A - B)\|_F^2 = 0$. Moreover, given a nice tree-decomposition of width $q - 1$ such that $k \geq q \cdot |D|^q$, such a matrix B can be computed in time $(n + d) \cdot |D|^{\mathcal{O}(q)}$.*

Proof Sketch. We will prove the second claim of the lemma, from which the first one follows. To this end, let us assume that we are given a nice tree-decomposition $\mathcal{T} = (T, \chi)$ of

G_I of width $q-1$ and $k \geq q \cdot |D|^q$. We denote the root of T by r and define C_t, V_t, $C_t^{\downarrow}$ and $V_t^{\downarrow}$ analogously to the proof of Theorem 1. We will explicitly construct $q \cdot |D|^q$ cluster centers yielding a zero sum of squares, so that after processing the node t, each of the centers is defined on coordinates $c \in C_t^{\downarrow}$. The remaining clusters will be empty, thus for the rest of the proof we assume $k = q \cdot |D|^q$.

Formally, we will attach to every node t the pair (R_t, α_t) where R_t is the matrix with the row labels $[k]$ and the column labels $C_t^{\downarrow}$, and α_t is a mapping from $V_t^{\downarrow}$ to $[k]$. Intuitively, R_t stores the restrictions of cluster centers to processed coordinates, while α_t assigns a cluster index to every row label $v \in V_t^{\downarrow}$. Throughout the dynamic programming procedure, we will maintain the following invariants:

1. $\alpha_t(u) \neq \alpha_t(v)$ for any two distinct u and v from V_t,
2. for every $\omega \in D^{C_t}$, there are precisely $\frac{k}{|D|^{|C_t|}}$ indices $i \in [k]$ such that $R_t[i][C_t] = \omega$, where $R_t[i][C_t]$ is the restriction of $R_t[i]$ to the coordinate set C_t,
3. $\sum_{v \in V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[v,c] \cdot (A[v,c] - R_t[\alpha_t(v), c])^2 = 0$.

To complete the proof, it remains to show that the records (R_t, α_t) can be computed in a leaf-to-root fashion via dynamic programming along the nodes of the tree-decomposition. Indeed, once we obtain the record (R_r, α_r) for the root node r, we can define the solution matrix B by setting $B[v] = R_r[\alpha_r(v)]$; this matrix has at most k distinct rows and the third invariant yields that $\sum_{v \in V_A} \sum_{c \in C_A} W[v,c] \cdot (A[v,c] - B[v,c])^2 = 0$. □

At this point, we can prove the main result of this section.

Theorem 3. BOUNDED-DOMAIN MCME *is fixed-parameter tractable when parameterized by* $\mathrm{tw}(G_I)$.

Proof. Given an instance of BOUNDED-DOMAIN MCME, we begin by constructing the incidence graph G_I and then applying the known 5-approximation algorithm to compute a tree-decomposition of G_I of width $q - 1 \leq 5\,\mathrm{tw}(G_I)$ in time $2^{\mathcal{O}(\mathrm{tw}(G_I))} \cdot (n+d)$. At this point we proceed by comparing q and k, as follows:

- if $k \geq q \cdot |D|^q$, then we can correctly output "Yes" (along with a suitable witness) by using Lemma 2;
- on the other hand, if $k \leq q \cdot |D|^q$ then we instead invoke Theorem 1 to solve the instance.

The running time in the former case is upper-bounded by $(n+d) \cdot |D|^{\mathcal{O}(\mathrm{tw}(G_I))}$, while in the latter case the bound is $(n+d) \cdot |D|^{\mathcal{O}(k \cdot \mathrm{tw}(G_I))} \leq (n+d) \cdot |D|^{|D|^{\mathcal{O}(\mathrm{tw}(G_I))}}$. □

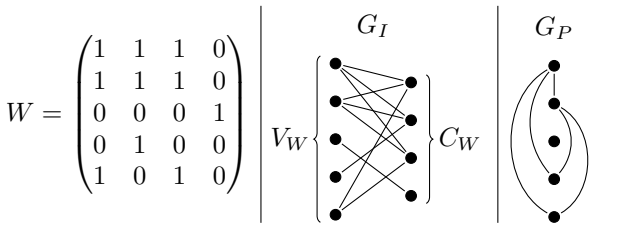

Figure 2. Example of a mask W (left) together with its incidence graph (middle) and its primal graph (right). Vertices from top to bottom correspond to rows from top to bottom and in the case of C_W columns from left to right. In this example $\mathrm{tw}(G_I) = 3$ and $\mathrm{tw}(G_P) = 2$.

4. Handling Real-Valued Matrices with Treewidth

We now turn our attention to REAL-VALUED MCME. As was mentioned in the Introduction, a fixed-parameter algorithm for the problem parameterized by the incidence treewidth of W would resolve a long-standing open problem as a special case, and so far remains beyond reach. However, we can still make tangible progress on the problem by considering the treewidth of a different representation of W. In particular, the *primal graph* G_P of W is a graph containing one vertex for each row of W, where two rows a, b are adjacent if and only if there is a column e such that $W[a,e] = W[b,e] = 1$. An example comparing primal and incidence graphs is provided in Figure 2.

Our aim in this section is to prove the next theorem.

Theorem 4 (⋆)**.** REAL-VALUED MCME *is fixed-parameter tractable when parameterized by* $\mathrm{tw}(G_P)$.

Before we proceed to the proof, we will first introduce some additional notation that will be useful in the context of real-valued matrices. For every $V' \subseteq V_A$, let $\mu(V', c) = \mathrm{argmin}_x f(x)$ for $f(x) = \sum_{v \in V'} W[v,c] \cdot (A[v,c] - x)^2$ and let $\Delta_c(V') = f(\mu(V', c))$. Intuitively, if vectors from V' form a cluster then $\Delta_c(V')$ is a minimal sum of distances to the cluster center along the coordinate c.

Observation 5 (⋆)**.** *For each coordinate* c*, if* $V'' = \{z \in V' \mid W[z,c] = 1\} \neq \emptyset$ *then*

$$\mu(V', c) = \frac{\sum_{v \in V''} A[v,c]}{|V''|}, \text{ and}$$

$$\Delta_c(V') = \sum_{v \in V''} (A[v,c])^2 - \frac{(\sum_{v \in V''} A[v,c])^2}{|V''|}.$$

Now we are ready to present the proof of Theorem 4.

Proof Sketch for Theorem 4. As our initial step, we once again apply the known approximation algorithm (Bodlaender et al., 2016) to compute a nice tree-decomposition (T, χ)

of G_P of width $q \leq 5\,\mathrm{tw}(G_P)$. We keep the notations r, V_t, and $V_t^{\downarrow}$ from the previous section. Note that now the bags in the tree-decomposition contain only vectors, so $V_t = \chi(t)$. In contrast to the previous section, for every node t we denote by $C_t^{\downarrow}$ the set of all *processed* in t coordinates, i.e., coordinates c such that $W[v_c, c] \neq 0$ for some $v_c \in V_t^{\downarrow} \setminus V_t$. A crucial observation is the following: if $c \in C_t^{\downarrow}$, then for all vectors $v \notin V_t^{\downarrow}$ it holds that $W[v, c] = 0$, otherwise v and v_c would appear in the same bag. In other words, values of the cluster centers on $c \in C_t^{\downarrow}$ are not meaningful for vectors introduced outside of T_t.

We will design a leaf-to-root dynamic programming algorithm which computes a set of records $\mathcal{R}_t$ at each node t of T. For each way p of partioning the vectors in the bag V_t into at most k clusters, the record $\mathcal{R}_t$ stores the minimum cost of clustering the vectors of $V_t^{\downarrow}$ in the coordinates of $C_t^{\downarrow}$, considering only the partitions of $V_t^{\downarrow}$ that extend p.

For the root r of T, $\mathcal{R}_r(\emptyset)$ is equal to the desired value $\min_B ||W \circ (A - B)||_F^2$, so it suffices to show how to compute the records at each node in T from the records of its children. Let t be an introduce node with a child t' such that $V_t = V_{t'} \cup \{v\}$, denote by p' the restriction of p to $V_{t'}$. We set $\mathcal{R}_t(p) = \mathcal{R}_{t'}(p')$ since $C_t^{\downarrow} = C_{t'}^{\downarrow}$ and $W[v, c] = 0$ for each $c \in C_{t'}^{\downarrow}$. If t is a join node with children t_1 and t_2, observe that $W[v, c] = 0$ for each $v \in V_{t_1}^{\downarrow} \setminus V_t$, $c \in C_{t_2}^{\downarrow}$, and for each $v \in V_{t_2}^{\downarrow} \setminus V_t$, $c \in C_{t_1}^{\downarrow}$. Furthermore, $C_{t_1}^{\downarrow}$ is necessarily disjoint from $C_{t_2}^{\downarrow}$, $C_t^{\downarrow} = C_{t_1}^{\downarrow} \cup C_{t_2}^{\downarrow}$, and $\mathcal{R}_t(p) = \mathcal{R}_{t_1}(p) + \mathcal{R}_{t_2}(p)$.

Finally, for a forget node t with a child t' such that $V_t \cup \{v\} = V_{t'}$, branch on all admissible partitions p' of $V_{t'}$ that extend p. Start with $\Delta(p') = 0$, for each coordinate $c \notin C_{t'}^{\downarrow}$ such that $W[v, c] = 1$, and for each equivalence class $[x]_{p'} \in [p']$ add $\Delta_c([x]_{p'})$, the cost of clustering of the rows $[x]_{p'}$ in the column c, to $\Delta(p')$. Since $C_t^{\downarrow} \setminus C_{t'}^{\downarrow}$ are precisely the coordinates above, and no row outside of the bag $V_{t'}$ contributes to these coordinates, the new partial weighted distance is $\mathcal{R}_{t'}(p') + \Delta(p')$, assuming that p' is the right choice. Therefore, we set $\mathcal{R}_t(p) = \min_{p'}(\mathcal{R}_{t'}(p') + \Delta(p'))$.

As for the running time, in each step there are at most $\mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))}$ partitions of the bag and hence the total running time required to process all forget nodes can in total be upper-bounded by $d \cdot \mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))} \leq (n + d) \cdot \mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))}$. The latter bound can then easily be shown to hold for all other kinds of nodes. □

5. An Incidence-Graph Based Algorithm for Real-Valued MCME

While Theorem 4 significantly expands the previously known boundaries of tractability for MCME, the algorithm's performance strongly depends on the structural properties of the primal graph. In general, primal graph representations are known to be denser than incidence graph representations, and this may make them unsuitable for the application of structure-based algorithms on certain instances (see, e.g., the example below Theorem 6).

As our final result, we show that although a fixed-parameter algorithm for Real-Valued MCME parameterized by $\mathrm{tw}(G_I)$ remains beyond our reach, we can exploit a different parameter of the incidence graph to achieve fixed-parameter tractability—namely, the local feedback edge number.

Theorem 6 (⋆). Real-Valued MCME *is fixed-parameter tractable when parameterized by* $\mathrm{lfen}(G_I)$.

We note that it is not difficult to show that $\mathrm{lfen}(G_I)$ and $\mathrm{tw}(G_P)$ are pairwise incomparable parameterizations. Indeed, an $n \times 1$ mask consisting only of "1" entries has $\mathrm{lfen}(G_I) = 0$ but $\mathrm{tw}(G_P) = n - 1$. On the other hand, consider, for some integer m, an $(m + 1) \times 2m$ mask W such that the first row v_0 of W consists only of "1" entries, while for each $i \in [m]$ the $(i + 1)$-th row has "1" entries on precisely two positions: i and $i + m$. Then G_P is a star with center in v_0, so $\mathrm{tw}(G_P) = 1$. However, G_I consists of m edge-disjoint cycles intersecting in v_0. It is then easy to observe that the local feedback edge number of v_0 with respect to any spanning tree of G_I is m.

We proceed by introducing some additional terminology that will be useful in the coming arguments. Let T be a fixed rooted spanning tree of G such that $\mathrm{lfen}(G, T) = \mathrm{lfen}(G)$, denote the root by r. For $t \in V(T)$, let T_t be the subtree of T rooted at t. We define the *boundary* $\delta(t)$ of t to be the set of endpoints of all edges in G with precisely one endpoint in $V(T_t)$ (observe that the boundary can never have a size of 1). t is called *closed* if $|\delta(t)| \leq 2$ and *open* otherwise.

Proposition 7 (Ganian and Korchemna (2021))**.**

1. *For every closed child t' of t in T, it holds that $\delta(t') = \{t, t'\}$ and tt' is the only edge between $V(T_{t'})$ and $V(G) \setminus V(T_{t'})$ in G.*
2. $|\delta(t)| \leq 2\,\mathrm{lfen}(G) + 2$.
3. *Let $\{t_i | i \in [j]\}$ be the set of all open children of t in T. Then $j \leq 2\,\mathrm{lfen}(G)$ and $\delta(t) \subseteq \bigcup_{i=1}^{j} \delta(t_i) \cup \{t\} \cup N_G(t)$.*

To prove Theorem 6, we will provide a leaf-to-root dynamic programming algorithm which stores information about optimal partitionings of $\delta(t)$ into clusters. On a very intuitive level, Point 1. of Proposition 7 allows the algorithm to handle the closed children in a greedy manner, Point 2. ensures that the size of the records is bounded, and Point 3. is used in the dynamic programming step to compute records for a parent based on the records of its children. Furthermore, one can observe that $|\bigcup_{i=1}^{j} \delta(t_i)|$ is upper-bounded by a

linear function of $\mathrm{lfen}(t)$, which will be useful to ascertain the runtime bound for the algorithm.

Observation 8. *For each node t in T, $|\bigcup_{i=1}^{j} \delta(t_i)| \leq 4\,\mathrm{lfen}(t) + 2$.*

Proof. By Point 2. of Proposition 7, the number of vertices in $\bigcup_{i=1}^{j} \delta(t_i)$ that belong to $\delta(t)$ is at most $|\delta(t)| \leq 2\,\mathrm{lfen}(G) + 2$. If $v \in \delta(t_i) \setminus (\delta(t))$ for some $i \in [j]$, then $v = t_i$. So the number of such v is at most $j \leq 2\,\mathrm{lfen}(G)$ by Point 3. of Proposition 7. In total, $|\bigcup_{i=1}^{j} \delta(t_i)| \leq 4\,\mathrm{lfen}(t) + 2$. □

We are now ready to establish the claimed result by providing an algorithm for REAL-VALUED MCME with running time $(kn^2 + d) \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$, assuming a spanning tree T of minimum local feedback edge number is provided. We note that such a spanning tree can be computed by a fixed-parameter algorithm (Ganian & Korchemna, 2021).

Proof Sketch for Theorem 6. We will design a leaf-to-root dynamic procedure computing the set of records for every node of T. Let $V_t^{\downarrow}$ and $C_t^{\downarrow}$ be the sets of vectors and coordinates of T_t correspondingly, we denote by V_t and C_t the restrictions of $\delta(t)$ to vectors and coordinates respectively, then $\delta(t) = V_t \sqcup C_t$. Analogously to the previous theorems, we define the record $\mathcal{R}_t$ of t as a mapping that associates each partition p of V_t with the optimal partial cost of clustering that agrees with p on $\delta(t)$. The partial cost is computed only on the rows of $V_t \cup V_t^{\downarrow}$ and the columns of $C_t^{\downarrow}$.

Recall that for the root r of T, the boundary is empty, in particular $V_r = \emptyset$, and $\mathcal{R}_r(\emptyset)$ is equal to the minimum value of $\|W \circ (A - B)\|_F^2$ that can be achieved by any real matrix B containing at most k distinct rows. Hence the instance is a YES-instance if and only if $\mathcal{R}_r(\emptyset) \leq \ell$. Thus, it only remains to carry out the computation of the records in each node from the records of its children, and this is the most technical part of the proof.

Intuitively, for a node t we consider separately its open and closed children, denote the former by $t_1, \ldots, t_j$, and the latter by $t_{j+1}, \ldots, t_m$. For each partition p of V_t, we enumerate all possible extensions of p to a partition p^* of $V_t \bigcup_{i \in [j]} V_{t_i}$; the crucial observation here is that the number of different p's and p^*'s is upper-bounded by $\mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$. Now, if t is a vertex node, every coordinate in $C_t^{\downarrow}$ is contained in the subtree of one of the children of t, and the cost of this coordinate is accounted-for in the record of the child. It thus only remains to sum up the records of the children, taking each time the respective partition on V_{t_i} for a child t_i. For an open child the partition is explicitly a restriction of p^* on V_{t_i}, and for a closed child, as $\delta(t_i) = \{t, t_i\}$, $V_{t_i} = \{t\}$ and the partition of V_{t_i} is always trivial.

More work has to be done if t is a coordinate node: we have to account for the distances in the new coordinate t. For a closed child t_i there is only interaction between the coordinate t and the row t_i, the rest of the subtree is independent. Therefore, the rows $t_{j+1}, \ldots, t_m$ can be clustered in any way, to minimize the cost in the coordinate t. This one-dimensional subproblem can be solved by a separate dynamic programming in time $\mathcal{O}(km^2)$. For the open children, the exact partioning on the boundary vectors is again fixed, and the dynamic programming can be enhanced to respect the given paritioning on these additional vectors, at the cost of an extra $2^{\mathcal{O}(\mathrm{lfen}(G_I))}$ time factor. □

6. Concluding Remarks

While our algorithmic results are specifically designed to deal with MEANS CLUSTERING WITH MISSING ENTRIES, it would be interesting to see whether the approaches and techniques developed here can be applied to other clustering variants or, e.g., the related task of low-rank matrix completion. Still, on the theoretical side the by far most prominent problem that is relevant to this research direction is the complexity of k-MEANS CLUSTERING for real-valued matrices when parameterized by the number of columns. A W[1]-hardness result for this problem would immediately exclude the existence of a fixed-parameter algorithm for REAL-VALUED MCME parameterized by the incidence treewidth of the mask, while a fixed-parameter algorithm could potentially open up the way towards tractability.

It would also be interesting to see how the considered sparsity parameters behave in practical settings. In particular, even though direct implementations of our exact algorithms with runtime guarantees are unlikely to outperform state-of-the-art heuristics, it may be possible to exploit these parameters to guide or otherwise improve existing methods.

Acknowledgements

Robert Ganian, Thekla Hamm, Viktoriia Korchemna, and Kirill Simonov acknowledge support by the Austrian Science Fund (FWF, projects Y1329 and P31336). Karolina Okrasa acknowledges support by the European Research Council, grant agreement No 714704; parts of this work were performed while visiting TU Wien, Vienna, Austria.

References

Aloise, D., Deshpande, A., Hansen, P., and Popat, P. Np-hardness of euclidean sum-of-squares clustering. *Mach. Learn.*, 75(2):245–248, 2009.

Bodlaender, H. L. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Discret. Math.*, 25(6):1305–1317, 1996.

Bodlaender, H. L., Jansen, B. M. P., and Kratsch, S. Preprocessing for treewidth: A combinatorial analysis through kernelization. *SIAM J. Discret. Math.*, 27(4):2108–2142, 2013.

Bodlaender, H. L., Drange, P. G., Dregi, M. S., Fomin, F. V., Lokshtanov, D., and Pilipczuk, M. A $c^k n$ 5-approximation algorithm for treewidth. *SIAM J. Comput.*, 45(2):317–378, 2016.

Cohen-Addad, V., de Mesmay, A., Rotenberg, E., and Roytman, A. The bane of low-dimensionality clustering. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 441–456. SIAM, 2018. ISBN 978-1-6119-7503-1. URL http://dl.acm.org/citation.cfm?id=3174304.3175300.

Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. *Parameterized Algorithms*. Springer, 2015. ISBN 978-3-319-21274-6. doi: 10.1007/978-3-319-21275-3. URL https://doi.org/10.1007/978-3-319-21275-3.

Dahiya, Y., Fomin, F. V., Panolan, F., and Simonov, K. Fixed-parameter and approximation algorithms for PCA with outliers. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2341–2351. PMLR, 2021.

Downey, R. G. and Fellows, M. R. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

Drineas, P., Frieze, A. M., Kannan, R., Vempala, S. S., and Vinay, V. Clustering large graphs via the singular value decomposition. *Mach. Learn.*, 56(1-3):9–33, 2004.

Dvorák, P., Eiben, E., Ganian, R., Knop, D., and Ordyniak, S. The complexity landscape of decompositional parameters for ILP: programs with few global variables and constraints. *Artif. Intell.*, 300:103561, 2021.

Eiben, E., Fomin, F. V., Golovach, P. A., Lochet, W., Panolan, F., and Simonov, K. EPTAS for *k*-means clustering of affine subspaces. In Marx, D. (ed.), *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pp. 2649–2659. SIAM, 2021a. doi: 10.1137/1.9781611976465.157. URL https://doi.org/10.1137/1.9781611976465.157.

Eiben, E., Ganian, R., Kanj, I., Ordyniak, S., and Szeider, S. The parameterized complexity of clustering incomplete data. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 7296–7304. AAAI Press, 2021b. URL https://ojs.aaai.org/index.php/AAAI/article/view/16896.

Fellows, M. R., Protti, F., Rosamond, F. A., da Silva, M. D., and Souza, U. S. Algorithms, kernels and lower bounds for the flood-it game parameterized by the vertex cover number. *Discret. Appl. Math.*, 245:94–100, 2018.

Fomin, F. V., Golovach, P. A., and Panolan, F. Parameterized low-rank binary matrix approximation. In Chatzigiannakis, I., Kaklamanis, C., Marx, D., and Sannella, D. (eds.), *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pp. 53:1–53:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

Fomin, F. V., Golovach, P. A., and Simonov, K. Parameterized *k*-clustering: Tractability island. *J. Comput. Syst. Sci.*, 117:50–74, 2021.

Ganian, R. and Korchemna, V. The complexity of bayesian network learning: Revisiting the superstructure. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 430–442. Curran Associates, Inc., 2021.

Ganian, R. and Ordyniak, S. The complexity landscape of decompositional parameters for ILP. *Artif. Intell.*, 257:61–71, 2018.

Ganian, R. and Ordyniak, S. Solving integer linear programs by exploiting variable-constraint interactions: A survey. *Algorithms*, 12(12):248, 2019. URL https://doi.org/10.3390/a12120248.

Ganian, R., Kanj, I. A., Ordyniak, S., and Szeider, S. Parameterized algorithms for the matrix completion problem. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1642–1651. PMLR, 2018. URL http://proceedings.mlr.press/v80/ganian18a.html.

Gaspers, S. and Najeebullah, K. Optimal surveillance of covert networks by minimizing inverse geodesic length. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pp. 533–540. AAAI Press, 2019.

Grüttemeier, N., Komusiewicz, C., and Morawietz, N. Efficient bayesian network structure learning via parameterized local search on topological orderings. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 12328–12335. AAAI Press, 2021.

Inaba, M., Katoh, N., and Imai, H. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the 10th annual Symposium on Computational Geometry (SoCG)*, pp. 332–339. ACM, 1994.

Kloks, T. *Treewidth: Computations and Approximations*. Springer, Berlin, 1994.

Koana, T., Froese, V., and Niedermeier, R. Parameterized algorithms for matrix completion with radius constraints. In Gørtz, I. L. and Weimann, O. (eds.), *31st Annual Symposium on Combinatorial Pattern Matching, CPM 2020, June 17-19, 2020, Copenhagen, Denmark*, volume 161 of *LIPIcs*, pp. 20:1–20:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Koana, T., Froese, V., and Niedermeier, R. Binary matrix completion under diameter constraints. In Bläser, M. and Monmege, B. (eds.), *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pp. 47:1–47:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

Lloyd, S. P. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136, 1982.

Mertzios, G. B., Nichterlein, A., and Niedermeier, R. The power of linear-time data reduction for maximum matching. *Algorithmica*, 82(12):3521–3565, 2020.

Moshkovitz, M., Dasgupta, S., Rashtchian, C., and Frost, N. Explainable k-means and k-medians clustering. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7055–7065. PMLR, 2020.

Nesetril, J. and de Mendez, P. O. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and Combinatorics*. Springer, 2012.

Ordyniak, S. and Szeider, S. Parameterized complexity results for exact bayesian network structure learning. *J. Artif. Intell. Res.*, 46:263–302, 2013.

Robertson, N. and Seymour, P. D. Graph minors. i. excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

Robertson, N. and Seymour, P. D. Graph minors. III. planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984.

Samer, M. and Szeider, S. Fixed-parameter tractability. In Biere, A., Heule, M., van Maaren, H., and Walsh, T. (eds.), *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pp. 425–454. IOS Press, 2009.

Samer, M. and Szeider, S. Constraint satisfaction with bounded treewidth revisited. *J. Comput. Syst. Sci.*, 76(2): 103–114, 2010.

Simonov, K., Fomin, F. V., Golovach, P. A., and Panolan, F. Refined complexity of PCA with outliers. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5818–5826. PMLR, 2019.

Wang, S., Li, M., Hu, N., Zhu, E., Hu, J., Liu, X., and Yin, J. K-means clustering with incomplete data. *IEEE Access*, 7:69162–69171, 2019. doi: 10.1109/ACCESS.2019.2910287.

The Complexity of k-Means Clustering when Little is Known (Appendix: Full Version)

Robert Ganian [*1] **Thekla Hamm** [*1] **Viktoriia Korchemna** [*1] **Karolina Okrasa** [*2] **Kirill Simonov** [*1]

Abstract

In the area of data analysis and arguably even in machine learning as a whole, few approaches have been as impactful as the classical k-means clustering. Here, we study the complexity of k-means clustering in settings where most of the data is not known or simply irrelevant. To obtain a more fine-grained understanding of the tractability of this clustering problem, we apply the parameterized complexity paradigm and obtain three new algorithms for k-means clustering of incomplete data: one for the clustering of bounded-domain (i.e., integer) data, and two incomparable algorithms that target real-valued data. Our approach is based on exploiting structural properties of a graphical encoding of the missing entries, and we show that tractability can be achieved using significantly less restrictive parameterizations than in the complementary case of few missing entries.

1. Introduction

k-means clustering is a fundamental task in machine learning and data analysis, one that has been the focus of extensive empirical as well as theoretical research. In general, the aim in k-means clustering is to partition the rows in an input matrix A into k clusters and compute one center per cluster so as to minimize the within-cluster sum of squares—or, when viewed as a decision problem, achieve a within-cluster sum of squares of at most a specified target value ℓ. Depending on the setting, A could be real-valued (Lloyd, 1982; Moshkovitz et al., 2020; Fomin et al., 2021) or contain integers from a bounded domain; for instance, Fomin et al. (Fomin et al., 2018) studied the case where A was binary, and Ganian et al. (Ganian et al., 2018) the bounded-domain case where $\ell = 0$.

It is well known that k-means clustering is NP-complete, even when restricted to $k = 2$ clusters (Drineas et al., 2004; Aloise et al., 2009). On the theoretical side, a prominent approach used to circumvent such lower bounds is the use of the *parameterized complexity*[2] paradigm, which provides a more fine-grained look into the complexity of problems. Since its inception, parameterized complexity has been successfully applied to a multitude of problems relevant to machine learning and artificial intelligence such as, e.g., Bayesian network learning (Ordyniak & Szeider, 2013; Grüttemeier et al., 2021; Ganian & Korchemna, 2021), integer linear programming (Ganian & Ordyniak, 2018; 2019; Dvorák et al., 2021), principal component analysis (Simonov et al., 2019; Dahiya et al., 2021), and of course clustering (Cohen-Addad et al., 2018; Fomin et al., 2018).

Apart from classical clustering problems, one direction that has been gaining traction in recent years is the study of clustering in settings where some of the data is not known or simply irrelevant—a task that essentially combines data completion with clustering (Wang et al., 2019). Indeed, several authors have studied the parameterized complexity of various clustering problems for data with missing entries (Ganian et al., 2018; Koana et al., 2020; 2021; Eiben et al., 2021b), and Eiben et al. (Eiben et al., 2021a) obtained a fixed-parameter approximation algorithm specifically for (k-)MEANS CLUSTERING WITH MISSING ENTRIES (MCME).

While their approaches and techniques differ, all of these works target the case where nearly all entries are known, i.e., where the unknown entries only occur in a precisely defined "sparse" way. On the other hand, very little is known about the complexity of MCME when the number of unknown entries is large; such situations are of practical relevance in, e.g., recommender systems and predictive analytics. For example, in the classical Netflix Prize challenge where the task was to predict user ratings for movies based on previous ratings, only about 1% of the user-movie pairs were orig-

[*]Equal contribution [1]Algorithms and Complexity Group, TU Wien, Austria [2]Institute of Informatics, University of Warsaw, Poland. Correspondence to: Robert Ganian <rganian@gmail.com>, Thekla Hamm <thamm@ac.tuwien.ac.at>, Viktoriia Korchemna <vkorchemna@ac.tuwien.ac.at>, Kirill Simonov <ksimonov@ac.tuwien.ac.at>, Karolina Okrasa <k.okrasa@mini.pw.edu.pl>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[2]See Preliminaries for a brief introduction.

inally supplied with a rating.[3] The central mission of this article is to push the boundaries of tractability for MCME to instances where *known* entries are sparse.

Contribution. A natural approach for handling instances of MCME with many missing entries would be to invert the parameterizations that have been developed for tackling instances where almost all entries are known. For instance, Eiben et al. (Eiben et al., 2021b) and Ganian et al. (Ganian et al., 2018) obtained several fixed-parameter clustering algorithms by using a parameter called the *covering number*, which is the minimum number of rows and columns needed to cover all the unknown entries. Equivalently, if we use an auxiliary binary matrix W (the *mask*) to specify which entries of A are relevant/known, the covering number is simply the minimum number of rows and columns needed to cover all the 0's in W. It would be tempting to instead parameterize by the number of rows and columns needed to cover all the 1's in W; such an "inverse covering number" would lead to fixed-parameter algorithms targeting instances where all known entries occur in a few columns and rows.

However, we show that in our setting it is in fact possible to do much better than that. Our approach is based on using the *incidence graph* representation of W, which is the graph containing one vertex for each column and for each row of W and where an edge connects row a with column b if and only if $W[a, b] = 1$. Observe that the inverse covering number considered in the previous paragraph would simply be the size of a minimum vertex cover in W, while the covering number is the size of a minimum vertex cover in the complement of W. It is worth noting that in graph-theoretic settings, the size of a minimum vertex cover is considered a highly restrictive parameterization, one that is used primarily when more desirable parameterizations fail. On the other hand, the by far most prominent parameter used for graphs is *treewidth* (Robertson & Seymour, 1984), which has found ubiquitous applications throughout computer science and which is smaller (i.e., less restrictive as a parameter) than the size of a minimum vertex cover. As our first result, we use the treewidth of the incidence graph of the mask W (the *incidence treewidth*) to prove:

Result 1: Bounded-domain MCME is fixed-parameter tractable when parameterized by the incidence treewidth of the mask.

Result 1 is noteworthy not only due to the use of treewidth instead of the inverse covering number, but also because incidence treewidth is the *only* parameter required to achieve tractability. In particular, unlike previous algorithms for clustering incomplete (bounded-domain or real-valued) data by Eiben et al. (Eiben et al., 2021a;b), Ganian et al. (Ganian et al., 2018), and Koana et al. (Koana et al., 2020; 2021), Result 1 can also be applied to instances where the number k of clusters is large. As we will see later, we will be able to retain this benefit in all our algorithms.

Unfortunately, an extension of Result 1 towards real-valued instances seems difficult at this point. Indeed, a fixed-parameter algorithm for real-valued MCME parameterized by the incidence treewidth of the mask would imply, as a special case, fixed-parameter tractability parameterized by the number d of columns. However, the existence of a fixed-parameter algorithm even for k-MEANS (corresponding to the restriction of MCME where all entries are known) when parameterized by d is a long-standing open problem dating back to Inaba et al.'s celebrated XP algorithm parameterized by $k + d$ (Inaba et al., 1994).

Instead, we show that one can obtain fixed-parameter algorithms for real-valued MCME by using the treewidth of a different graph representation of W. In particular, we use the *primal graph*—a well-established counterpart to the incidence graph representation in settings such as Boolean satisfiability (Samer & Szeider, 2009), constraint satisfaction (Samer & Szeider, 2010) or integer programming (Ganian & Ordyniak, 2019). In our context, the primal graph of W contains a vertex for each row of W, with edges connecting pairs of rows that share at least one coordinate which is known/relevant according to the mask, and the primal treewidth is simply the treewidth of this graph.

Result 2: Real-valued MCME is fixed-parameter tractable when parameterized by the primal treewidth of the mask.

Note that, due to the nature of the primal graph representation, the primal treewidth is a more restrictive parameter than the incidence treewidth (in the sense of being bounded on fewer classes of instances). In fact, the primal treewidth of a mask may in certain cases be unboundedly large even when its incidence graph is just a tree. It would hence be useful to also have an algorithm for real-valued MCME that can exploit properties of the incidence graph, even for properties that are more restrictive than incidence treewidth.

However, the same obstacle towards extending Result 1 to the real-valued setting also applies to all traditional structural parameters that are, intuitively, based on small vertex separators in the graph—including *pathwidth* (Robertson & Seymour, 1983), *treedepth* (Nesetril & de Mendez, 2012), the *feedback vertex number* e.g., (Mertzios et al., 2020) and even the *vertex cover number* e.g., (Gaspers & Najeebullah, 2019; Fellows et al., 2018), which is the most restrictive vertex separator based parameter. In spite of this, as our third and final result we identify a parameter of the incidence graph that yields fixed-parameter tractability of real-valued MCME: the recently introduced *local feedback edge number* (lfen) (Ganian & Korchemna, 2021). On a high level,

[3]The dataset is available at https://www.kaggle.com/netflix-inc/netflix-prize-data.

lfen can be seen as an edge-based restriction of treewidth, and as a less restrictive parameter than the edge deletion distance to acyclicity (i.e., the *feedback edge number*).

Result 3: Real-valued MCME is fixed-parameter tractable when parameterized by the local feedback edge number of the incidence graph of the mask.

We note that the parameters used in Results 2 and 3 are orthogonal and the gap between them may be arbitrarily large. Because of this, both results are incomparable and give rise to different tractable classes for the problem.

2. Preliminaries

For an integer i, we let $[i] = \{1, 2, \ldots, i\}$ and $[i]_0 = [i] \cup \{0\}$. On the other hand, if p is an equivalence relation on domain X, we denote by $[p]$ the set of equivalence classes of p and by $|[p]|$ the cardinality of $[p]$, i.e., the number of equivalence classes of p. For every $x \in X$, $[x]_p \in [p]$ is the equivalence class containing x.

Notation and Problem Formulation. Let A be a matrix with n rows and d columns. We use V_A and C_A to denote the set of row and column indices of A, i.e., $V_A = [n]$ and $C_A = [d]$. Let $M_A = \max_{v \in V_A} \max_{c \in C_A} |A[v, c]|$. For two matrices A, B, we denote by $A - B$ the entry-wise subtraction of the two matrices, i.e., $(A - B)[i, j] = A[i, j] - B[i, j]$, and similarly by $A \circ B$ the entry-wise product of the two matrices, i.e., $(A \circ B)[i, j] = A[i, j] \cdot B[i, j]$. For a matrix $A \in \mathbb{R}^{n \times d}$ we denote its squared Frobenius norm by $||A||_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{d} A[i, j]^2$. We can now formalize our problems of interest (abbreviated as MCME):

MEANS CLUSTERING WITH MISSING ENTRIES

Input: Matrix $A \in D^{n \times d}$ over a domain $D \subseteq \mathbb{R}$, binary matrix W (the *mask*), integers k and ℓ.

Task: Determine if there exists a matrix B over D containing at most k distinct rows such that $\|W \circ (A - B)\|_F^2 \leq \ell$.

We distinguish between BOUNDED-DOMAIN MCME and REAL-VALUED MCME depending on whether $D = [z]_0$ for some fixed integer z, or D is the set of reals. In particular, for $z = 1$ this coincides with BINARY k-MEANS CLUSTERING (Fomin et al., 2018). In this case, minimizing the Frobenius norm also minimizes the sum of Hamming distances for each row to its cluster center. Using the Frobenius norm also for $z > 1$ is not only consistent with the continuous case, but also allows us to capture further applications, e.g., when the data matrix to cluster is composed of user ratings each taking a small numeric value. While we state MCME as a decision problem for complexity-theoretic reasons, every algorithm presented here is constructive and can output a solution matrix B if one exists.

In this formulation, the distinct rows of B are the centers of the sought-after clusters while ℓ is the target upper-bound on the sum of squares. For $V' \subseteq V_A$, we call a mapping $\varphi : V' \to [k]$ a *cluster-assignment* w.r.t. B if for every $v_1, v_2 \in V'$ such that $\varphi(v_1) = \varphi(v_2)$, it holds that $B[v_1] = B[v_2]$. Notice that the existence of a cluster assignment w.r.t. a matrix B' implies that B' has at most k distinct rows. The *clusters* associated with φ are then sets of the form $\{v \in V \mid \varphi(v) = i\}$ for some $i \in [k]$.

In our algorithms, we often construct the solution matrix dynamically by "merging" smaller matrices; we formalize this procedure below. Let $\{B_i \mid i \in [m]\}$ be a set of matrices, where each B_i has row labels V_i, column labels C_i and is associated with a cluster-assignment $\varphi_i : V_i \to [k]$ and the following two *consistency conditions* hold:

- $\varphi_i(v) = \varphi_j(v)$ for every $v \in V_i \cap V_j$, $i, j \in [m]$,
- if $\varphi_i(v_i) = \varphi_j(v_j)$ for some $v_i \in V_i$ and $v_j \in V_j$, then $B_i[v_i, c] = B_j[v_j, c]$ for every $c \in C_i \cap C_j$.

Let $\varphi = \bigcup_{i \in [m]} \varphi_i$. We define the *composition* of (B_i, φ_i), $i \in [m]$, to be the matrix B^* with row labels $V = \bigcup_{i \in [m]} V_i$, column labels $C = \bigcup_{i \in [m]} C_i$ and entries as follows. For every $v \in V$ and $c \in C$, pick any $i \in [m]$ such that $c \in C_i$ and there exists $v_i \in V_i$ with $\varphi(v_i) = \varphi(v)$. We set $B^*[v, c_i] = B_i[v_i, c_i]$. If there is no such $i \in [m]$, we simply set $B^*[v, c]$ to some arbitrary but uniform default value—in our case, we will always use 0. Observe that if both consistency conditions hold, then both B^* and φ are well-defined and that φ is a cluster-assignment w.r.t. B^*.

Claim 1. *B^* is well-defined and admits the cluster-assignment φ.*

Proof. To see that B^* is well-defined, pick $v \in V$ and $c \in C_i \cap C_j$. If there exist $v_i \in V_i$ and $v_j \in V_j$ such that $\varphi(v_i) = \varphi(v)$ and $\varphi(v_j) = \varphi(v)$, then $\varphi(v_i) = \varphi(v_j)$, so $B_i[v_i, c] = B_j[v_j, c]$ by the consistency conditions. Moreover, φ is the cluster-assignment w.r.t. B^*. Indeed, consider $u, v \in V$ with $\varphi(u) = \varphi(v)$ and $c \in C$. If $c \in C_i$ for some $i \in [m]$ such that there exists $v_i \in V_i$ with $\varphi(v_i) = \varphi(v) = \varphi(u)$, then $B^*[v, c] = B_i[v_i, c] = B^*[u, c]$. If there is no such $i \in [m]$, we have $B^*[v, c] = 0 = B^*[u, c]$. □

Observe that if $C_i \cap C_j = \emptyset$ for every $i \neq j$, we can skip the second consistency condition: in this case it just requires $B_i[v] = B_i[u]$ whenever $\varphi_i(v) = \varphi_i(u)$, which holds as φ_i is the cluster-assignment w.r.t. B_i.

Parameterized Complexity. In parameterized algorithmics (Downey & Fellows, 2013; Cygan et al., 2015), the running-time of an algorithm is studied with respect to a parameter $k \in \mathbb{N}_0$ and input size n. The basic idea is to find a parameter that describes the structure of the instance

such that the combinatorial explosion can be confined to this parameter. In this respect, the most favorable complexity class is FPT (*fixed-parameter tractable*), which contains all problems that can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$, where f is a computable function. Algorithms with this running-time are called *fixed-parameter algorithms*.

Sparse Matrices. It follows from definition of MCME that entries of the input matrix A and the target matrix B are only relevant when the respective entry of the mask W is set to "1". We thus assume that only those entries are stored explicitly; in particular, the size of the input is therefore linear in the number of non-zero entries of W. All the structural parameters in our considerations automatically imply that the mask W is sparse, in the sense that it has at most $\mathcal{O}_k(n+d)$ non-zero entries, where the constant under $\mathcal{O}_k(\cdot)$ depends on the respective parameter k. All algorithms in this paper except one are linear-time in the number of non-zero entries of the mask, and the above allows us to state their running times in the form $(n+d) \cdot f(k)$, where f is a certain function of the respective parameter k.

Treewidth. A *nice tree-decomposition* $\mathcal{T}$ of a graph $G = (V, E)$ is a pair (T, χ), where T is a tree (whose vertices we call *nodes*) rooted at a node r and χ is a function that assigns each node t a set $\chi(t) \subseteq V$ such that:

- For every $uv \in E$ there is a node t where $u, v \in \chi(t)$.
- For every vertex $v \in V$, the set of nodes t satisfying $v \in \chi(t)$ forms a subtree of T.
- $|\chi(\ell)| = 1$ for every leaf ℓ of T and $|\chi(r)| = 0$.
- There are only three kinds of non-leaf nodes in T:
  - **Introduce node:** a node t with exactly one child t' such that $\chi(t) = \chi(t') \cup \{v\}$ for some vertex $v \notin \chi(t')$.
  - **Forget node:** a node t with exactly one child t' such that $\chi(t) = \chi(t') \setminus \{v\}$ for some vertex $v \in \chi(t')$.
  - **Join node:** a node t with two children t_1, t_2 such that $\chi(t) = \chi(t_1) = \chi(t_2)$.

The *width* of a nice tree-decomposition (T, χ) is $\max_{t \in V(T)}(|\chi(t)| - 1)$, and the *treewidth* of the graph G, denoted $\mathrm{tw}(G)$, is the minimum width of a nice tree-decomposition of G. We let T_t denote the subtree of T rooted at a node t, and we use $\chi_t^{\downarrow}$ to denote the set $\bigcup_{t' \in V(T_t)} \chi(t')$. The sets $\chi(t)$ are commonly called *bags*.

Fixed-parameter algorithms are known for computing nice tree-decompositions of optimal width and linearly many nodes (Bodlaender, 1996; Kloks, 1994), albeit more efficient fixed-parameter approximation algorithms are often used to achieve better running times (Bodlaender et al., 2016).

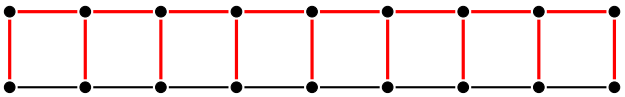

Figure 1. Example of a graph G with the spanning tree T (marked in thick red) such that $\mathrm{lfen}(G) = \mathrm{lfen}(G, T) = 2$. The feedback edge number of G, i.e., its edge deletion distance to acyclicity, is exactly the number of black edges and can be made arbitrarily large in this fashion while preserving $\mathrm{lfen}(G) = 2$.

Local Feedback Edge Number. The *local feedback edge number* was recently introduced by Ganian and Korchemna (2021) as an edge-cut based restriction of treewidth that acts as a "local" measure of the size of a feedback edge set; they used the parameter to obtain fixed-parameter algorithms for learning Bayesian networks and polytrees.

For a connected graph $G = (V, E)$ and a spanning tree T of G, the *local feedback edge set* at $x \in V$ is defined as $E^T_{\mathrm{loc}}(x) = \{yz \in E \setminus E(T) \mid$ the unique path between y and z in T contains $x\}$. The *local feedback edge number of* (G, T) (denoted $\mathrm{lfen}(G, T)$) is then equal to $\max_{x \in V} |E^T_{\mathrm{loc}}(x)|$, and the *local feedback edge number* of G is simply the smallest local feedback edge number among all possible spanning trees of G, i.e., $\mathrm{lfen}(G) = \min_{T \text{ is a spanning tree of } G} \mathrm{lfen}(G, T)$. An example of a graph G with $\mathrm{lfen}(G) = 2$ is provided in Figure 1.

3. Solving Bounded-Domain MCME by Exploiting the Mask

Since BOUNDED-DOMAIN MCME is NP-complete (Drineas et al., 2004; Aloise et al., 2009), it is natural to attempt and circumvent this lower bound by exploiting structural measures of the inputs to obtain fixed-parameter tractability. Eiben et al. (Eiben et al., 2021a) recently obtained a fixed-parameter approximation algorithm for BOUNDED-DOMAIN requiring the simultaneous parameterization by three measures: the number of clusters, the desired approximation ratio, and a technical measure that captures the sparsity of the missing data (i.e., the "0" entries in W).

In this section, we will present a fixed-parameter algorithm for BOUNDED-DOMAIN MCME which is aimed at the complementary case where the relevant data is sparse. To formalize this, we consider the *incidence graph* representation G_I of W which is defined as follows: $V(G_I) = V_W \cup C_W$, and $E(G_I) = \{ab \mid a \in V_W, b \in C_W, W[a, b] = 1\}$. Figure 2 later on can also be used as an example of the representation.

Intuitively, each "1" entry in W will correspond to an edge in G_I, and hence "structurally sparse" incidence graphs correspond to settings where most data is unknown. There is a well-studied hierarchy of structural graph parameters

which measure, in a certain sense, the sparsity of graphs (see, e.g., Figure 1 in (Bodlaender et al., 2013)). Treewidth will be our parameter of choice here, as the best known and also most general parameter in this hierarchy. However, it is worth noting that the use of the incidence graph may provide a new perspective on previous algorithms for clustering problems—for instance, the so-called *covering number* parameter (Ganian et al., 2018; Eiben et al., 2021b) is simply the vertex cover number of the complement of G_I.

The main goal of this section is to establish the fixed-parameter tractability of MCME parameterized by $\mathrm{tw}(G_I)$. As a first step towards this goal, we obtain a dynamic programming algorithm that handles the simpler case where k is also part of the parameterization.

Theorem 1. BOUNDED-DOMAIN MCME *is fixed-parameter tractable when parameterized by* $k + \mathrm{tw}(G_I)$.

Proof. We begin by applying the well-established 5-approximation algorithm for treewidth (Bodlaender et al., 2016) to compute a nice tree-decomposition of G_I of width $q \leq 5\,\mathrm{tw}(G_I)$ in time $2^{\mathcal{O}(\mathrm{tw}(G_I))}(n+d)$. Let r be the root of T. Given a node t of T, let V_t and C_t be the sets of vectors and coordinates in $\chi(t)$, respectively. Moreover, let $C_t^{\downarrow}$ and $V_t^{\downarrow}$ be the restrictions of $\chi_t^{\downarrow}$ to sets of coordinates and vectors correspondingly.

To prove the theorem, we will design a leaf-to-root dynamic programming algorithm which will compute and store a set of records at each node t of T, whereas once we ascertain the records for r we will have the information required to output a correct answer. Intuitively, the records will store the cluster centers restricted to the coordinates that appear in the bag, the partition of the vectors in the bag into clusters and the sum of minimum distances from vectors in $V_t^{\downarrow}$ to the cluster centers along the coordinates in $C_t^{\downarrow}$.

Formally, the records will have the following structure. We call a pair $(\mathrm{cent}, \mathrm{part})$ a *snapshot* in t if the following holds:

- $\mathrm{cent} : [k] \times C_t \to D$,
- $\mathrm{part} : V_t \to [k]$.

Let $S(t)$ be the set of all snapshots of t. The record $\mathcal{R}_t$ of t is then a mapping from $S(t)$ to the set $\mathbb{R}^+$ of non-negative reals. Observe that $|S(t)| \leq |D|^{k(q+1)}k^{q+1}$.

To introduce the semantics of our records, let $\mathcal{B}_t$ be the set of all matrices with row labels $V_t^{\downarrow}$, column labels $C_t^{\downarrow}$ and entries of domain D. Let B_t be a matrix in $\mathcal{B}_t$. We define the *partial weighted distance* from B_t to A in t as follows:

$$\mathrm{pwd}(B_t, t) = \sum_{v \in V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[v,c] \cdot (A[v,c] - B_t[v,c])^2.$$

For $B_t \in \mathcal{B}_t$, we say that $(\mathrm{cent}, \mathrm{part}) \in S(t)$ is the *snapshot of* B_t in t if there is a cluster-assignment φ w.r.t. B_t such that the following conditions hold:

- $\mathrm{part} = \varphi|_{V_t}$,
- for every $c \in C_t$ and $v \in V_t^{\downarrow}$, $B_t[v,c] = \mathrm{cent}[\varphi(v), c]$.

Recall that the existence of a cluster-assignment implies, in particular, that B_t has at most k distinct rows. We are now ready to define the record $\mathcal{R}_t$. For each snapshot $(\mathrm{cent}, \mathrm{part}) \in S(t)$, we set $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$ if there exists $B_t \in \mathcal{B}_t$ such that:

- $(\mathrm{cent}, \mathrm{part})$ is the snapshot of B_t in t,
- $\mathrm{pwd}(B_t, t) = \tau$, and
- $\forall B_t' \in \mathcal{B}_t$ such that $(\mathrm{cent}, \mathrm{part})$ is the snapshot of B_t', $\mathrm{pwd}(B_t', t) \geq \tau$.

In this case we say that B_t *witnesses* $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$.

Recall that for the root r, we assume $\chi(r) = \emptyset$. Hence $S(r)$ contains only one element $(\emptyset, \emptyset)$, and $\mathcal{R}_r(\emptyset, \emptyset)$ is equal to the minimum value of $\|W \circ (A - B)\|_F^2$ that can be achieved by any matrix B with entries of domain D containing at most k distinct rows. In other words, the instance is a YES-instance if and only if $\mathcal{R}_r(\emptyset, \emptyset) \leq \ell$. To prove the theorem, it now suffices to show that the records can be computed in a leaf-to-root fashion by proceeding along the nodes of T.

We distinguish the following cases:

t is a leaf node. Let $\chi(t) = \{v\}$ where v is a vector. By definition, $S(t) = \{(\emptyset, \mathrm{part}) | \, \mathrm{part} : [1] \to [k]\}$ and $\mathcal{R}_t(\emptyset, \mathrm{part}) = 0$ for every $(\emptyset, \mathrm{part}) \in S(t)$ as there are no coordinates to sum over. Let $\chi(t) = \{c\}$ for some coordinate c. Then $S(t) = \{(\mathrm{cent}, \emptyset) | \, \mathrm{cent} : [k] \times [1] \to D\}$ and $\mathcal{R}_t(\mathrm{cent}, \emptyset) = 0$ for each $(\mathrm{cent}, \emptyset) \in S(t)$.

t is a forget node. Let t' be the child of t in T and $\chi(t) = \chi(t') \setminus \{v\}$ for some vector v. We set $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) := \min_{i \in [k]} \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i))$ for each $(\mathrm{cent}, \mathrm{part}) \in S(t)$. For correctness, it will be useful to observe that $\mathcal{B}_t = \mathcal{B}_{t'}$. If $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$, then there exists a matrix B_t which witnesses this. But then B_t also admits a snapshot $(\mathrm{cent}, \mathrm{part} \cup (v,i))$ at t' for some $i \in [k]$ and witnesses $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i)) \leq \tau$. So in our algorithm $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) \leq \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i)) \leq \tau$. If on the other hand $\mathcal{R}_t^0(\mathrm{cent}, \mathrm{part}) = \tau$, then there exists a snapshot $(\mathrm{cent}, \mathrm{part} \cup (v,i))$ at t' for some $i \in [k]$ such that $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i)) = \tau$. $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$ now follows from the existence of a matrix witnessing the value of $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part} \cup (v,i))$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}_t^0$.
If $\chi(t) = \chi(t') \setminus \{c\}$ for some coordinate c, we set $\mathcal{R}_t(\mathrm{cent}, \mathrm{part})$ equal to $\min\{\mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) | \, \mathrm{cent} \text{ is a restriction of } \mathrm{cent}' \text{ to all coordinates except } c\}$. Correctness can be argued similarly to the case of a forgotten vector.

t is an introduce node (introducing a vector). Let t' be

the child of t in T and let $\chi(t) = \chi(t') \cup \{v_0\}$ for some vector v_0. Fix a snapshot $(\mathrm{cent}, \mathrm{part})$ in $S(t)$, let $i = \mathrm{part}(v_0)$ and $\mathrm{part}' = \mathrm{part} \setminus (v_0, i)$. We will denote by Δ_0 the sum of distances from v_0 to the i-th cluster center along the coordinates in C_t, i.e., $\Delta_0 = \sum_{c \in C_t} W[v_0, c] \cdot (A[v_0, c] - \mathrm{cent}[i, c])^2$. We set $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0$. For correctness, assume that $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \tau = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0$. Construct a matrix B_t from the witness $B_{t'}$ of $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}')$ with a cluster-assignment φ' by adding a new row with a label v_0 as follows. If $i = \varphi'(v)$ for some $v \in V^{\downarrow}_{t'}$, we set $B_t[v_0] := B_t[v]$. Otherwise we define $B_t[v_0, c] := \mathrm{cent}[i, c]$ for $c \in C_t$ and $B_t[v_0, c] := 0$ in the rest of coordinates c. Note that in both cases $B_t[v_0, c] = \mathrm{cent}[i, c]$ for every $c \in C_t$: in the second case it follows from the definition, while in the first one we have $B_t[v_0, c] := B_t[v, c] = B_{t'}[v, c] = \mathrm{cent}[\varphi'(v), c] = \mathrm{cent}[i, c]$. The matrix B_t with a cluster-assignment $\varphi = \varphi' \cup (v_0, i)$ has a snapshot $(\mathrm{cent}, \mathrm{part})$ in t and $\mathrm{pwd}(B_t, t) = \sum_{v \in V^{\downarrow}_t} \sum_{c \in C^{\downarrow}_t} W[v, c] \cdot (A[v, c] - B_t[v, c])^2 = \mathrm{pwd}(B_{t'}, t') + \sum_{c \in C^{\downarrow}_t} W[v_0, c] \cdot (A[v_0, c] - B_t[v_0, c])^2$. As $\mathrm{pwd}(B_{t'}, t') = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}')$, $W[v_0, c] = 0$ for every forgotten coordinate c and $B_t[v_0, c] := \mathrm{cent}[i, c]$ for every $c \in C_t$, we have $\mathrm{pwd}(B_t, t) = \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0 = \tau$. So B_t witnesses that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$.

On the other hand, assume that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$. Then there exists a matrix B_t in $\mathcal{B}_t$ admitting the snapshot $(\mathrm{cent}, \mathrm{part})$ in t such that $\mathrm{pwd}(B_t, t) = \tau$. Let φ be the corresponding cluster-assignment w.r.t. B_t. We construct $B_{t'}$ from B_t by deletion of the row v_0. Then $B_{t'}$ with cluster-assignment $\varphi' = \varphi \setminus \{(v_0, i)\}$ has a snapshot $(\mathrm{cent}, \mathrm{part}')$ in t and witnesses that $\mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') \leq \mathrm{pwd}(B_{t'}, t') = \tau - \Delta_0$. Therefore in our algorithm $\mathcal{R}^0_t \leq \mathcal{R}_{t'}(\mathrm{cent}, \mathrm{part}') + \Delta_0 \leq \tau$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}^0_t$.

t is an introduce node (introducing a coordinate). Let t' be the child of t in T and let $\chi(t) = \chi(t') \cup \{c_0\}$ for some coordinate c_0. Fix a snapshot $(\mathrm{cent}, \mathrm{part})$ in $S(t)$, we wil denote by Δ_0 the sum of distances from $v \in V^{\downarrow}_t$ to the corresponding cluster center along the coordinate c_0, i.e., $\Delta_0 = \sum_{v \in V_t} W[v, c_0] \cdot (A[v, c_0] - \mathrm{cent}[\mathrm{part}(v), c_0])^2$. Let cent' be the restriction of cent to all the coordinates except c_0. We set $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0$. For correctness, assume that $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \tau = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0$. We construct a matrix B_t from a witness $B_{t'}$ of $\mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part})$ with cluster-assignment φ' by adding a column c_0. For every $v \in V^{\downarrow}_t$, we set $B_t[v, c] = B_{t'}[v, c]$ for all $c \in C^{\downarrow}_{t'}$ and $B_t[v, c_0] = \mathrm{cent}[\varphi'(v), c_0]$. Then B_t with the same cluster-assignment φ' has a snapshot $(\mathrm{cent}, \mathrm{part})$ in t and $\mathrm{pwd}(B_t, t) = \mathrm{pwd}(B_{t'}, t') + \Delta_0 = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0 = \tau$. Therefore B_t witnesses that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$. On the other hand, assume that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$. Then there exists a matrix B_t in $\mathcal{B}_t$ with the snapshot $(\mathrm{cent}, \mathrm{part})$ in t such that $\mathrm{pwd}(B_t, t) = \tau$. Let $B_{t'}$ be obtained from B_t by deletion of the column with the label c_0. Then $B_{t'}$ witnesses $\mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) \leq \tau - \Delta_0$, so in our algorithm $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t'}(\mathrm{cent}', \mathrm{part}) + \Delta_0 \leq \tau$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}^0_t$.

t is a join node. Let t_1, t_2 be the two children of t in T, recall that $\chi(t_1) = \chi(t_2) = \chi(t)$ and $\chi^{\downarrow}_{t_1} \cap \chi^{\downarrow}_{t_2} = \chi(t)$. For every $(\mathrm{cent}, \mathrm{part})$ in $S(t)$ we set $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t_1}(\mathrm{cent}, \mathrm{part}) + \mathcal{R}_{t_2}(\mathrm{cent}, \mathrm{part}) - \mathrm{doublecount}$, where $\mathrm{doublecount} = \sum_{v \in V_t} \sum_{c \in C_t} W[v, c] \cdot (A[v, c] - \mathrm{cent}[\mathrm{part}(v), c])^2$.

For correctness, assume that $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \tau = \tau_1 + \tau_2 - \mathrm{doublecount}$, where $\tau_1 = \mathcal{R}_{t_1}(\mathrm{cent}, \mathrm{part})$, $\tau_2 = \mathcal{R}_{t_2}(\mathrm{cent}, \mathrm{part})$. Let B_i with the cluster-assignment φ_i be the witness of $\mathcal{R}_{t_i}(\mathrm{cent}, \mathrm{part}) = \tau_i$, $i = 1, 2$. We obtain a matrix B_t with cluster-assignment $\varphi = \varphi_1 \cup \varphi_2$ as a composition of (B_1, φ_1) and (B_2, φ_2). To check the consistency conditions, observe that the sets of common row and column labels of B_1 and B_2 are V_t and C_t correspondingly. Recall that $\varphi_1|_{V_t} = \mathrm{part} = \varphi_2|_{V_t}$. Moreover, if $v_1 \in V_1$ and $v_2 \in V_2$ are such that $\varphi_1(v_1) = \varphi_2(v_2)$, then we have $B_1[v_1, c] = \mathrm{cent}[\varphi_1(v_1), c] = \mathrm{cent}[\varphi_2(v_2), c] = B_2[v_2, c]$ for every $c \in C_t$.

Note that for every $v \in V_t$, $\varphi(v) = \varphi_1(v) = \mathrm{part}$. Pick $c \in C_t$ and $v \in V^{\downarrow}_t$, then $v \in V^{\downarrow}_{t_i}$ for some $i \in \{1, 2\}$ and so $B_t[v, c] = B_i[v, c] = \mathrm{cent}[\varphi_i(v), c] = \mathrm{cent}[\varphi(v), c]$. Therefore $(\mathrm{cent}, \mathrm{part})$ is a snapshot of B_t in t. Recall that $\mathrm{pwd}(B_t, t) = \sum_{v \in V^{\downarrow}_t} \sum_{c \in C^{\downarrow}_t} W[v, c] \cdot (A[v, c] - B_t[v, c])^2$. Here $W[v, c] = 0$ for every $v \in V^{\downarrow}_{t_i}$ and $c \notin C^{\downarrow}_{t_i}$, $i = 1, 2$. So $\mathrm{pwd}(B_t, t) = \sum_{v \in V^{\downarrow}_{t_1}} \sum_{c \in C^{\downarrow}_{t_1}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2 + \sum_{v \in V^{\downarrow}_{t_2}} \sum_{c \in C^{\downarrow}_{t_2}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2 - \sum_{v \in V_t} \sum_{c \in C_t} W[v, c] \cdot (A[v, c] - B_t[v, c])^2 = \mathrm{pwd}(B_1, t_1) + \mathrm{pwd}(B_2, t_2) - \mathrm{doublecount} = \tau_1 + \tau_2 - \mathrm{doublecount} = \tau$. Hence B_t witnesses $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) \leq \tau$.

For the converse, assume that $\mathcal{R}_t(\mathrm{cent}, \mathrm{part}) = \tau$ and B_t is a matrix witnessing this. Let B_i be the restriction of B_t to rows $V^{\downarrow}_{t_i}$ and columns $C^{\downarrow}_{t_i}$, $i = 1, 2$. Then B_1 and B_2 have a snapshot $(\mathrm{cent}, \mathrm{part})$ and in our algorithm $\mathcal{R}^0_t(\mathrm{cent}, \mathrm{part}) = \mathcal{R}_{t_1}(\mathrm{cent}, \mathrm{part}) + \mathcal{R}_{t_2}(\mathrm{cent}, \mathrm{part}) - \mathrm{doublecount} \leq \mathrm{pwd}(B_1) + \mathrm{pwd}(B_2) - \mathrm{doublecount} = \mathrm{pwd}(B) = \tau$. Hence the resulting record $\mathcal{R}_t = \mathcal{R}^0_t$ is correct, which concludes the correctness proof of the algorithm.

To bound the runtime of the algorithm, observe that at each node t we compute the record $\mathcal{R}_t$ for $|S(t)| \leq |D|^{kq} k^q = |D|^{\mathcal{O}(k \cdot \mathrm{tw}(G_I))}$ entries, where each entry is computed in time at most quadratic in $\mathrm{tw}(G_I)$. Since the tree-decomposition is nice and has at most $\mathcal{O}(n + d)$

nodes, the runtime of the algorithm is upper-bounded by $(n+d)\cdot|D|^{\mathcal{O}(k\cdot\mathrm{tw}(G_I))}$. □

Towards proving fixed-parameter tractability without involving k, we consider the case where k is significantly larger than $\mathrm{tw}(G_I)$. We prove that this case always admits a "perfect" solution, which can also be computed efficiently.

Lemma 2. *If $k \geq (\mathrm{tw}(G_I)+1)\cdot|D|^{\mathrm{tw}(G_I)+1}$ then there exists a matrix B over domain D containing at most k distinct rows such that $\|W\circ(A-B)\|_F^2 = 0$. Moreover, given a nice tree-decomposition of width $q-1$ such that $k \geq q\cdot|D|^q$, such a matrix B can be computed in time $(n+d)\cdot|D|^{\mathcal{O}(q)}$.*

Proof. We will prove the second claim of the lemma, from which the first one follows. To this end, let us assume that we are given a nice tree-decomposition $\mathcal{T}=(T,\chi)$ of G_I of width $q-1$ and $k \geq q\cdot|D|^q$. We denote the root of T by r and define C_t, V_t, $C_t^\downarrow$ and $V_t^\downarrow$ analogously to the proof of Theorem 1. We will explicitly construct $q\cdot|D|^q$ cluster centers yielding a zero sum of squares, so that after processing the node t, each of the centers is defined on coordinates $c \in C_t^\downarrow$. The remaining clusters will be empty, thus for the rest of the proof we assume $k = q\cdot|D|^q$.

Formally, we will attach to every node t the pair (R_t,α_t) where R_t is the matrix with the row labels $[k]$ and the column labels $C_t^\downarrow$, and α_t is a mapping from $V_t^\downarrow$ to $[k]$. Intuitively, R_t stores the restrictions of cluster centers to processed coordinates, while α_t assigns a cluster index to every row label $v \in V_t^\downarrow$. Throughout the dynamic programming procedure, we will maintain the following invariants:

1. $\alpha_t(u) \neq \alpha_t(v)$ for any two distinct u and v from V_t,
2. for every $\omega \in D^{C_t}$, there are precisely $\frac{k}{|D|^{|C_t|}}$ indices $i \in [k]$ such that $R_t[i][C_t] = \omega$, where $R_t[i][C_t]$ is the restriction of $R_t[i]$ to the coordinate set C_t,
3. $\sum_{v\in V_t^\downarrow}\sum_{c\in C_t^\downarrow} W[v,c]\cdot(A[v,c]-R_t[\alpha_t(v),c])^2 = 0$.

Assume that the invariants hold in the root r, let us define the matrix B by setting $B[v] = R_r[\alpha_r(v)]$. Then B has at most k distinct rows and the third invariant yields that $\sum_{v\in V_A}\sum_{c\in C_A} W[v,c]\cdot(A[v,c]-B[v,c])^2 = 0$. In particular, for $q = \mathrm{tw}(G_I)+1$ we get the statement of the lemma. Hence it is sufficient to construct (R_t,α_t) in a leaf-to-root fashion, for each of the following cases.

t is a leaf node. If $\chi(t) = \{v_0\}$ for some row label v_0, we set $\alpha_t(v_0) = 1$ and $R_t[i] = \emptyset$ for every $i \in [k]$. Otherwise, $\chi(t) = \{c\}$ for a coordinate c. Then α_t should be the empty mapping, for every $i \in [k]$ we set $R_t[i,c]$ equal to some $\gamma \in D$ so that every $\gamma \in D$ appears in R_t precisely $\frac{k}{|D|}$ times. It is easy to see that all the invariants are satisfied.

t is a forget node. Let t' be the child of t in T, we keep $R_t = R_{t'}$ and $\alpha_t = \alpha_{t'}$. The first and third invariants are obviously preserved. In case $\chi(t) = \chi(t')\setminus\{c_0\}$ for a coordinate c_0, we have $C_{t'} = C_t \cup \{c_0\}$. By the second invariant for t', for every $\gamma \in D$ and $\omega \in D^{C_t}$ the number of $i \in [k]$ with $R_{t'}[i][C_t] = \omega$ and $R_{t'}[i][c_0] = \gamma$ is $\frac{k}{|D|^{|C_{t'}|}}$. Therefore the number of $i \in [k]$ with $R_t[i][C_t] = \omega$ is precisely $|D|\cdot\frac{k}{|D|^{|C_{t'}|}} = \frac{k}{|D|^{|C_t|}}$.

t is an introduce node (introducing a vector). Let t' be the child of t in T and $\chi(t) = \chi(t')\cup\{v_0\}$ for some row label v_0. We then set $R_t = R_{t'}$ and $\alpha_t = \alpha_{t'}\cup(v_0,i_0)$ where $i_0 \in [k]$ is such that $R_{t'}[i_0][C_t] = A[v_0][C_t]$ and $\alpha_{t'}(v) \neq i_0$ for any $v \in V_{t'}$. Such an index always exists: by the second invariant for t' there are $\frac{k}{|D|^{|C_t|}} \geq \frac{k}{|D|^q} = q > |V_{t'}|$ indices i such that $R_{t'}[i][C_t] = A[v_0][C_t]$, so at least one of them is not equal to any $\alpha_{t'}(v)$, $v \in V_{t'}$. As the third invariant holds in t', we have $\sum_{v\in V_t^\downarrow}\sum_{c\in C_t^\downarrow} W[v,c]\cdot(A[v,c]-R_t[\alpha_t(v),c])^2 = \sum_{c\in C_t^\downarrow} W[v_0,c]\cdot(A[v_0,c]-R_t[\alpha_t(v_0),c])^2$. Here $W[v_0,c] = 0$ for every $c \in C_t^\downarrow\setminus C_t$ because such c never appears in the same bag with v_0. Moreover, for $c \in C_t$, $R_t[\alpha_t(v_0),c] = R_t[i_0,c] = A[v_0,c]$, so every summand is equal to zero.

t is an introduce node (introducing a coordinate). In case if t' is the child of t in T and $\chi(t) = \chi(t')\cup\{c_0\}$ for some coordinate c_0, we set $\alpha_t = \alpha_{t'}$, $R_t[i][C_{t'}] = R_{t'}[i][C_{t'}]$ for every $i \in [k]$ and define R_t in the column c_0 as follows. For every vector $\omega \in D^{C_{t'}}$, let g^ω be the set of indices $i \in [k]$ such that $R_{t'}[i][C_{t'}] = \omega$. Recall that by the second invariant for t', $|g^\omega| = \frac{k}{|D|^{|C_t-1|}}$. We subdivide g^ω into $|D|$ groups $\{g^\omega_\gamma \mid \gamma \in D\}$ of the same size $\frac{k}{|D|^{|C_t|}}$ so that if $\alpha_t(v) \in g^\omega$ for some $v \in V_t$ then $\alpha_t(v) \in g^\omega_\gamma$ with $\gamma = A[v,c_0]$. It is always possible as α_t is injective on V_t (so the index γ is determined uniquely for $\alpha_t(v) \in g^\omega$) and $\frac{k}{|D|^{|C_t|}} \geq \frac{k}{|D|^q} = q \geq |V_t|$ (so potentially one of the groups can contain $\alpha_t(v)$ for all $v \in V_t$). For every $\gamma \in D$ and $i \in g^\omega_\gamma$ we then set $R_t[i,c_0] = \gamma$ and by this satisfy the second invariant in t. The following validates the third one: $\sum_{v\in V_t^\downarrow}\sum_{c\in C_t^\downarrow} W[v,c]\cdot(A[v,c]-R_t[\alpha_t(v),c])^2 = \sum_{v\in V_t^\downarrow} W[v,c_0]\cdot(A[v,c_0]-R_t[\alpha_t(v),c_0])^2 = \sum_{v\in V_t} W[v,c_0]\cdot(A[v,c_0]-R_t[\alpha_t(v),c_0])^2$. Fix $v \in V_t$, let $\gamma = A[v][C_{t'}]$. Then $\alpha_t(v) \in g^\omega$, so $\alpha_t(v) \in g^\omega_\gamma$ with $\gamma = A[v,c_0]$. Therefore $R_t[\alpha_t(v),c_0] = \gamma = A[v,c_0]$, i.e., every summand is equal to zero.

t is a join node. Let t_1,t_2 be the two children of t in T, recall that $\chi(t_1) = \chi(t_2) = \chi(t)$. By the well-known separation property of tree-decompositions, $\chi^\downarrow_{t_1}\cap\chi^\downarrow_{t_2} = \chi(t)$ (Downey & Fellows, 2013; Cygan et al., 2015). Without loss of generality we may assume that α_{t_1} and α_{t_2} coincide on V_t and $R_{t_1}[i,c] = R_{t_2}[i,c]$ for every $i \in [k]$ and

$c \in C_t$. In the case it does not hold, we achieve it by permuting the rows of R_{t_2}: by the second invariant, the sets of rows R_{t_1} and R_{t_2} are the same when restricted to C_t, thus there is a permutation that achieves $R_{t_1}[i, c] = R_{t_2}[i, c]$ for every $i \in [k]$ and $c \in C_t$. Furthermore, by the third invariant, for each $v \in V_t$, α_{t_2} maps v to a row that coincides with v on C_t. Together with the first invariant it means that for each $\omega \in D^{C_t}$, α_{t_2} acts injectively from $V_t^\omega = \{v \in V_t : A[v][C_t] = \omega\}$ to the set of rows of R_{t_2} that coincide with ω on C_t. Since the same holds for α_{t_1} and R_{t_1}, for each $\omega \in D^{C_t}$ there is a permutation on the rows of R_{t_2} equal to ω on C_t that makes α_{t_1} and α_{t_2} agree on V_t^ω. Combinining the above permutations, applying the resulting permutation π_t of $[k]$ on the rows of R_{t_2}, and replacing α_{t_2} with $\pi_t \circ \alpha_{t_2}$ leads to the desired property.

Then, we can correctly define $\alpha_t = \alpha_{t_1} \cup \alpha_{t_2}$ and construct R_t as follows:

- for every $i \in [k]$ and $c \in C_{t_1}^\downarrow$, $R_t[i, c] = R_{t_1}[i, c]$,
- for every $i \in [k]$ and $c \in C_{t_2}^\downarrow$, $R_t[i, c] = R_{t_2}[i, c]$.

(R_t, α_t) obviously satisfies the first and second invariants, let us verify the third one. Note that $W[v, c] = 0$ for every $v \in V_{t_j}^\downarrow$ and $c \notin C_{t_j}^\downarrow$, $j = 1, 2$. Therefore $\sum_{v \in V_t^\downarrow} \sum_{c \in C_t^\downarrow} W[v, c] \cdot (A[v, c] - R_t[\alpha_t(v), c])^2 \leq \sum_{v \in V_{t_1}^\downarrow} \sum_{c \in C_{t_1}^\downarrow} W[v, c] \cdot (A[v, c] - R_{t_1}[\alpha_{t_1}(v), c])^2 + \sum_{v \in V_{t_2}^\downarrow} \sum_{c \in C_{t_2}^\downarrow} W[v, c] \cdot (A[v, c] - R_{t_2}[\alpha_{t_2}(v), c])^2 = 0 + 0 = 0$ as desired.

We now estimate the running time of our procedure. To show the desired time bound that is linear in terms of $n + d$, we further specify how the entries (R_t, α_t) are stored. For a node t, we store separately the mapping α_t on the forgotten row indices $V_t^\downarrow \setminus V_t$ as a linked list of pairs $(v, \alpha_t(v))$ where $v \in V_t^\downarrow \setminus V_t$, and separately the values of α on the row indices of V_t; for the V_t part, the precise method of storage is not relevant for our desired running time bound as there are at most q such entries in each node. Analogously, for each $i \in [k]$, the row $R_t[i]$ is stored separately on the columns of C_t and $C_t^\downarrow \setminus C_t$, where the latter is stored as a linked list of pairs $(c, R_t[i, c])$, $c \in C_t^\downarrow \setminus C_t$.

With the storage method above in mind, observe first that for all nodes except join nodes the manipulations on the entries in the bag take time at most $\mathcal{O}(kq)$, while the entries for the forgotten row and column indices are simply copied from the child node. Since the entry at a child node is never reused further in the algorithm, the corresponding linked lists can be copied by reference in constant time each.

For a join node, finding a suitable permutation of R_{t_2} that matches with R_{t_1} on C_t takes time $\mathcal{O}(k^2 q)$ by comparing the values in the columns of C_t for each pair of rows from R_{t_1} and R_{t_2}, and the same time bound covers tweaking this permutation to obtain π_t that also makes α_{t_1} and α_{t_2} agree on V_t. Applying π_t to rearrange the rows of R_{t_2} takes time $\mathcal{O}(kq)$ since we pass the representations of $R_{t_2}[i][C_t^\downarrow \setminus C_t]$ by reference. Now, to compose π_t with α_{t_2}, instead of computing the image value-by-value which would take linear time, we do the following constant-time deferred application. For the linked list that stores the values of α_{t_2} on $V_t^\downarrow \setminus V_t$, we consider another type of node that contains a single permutation π_t of $[k]$. This node in interpreted as follows: any pair (v, i) that comes after this node is deemed to represent the pair $(v, \pi_t(i))$ instead. Using permutation nodes, π_t can by applied to α_{t_2} in $\mathcal{O}(k + q)$ by computing the values on V_t explicitly and putting a permutation node π_t at the beginning of the linked list for $V_t^\downarrow \setminus V_t$, and another permutation node π_t^{-1} at the end of this list. Finally, the union of α_{t_1} and α_{t_2} is computed in time $\mathcal{O}(q)$ by appending the linked list that stores the images of α_{t_2} on $V_{t_2}^\downarrow \setminus V_{t_2}$ to the linked list of α_{t_1} on $V_{t_1}^\downarrow \setminus V_{t_1}$, and storing the values on V_t separately. In the same fashion, for each $i \in [k]$ we construct the row $R_t[i]$ by concatenating the respective linked lists of $R_{t_1}[i]$ and $R_{t_2}[i]$.

After processing the root node r, the entries of B can be restored efficiently in the following way. Recall that at most $\mathcal{O}((n + d) \cdot q)$ entries of W are "1", as the graph G_I has a tree-decomposition of width $q - 1$; only the corresponding $\mathcal{O}((n + d) \cdot q)$ entries of B are relevant, others need not to be restored. First, in time $\mathcal{O}(kd)$ we retrieve the rows of R_r from the computed representation and store each row as an array indexed by the column indices. The cluster assignment can be retrieved from the representation of α_r in time $\mathcal{O}((n + d) \cdot k)$ by traversing through the corresponding linked list while maintaining the composition of all occured permutation nodes (observe that at most $2(n+d)$ permutation nodes were created throughout the execution). Afterwards each relevant entry $B[v, c]$ can be computed in constant time, by retrieving the cluster assignment $\alpha_r(v)$ of v and the value $R_r[\alpha_r(v), c]$ from the respective row array. Since the number of nodes in the tree-decomposition is bounded by $\mathcal{O}(n + d)$ and processing of each node takes time $\mathcal{O}(k^2 q)$, the total running time is thus bounded by

$$\mathcal{O}((n + d) \cdot k^2 q) = (n + d) \cdot |D|^{\mathcal{O}(q)},$$

as desired. □

At this point, we can prove the main result of this section.

Theorem 3. BOUNDED-DOMAIN MCME *is fixed-parameter tractable when parameterized by* $\mathrm{tw}(G_I)$.

Proof. Given an instance of BOUNDED-DOMAIN MCME, we begin by constructing the incidence graph G_I and then applying the known 5-approximation algorithm to compute a tree-decomposition of G_I of width $q - 1 \leq 5\,\mathrm{tw}(G_I)$

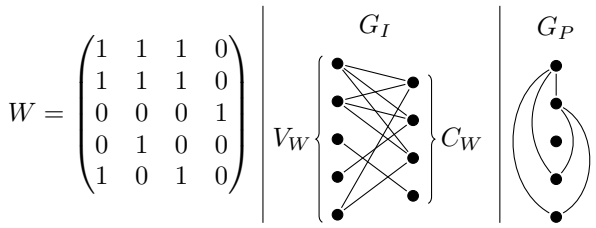

Figure 2. Example of a mask W (left) together with its incidence graph (middle) and its primal graph (right). Vertices from top to bottom correspond to rows from top to bottom and in the case of C_W columns from left to right. In this example $\mathrm{tw}(G_I) = 3$ and $\mathrm{tw}(G_P) = 2$.

in time $2^{\mathcal{O}(\mathrm{tw}(G_I))} \cdot (n + d)$. At this point we proceed by comparing q and k, as follows:

- if $k \geq q \cdot |D|^q$, then we can correctly output "Yes" (along with a suitable witness) by using Lemma 2;
- on the other hand, if $k \leq q \cdot |D|^q$ then we instead invoke Theorem 1 to solve the instance.

The running time in the former case is upper-bounded by $(n + d) \cdot |D|^{\mathcal{O}(\mathrm{tw}(G_I))}$, while in the latter case the bound is $(n + d) \cdot |D|^{\mathcal{O}(k \cdot \mathrm{tw}(G_I))} \leq (n + d) \cdot |D|^{|D|^{\mathcal{O}(\mathrm{tw}(G_I))}}$. □

4. Handling Real-Valued Matrices with Treewidth

We now turn our attention to REAL-VALUED MCME. As was mentioned in the Introduction, a fixed-parameter algorithm for the problem parameterized by the incidence treewidth of W would resolve a long-standing open problem as a special case, and so far remains beyond reach. However, we can still make tangible progress on the problem by considering the treewidth of a different representation of W. In particular, the *primal graph* G_P of W is a graph containing one vertex for each row of W, where two rows a, b are adjacent if and only if there is a column e such that $W[a, e] = W[b, e] = 1$. An example comparing primal and incidence graphs is provided in Figure 2.

Our aim in this section is to prove the next theorem.

Theorem 4. REAL-VALUED MCME *is fixed-parameter tractable when parameterized by* $\mathrm{tw}(G_P)$.

Before we proceed to the proof, we will first introduce some additional notation that will be useful in the context of real-valued matrices. For every $V' \subseteq V_A$, let $\mu(V', c) = \mathrm{argmin}_x f(x)$ for $f(x) = \sum_{v \in V'} W[v, c] \cdot (A[v, c] - x)^2$ and let $\Delta_c(V') = f(\mu(V', c))$. Intuitively, if vectors from V' form a cluster then $\Delta_c(V')$ is a minimal sum of distances to the cluster center along the coordinate c.

Observation 5. *For each coordinate c, if $V'' = \{z \in V' \mid W[z, c] = 1\} \neq \emptyset$ then*

$$\mu(V', c) = \frac{\sum_{v \in V''} A[v, c]}{|V''|}, \text{ and}$$

$$\Delta_c(V') = \sum_{v \in V''} (A[v, c])^2 - \frac{(\sum_{v \in V''} A[v, c])^2}{|V''|}.$$

Proof. As W is a binary matrix, we can rewrite $f(x) = \sum_{v \in V''} (A[v, c] - x)^2 = |V''|x^2 - 2 \sum_{v \in V''} A[v, c] \cdot x + \sum_{v \in V''} (A[v, c])^2$. By assumption, $|V''| \neq 0$, so $f(x)$ is a quadratic function and attains a minimum at an extreme point, i.e., at $x = \frac{\sum_{v \in V''} A[v,c]}{|V''|}$. The minimal value of f can be computed as $\sum_{v \in V''} (A[v, c])^2 - \frac{(\sum_{v \in V''} A[v,c])^2}{|V''|}$. □

Now we are ready to present the proof of Theorem 4.

Proof of Theorem 4. As our initial step, we once again apply the known approximation algorithm (Bodlaender et al., 2016) to compute a nice tree-decomposition (T, χ) of G_P of width $q \leq 5\,\mathrm{tw}(G_P)$. We keep the notations r, V_t, and $V_t^{\downarrow}$ from the previous section. Note that now the bags in the tree-decomposition contain only vectors, so $V_t = \chi(t)$. In contrast to the previous section, for every node t we denote by $C_t^{\downarrow}$ the set of all *processed* in t coordinates, i.e., coordinates c such that $W[v_c, c] \neq 0$ for some $v_c \in V_t^{\downarrow} \setminus V_t$. A crucial observation is the following: if $c \in C_t^{\downarrow}$, then for all vectors $v \notin V_t^{\downarrow}$ it holds that $W[v, c] = 0$, otherwise v and v_c would appear in the same bag. In other words, values of the cluster centers on $c \in C_t^{\downarrow}$ are not meaningful for vectors introduced outside of T_t.

We will design a leaf-to-root dynamic programming algorithm which computes a set of records $\mathcal{R}_t$ at each node t of T. For each way p of partioning the vectors in the bag V_t into at most k clusters, the record $\mathcal{R}_t$ stores the minimum cost of clustering the vectors of $V_t^{\downarrow}$ in the coordinates of $C_t^{\downarrow}$, considering only the partitions of $V_t^{\downarrow}$ that extend p.

Formally, for a node t of T let $\mathcal{P}_t$ be the set of all equivalence relations $p \subseteq V_t \times V_t$ with at most k equivalence classes. The record $\mathcal{R}_t$ of t is a mapping from $\mathcal{P}_t$ to $\mathbb{R}^+$. Observe that $|\mathcal{P}_t| \leq (q + 1)^{q+1}$.

To introduce the semantics of our records, let $\mathcal{B}_t$ be the set of all real matrices with the row labels $V_t^{\downarrow}$ and the column labels $C_t^{\downarrow}$. Let B_t be a matrix in $\mathcal{B}_t$. We define the *partial weighted distance* from B_t to A in t as follows:

$$\mathrm{pwd}(B_t, t) = \sum_{v \in V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2.$$

We also denote the number of *forgotten clusters* by $\mathrm{fcl}(B_t, t)$, that is, the number of distinct rows in B_t not

equal to any $B_t[v]$, $v \in V_t$. We say that $p \in \mathcal{P}_t$ with $|[p]| \leq k$ equivalence classes is the *t-partition* of B_t if the following conditions hold:

- for every $v, w \in V_t$ such that $(v, w) \in p$, $B_t[v] = B_t[w]$,
- $\mathrm{fcl}(B_t, t)$ is at most $k - |[p]|$.

The last two conditions are in fact equivalent to existence of a cluster-assignment φ w.r.t. B_t such that for every $u, v \in V_t$, $\varphi(u) = \varphi(v)$ if and only if $(u, v) \in p$.

We are now ready to define the record $\mathcal{R}_t$. For each $p \in \mathcal{P}_t$, we set $\mathcal{R}_t(p) = \tau$ if there exists $B_t \in \mathcal{B}_t$ such that:

- p is the t-partition of B_t,
- $\mathrm{pwd}(B_t, t) = \tau$, and
- $\forall B'_t \in \mathcal{B}_t$ such that p is the t-partition of B'_t, $\mathrm{pwd}(B'_t, t) \geq \tau$.

In this case we say that B_t *witnesses* that $\mathcal{R}_t(p) = \tau$. Observe that $\mathcal{R}_t(p) < \infty$ since any $p \in \mathcal{P}_t$ is a t-partition of the zero matrix, and that $\mathcal{R}_t(p)$ is well-defined (in the sense that $\inf_{B_t \in \mathcal{B}_t} \mathrm{pwd}(B_t, t)$ is achieved on $\mathcal{B}_t$), since $\mathrm{pwd}(B_t, t)$ is minimized on a matrix constructed by Observation 5 from an optimal cluster-assignment, and there is only a finite number of those.

Recall that for the root r of T, we assume $V_r = \emptyset$. Hence $\mathcal{P}_r$ contains only one element $\emptyset$, and $\mathcal{R}_r(\emptyset)$ is equal to the minimum value of $\|W \circ (A - B)\|_F^2$ that can be achieved by any real matrix B containing at most k distinct rows. Indeed, if some coordinate c is not processed in r, then $W[v, c] = 0$ for every row label v. Let us extend a witness B_r of $\mathcal{R}_r$ to a matrix B with column set C_A by setting $B[v, c] = 0$ for every row label v and every $c \notin C_r^{\downarrow}$. Then $\mathcal{R}_r(\emptyset) = \sum_{v \in V_r^{\downarrow}} \sum_{c \in C_r^{\downarrow}} W[v, c] \cdot (A[v, c] - B_r[v, c])^2 = \sum_{v \in V_A} \sum_{c \in C_A} W[v, c] \cdot (A[v, c] - B[v, c])^2$. Hence the instance is a YES-instance if and only if $\mathcal{R}_r(\emptyset) \leq \ell$.

We will show how to compute the records in a leaf-to-root fashion by proceeding along the nodes of T, in each of the following cases:

t is a leaf node. Let $V_t = \{v\}$, then $\mathcal{P}_t$ contains only one equivalence relation $p = \{(v, v)\}$ and $\mathcal{R}_t(p) = 0$ as there are no processed coordinates.

t is a forget node. Let t' be the child of t in T and $V_t = V_{t'} \setminus \{v_0\}$ for some vector v_0, we denote $C_{\mathrm{new}} = C_t^{\downarrow} \setminus C_{t'}^{\downarrow}$. By branching over all $p' \in \mathcal{P}(t')$ such that $p'|_{V_t \times V_t} = p$, we compute

$$\mathcal{R}_t^0(p) = \min_{p'} \left(\mathcal{R}_{t'}(p') + \Delta(p') \right),$$

where

$$\Delta(p') = \sum_{[v] \in [p']} \sum_{c \in C_{\mathrm{new}}} \Delta_c([v]).$$

For corectness, assume that $\mathcal{R}_t^0(p) = \tau = \mathcal{R}_{t'}(p') + \Delta(p')$ for some $p' \in \mathcal{P}(t')$ such that $p'|_{V_t \times V_t} = p$. Construct a matrix B_t from the witness $B_{t'}$ of $\mathcal{R}_{t'}(p')$ by adding a new column for every label $c \in C_{\mathrm{new}}$ as follows. Fot every $v \in V_{t'}$, we set $B_t[v, c] = \mu([v]_{p'}, c)$. If $u \in V_{t'}^{\downarrow} \setminus V_{t'}$ is such that $B_{t'}[u] = B_{t'}[v]$ for some $v \in V_{t'}$, we set $B_t[u, c] = \mu([v]_{p'}, c)$. Otherwise we set $B_{t'}[u, c] = 0$. To see that p is a t-partition of B_t, assume firstly that there is $v \neq v_0$ such that $(v_0, v) \in p'$, then $\mathrm{fcl}(B_t, t) = \mathrm{fcl}(B_{t'}, t')$ and $|[p]| = |[p']|$. Otherwise $\mathrm{fcl}(B_t, t) \leq \mathrm{fcl}(B_{t'}, t') + 1$ and $|[p]| = |[p']| - 1$; in any case p is a t-partition of B_t. Recall that

$$\mathrm{pwd}(B_t, t) = \sum_{v \in V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2$$
$$= \mathrm{pwd}(B_{t'}, t') + \sum_{v \in V_t^{\downarrow}} \sum_{c \in C_{\mathrm{new}}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2.$$

As $C_{\mathrm{new}} \cap C_{t'}^{\downarrow} = \emptyset$, we have $W[v, c] = 0$ for all $c \in C_{\mathrm{new}}$ and $v \in V_{t'}^{\downarrow} \setminus V_{t'}$. Therefore,

$$\mathrm{pwd}(B_t, t) = \mathrm{pwd}(B_{t'}, t') + \sum_{v \in V_{t'}} \sum_{c \in C_{\mathrm{new}}} W[v, c] \cdot (A[v, c] - \mu([v]_{p'}, c))^2$$
$$= \mathcal{R}_{t'}(p') + \Delta(p') = \tau,$$

and thus B_t witnesses that $\mathcal{R}_t(p) \leq \tau$.

On the other hand, assume that $\mathcal{R}_t(p) = \tau$ and B_t is a matrix in $\mathcal{B}_t$ witnessing this. Let $B_{t'}$ be obtained from B_t by deleting the columns with labels in C_{new}. We construct $p' \in \mathcal{P}(t')$ from p by adding v_0 to an equivalence class of arbitrary $v \in V_{t'}$ such that $B_t[v_0] = B_t[v]$. If there is no such v, we create a new equivalence class $\{v_0\}$. In any case p' is a t'-partition of $B_{t'}$: if $|[p']| = |[p]|$ then $\mathrm{fcl}(B_{t'}, t') = \mathrm{fcl}(B_t, t)$, in case $|[p']| = |[p]| + 1$ we have $\mathrm{fcl}(B_{t'}, t') = \mathrm{fcl}(B_t, t) - 1$. Therefore $B_{t'}$ witnesses that

$$\mathcal{R}_{t'}(p') \leq \mathrm{pwd}(B_{t'}, t')$$
$$= \tau - \sum_{v \in V_{t'}} \sum_{c \in C_{\mathrm{new}}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2$$
$$\leq \tau - \sum_{[v] \in [p']} \sum_{c \in C_{\mathrm{new}}} \Delta_c([v]) = \tau - \Delta(p').$$

So in our algorithm $\mathcal{R}_t^0(p) \leq \mathcal{R}_{t'}(p') + \Delta(p') \leq \tau$. Hence, we can correctly set $\mathcal{R}_t = \mathcal{R}_t^0$.

t is an introduce node. Let t' be the child of t in T and let $V_t = V_{t'} \cup \{v_0\}$ for some vector v_0. For every p in $\mathcal{P}(t)$, we set $\mathcal{R}_t^0(p) = \mathcal{R}_{t'}(p')$ where $p' = p|_{V_{t'} \times V_{t'}}$. For correctness, it will be useful to observe that $C_t^{\downarrow} = C_{t'}^{\downarrow}$. Recall from

the first paragraph of the proof that $W[v_0, c] = 0$ for every $c \in C^{\downarrow}_{t'}$. Assume $\mathcal{R}_t(p) = \tau$ with a witness B_t, let us obtain $B_{t'}$ from B_t by deleting the row with label v_0. Then $B_{t'}$ admits the t'-partition p' and $\mathrm{pwd}(B_{t'}, t') = \mathrm{pwd}(B_t, t)$, so $B_{t'}$ witnesses that $\mathcal{R}_{t'}(p') \leq \tau$.

If on the other hand $\mathcal{R}_{t'}(p') = \tau$ and $B_{t'}$ is a matrix witnessing this, let us add to $B_{t'}$ a row with the label v_0 so that p is a t-partition of the resulting matrix B_t. For this, if $(v_0, v) \in p$ for some $v \in V_{t'}$, we set $B_t[v_0] := B_{t'}[v]$, which results in $|[p]| = |[p']|$ and $\mathrm{fcl}(B_t, t) = \mathrm{fcl}(B_{t'}, t')$. Otherwise $|[p]| = |[p']| + 1$; we make $B_t[v_0]$ equal to any row of $B_{t'}$ that is not among $\{B_{t'}[v] \mid v \in V_{t'}\}$ (if such a row exists) and by this achieve $\mathrm{fcl}(B_t, t) = \mathrm{fcl}(B_{t'}, t') - 1$. If all the rows of $B_{t'}$ are among $\{B_{t'}[v] \mid v \in V_{t'}\}$, we set $B_t[v_0]$ equal to any of them. Then p is a t-partition of B_t and B_t witnesses $\mathcal{R}_t(p) \leq \tau$, so we can correctly set $\mathcal{R}_t = \mathcal{R}^0_t$.

t is a join node. Let t_1, t_2 be the children of t in T. For every p in $\mathcal{P}_t$ we set $\mathcal{R}^0_t(p) = \mathcal{R}_{t_1}(p) + \mathcal{R}_{t_2}(p)$. For correctness, assume that $\mathcal{R}^0_t(p) = \tau = \tau_1 + \tau_2$ where $\tau_1 = \mathcal{R}_{t_1}(p)$, $\tau_2 = \mathcal{R}_{t_2}(p)$. Let B_i be a witness of $\mathcal{R}_{t_i}(p) = \tau_i$ and let φ_i be the cluster-assignment w.r.t. B_i such that for every $u, v \in V_t$, $\varphi_i(u) = \varphi_i(v)$ if and only if $(u, v) \in p$, $i = 1, 2$. Without loss of generality we may assume that $\varphi_1|_{V_t} = \varphi_2|_{V_t}$. As $V^{\downarrow}_{t_1} \cap V^{\downarrow}_{t_2} = V_t$ and $C^{\downarrow}_{t_1} \cap C^{\downarrow}_{t_2} = \emptyset$, there exists a composition of (B_1, φ_1) and (B_2, φ_2), we denote it by B_t. Recall that B_t admits the cluster-assignment $\varphi = \varphi_1 \cup \varphi_2$, in particular, p is a t-partition of B_t. As $W[v, c] = 0$ for every $c \in C^{\downarrow}_{t_i}$ and $v \notin V^{\downarrow}_{t_i}$, $i = 1, 2$, we have

$$\begin{aligned}\mathrm{pwd}(B_t, t) = &\sum_{v \in V^{\downarrow}_{t_1}} \sum_{c \in C^{\downarrow}_{t_1}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2 \\ &+ \sum_{v \in V^{\downarrow}_{t_2}} \sum_{c \in C^{\downarrow}_{t_2}} W[v, c] \cdot (A[v, c] - B_t[v, c])^2 \\ &= \mathrm{pwd}(B_1, t_1) + \mathrm{pwd}(B_2, t_2) = \tau_1 + \tau_2 = \tau.\end{aligned}$$

Hence B_t witnesses that $\mathcal{R}_t(p) \leq \tau$.

For the converse, assume that $\mathcal{R}_t(p) = \tau$ and B_t is a matrix witnessing this. Let B_i be the restriction of B to rows $V^{\downarrow}_{t_i}$ and columns $C^{\downarrow}_{t_i}$, $i = 1, 2$. Then p is a t-partition of B_1 and B_2 and in our algorithm $\mathcal{R}^0_t(p) = \mathcal{R}_{t_1}(p) + \mathcal{R}_{t_2}(p) \leq \mathrm{pwd}(B_1) + \mathrm{pwd}(B_2) = \mathrm{pwd}(B_t) = \tau$. So the resulting record $\mathcal{R}_t = \mathcal{R}^0_t$ is correct, which concludes the correctness proof of the algorithm.

To estimate the time complexity, recall that in a forget node we branch over at most $(q+1)^{q+1} = \mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))}$ partitions p'. Computing $\Delta(p')$ requires time at most $|C_{\mathrm{new}}| \cdot (q+1)^2 \leq |C_{\mathrm{new}}| \cdot \mathcal{O}(\mathrm{tw}(G_P)^2)$. Observe that each column appears in C_{new} of at most one forget node in T, as all rows that have a value of "1" in the same column form a clique. Thus, the processing time of all forget nodes is upper-bounded by $d \cdot \mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))}$. For the rest of the nodes, it is easy to see that the processing time of each node is upper-bounded by $\mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))}$. Since the nice tree-decomposition $\mathcal{T}$ has $\mathcal{O}(n)$ nodes, the total running time is at most $(n + d) \cdot \mathrm{tw}(G_P)^{\mathcal{O}(\mathrm{tw}(G_P))}$. □

5. An Incidence-Graph Based Algorithm for Real-Valued MCME

While Theorem 4 significantly expands the previously known boundaries of tractability for MCME, the algorithm's performance strongly depends on the structural properties of the primal graph. In general, primal graph representations are known to be denser than incidence graph representations, and this may make them unsuitable for the application of structure-based algorithms on certain instances (see, e.g., the example below Theorem 6).

As our final result, we show that although a fixed-parameter algorithm for REAL-VALUED MCME parameterized by $\mathrm{tw}(G_I)$ remains beyond our reach, we can exploit a different parameter of the incidence graph to achieve fixed-parameter tractability—namely, the local feedback edge number.

Theorem 6. REAL-VALUED MCME *is fixed-parameter tractable when parameterized by* $\mathrm{lfen}(G_I)$.

We note that it is not difficult to show that $\mathrm{lfen}(G_I)$ and $\mathrm{tw}(G_P)$ are pairwise incomparable parameterizations. Indeed, an $n \times 1$ mask consisting only of "1" entries has $\mathrm{lfen}(G_I) = 0$ but $\mathrm{tw}(G_P) = n - 1$. On the other hand, consider, for some integer m, an $(m+1) \times 2m$ mask W such that the first row v_0 of W consists only of "1" entries, while for each $i \in [m]$ the $(i+1)$-th row has "1" entries on precisely two positions: i and $i + m$. Then G_P is a star with center in v_0, so $\mathrm{tw}(G_P) = 1$. However, G_I consists of m edge-disjoint cycles intersecting in v_0. It is then easy to observe that the local feedback edge number of v_0 with respect to any spanning tree of G_I is m.

We proceed by introducing some additional terminology that will be useful in the coming arguments. Let T be a fixed rooted spanning tree of G such that $\mathrm{lfen}(G, T) = \mathrm{lfen}(G)$, denote the root by r. For $t \in V(T)$, let T_t be the subtree of T rooted at t. We define the *boundary* $\delta(t)$ of t to be the set of endpoints of all edges in G with precisely one endpoint in $V(T_t)$ (observe that the boundary can never have a size of 1). t is called *closed* if $|\delta(t)| \leq 2$ and *open* otherwise.

Proposition 7 (Ganian and Korchemna (2021))**.**

1. *For every closed child t' of t in T, it holds that $\delta(t') = \{t, t'\}$ and tt' is the only edge between $V(T_{t'})$ and $V(G) \setminus V(T_{t'})$ in G.*
2. *$|\delta(t)| \leq 2\,\mathrm{lfen}(G) + 2$.*
3. *Let $\{t_i | i \in [j]\}$ be the set of all open children of t in T.*

Then $j \leq 2\,\mathrm{lfen}(G)$ *and* $\delta(t) \subseteq \bigcup_{i=1}^{j} \delta(t_i) \cup \{t\} \cup N_G(t)$.

To prove Theorem 6, we will provide a leaf-to-root dynamic programming algorithm which stores information about optimal partitionings of $\delta(t)$ into clusters. On a very intuitive level, Point 1. of Proposition 7 allows the algorithm to handle the closed children in a greedy manner, Point 2. ensures that the size of the records is bounded, and Point 3. is used in the dynamic programming step to compute records for a parent based on the records of its children. Furthermore, one can observe that $|\bigcup_{i=1}^{j} \delta(t_i)|$ is upper-bounded by a linear function of $\mathrm{lfen}(t)$, which will be useful to ascertain the runtime bound for the algorithm.

Observation 8. *For each node* t *in* T, $|\bigcup_{i=1}^{j} \delta(t_i)| \leq 4\,\mathrm{lfen}(t) + 2$.

Proof. By Point 2. of Proposition 7, the number of vertices in $\bigcup_{i=1}^{j} \delta(t_i)$ that belong to $\delta(t)$ is at most $|\delta(t)| \leq 2\,\mathrm{lfen}(G) + 2$. If $v \in \delta(t_i) \setminus (\delta(t))$ for some $i \in [j]$, then $v = t_i$. So the number of such v is at most $j \leq 2\,\mathrm{lfen}(G)$ by Point 3. of Proposition 7. In total, $|\bigcup_{i=1}^{j} \delta(t_i)| \leq 4\,\mathrm{lfen}(t) + 2$. □

We are now ready to establish the claimed result by providing an algorithm for REAL-VALUED MCME with running time $(kn^2 + d) \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$, assuming a spanning tree T of minimum local feedback edge number is provided. We note that such a spanning tree can be computed by a fixed-parameter algorithm (Ganian & Korchemna, 2021).

Proof of Theorem 6. We will design a leaf-to-root dynamic procedure computing the set of records for every node of T. Intuitively, the record in t will store the minimum sum of distances of processed vectors, computed along the processed coordinates, to their cluster centers for every possible partition of vectors in $\delta(t)$ into clusters.

Formally, let $V_t^{\downarrow}$ and $C_t^{\downarrow}$ be the sets of vectors and coordinates of T_t correspondingly, we denote by V_t and C_t the restrictions of $\delta(t)$ to vectors and coordinates respectively, then $\delta(t) = V_t \sqcup C_t$. Let $\mathcal{P}_t$ be the set of all equivalence relations $p \subseteq V_t \times V_t$ with at most k equivalence classes. The record $\mathcal{R}_t$ of t is a mapping from $\mathcal{P}_t$ to $\mathbb{R}^+$. Observe that $|\mathcal{P}_t| \leq \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$.

To introduce the semantics of our records, let $\mathcal{B}_t$ be the set of all real matrices with the row labels $V_t^{\downarrow} \cup V_t$ and the column labels $C_t^{\downarrow}$. Let B_t be a matrix in $\mathcal{B}_t$. We define the *partial weighted distance* from B_t to A in t as follows:

$$\mathrm{pwd}(B_t, t) = \sum_{v \in V_t \cup V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[v,c] \cdot (A[v,c] - B_t[v,c])^2.$$

For a matrix B_t with the row labels $V_t^{\downarrow} \cup V_t$ and the column labels $C_t^{\downarrow}$, we say that $p \in \mathcal{P}_t$ is the *t-partition* of B_t if the following conditions hold:

- for every $v, w \in V_t$ such that $(v, w) \in p$, $B_t[v] = B_t[w]$,
- the number $\mathrm{fcl}(B_t, t)$ of pairwise different rows in B_t not equal to any $B[v]$, $v \in V_t$, is at most $k - |[p]|$.

Note that the existence of a t-partition automatically implies that B_t has at most k distinct rows. We are now ready to define the record $\mathcal{R}_t$. For each $p \in \mathcal{P}_t$, we set $\mathcal{R}_t(p) = \tau$ if there exists $B_t \in \mathcal{B}_t$ such that:

- p is the t-partition of B_t,
- $\mathrm{pwd}(B_t, t) = \tau$, and
- $\forall B'_t \in \mathcal{B}_t$ such that p is the t-partition of B'_t, $\mathrm{pwd}(B'_t, t) \geq \tau$.

In this case we say that B_t *witnesses* that $\mathcal{R}_t(p) = \tau$.

Recall that for the root $r \in T$, the boundary is empty, in particular $V_r = \emptyset$, so $\mathcal{P}_r$ contains only one element $\emptyset$, and $\mathcal{R}_r(\emptyset)$ is equal to the minimum value of $\|W \circ (A - B)\|_F^2$ that can be achieved by any real matrix B containing at most k distinct rows. Hence the instance is a YES-instance if and only if $\mathcal{R}_r(\emptyset) \leq \ell$. We will compute the records in a leaf-to-root fashion by proceeding along the nodes of T. Note that for any leaf vector v, there are no coordinates in T_v and therefore $\mathcal{R}_v(p) = 0$ for every $p \in \mathcal{P}_v$.

Let $t = c$ be a (maybe leaf) coordinate with children $v_1, \ldots, v_m$ where $v_1, \ldots, v_j$ are open and the rest are closed. Recall from Proposition 7 that $j \leq 2\,\mathrm{lfen}(G_I)$. We asssume that the closed children are sorted by the coordinate c, i.e., $A[v_i, c] \leq A[v_{i+1}, c]$, $i \in [m-1] \setminus [j]$. Let us denote $V_0 = \{v | (v, c) \in E(G_I) \setminus E(T)\}$, and for each $i \in [m]$, $V_i = V_{v_i}$, $C_i = C_{v_i}$, $V_i^{\downarrow} = V_{v_i}^{\downarrow}$, and $C_i^{\downarrow} = C_{v_i}^{\downarrow}$. Observe that $V_t \subseteq \bigcup_{i \in [j]_0} V_i$.

For every $p_0 \in \mathcal{P}_t$, we initiate by setting $\mathcal{R}_t(p_0) = dnM_A + 1$ and branch over all equivalence relations p on $\bigcup_{i \in [j]_0} V_i$ such that $|[p]| \leq k$ and $p|_{V_t} = p_0$. In every branch we construct a partition p^* on the domain $D^* = \bigcup_{i \in [m]_0} V_i$ extending p to the set of closed children. Intuitively, p^* will provide a subdivision of the interval $[j+1, m]$ into at most k subintervals such that closed children v_{i_1} and v_{i_2} belong to the same cluster if i_1 and i_2 belong to the same subinterval.

To find the optimal (i.e., minimizing the sum of distances in the coordinate c) subdivision, we will compute the dynamic programming table $\mathcal{M}(i, S, k_{\mathrm{cl}})$ for every subset $S \subseteq [p]$ and non-negative integers $i \in [j, m]$ and $k_{cl} \leq k - |[p]|$, that stores the minimum sum of distances along the coordinate c for the closed children $v_{j+1} \ldots v_i$ if they are partitioned into $|S| + k_{\mathrm{cl}}$ clusters with the following property: k_{cl} clusters do not contain any open children, and for each equivalence

class $[v] \in S$, there is a cluster intersecting the set of open children by exactly $[v]$. We initiate by setting $\mathcal{M}(j, \emptyset, 0) = 0$ and $\mathcal{M}(j, S, k_{\mathrm{cl}}) = dnM_A + 1$ whenever $S \neq \emptyset$ or $k_{\mathrm{cl}} \neq 0$. Further, we set $\mathcal{M}(i, \emptyset, 0) = dnM_A + 1$ for $i > j$ and then for remaining triples $S \subseteq [p]$, $i \in [j+1, m]$ and $k_{cl} \leq k - |[p]|$ compute:

$$\mathcal{M}_1(i, S, k_{\mathrm{cl}}) = \min_{j \leq i' < i} \Big(\mathcal{M}(i', S, k_{\mathrm{cl}} - 1) + \Delta_c(v_{(i',i]}) \Big),$$

$$\mathcal{M}_2(i, S, k_{\mathrm{cl}}) = \min_{j \leq i' \leq i} \min_{[v] \in S} \Big(\mathcal{M}(i', S \setminus \{[v]\}, k_{\mathrm{cl}}) + \Delta_c([v] \cup v_{(i',i]}) \Big),$$

where $v_{(i',i]} = \{v_y : y \in [i] \setminus [i']\}$. As special cases, we set $\mathcal{M}_1(i, S, k_{\mathrm{cl}}) = dnM_A + 1$ if $k_{\mathrm{cl}} = 0$, and $\mathcal{M}_2(i, S, k_{\mathrm{cl}}) = dnM_A + 1$ if $S = \emptyset$. Intuitively, $\mathcal{M}_1$ corresponds to the case where the children in $v_{(i',i]}$ form a separate cluster, while $\mathcal{M}_2$ represents the addition of $v_{(i',i]}$ to the cluster of open children from $[v]$. $\mathcal{M}_2$ also captures an option when no closed children are added to the cluster of $[v]$, namely, when $i = i'$. We set $\mathcal{M}(i, S, k_{\mathrm{cl}}) = \min(\mathcal{M}_1(i, S, k_{\mathrm{cl}}), \mathcal{M}_2(i, S, k_{\mathrm{cl}}))$.

When all the entries of $\mathcal{M}$ are computed, let $\Delta_c(p) = \min_{0 \leq k_{\mathrm{cl}} \leq k - |[p]|} \mathcal{M}(m, [p], k_{\mathrm{cl}}) = \mathcal{M}(m, [p], k^0_{\mathrm{cl}})$. A simple back-tracking allows to construct the partition p^* of D^* extending p such that $|[p^*]| = |[p]| + k^0_{\mathrm{cl}}$ and $\sum_{[v] \in [p^*]} \Delta_c([v]) = \Delta_c(p)$. We try to improve $\mathcal{R}^0_t(p_0) := \min(\mathcal{R}^0_t(p_0), \sum_{i \in [m]} \mathcal{R}_{v_i}(p^*|_{V_i}) + \Delta_c(p))$.

For correctness, assume that in our algorithm $\mathcal{R}^0_v(p) = \tau = \sum_{i \in [m]} \mathcal{R}_{v_i}(p^*|_{V_i}) + \Delta_c(p)$. For every $i \in [j]$, let $p_i = p^*|_{V_i}$ and let B_i be the witness of $\mathcal{R}_{v_i}(p_i)$. We define the matrix B_0 with the only column label c and row labels V_0 by setting $B_0[v, c] = \mu([v]_{p^*}, c)$. Then, in particular, $\sum_{u \in [v] \cap V_0} W[u, c] \cdot (A(u, c) - B_0(u, c))^2 = \Delta_c([v])$ for every $[v] \in [p^*]$. For every $i \in [m]_0$, $p_i = p^*|_{V_i}$ and therefore we can construct cluster-assignments φ_i w.r.t. B_i such that

- for every $u, v \in V_i$, $\varphi_i(u) = \varphi_i(v)$ if and only if $(u, v) \in p^*$, $i \in [m]_0$;
- for every $v \in V_{i_1} \cap V_{i_2}$, $\varphi_{i_1}(v) = \varphi_{i_2}(v)$, $i_1, i_2 \in [m]_0$.

Note that the sets of column labels of B_i are pairwise disjoint, so there exists a composition B_t of (B_i, φ_i), $i \in [m]_0$. Recall that B_t admits the cluster-assignment $\varphi = \bigcup_{i \in [m]_0} \varphi_i$, has row labels $\bigcup_{i \in [m]} (V_i \cup V_i^{\downarrow}) \cup V_0 = V_t \cup V_t^{\downarrow}$ and column labels $\bigcup_{i \in [m]} C_i^{\downarrow} \cup \{c\} = C_t^{\downarrow}$. Let us compute $\mathrm{pwd}(B_t, t) = \sum_{v \in V_t \cup V_t^{\downarrow}} \sum_{c' \in C_t^{\downarrow}} W[v, c'] \cdot (A[v, c'] - B_t[v, c'])^2$.

Observe that if $c' \in C_i^{\downarrow}$ and $v \notin V_i \cup V_i^{\downarrow}$, then $W[v, c'] = 0$, $i \in [m]$. The set of v such that $W[v, c] = 1$ is precisely V_0. Therefore $\mathrm{pwd}(B_t, t)$ can be computed as $\sum_{i \in [m]} \sum_{v \in V_i \cup V_i^{\downarrow}} \sum_{c' \in C_i^{\downarrow}} W[v, c'] \cdot (A[v, c'] - B_i[v, c'])^2 + \sum_{v \in V_0} W[v, c] \cdot (A[v, c] - B_0[v, c])^2 = \sum_{i \in [m]} \mathrm{pwd}(B_i, v_i) + \sum_{[v] \in [p^*]} \Delta_c([v]) = \sum_{i \in [m]} \mathcal{R}_{v_i}(p_i) + \Delta_c(p) = \tau$. Hence B_t witnesses that $\mathcal{R}^0_v(p) \leq \tau$.

For the converse, assume that $\mathcal{R}_t(p_0) = \tau$ and B_t is a matrix witnessing this. There exists a cluster-assignment φ w.r.t. B_t such that for every $u, v \in V_t$, $\varphi(u) = \varphi(v)$ if and only if $(u, v) \in p_0$. We define the equivalence relation p^* on domain D^* as follows: $(u, v) \in p^*$ if and only if $\varphi(u) = \varphi(v)$. In particular, p^* extends p_0 and $|[p]| \leq k$.

Claim 2. *If u and v are closed children of c and $A[u, c] \leq A[v, c]$ but $B_t[u, c] \geq B_t[v, c]$, then there exists a witness $B'_t \in \mathcal{B}_t$ of $\mathcal{R}_t(p_0) \leq \tau$ with cluster-assignment φ' such that:*

- *$\varphi'(u) = \varphi(v)$, $\varphi'(v) = \varphi(u)$;*
- *φ' coincides with φ outside of $V_u^{\downarrow} \cup V_v^{\downarrow}$;*
- *$B'_t[u, c] = B_t[v, c]$, $B'_t[v, c] = B_t[u, c]$;*
- *$B'_t[v_i, c] = B_t[v_i, c]$ for every $v_i \notin \{u, v\}$, $i \in [m]$;*

Proof. Let us define φ' as follows:

- if $w \in V_u^{\downarrow} \cup V_v^{\downarrow}$ and $\varphi(w) = \varphi(u)$, then $\varphi'(w) = \varphi(v)$;
- if $w \in V_u^{\downarrow} \cup V_v^{\downarrow}$ and $\varphi(w) = \varphi(v)$, then $\varphi'(w) = \varphi(u)$;
- for the rest of $w \in V_t \cup V_t^{\downarrow}$, $\varphi'(w) = \varphi(w)$;

We define the matrix B'_t with row and column labels $V_t \cup V_t^{\downarrow}$ and $C_t^{\downarrow}$:

- if $\varphi'(w) \neq \varphi'(v)$ and $\varphi'(w) \neq \varphi'(u)$, we set $B'_t[w] = B_t[w]$, $w \in V_t \cup V_t^{\downarrow}$;
- $B'_t[v, c'] = B_t[v, c']$ for $c' \in C_v^{\downarrow} \cup C_u^{\downarrow}$; $B'_t[v, c'] = B_t[u, c']$ for the rest of c';
- $B'_t[u, c'] = B_t[u, c']$ for $c' \in C_v^{\downarrow} \cup C_u^{\downarrow}$; $B'_t[u, c'] = B_t[v, c']$ for the rest of c';
- if $\varphi'(w) = \varphi'(x)$, we set $B'_t[w] = B'_t[x]$, $x \in \{u, v\}$.

From the last definition it is clear that φ' is a cluster-assignment w.r.t. B'_t. Observe that φ coincides with φ' on V_t, in particular, p_0 is a t-partition of B'_t. Let us compare $\mathrm{pwd}(B_t, t)$ and $\mathrm{pwd}(B'_t, t)$. Summands corresponding to w with $\varphi'(w) \neq \varphi'(v)$ and $\varphi'(w) \neq \varphi'(u)$ obviously coincide. Consider $w \neq v$ with $\varphi'(w) = \varphi'(v)$. If $w \in V_v^{\downarrow} \cup V_u^{\downarrow}$, then $\varphi(w) = \varphi(v)$ and therefore for $c' \in C_v^{\downarrow} \cup C_u^{\downarrow}$ we have $B'_t[w, c'] = B'_t[v, c'] = B_t[v, c'] = B_t[w, c']$. For $c' \notin C_v^{\downarrow} \cup C_u^{\downarrow}$, $W[w, c'] = 0$ as v and u are closed children. Assume that $w \notin V_v^{\downarrow} \cup V_u^{\downarrow}$, then $\varphi(w) = \varphi(u)$. For $c' \in C_v^{\downarrow} \cup C_u^{\downarrow}$, we have $W[w, c'] = 0$. For the rest of c', $B'_t[w, c'] = B'_t[v, c'] = B_t[u, c'] = B_t[w, c']$, so all the

summands corresponding to w coincide. Similarly they coincide for $w \neq u$ with $\varphi'(w) = \varphi'(u)$. In particular, $B'_t[v_i, c] = B_t[v_i, c]$ for every $v_i \notin \{u, v\}$, $i \in [m]$. The only difference may occure in the summands corresponding to u and v. Moreover, the only coordinate c' where $W[v, c'] = 1$ and $B'_t[v, c'] \neq B_t[v, c']$, is $c' = c$, similarly for u. So comparing $\mathrm{pwd}(B_t, t)$ and $\mathrm{pwd}(B'_t, t)$ is in fact comparing $(A[v, c] - B_t[v, c])^2 + (A[u, c] - B_t[u, c])^2$ and $(A[v, c] - B_t[u, c])^2 + (A[u, c] - B_t[v, c])^2$. It is easy to see that the second sum is not larger then the first, see Figure 3 for the illustration. □

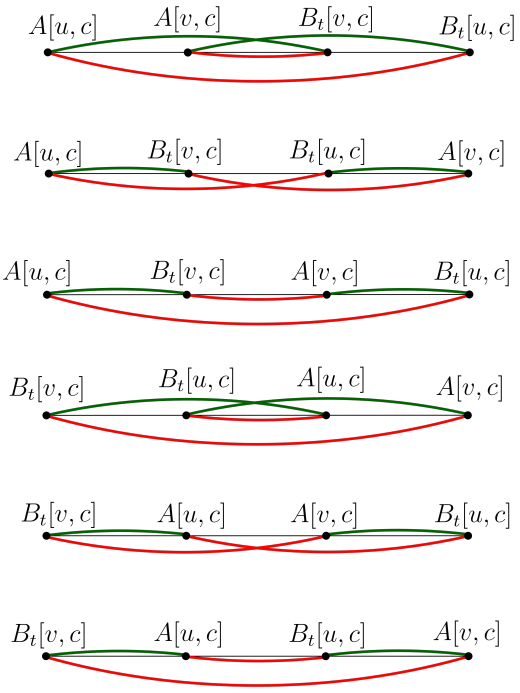

Figure 3. Red and green segments represent contribution of v and u along c into the total sum, for B_t and B'_t correspondingly. For any arrangement of $A[u, c]$, $A[v, c]$, $B_t[u, c]$ and $B_t[v, c]$, the sum of squares of green segments is not larger then the sum of squares of red ones.

In fact, Claim 2 allows us to assume without loss of generality that every cluster intersects the set of closed children of c by some $v_{(i_1,i_2]}$, $i_1 \leq i_2$. Indeed, pick any closed children v_i and $v_{i'}$ of c with $i < i'$, then $A[v_i, c] \leq A[v_{i'}, c]$. We will say that the pair $(v_i, v_{i'})$ is *bad* if one of the following holds:

- $B_t[v_i, c] > B_t[v_{i'}, c]$;
- $B_t[v_i, c] = B_t[v_{i'}, c]$, $\varphi(v_i) \neq \varphi(v_{i'})$ but there exists $i'' > i'$ such that $\varphi(v_i) = \varphi(v_{i''})$.

Observe that if there are any bad pairs, we can chose one of them so that application of the Claim 2 to it decreases the number of bad pairs. Applying Claim 2 iteratively until there remain no bad pairs, we finally obtain a witness B^*_t of $\mathcal{R}_t(p_0) = \tau$ with cluster-assignment φ^* such that any cluster of φ^* intersects the set of closed children of c in some $v_{(i_1,i_2]}$. Further we assume $B^*_t = B_t$, $\varphi^* = \varphi$. By optimality of B_t, we conclude that $\sum_{u \in [v]} W[u, c] \cdot (A[u, c] - B_t[u, c])^2 = \Delta_c([v])$ for every $[v] \in [p^*]$. Every such case is captured by the computation scheme of the dynamic table $\mathcal{M}$, so we have $\mathcal{M}(m, [p], k^0_{\mathrm{cl}}) \leq \sum_{[v] \in [p^*]} \Delta_c([v])$ where p is the restriction of p^* to $D^* \setminus v_{(j,m]}$ and $k^0_{\mathrm{cl}} = |[p^*]| - |[p]|$. Let B_i be the restriction of B_t to rows $V_i \cup V_i^{\downarrow}$ and columns $C_i^{\downarrow}$, $i \in [m]$. Then $p^*|_{V_i}$ is v_i-partition of B_i and hence in our algorithm

$$\begin{aligned}\mathcal{R}^0_t(p_0) &\leq \sum_{i \in [m]} \mathcal{R}_{v_i}(p^*|_{V_i}) + \mathcal{M}(m, [p], k^0_{\mathrm{cl}}) \\ &\leq \sum_{i \in [m]} \mathrm{pwd}(B_i, v_i) + \sum_{[v] \in [p^*]} \Delta_c([v]) \\ &= \mathrm{pwd}(B_t, t) = \tau.\end{aligned}$$

It remains to consider the case where $t = v$ is a vector with children $c_1, \ldots, c_m$ in T, where $c_1, \ldots, c_j$ are open and the rest are closed. We will use the notations $V_i = V_{c_i}$, $C_i = C_{c_i}$, $V_i^{\downarrow} = V_{c_i}^{\downarrow}$ and $C_i^{\downarrow} = C_{c_i}^{\downarrow}$, $i \in [m]$. For every $p_0 \in \mathcal{P}_t$, we initiate by setting $\mathcal{R}_t(p_0) = ndM_A + 1$ and then branch over equivalence relations p on $\bigcup_{i \in [j]} V_i \cup \{v\} \supseteq V_t$ such that $|[p]| \leq k$ and $p|_{V_t} = p_0$. In every branch we set $\mathcal{R}^0_v(p_0) := \min(\mathcal{R}^0_v(p_0), \sum_{i \in [m]} \mathcal{R}_{c_i}(p_i))$ where $p_i = p|_{V_i}$ for $i \in [j]$ and $p_i = \{(v, v)\}$ for $i \in [m] \setminus [j]$.

For correctness, assume that in our algorithm $\mathcal{R}^0_v(p_0) = \tau = \sum_{i \in [m]} \mathcal{R}_{c_i}(p_i)$ and let B_i be the witness of $\mathcal{R}_{c_i}(p_i)$. We define φ on the domain of p and with values in $[k]$ by setting $\varphi(u) = \varphi(w)$ if and only if $(u, w) \in p$. Recall that $p_i = p|_{V_i}$ for $i \in [j]$ and $p_i = \{(v, v)\}$ for $i \in [m] \setminus [j]$, so there exist cluser-assignments φ_i w.r.t. B_i which agree with φ on intersections of their domains. Sets of columns of B_i are pairwise disjoint, so there exists a composition matrix B_t with cluster-assignment $\bigcup_{i \in [m]} \varphi_i = \varphi$, row labels $\bigcup_{i \in [m]} (V_i \cup V_i^{\downarrow}) = V_t \cup V_t^{\downarrow}$ and column labels $\bigcup_{i \in [m]} C_i^{\downarrow} = C_t^{\downarrow}$. Let us compute $\mathrm{pwd}(B_t, t) = \sum_{u \in V_t \cup V_t^{\downarrow}} \sum_{c \in C_t^{\downarrow}} W[u, c] \cdot (A[u, c] - B_t[u, c])^2$. Here $C_t^{\downarrow} = \sqcup_{i \in [m]} C_i^{\downarrow}$ and $W[u, c] = 0$ whenever $c \in C_i^{\downarrow}$ and $u \notin V_i \cup V_i^{\downarrow}$. Therefore $\mathrm{pwd}(B_t, t) = \sum_{i \in [m]} \sum_{u \in V_i \cup V_i^{\downarrow}} \sum_{c \in C_i^{\downarrow}} W[u, c] \cdot (A[u, c] - B_i[u, c])^2 = \sum_{i \in [m]} \mathrm{pwd}(B_i, v_i) = \sum_{i \in [m]} \mathcal{R}_{c_i}(p_i) = \tau$. Hence B_t witnesses that $\mathcal{R}_v(p) \leq \tau$.

For the converse, assume that $\mathcal{R}_t(p_0) = \tau$ and B_t is a matrix witnessing this. Let φ be the cluster-assignment w.r.t. B_t such that for every $u, v \in V_t$, $\varphi(u) = \varphi(v)$ if and only if $(u, v) \in p_0$. We extend p_0 to the equivalence relation p on domain $\bigcup_{i \in [j]} V_i \cup \{v\}$ defined as follows: $(u, v) \in p$ if and only if $\varphi(u) = \varphi(v)$. Let B_i be the restriction of B_t to

rows $V_i \cup V_i^{\downarrow}$ and columns $C_i^{\downarrow}$, $i \in [m]$, then $p_i = p|_{V_i}$ is c_i-partition of B_i, $i \in [j]$. Hence in our algorithm $\mathcal{R}_t^0(p_0) \leq \sum_{i\in[j]} \mathcal{R}_{c_i}(p_i) + \sum_{i\in[m]\setminus[j]} \mathcal{R}_{c_i}(\{(v,v)\}) \leq \sum_{i\in[m]} \mathrm{pwd}(B_i, c_i) = \mathrm{pwd}(B_t, t) = \tau$. Hence the resulting record $\mathcal{R}_t = \mathcal{R}_t^0$ is correct, which concludes the correctness proof of the algorithm.

Let us estimate the time for processing coordinates $t = c$. For a fixed c with m_c children, we branch over equivalence relations p on domain $\bigcup_{i\in[j]_0} V_i \subseteq \delta(c) \cup \bigcup_{i\in[j]} \delta(v_i)$. By Observation 8, the size of the domain is upper-bounded by $\mathcal{O}(\mathrm{lfen}(G_I))$, so there are at most $\mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$ branches. In every branch, we start from computing $\Delta_c([v] \cup v_{(i',i]})$ and $\Delta_c(v_{(i',i]})$ for each $[v] \in [p]$ and $i' < i$ from $[m_c] \setminus [j]$. Taking into account Observation 5, all the computations can be performed in time at most $m_c^2 \cdot |[p]|$. After this, every entry of $\mathcal{M}$ can be calculated in time $\mathcal{O}(m_c \cdot |[p]|) \leq m_c \cdot \mathcal{O}(\mathrm{lfen}(G_I))$. $\mathcal{M}$ has size at most $m_c k \cdot 2^{|[p]|} \leq m_c k \cdot 2^{\mathcal{O}(\mathrm{lfen}(G_I))}$ and therefore can be computed for all the branches in time $m_c^2 k \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$. As $\sum_{c\in C_A} m_c^2 \leq (\sum_{c\in C_A} m_c)^2 \leq n^2$, processing all the coordinates c takes time of at most $n^2 k \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$. Record for a vector v with m_v children can be computed in $m_v \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$, which yields the time $d \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$ for processng all the vectors. Therefore the total runnung time of the algorithm is upper-bounded by $(kn^2 + d) \cdot \mathrm{lfen}(G_I)^{\mathcal{O}(\mathrm{lfen}(G_I))}$. □

6. Concluding Remarks

While our algorithmic results are specifically designed to deal with MEANS CLUSTERING WITH MISSING ENTRIES, it would be interesting to see whether the approaches and techniques developed here can be applied to other clustering variants or, e.g., the related task of low-rank matrix completion. Still, on the theoretical side the by far most prominent problem that is relevant to this research direction is the complexity of k-MEANS CLUSTERING for real-valued matrices when parameterized by the number of columns. A W[1]-hardness result for this problem would immediately exclude the existence of a fixed-parameter algorithm for REAL-VALUED MCME parameterized by the incidence treewidth of the mask, while a fixed-parameter algorithm could potentially open up the way towards tractability.

It would also be interesting to see how the considered sparsity parameters behave in practical settings. In particular, even though direct implementations of our exact algorithms with runtime guarantees are unlikely to outperform state-of-the-art heuristics, it may be possible to exploit these parameters to guide or otherwise improve existing methods.

Acknowledgements

Robert Ganian, Thekla Hamm, Viktoriia Korchemna, and Kirill Simonov acknowledge support by the Austrian Science Fund (FWF, projects Y1329 and P31336). Karolina Okrasa acknowledges support by the European Research Council, grant agreement No 714704; parts of this work were performed while visiting TU Wien, Vienna, Austria.

References

Aloise, D., Deshpande, A., Hansen, P., and Popat, P. Np-hardness of euclidean sum-of-squares clustering. *Mach. Learn.*, 75(2):245–248, 2009.

Bodlaender, H. L. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Discret. Math.*, 25(6):1305–1317, 1996.

Bodlaender, H. L., Jansen, B. M. P., and Kratsch, S. Preprocessing for treewidth: A combinatorial analysis through kernelization. *SIAM J. Discret. Math.*, 27(4):2108–2142, 2013.

Bodlaender, H. L., Drange, P. G., Dregi, M. S., Fomin, F. V., Lokshtanov, D., and Pilipczuk, M. A $c^k n$ 5-approximation algorithm for treewidth. *SIAM J. Comput.*, 45(2):317–378, 2016.

Cohen-Addad, V., de Mesmay, A., Rotenberg, E., and Roytman, A. The bane of low-dimensionality clustering. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 441–456. SIAM, 2018. ISBN 978-1-6119-7503-1. URL http://dl.acm.org/citation.cfm?id=3174304.3175300.

Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. *Parameterized Algorithms*. Springer, 2015. ISBN 978-3-319-21274-6. doi: 10.1007/978-3-319-21275-3. URL https://doi.org/10.1007/978-3-319-21275-3.

Dahiya, Y., Fomin, F. V., Panolan, F., and Simonov, K. Fixed-parameter and approximation algorithms for PCA with outliers. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2341–2351. PMLR, 2021.

Downey, R. G. and Fellows, M. R. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

Drineas, P., Frieze, A. M., Kannan, R., Vempala, S. S., and Vinay, V. Clustering large graphs via the singular value decomposition. *Mach. Learn.*, 56(1-3):9–33, 2004.

Dvorák, P., Eiben, E., Ganian, R., Knop, D., and Ordyniak, S. The complexity landscape of decompositional parameters for ILP: programs with few global variables and constraints. *Artif. Intell.*, 300:103561, 2021.

Eiben, E., Fomin, F. V., Golovach, P. A., Lochet, W., Panolan, F., and Simonov, K. EPTAS for *k*-means clustering of affine subspaces. In Marx, D. (ed.), *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pp. 2649–2659. SIAM, 2021a. doi: 10.1137/1.9781611976465.157. URL https://doi.org/10.1137/1.9781611976465.157.

Eiben, E., Ganian, R., Kanj, I., Ordyniak, S., and Szeider, S. The parameterized complexity of clustering incomplete data. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 7296–7304. AAAI Press, 2021b. URL https://ojs.aaai.org/index.php/AAAI/article/view/16896.

Fellows, M. R., Protti, F., Rosamond, F. A., da Silva, M. D., and Souza, U. S. Algorithms, kernels and lower bounds for the flood-it game parameterized by the vertex cover number. *Discret. Appl. Math.*, 245:94–100, 2018.

Fomin, F. V., Golovach, P. A., and Panolan, F. Parameterized low-rank binary matrix approximation. In Chatzigiannakis, I., Kaklamanis, C., Marx, D., and Sannella, D. (eds.), *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pp. 53:1–53:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

Fomin, F. V., Golovach, P. A., and Simonov, K. Parameterized *k*-clustering: Tractability island. *J. Comput. Syst. Sci.*, 117:50–74, 2021.

Ganian, R. and Korchemna, V. The complexity of bayesian network learning: Revisiting the superstructure. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 430–442. Curran Associates, Inc., 2021.

Ganian, R. and Ordyniak, S. The complexity landscape of decompositional parameters for ILP. *Artif. Intell.*, 257: 61–71, 2018.

Ganian, R. and Ordyniak, S. Solving integer linear programs by exploiting variable-constraint interactions: A survey. *Algorithms*, 12(12):248, 2019. URL https://doi.org/10.3390/a12120248.

Ganian, R., Kanj, I. A., Ordyniak, S., and Szeider, S. Parameterized algorithms for the matrix completion problem. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1642–1651. PMLR, 2018. URL http://proceedings.mlr.press/v80/ganian18a.html.

Gaspers, S. and Najeebullah, K. Optimal surveillance of covert networks by minimizing inverse geodesic length. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pp. 533–540. AAAI Press, 2019.

Grüttemeier, N., Komusiewicz, C., and Morawietz, N. Efficient bayesian network structure learning via parameterized local search on topological orderings. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 12328–12335. AAAI Press, 2021.

Inaba, M., Katoh, N., and Imai, H. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the 10th annual Symposium on Computational Geometry (SoCG)*, pp. 332–339. ACM, 1994.

Kloks, T. *Treewidth: Computations and Approximations*. Springer, Berlin, 1994.

Koana, T., Froese, V., and Niedermeier, R. Parameterized algorithms for matrix completion with radius constraints. In Gørtz, I. L. and Weimann, O. (eds.), *31st Annual Symposium on Combinatorial Pattern Matching, CPM 2020, June 17-19, 2020, Copenhagen, Denmark*, volume 161 of *LIPIcs*, pp. 20:1–20:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Koana, T., Froese, V., and Niedermeier, R. Binary matrix completion under diameter constraints. In Bläser, M. and Monmege, B. (eds.), *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pp. 47:1–47:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

Lloyd, S. P. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136, 1982.

Mertzios, G. B., Nichterlein, A., and Niedermeier, R. The power of linear-time data reduction for maximum matching. *Algorithmica*, 82(12):3521–3565, 2020.

Moshkovitz, M., Dasgupta, S., Rashtchian, C., and Frost, N. Explainable k-means and k-medians clustering. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7055–7065. PMLR, 2020.

Nesetril, J. and de Mendez, P. O. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and Combinatorics*. Springer, 2012.

Ordyniak, S. and Szeider, S. Parameterized complexity results for exact bayesian network structure learning. *J. Artif. Intell. Res.*, 46:263–302, 2013.

Robertson, N. and Seymour, P. D. Graph minors. i. excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

Robertson, N. and Seymour, P. D. Graph minors. III. planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984.

Samer, M. and Szeider, S. Fixed-parameter tractability. In Biere, A., Heule, M., van Maaren, H., and Walsh, T. (eds.), *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pp. 425–454. IOS Press, 2009.

Samer, M. and Szeider, S. Constraint satisfaction with bounded treewidth revisited. *J. Comput. Syst. Sci.*, 76(2): 103–114, 2010.

Simonov, K., Fomin, F. V., Golovach, P. A., and Panolan, F. Refined complexity of PCA with outliers. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5818–5826. PMLR, 2019.

Wang, S., Li, M., Hu, N., Zhu, E., Hu, J., Liu, X., and Yin, J. K-means clustering with incomplete data. *IEEE Access*, 7:69162–69171, 2019. doi: 10.1109/ACCESS.2019.2910287.