On the Parameterized Complexity of Clustering Incomplete Data into Subspaces of Small Rank

Robert Ganian,[1] **Iyad Kanj,**[2] **Sebastian Ordyniak,**[3] **Stefan Szeider**[1]
[1]Algorithms and Complexity Group, TU Wien, Austria. Emails: rganian@gmail.com, stefan@szeider.net
[2]School of Computing, DePaul University, USA. Email: ikanj@cs.depaul.edu
[3]Department of Computer Science, The University of Sheffield, UK. Email: sordyniak@gmail.com

Abstract

We consider a fundamental matrix completion problem where we are given an incomplete matrix and a set of constraints modeled as a CSP instance. The goal is to complete the matrix subject to the input constraints and in such a way that the complete matrix can be clustered into few subspaces with low rank. This problem generalizes several problems in data mining and machine learning, including the problem of completing a matrix into one with minimum rank. In addition to its ubiquitous applications in machine learning, the problem has strong connections to information theory, related to binary linear codes, and variants of it have been extensively studied from that perspective. We formalize the problem mentioned above and study its classical and parameterized complexity. We draw a detailed landscape of the complexity and parameterized complexity of the problem with respect to several natural parameters that are desirably small and with respect to several well-studied CSP fragments.

Introduction

Problem Definition and Motivation Motivated by a wide range of applications from data completion, clustering, and prediction, we study the computational complexity of the following fundamental COMPLETION TO SUBSPACE CLUSTERING problem (CSC):

> Given an incomplete matrix $\mathbf{M}$ over some fixed finite field, a set $\mathcal{C}$ of constraints, and $t, d \in \mathbb{N}$, find a completion of $\mathbf{M}$ satisfying all constraints in $\mathcal{C}$ and a partitioning of its rows into at most t subspaces, each of rank at most d.

CSC generalizes and/or has connections to several well-studied matrix completion problems. The first problem it generalizes is referred to as the LOW-RANK MATRIX COMPLETION problem, in which the goal is to complete the matrix into one with minimum rank, whose decision version corresponds to the constant parameter value $t = 1$ in the CSC problem. The LOW-RANK MATRIX COMPLETION problem has been extensively studied (Candès and Plan 2010; Candès and Recht 2009; Candès and Tao 2010; Fazel 2002; Hardt et al. 2014; Keshavan, Montanari, and Oh 2010a; 2010b; Recht 2011) and is known to be **NP**-hard (Peeters 1996) even for binary matrices (*i.e.*, over GF(2)) with $d = 3$.

The second problem generalized by CSC is the LOW IDENTICAL ROW MATRIX COMPLETION PROBLEM (Ganian et al. 2018); the decision version of this problem corresponds to the case of $d = 1$ in the CSC problem and is **NP**-hard already for matrices over GF(2).

CSC also has strong connections to the RANK PARTITION problem: partition the rows of a given binary matrix into two submatrices of specified sizes in a way that minimizes the sum of the ranks of the two submatrices. RANK PARTITION is closely related to the notion of the *Trellis complexity* of binary linear codes, and has been extensively studied in information theory (Horn and Kschischang 1996; Kashyap 2008; Vardy 1997a; 1997b; Jain, Mandoiu, and Vazirani 1998); in fact, settling the complexity of these problems and their variants was a long-standing open problem in that field.

Moreover, CSC reflects a recent line of research in the area of ranking problems over incomplete data, pioneered by Choi, den Broeck, and Darwiche (2015), and which has shown great promise in subsequent work (Choi, Tavabi, and Darwiche 2016; Chen et al. 2016; Choi, Shen, and Darwiche 2017; Yao, Choi, and Darwiche 2017). Finally, the subspace clustering problem, in the two settings where the matrix is complete or incomplete, has been the subject of a vast amount of research works (see, *e.g.*, Eriksson, Balzano, and Nowak, 2012; Li and Vidal, 2016; Pimentel-Alarcón et al., 2016).

Related results are also presented in very recent works which investigated the complexity of different matrix editing and clustering problems from the parameterized and approximation perspectives (Fomin, Golovach, and Panolan 2018; Fomin et al. 2019; Eiben et al. 2019). All of these papers except for the last one are concerned with the complete data setting.

Contribution We initiate the study of the complexity landscape of CSC not only from the classical viewpoint, but also from the perspective of *parameterized complexity*—a modern paradigm that allows us to make more precise statements about the asymptotic performance of algorithms and corresponding lower bounds[1]. In the parameterized setting, we

[1]We refer to the respective books for an introduction to parameterized complexity (Downey and Fellows 2013; Cygan et al. 2015)

domain	$\mathcal{C}$	d	t	k	complexity	result
2	$\emptyset$	3	1	∞	NPc	Peeters (1996)
2	$\emptyset$	2	∞	0	NPc	Theorem 1
2	$\emptyset$	∞	2	0	NPc	Theorem 2
$O(1)$	$\emptyset$	parm	parm	0	FPT	Corollary 4
$O(1)$	LinEq	parm	parm	parm	FPT	Theorem 6
2	Horn	3	1	4	NPc	Theorem 7
$O(1)$	LinEq	∞	1	parm	FPT_R	Theorem 10
$O(1)$	$\mathcal{C}^{\mathsf{st}}$	1	∞	parm	FPT	Theorem 13

Table 1: An overview of the results for CSC[$\mathcal{C}$]. Column domain: 2 means the domain size is 2, $O(1)$ means that the domain size is bounded by any constant. Column $\mathcal{C}$: $\emptyset$ means there are no constraints in place, LinEq means the CSP is a conjunction of linear equations, Horn means the CSP is a Horn formula, and $\mathcal{C}^{\mathsf{st}}$ means that the CSP belongs to a strongly tractable class. Columns d, t and k: numbers mean the respective value is set to these constants, ∞ means that the respective value is unbounded, and parm means that the value is taken as a parameter. Column complexity: the problem corresponding to the respective line is either NP-complete (NPc), fixed-parameter tractable (FPT), or randomized fixed-parameter tractable (FPT_R).

consider the complexity of a problem modulo the assumption that some parts of the input, referred to as the *parameters*, are expected/desired to be small, and the aim is to obtain algorithms which run in time $f(k') \cdot n^{\mathcal{O}(1)}$ for some computable function f of the sum k' of the parameters sum, and input size n). Parameterized problems admitting such algorithms are called *fixed-parameter tractable* (**FPT**).

We study the complexity of CSC with respect to the following three dimensions:

(1) the set of constraints $\mathcal{C}$, modeled as an instance of the constraint satisfaction problem (CSP), used to constrain the completion of the matrix;

(2) the natural parameters d and t that define the rank and the number, respectively, of the resulting subspaces; and

(3) the restrictions on the occurrences of missing entries in the incomplete matrix.

For (1), we consider several natural and well-studied types of constraints, notably linear equations (CSC[LinEq]) and various other tractable fragments of CSP.

In order to formally capture (3), we follow up on the work of Ganian et al. (2018), who introduced and motivated the *covering number* k—a natural restriction on the occurrence of missing entries in matrix completion instances.

We begin by showing that CSC remains **NP**-complete even in severely restricted settings: when there are no constraints, no missing entries (and hence the aim is merely to partition the matrix), and either $t = 2$ or $d = 2$. These lower bounds are tight, in the sense that the considered fragments become tractable for $t = 1$ or $d = 1$.

On the positive side, we show that CSC[LinEq] is **FPT** parameterized by t, d, and k, and this parameterization is in fact tight: one cannot drop any of these three parameters without losing tractability. As for the choice of constraints, we also show that the **FPT** result cannot be extended to arbitrary tractable constraints—for instance, CSC[Horn] (*i.e.*, binary instances with Horn constraints) is **NP**-hard already for $t = 1$, $d = 3$, and $k \leq 4$.

We then turn our attention to the two special cases of CSC that have been studied in previous work, namely low-rank matrix completion and distinct row minimization. In the former setting (*i.e.*, when $t = 1$), we show that the **FPT** result for CSC[LinEq] can be transferred to the setting of low-rank matrix completion without taking the target rank d as a parameter. Our result in the latter setting (*i.e.*, when $d = 1$) is even more surprising, as we show that: for *any* tractable class $\mathcal{C}$ of constraints, CSC[$\mathcal{C}$] is **FPT** parameterized by t and k. A summary of our results is provided in Table 1.

Preliminaries

Matrices For positive integers i and $j > i$, we write $[i]$ for the set $\{1, 2, \ldots, i\}$, and $i : j$ for the set $\{i, i + 1, \ldots, j\}$. For an $m \times n$ matrix $\mathbf{M}$ (*i.e.*, a matrix with m rows and n columns), and for $i \in [m]$ and $j \in [n]$, $\mathbf{M}[i, j]$ denotes the element in the i-th row and j-th column of $\mathbf{M}$. Similarly, for a vector d, we write $d[i]$ for the i-th coordinate of d. We write $\mathbf{M}[*, j]$ for the *column-vector* $(\mathbf{M}[1, j], \mathbf{M}[2, j], \ldots, \mathbf{M}[m, j])$, and $\mathbf{M}[i, *]$ for the *row-vector* $(\mathbf{M}[i, 1], \mathbf{M}[i, 2], \ldots, \mathbf{M}[i, n])$. $|\mathbf{M}|$ denotes the number of columns of $\mathbf{M}$.

The *domain* of a matrix is the set of elements that the matrix's entries belong to. We mostly consider matrices where the domain is the *finite field* GF(p) of order p; recall that if p is a prime number, such a field can be equivalently represented as the set of integers modulo p.

The *row-rank* (resp. *column-rank*) of a matrix $\mathbf{M}$ is the maximum number of linearly independent rows (resp. *columns*) in $\mathbf{M}$. It is well known that the row-rank of a matrix is equal to its column-rank, and this number is referred to as the *rank* of the matrix. We let $\mathsf{rk}(\mathbf{M})$ and $\mathsf{dr}(\mathbf{M})$ denote the rank and the number of distinct rows of the matrix $\mathbf{M}$, respectively.

An *incomplete matrix* over GF(p) is a matrix that may contain not only elements from GF(p) but also the special symbol $\bullet$. An entry is a *missing* entry if it contains $\bullet$. A (possibly incomplete) $m \times n$ matrix $\mathbf{M}'$ is *consistent* with an $m \times n$ matrix $\mathbf{M}$ if and only if, for each $i \in [m]$ and $j \in [n]$, either $\mathbf{M}'[i, j] = \mathbf{M}[i, j]$ or $\mathbf{M}'[i, j] = \bullet$.

Constraint Satisfaction Problems We will consider a variety of very general classes of *constraint satisfaction prob-*

lems, which we will define in this subsection.

An instance $\mathcal{I} = (V, D, C)$ of the constraint satisfaction problem (CSP) consists of a set V of variables, a finite domain D of values, and a set C of constraints, each $c \in C$ specifies allowed combinations of values for some subset $scope(C) \subseteq V$. The domain of considered CSP instances will be equal to the domain of the corresponding matrices.

A *partial instantiation* is an assignment $\alpha : V' \to D$ defined on some subset $V' \subseteq V$. If $V' = V$ then α is *total*. A constraint $c \in C$ can be specified by a table with all allowed instantiations or in terms of a global constraint (van Hoeve and Katriel 2006). A partial instantiation α *satisfies* a constraint c if α restricted to $scope(c)$ is allowed by c. A CSP instance $\mathcal{I}$ is *satisfiable* (or *consistent*) if there exists a total instantiation α which satisfies all constraints in C.

A class $\mathcal{C}$ of CSP instances is *strongly tractable* if for each partial instantiation α we can determine in polynomial time whether α can be extended to a total instantiation that satisfies $\mathcal{I}$. We note that most known tractable classes $\mathcal{C}$ are strongly tractable. We denote by LinEq $\subseteq$ CSP the set of all CSP instances defined via a system of linear equations over GF(p). Further, we denote by Horn the set of all Boolean CSP instances where each constraint is a Horn clause (*i.e.*, is equivalent to a disjunction of literals where at most one of them is positive). It is well-known that both LinEq and Horn are strongly tractable classes (Carbonnel and Cooper 2016).

Problem Formulation and Parameters

With the above definitions and notation of matrices and CSP at hand, we can now formally define the general matrix completion problem that we consider. Let $\mathcal{C} \subseteq$ CSP be a class of CSP instances, and p be a fixed prime.

COMPLETION TO SUBSPACE CLUSTERING (CSC[$\mathcal{C}$])	
Input:	An incomplete matrix $\mathbf{M}$ over GF(p), a CSP instance $\mathcal{I} = (\{x_1, \ldots, x_n\}, \text{GF}(p), C) \in \mathcal{C}$ where $n = \lvert\mathbf{M}\rvert$, and $d, t \in \mathbb{N}$.
Task:	Find a matrix $\mathbf{M}'$ such that (i) $\mathbf{M}'$ is consistent with $\mathbf{M}$; (ii) the rows of $\mathbf{M}'$ can be partitioned into at most t submatrices each of rank at most d; and (iii) for each row vector $\mathbf{M}'[i, *]$, the total instantiation $\alpha : x_j \mapsto \mathbf{M}'[i, j]$ satisfies $\mathcal{I}$.

Without loss of generality, we assume that the rows of the input matrix are pairwise distinct. To avoid any confusion, we remark that while the focus lies on the completion part of the problem (*i.e.*, finding $\mathbf{M}'$), all our algorithms can also output a valid partitioning satisfying property (ii).

It is easy to observe that CSC[$\mathcal{C}$] is at least as hard as $\mathcal{C}$. Indeed, an instance $\mathcal{I} \in \mathcal{C}$ is satisfiable if and only if the $1 \times m$ matrix with all entries containing $\bullet$ is a yes-instance of CSC[$\mathcal{C}$]. Hence, it is necessary to restrict $\mathcal{C}$ to a tractable class of instances. By a similar argument, it follows that $\mathcal{C}$ must—in fact—be strongly tractable (in particular, one can model partial instantiations by replacing $\bullet$ with a specific value from D). We will use the notation CSC[$\emptyset$] to refer to instances of CSC with no constraints.

Problem Parameterizations As mentioned earlier, we will require a parameter that restricts the placement of missing entries in the input matrix. Such a restriction is necessary since even the simplest matrix completion problems become intractable when missing entries are unrestricted.

The parameter we consider here is the *covering number* (Ganian et al. 2018), which we will henceforth denote as k. The covering number of a matrix is the minimum number of rows and columns required to cover all missing entries in the matrix[2]. The parameter has recently been used to obtain a complexity map for two subcases of CSC without constraints (Ganian et al. 2018), and is motivated by situations where a known matrix is extended by a few new rows or columns for which only partial information is available.

It is known that k can be computed in polynomial time (Ganian et al. 2018, Proposition 2).

The Complexity of Subspace Partitioning

In this section, we draw a parameterized complexity landscape for CSC. For our initial lower bounds, we consider the restriction of CSC[$\emptyset$] over GF(2) where $k = 0$; that is, there are no missing entries, and the problem merely asks for a partitioning of the matrix rows into at most t subspaces, each of rank at most d. We will refer to this problem as BASIC SUBSPACE CLUSTERING (BSC). Note that the hardness results we obtain can trivially be lifted to the more general settings of CSC.

We start by showing that BSC remains **NP**-hard even when $d = 2$, and that it also remains **NP**-hard even when $t = 2$.

Theorem 1. BSC *is* **NP**-*hard for* $d = 2$.

Proof. We prove the theorem by giving a polynomial-time reduction from the **NP**-hard problem (Holyer 1981) EDGE-PARTITION INTO TRIANGLES: Given an undirected graph G, decide whether $E(G)$ can be partitioned into triangles. Given an instance G of EDGE-PARTITION INTO TRIANGLES, where $V(G) = \{v_1, \ldots, v_n\}$ and $E(G) = \{e_1, \ldots, e_m\}$, we construct an instance $\mathcal{I}$ of BSC as follows. The matrix $\mathbf{M}$ has m rows and n columns, corresponding to the edges and vertices of G, respectively; w.l.o.g., we label the rows and columns by the indices of their corresponding edges and vertices, respectively. The matrix $\mathcal{I}$ is basically the characteristic matrix of $E(G)$ w.r.t. $V(G)$, in which $\mathbf{M}[i, j] = 1$ iff e_i is incident to v_j in G. We set $d = 2$ and $t = m/3$. This completes the construction of $\mathcal{I}$, which clearly can be performed in polynomial time.

Observe that each row in $\mathbf{M}$ contains exactly two 1's, and that a set of 3 rows in $\mathbf{M}$ has rank 2 iff the edges corresponding to the 3 rows form a triangle/cycle in G. With the aforementioned observation in mind, it is now easy to verify that a partitioning of $E(G)$ into $m/3$ triangles corresponds to a partitioning of the rows of $\mathbf{M}$ into $m/3$ subspaces each of rank exactly 2. On the other hand, if the rows of $\mathbf{M}$ can be partitioned into at most $m/3$ subspaces each of rank at most 2, then from the above observation combined with the fact that any 4 rows of $\mathbf{M}$ form a subspace with rank greater than 2, it follows that the rows of $\mathbf{M}$ can be partitioned into exactly $m/3$ subspaces each of rank exactly 2; this partitioning corresponds to a partitioning of $E(G)$ into $m/3$ triangles. □

[2] An entry at position $\mathbf{M}[i, j]$ is *covered* by row i and column j.

Theorem 2. BSC *is* **NP**-*hard even for* $t = 2$.

Proof. (Sketch) The polynomial-time reduction is from an **NP**-hard restriction of MAX CUT, and is an adaptation of the reduction from MAX CUT given by Horn and Kschischang (1996) to show that the $n/2$-PARTITION RANK PERMUTATION problem ($n/2$-PRP) is **NP**-hard, which is, in turn, an adaptation of a reduction given by Garey, Johnson, and Stockmeyer (1976) to show that MINIMUM CUT INTO EQUAL-SIZED SUBSETS is **NP**-hard. (Recall that in the unweighted MAX CUT problem, we are given an undirected graph G and $w \in \mathbb{N}$, and the question is whether the vertex-set of G can be partitioned into two parts such that the number of edges across the partition is at least w.) In the $n/2$-PRP problem, we are given an $m \times n$ binary matrix and $w \in \mathbb{N}$, and the question is whether the columns of the matrix can be partitioned (or permuted) into two equal-size parts, each with $n/2$ columns, such that the sum of the ranks of the two submatrices induced by the two parts is at most w. Since in an instance of BSC with $t = 2$ we can transpose the matrix and instead ask whether the columns of the transpose matrix can be partitioned into two subspaces each with rank at most d, the only differences between BSC and $n/2$-PRP are the requirement that the two submatrices have equal number of columns and the requirement that the sum of their ranks is upper bounded by a given number, as opposed to that each of their ranks is upper bounded by the same given number. We will sketch how the proof of the **NP**-hardness of $n/2$-PRP can be modified to work for the restriction of BSC to $t = 2$. As noted above, in what follows, we may assume that, for an instance of BSC, we ask for a partition of the matrix columns (not the rows) into two subspaces each of rank at most d; denote this restriction of BSC as 2-BSC.

The reduction is from a restriction of MAX CUT to instances (G, w) satisfying three properties: (i) The edge-complement graph of G is connected, (ii) $|E(G)| - |V(G)|$ is even, and (iii) $|V(G)|^2 - w$ is even. Such a restriction can be easily shown to be **NP**-hard. For instance, one can start from an instance of the **NP**-hard problem (Garey and Johnson 1990) MAX CUT ON CUBIC GRAPHS, which can be easily verified to satisfy (i), and add a small gadget to make it satisfy (ii) and (iii), in case it does not already satisfy them. (For example, assuming that it does not satisfy (ii), to make it satisfy it without violating (i), a triangle and an edge joining a vertex of the triangle to a vertex in G can be added, and we increase w by 3. Now assuming that the resulting graph does not satisfy (iii), to make it satisfy it without violating (i) and (ii), we can add a new triangle and two edges between two vertices of the triangle and the same vertex in the graph, and increase w by 4.) This certainly results in an **NP**-hard restriction of MAX CUT; denote this restriction as RES-MAX CUT. The reason for using such a restriction of MAX CUT (as opposed to MAX CUT) is that (i) it is crucial for an argument in the adapted **NP**-hardness proof by Horn and Kschischang, (ii) it simplifies the construction (as there will be no need anymore for distinguishing two cases in the construction), and (iii) it is needed for ensuring that the upper bound on the sum of the ranks is even, and hence, can be split equally into an upper bound on the rank of each subspace.

Next, we briefly discuss the required additional changes in the **NP**-hardness proof for $n/2$-PRP to make it work for 2-BSC. We follow the terminology of Horn and Kschischang as much as possible. Let (G, w) be an instance of RES-MAX CUT, where G has M vertices and N edges. We construct the following graph G' from G, which is the same construction as that of Horn and Kschischang, albeit without the need to distinguish the two cases based on whether or not $|E(G)| - |V(G)|$ is even. Let $V(G) = \{v_1, \ldots, v_M\}$. Introduce a new set of M vertices $\{v_{M+1}, \ldots, v_{2M}\}$. Start with $V(G) \cup \{v_{M+1}, \ldots, v_{2M}\}$ (and no edges), and add the following edges: (1) Form a clique on $\{v_{M+1}, \ldots, v_{2M}\}$; (2) form a complete bipartite graph with $V(G)$ as one part and $\{v_{M+1}, \ldots, v_{2M}\}$ as the other; and (3) add the complement of the edge-set of G between the vertices in $V(G)$. Finally, replace each v_i, $i \in [2M]$, with a clique C_i on M^3 many vertices $\{c_{i,j} \mid j \in [M^3]\}$ and connect vertex $c_{i,j}$ in C_i to vertex $c_{j,i}$ in C_j, for $i, j \in [2M]$, iff v_i and v_j are connected. Let the resulting graph be G'. Finally, let $\mathbf{M}$ be the incident binary matrix whose rows correspond to the vertices of G' and columns to the edges of G, and such that an entry in $\mathbf{M}$ is 1 iff the corresponding vertex and edge are incident in G'; set $t = 2$ and $d = M^4 - (M^2 - w)/2 - 1$.

From this point on, the proof of Horn and Kschischang follows with some minor modifications. □

The above results imply the parameterized intractability (*i.e.*, para-**NP**-hardness) of BSC w.r.t. each of the parameterizations by d and t. This begs the question about the parameterized complexity of BSC parameterized by both d and t (*i.e.*, by $d + t$). The following simple observation helps us answer the aforementioned question:

Observation 3. *Let* $\mathbf{M}$ *be a complete matrix with distinct rows over some finite domain* Ω. *Then any subspace of* $\mathbf{M}$ *of rank at most* d *contains at most* $|\Omega|^d$ *rows.*

The above observation follows by fixing a basis of the subspace of rank at most d, and noting that each vector/row in the subspace (including the basis vectors) can be written as a linear combination of the (at most) d vectors in the basis.

Corollary 4. BSC *is* **FPT** *parameterized by* $d + t$.

Proof. Observation 3 implies that the input matrix $\mathbf{M}$ in any yes-instance of BSC must have at most $t \cdot |\Omega|^d$ rows; otherwise, the instance can be rejected. This means that the instance can be solved by brute force in **FPT**-time. □

Next, we consider the possibility of lifting this **FPT** result to the more general setting of CSC[$\mathcal{C}$]. However, even for the case when $\mathcal{C} = \emptyset$, the result of Peeters (1996) implies the para-**NP**-hardness of the problem parameterized by $d + t$, as they show the **NP**-hardness of the problem of completing a binary matrix into one of rank 3. This implies:

Observation 5. CSC[$\emptyset$] *is* **NP**-*hard even for* $t = 1$ *and* $d = 3$.

It follows from the above observation that restrictions must be imposed on the missing entries in the matrix if any **FPT** results are to be obtained. As the main positive result for this section, we show that parameterizing by $d + t + k$ allows us

to obtain a fixed-parameter algorithm not only for CSC[$\emptyset$], but also in the presence of linear equations.

Theorem 6. CSC[LinEq] *is* **FPT** *parameterized by* $d+t+k$.

Proof. We give an **FPT** algorithm for CSC[LinEq] parameterized by $d + t + k$. Let $(\mathbf{M}, \Gamma, t, d)$ be an instance of CSC[LinEq], where $\Gamma \in$ LinEq is a set of linear constraints (equations) having to hold at each row, and as before, let Ω denote the domain from which the matrix values are drawn. Let R and C denote the sets of the rows and columns in $\mathbf{M}$, respectively, that cover the missing entries, and note that $|R| + |C| \leq k$.

First, we upper bound the number of rows of $\mathbf{M}$ by a function of the parameter, in any yes-instance $(\mathbf{M}, \Gamma, t, d)$ of CSC[LinEq]. It suffices to upper bound $|\overline{R}|$ by a function of the parameter, where $\overline{R}$ is the set of rows in $\mathbf{M}$ that are not in R. Partition $|\overline{R}|$ into groups such that all rows in the same group agree on all the entries in the columns in C. The number of resulting groups is at most $(|\Omega| + 1)^{|C|} \leq (|\Omega| + 1)^k$, as each entry whose column is in C contains either $\bullet$ or a domain value. Fix a group Y. Since $\mathbf{M}$ does not contain identical rows, any two rows in Y must differ on at least one column not in C, and hence, must be completed into distinct rows in any solution of $(\mathbf{M}, \Gamma, t, d)$. By Observation 3, the number of rows in any subspace of $\mathbf{M}$ of rank at most d is $|\Omega|^d$, and hence the total number of rows in any completion of $\mathbf{M}$ for a yes-instance $(\mathbf{M}, \Gamma, t, d)$ is at most $t \cdot |\Omega|^d$. We conclude that the number of rows in group Y is at most $t \cdot |\Omega|^d$ in any yes-instance of the problem. It follows that the total number of rows in $|\overline{R}|$ is at most $t \cdot |\Omega|^d \cdot (|\Omega| + 1)^{|C|}$, and hence the number of rows in $\mathbf{M}$ is at most $t \cdot |\Omega|^d \cdot (|\Omega| + 1)^{|C|} + k$, which is a function of the parameter, in any yes-instance $(\mathbf{M}, \Gamma, t, d)$ of CSC[LinEq]; otherwise, we can reject the instance.

Suppose now that $\mathbf{M}$ meets the above upper bound on the number of rows. Next, we enumerate all partitions of the rows of $\mathbf{M}$ into t parts. Clearly, this enumeration takes **FPT**-time. Let these parts be $R_1, \ldots, R_s$, where $s \leq t$.

As the last step, for an enumeration $R_1, \ldots, R_s$, we need to check if each R_i, $i \in [s]$, has rank at most d; if this is the case, we accept the instance. If no enumeration leads to acceptance, we reject the instance. To check whether a subset R_i, $i \in [s]$, of vectors has rank at most d, we enumerate each subset B of at most d vectors in R_i as basis for R_i; note that the total number of vectors in R_i is upper bounded by a function of the parameter, and hence so is the number of subsets that needs to be enumerated. We introduce a variable (over Ω) for each missing entry in a row of R_i; let X be the set of the introduced variables. For each (remaining) vector $\vec{v} \in R_i \setminus B$, we enumerate the at most d coefficients over Ω of $\vec{v}$ that result from writing $\vec{v}$ as a linear combination of the vectors in B. We introduce n linear equations, over (a subset of) the variables in X, corresponding to the equations resulting from writing each entry in $\vec{v}$ as a linear combination of the corresponding entries in the vectors in B, w.r.t. the enumerated $\leq d$ coefficients for $\vec{v}$. Let Γ_0 be the system of linear equations obtained over all vectors $\vec{v} \in R_i \setminus B$. Finally, for each row $\vec{v} \in R_i$, we add copies of the equations in Γ (over the terms corresponding to the entries of $\vec{v}$) to ensure that every row satisfies the constraints. We solve Γ_0 together with the copies of Γ for each row in R_i in polynomial time (*e.g.*, using Gaussian elimination). Clearly, R_i has rank at most d with each row satisfying the constraints in Γ iff one of the resulting linear systems, over all enumerations, has a solution. This step takes **FPT**-time, and so does the whole algorithm. □

	x_1	x_2	x_3	x_4	x_5	$n+1$	$n+2$	$n+3$
1.	$\bullet$	$\bullet$	$\bullet$	$\bullet$	$\bullet$	0	0	1
2.	$\bullet$	$\bullet$	$\bullet$	$\bullet$	$\bullet$	1	1	1
3.	1	1	0	1	0	1	1	1
4.	0	0	0	0	0	0	1	1

Figure 1: The matrix M in the construction used in Theorem 7 for the instance of SAT^R with variables $x_1, \ldots, x_5$, one positive clause $\{x_1, x_2, x_4\}$, and two negative clauses $\{\overline{x_1}, \overline{x_2}, \overline{x_4}\}$ and $\{\overline{x_3}, \overline{x_5}\}$.

Theorem 6 begs the question of whether there is something specific about linear equations in this setting, or whether the result can be lifted to any strongly tractable class of CSPs. As our last result in this section, we show that the latter is not possible—already for the highly restrictive class of Horn CSPs, CSC[Horn] becomes **NP**-hard even when $t = 1$ and the number of rows (which naturally upper-bounds k and d) is at most 4.

Theorem 7. CSC[Horn] *is* **NP**-*hard even when* $t = 1$ *and the input matrix has 4 rows.*

We will prove the above theorem via a polynomial-time reduction from a restriction of SAT, referred to as SAT^R, which we first show to be **NP**-hard. Call a clause in a CNF formula *positive* (resp. *negative*) if it consists of only positive (resp. negative) literals. An instance of SAT^R consists of a CNF formula F satisfying the following three properties: (i) each clause in F is either positive or negative; (ii) the positive clauses are pairwise disjoint; and (iii) for each positive clause $C = \{x_1, \ldots, x_r\}$ there is a negative clause $C' = \{\overline{x_1}, \ldots, \overline{x_r}\}$ in F over the same variables, referred to as the *dual* of C.

Lemma 8. SAT^R *is* **NP**-*complete.*

Proof Sketch for Theorem 7. Let F be an instance of SAT^R over n variables $x_1, \ldots, x_n$. Denote by P and N the sets of positive and negative clauses in F, respectively. We construct a matrix $\mathbf{M}$ with 4 rows and $n + 3$ columns, where column i of $\mathbf{M}$ corresponds to variable x_i, for $i \in [n]$. The entries of $\mathbf{M}$ are defined as follows. First, the 4 entries in column $n + 3$ are all set to 1. In row 1, the entries in the first n columns (corresponding to the variables) are set to $\bullet$, and the two entries in columns $n + 1$ and $n + 2$ are set to 0. In row 2, the entries in the first n columns are set to $\bullet$, the entry in column $n + 1$ is set to 1, and the entry in column $n + 2$ is set to 0. In row 3, each entry corresponding to a variable that appears in P is set to 1, all other entries in columns $1, \ldots, n$ (corresponding to variables) are set to 0, and both entries in columns $n + 1$ and $n + 2$ are set to 1. Finally, in

row 4, all entries in columns $1, \ldots, n+1$ are set to 0, and the entry in column $n+2$ is set to 1. This completes the construction of $\mathbf{M}$. We refer to Figure 1 for an example of this construction.

The Horn formula H associated with the instance of CSC[Horn], is defined as follows. The variables of F are also variables in H, where variable x_i is associated with column i in $\mathbf{M}$. We create the new Boolean variables x_{n+1}, x_{n+2} in H that are associated with columns $n+1, n+2$ of $\mathbf{M}$, respectively. The clauses of H are defined as follows. For each clause $C \in N$, create the clause $C \cup \{x_{n+1}\}$ and add it to H; let N' be the set of clauses in H created this way. For each clause $C \in P$, create the clause $C' \cup \{x_{n+2}\}$ and add it to H, where C' is the dual of C (*i.e.*, the clause consisting of the negations of the positive literals in C); let P' be the set of clauses in H created this way. This completes the construction of H. Finally, we set $d = 3$. Let $(\mathbf{M}, H, d)$ be the resulting instance of CSC[Horn]. Clearly, $(\mathbf{M}, H, d)$ can be constructed from F in polynomial time. We have:

Claim 9. *In any valid completion of* $\mathbf{M}$ *into a matrix* $\mathbf{M}'$*, we have* $\mathsf{rk}(\mathbf{M}') = 3$ *or* $\mathsf{rk}(M') = 4$*. Moreover,* $\mathsf{rk}(\mathbf{M}') = 3$ *iff* $\mathbf{M}'[1,*] = \mathbf{M}'[2,*] + \mathbf{M}'[3,*] + \mathbf{M}'[4,*]$ *(addition in GF*(2)*), which is equivalent to saying that* $\mathbf{M}'[1,i] = \mathbf{M}'[2,i]$ *iff* $\mathbf{M}'[3,i] = 0$*, for every* $i \in [n]$*.*

To show the correctness of the above claim, we make the following observations. Since (the complete) rows 3 and 4 of $\mathbf{M}$ are independent, and since adding any two rows of $\mathbf{M}$ results in a 0 entry in column $n+3$, which is 1 for all rows of $\mathbf{M}$, any completion of $\mathbf{M}$ results in a matrix of rank at least 3, and hence, of rank 3 or 4. This shows the first part of the claim. Now suppose that $\mathbf{M}$ has a valid completion into a matrix $\mathbf{M}'$ of rank 3. By the same token as above, we can assume that the completed rows 2, 3, and 4 of $\mathbf{M}'$ form a basis for the rows of $\mathbf{M}'$, and hence we have $\mathbf{M}'[1,*] = \mathbf{M}'[2,*] + \mathbf{M}'[3,*] + \mathbf{M}'[4,*]$. Now since row 4 of $\mathbf{M}'$ contains all 0's in columns $1, \ldots, n$, it follows from the above equation that $\mathbf{M}'[1,i]$ and $\mathbf{M}'[2,i]$ agree on precisely those columns $i \in [n]$ for which $\mathbf{M}'[3,i] = 0$.

Now suppose that F is satisfiable, and let τ be a satisfying assignment for F. Consider the completion of $\mathbf{M}$ into a matrix $\mathbf{M}'$ that assigns to entry $\mathbf{M}[1,i]$, for $i \in [n]$, the value assigned by τ to x_i, and completes row 2 of $\mathbf{M}$ in accordance with the equation $\mathbf{M}'[1,i] = \mathbf{M}'[2,i]$ iff $\mathbf{M}'[3,i] = 0$, for $i \in [n]$. Clearly, because the previous equation is satisfied, we have $\mathsf{rk}(\mathbf{M}') = 3$. It is not difficult now to show that each row in $\mathbf{M}'$ satisfies H.

To prove the converse, suppose that for the instance $(\mathbf{M}, H, d)$ of CSC[Horn] the matrix $\mathbf{M}$ has a valid completion $\mathbf{M}'$ with $\mathsf{rk}(\mathbf{M}') = 3$. Let τ be the truth assignment to F that assigns variable x_i the value $\mathbf{M}'[1,i]$, for $i \in [n]$. It can be easily verified that τ satisfies F. □

Special Cases of CSC

In the second part of our paper, we turn our attention to the two notable special cases of CSC that have been studied in previous work: low-rank matrix completion and distinct row minimization.

$$\begin{pmatrix} \bullet & 7 & \bullet & 6 & 9 & \bullet \\ 6 & 7 & 0 & 6 & \bullet & 7 \\ \bullet & 7 & \bullet & \bullet & 9 & \bullet \\ 6 & 7 & 0 & \bullet & 9 & 8 \\ \bullet & 7 & \bullet & 6 & 6 & \bullet \end{pmatrix}$$

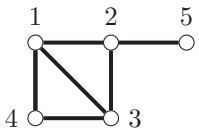

Figure 2: Illustration of a matrix and its compatibility graph. The vertex label indicates the corresponding row number.

Low-Rank Matrix Completion We consider the special case where $t = 1$, *i.e.*, the task of minimizing the rank of the matrix. We will refer to this problem as LOW-RANK MATRIX COMPLETION (CSC_R). Note that the lower bound presented in Theorem 7 immediately carries over to this setting as well, implying that CSC_R[Horn] is intractable. However, we will show that in contrast to the more general case of CSC[LinEq] (which requires all three parameters d, t, and k), CSC_R[LinEq] is already fixed-parameter tractable parameterized only by k; note that $\text{CSC}_\text{R}[\emptyset]$ is **NP**-hard due to Observation 5.

Theorem 10. CSC_R[LinEq] *parameterized by* k *is* **FPT**.

Proof Sketch. Let $\mathbf{M}$ be the input matrix and Γ be the set of linear constraints imposed over the rows of $\mathbf{M}$. At a high level, we follow a similar strategy as that of Ganian et al. (2018) to establish the tractability of $\text{CSC}_\text{R}[\emptyset]$. In particular, we will compute sets R and C of covering rows and columns (where $|R| + |C| = k$), and branch over certain *signatures* that capture information about the dependencies among the rows in R and columns in C. In each branch, we obtain a system of equations that needs to be solved in order to determine whether the signatures are valid—*i.e.*, whether it is possible to choose dependent rows and columns in the way specified by the signature while satisfying all constraints. A key distinction is that when checking for the validity of a signature, here we also need to make sure that all equalities in Γ are satisfied. Once we determine which signatures are valid, we choose one that minimizes the total rank. □

Distinct Row Clustering Finally, we turn to the special case where $d = 1$, *i.e.*, the task of minimizing the number of distinct rows in the matrix. We will refer to this problem as DISTINCT ROW CLUSTERING (CSC_DR). Here, we can obtain a result which is surprisingly generic: for any strongly tractable class $\mathcal{C}$ of CSP, $\text{CSC}_\text{DR}[\mathcal{C}]$ is **FPT** parameterized merely by k. We start with a brief introduction of the compatibility graph and treewidth, two concepts that are key tools for our result.

Let $\mathbf{M}$ be an incomplete matrix over GF(p). We say that two rows of $\mathbf{M}$ are *compatible* if whenever the two rows differ at some entry, then one of the rows has a $\bullet$ at that entry. The *compatibility graph* of $\mathbf{M}$ (Ganian et al. 2018), denoted by $G(\mathbf{M})$, is the undirected graph whose vertices correspond to the row indices of $\mathbf{M}$ and in which there is an edge between two vertices if and only if their two corresponding rows are compatible. An illustration is provided in Figure 2.

Let $\mathcal{C}$ be a class of CSP instances and $\mathcal{I} = (\mathbf{M}, I_C, t)$ be an instance of $\text{CSC}_\text{DR}[\mathcal{C}]$. We say that a set of rows R of $\mathbf{M}$ is *compatible* if all pairs of rows in R are pairwise compatible.

For a set of rows R and a column index c, let $E(R, c)$ be the set of all values occurring at column c in any row in R, *i.e.*, $E(R, c) = \{ r[c] : r \in R \}$. Note that if R is a set of compatible rows, then $E(R, c)$ contains at most one value other than $\bullet$ for every column index c. Hence, for a set R of compatible rows and a column index c, we can define $U(R, c)$ to be equal to the unique value in $E(R, c) \setminus \{\bullet\}$ if $E(R, c) \setminus \{\bullet\} \neq \emptyset$ and equal to $\bullet$, otherwise. Moreover, we denote by $U(R)$ the unique row defined by $U(R)[c] = U(R, c)$ for every column index c.

Observation 11. *A set R of rows of $\mathbf{M}$ can be completed to the same row if and only if $G(\mathbf{M}[R, *])$ forms a clique and the partial instantiation given by $U(R)$ can be extended to a total instantiation that satisfies I_C.*

The above observation implies that a solution for $\mathcal{I}$ can be thought of as a consistent partition $\mathcal{P}$ of the vertex set of $G(\mathbf{M})$ into cliques, where *consistent* means that the (partial) instantiation represented by $U(\alpha(P))$ can be extended to a total instantiation satisfying I_C, for every $P \in \mathcal{P}$, where α denotes the natural bijection from the set of vertices of $G(\mathbf{M})$ to the set R of rows of $\mathbf{M}$.

A *tree-decomposition* $\mathcal{T}$ of a graph $G = (V, E)$ is a tuple (T, χ), where T is a tree and χ is a function that assigns each tree node x a set $\chi(x) \subseteq V$ of vertices such that the following conditions are met: (i) For every vertex $v \in V(G)$, the set of tree nodes x with $v \in \chi(x)$ forms a non-empty subtree of T. (ii) For every edge $uv \in E(G)$ there is a tree node x such that $u, v \in \chi(x)$. We call the sets $\chi(x)$ *bags*, where $\chi(x)$ is the bag associated with x. The *width* of a tree-decomposition (T, χ) is the size of a largest bag minus 1. A tree-decomposition of minimum width is called *optimal*. The *treewidth* of a graph G, denoted by $\mathsf{tw}(G)$, is the width of an optimal tree decomposition of G.

The following lemma provides us with the main tool needed for our tractability result as it allows us to reduce DISTINCT ROW CLUSTERING to the task of obtaining an upper bound on the treewidth of the compatibility graph.

Lemma 12. *Let $\mathcal{C}^{\mathsf{st}}$ be a strongly tractable class of CSP instances. Then $\mathrm{CSC}_{\mathrm{DR}}[\mathcal{C}^{\mathsf{st}}]$ parameterized by the treewidth of the compatibility graph is fixed-parameter tractable.*

Sketch of Proof. Let $\mathcal{I} = (\mathbf{M}, I_C, t)$ with $I_C = (\{x_1, \ldots, x_n\}, D, C)$ be the given instance of $\mathrm{CSC}_{\mathrm{DR}}[\mathcal{C}^{\mathsf{st}}]$ and let G be its associated compatibility graph, *i.e.*, $G = G(\mathbf{M})$. We will show the lemma using a dynamic programming algorithm on a tree-decomposition of G. Since it is well-known (Kloks 1994; Bodlaender 1996; Bodlaender et al. 2016) that a tree decomposition of width ω can be computed in fixed-parameter tractable-time parameterized by ω, we can in the following assume that we are given a tree decomposition (T, χ) of G of width ω.

For a subgraph H of G, we say that $\mathcal{P}$ is a *partition of a H into cliques* if $\{V(P) : P \in \mathcal{P}\}$ partitions the vertex set of $V(H)$ and $H[P]$ is a clique for every $P \in \mathcal{P}$. If it holds additionally that the partial instantiation given by $U(\alpha(V(P)))$ can be extended to a total instantiation satisfying I_C, then we say that $\mathcal{P}$ is a *consistent* partition of H into cliques. For every node $x \in V(T)$, we will compute the set $\mathcal{R}(x)$ of records containing all pairs $(\mathcal{P}, c)$ such that: (i) $\mathcal{P}$ is a consistent partition of $G[\chi(x)]$ into cliques, and (ii) c is the minimum integer such that $G[\chi(T_x)]$ has a consistent partition $\mathcal{P}'$ into c cliques with $\mathcal{P} = (\{ P' \cap \chi(n) : P' \in \mathcal{P}' \} \setminus \{\emptyset\})$. Note that given $\mathcal{R}(x)$ for every node $x \in V(T)$, we can easily obtain a solution for $\mathcal{I}$. In particular, $\mathcal{I}$ is a yes-instance if and only if $\mathcal{R}(r)$, where r is the root of T, contains a record $(\emptyset, t')$ with $t' \leq t$. □

We can now show the main result of this section.

Theorem 13. *Let $\mathcal{C}_D^{\mathsf{st}}$ be a class of strongly tractable CSP instances over a finite domain Ω. Then $\mathrm{CSC}_{\mathrm{DR}}[\mathcal{C}^{\mathsf{st}}]$ parameterized by k is* **FPT**.

Proof. Let $\mathcal{I} = (\mathbf{M}, I_C, t)$ with $I_C = (\{x_1, \ldots, x_n\}, D, C)$ be the given instance of $\mathrm{CSC}_{\mathrm{DR}}[\mathcal{C}]$, let G be its associated compatibility graph, *i.e.*, $G = G(\mathbf{M})$. We begin by computing a set $R_\bullet$ of rows and $C_\bullet$ of columns such that $|R_\bullet \cup C_\bullet| \leq k$ and every occurrence of $\bullet$ in $\mathbf{M}$ is contained in a row or column in $R_\bullet \cup C_\bullet$. Let R and C be the set of rows and columns of $\mathbf{M}$, respectively. Let $\mathcal{P}$ be the unique partition of $R \setminus R_\bullet$ such that two rows r and r' belong to the same set in $\mathcal{P}$ if and only if they are identical on all columns in $C \setminus C_\bullet$. Then $|P| \leq (|\Omega| + 1)^k$, for every $P \in \mathcal{P}$, since two rows in P can differ on at most $|C_\bullet| \leq k$ entries, each having $(|\Omega|+1)$ values to be chosen from. Moreover, any two rows in $R \setminus R_\bullet$ that are not contained in the same set in $\mathcal{P}$ are not compatible, which implies that they appear in different components of $G \setminus R_\bullet$ and hence the set of vertices in every component of $G \setminus R_\bullet$ is a subset of P, for some $P \in \mathcal{P}$. It is now straightforward to show that $\mathsf{tw}(G) \leq k + (|\Omega| + 1)^k$, and hence, $\mathsf{tw}(G)$ is bounded by a function of the parameter k. The theorem now follows by Lemma 12. □

Conclusion

We initiated the study of a fundamental matrix clustering problem, in the incomplete data setting, and subject to constraints imposed on the completed matrix. Here, the addition of constraints expands the applications of the problem in a similar manner as in preference learning (Choi, den Broeck, and Darwiche 2015).

We investigated the parameterized complexity of the problem with respect to natural parameters and painted a detailed landscape of its complexity. Our findings give tight parameterized complexity results with respect to the parameters under consideration, as well as show the **NP**-completeness of several important matrix partitioning problems. Many of the obtained fixed-parameter tractability results can be lifted to the setting where the completion is subject to a tractable CSP that satisfies mild additional restrictions.

We hope that our encouraging results will evoke further research on this general topic, as there is much room for generalization and extension. For instance, a natural extension is to consider the case where the domain is part of the input, as this would allow the use of global constraints such as the all-different and permutation constraints. Moreover, a natural open problem that ensues from our work is to determine the parameterized complexity of $\mathrm{CSC}[\mathcal{C}]$ where $\mathcal{C}$ is the class of bijunctive constraints.

Acknowledgements Robert Ganian acknowledges support from the Austrian Science Fund (FWF, Project P 31336: **NFPC**). Stefan Szeider acknowledges the support of the Austrian Research Funds (FWF), Project P 32441.

References

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

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

Candès, E. J., and Plan, Y. 2010. Matrix completion with noise. *Proceedings of the IEEE* 98(6):925–936.

Candès, E. J., and Recht, B. 2009. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* 9(6):717–772.

Candès, E. J., and Tao, T. 2010. The power of convex relaxation: near-optimal matrix completion. *IEEE Trans. Information Theory* 56(5):2053–2080.

Carbonnel, C., and Cooper, M. C. 2016. Tractability in constraint satisfaction problems: a survey. *Constraints* 21(2):115–144.

Chen, E. Y.; Shen, Y.; Choi, A.; and Darwiche, A. 2016. Learning Bayesian networks with ancestral constraints. In *NIPS 2016*, 2325–2333.

Choi, A.; den Broeck, G. V.; and Darwiche, A. 2015. Tractable learning for structured probability spaces: A case study in learning preference distributions. In *IJCAI 2015*, 2861–2868.

Choi, A.; Shen, Y.; and Darwiche, A. 2017. Tractability in structured probability spaces. In *NIPS 2017*, 3480–3488.

Choi, A.; Tavabi, N.; and Darwiche, A. 2016. Structured features in naive Bayes classification. In *AAAI 2016*, 3233–3240.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer.

Downey, R. G., and Fellows, M. R. 2013. *Fundamentals of Parameterized Complexity*. Springer.

Eiben, E.; Ganian, R.; Kanj, I.; Ordyniak, S.; and Szeider, S. 2019. On clustering incomplete data. *CoRR* abs/1911.01465.

Eriksson, B.; Balzano, L.; and Nowak, R. 2012. High-rank matrix completion. In *AISTATS 2012*, 373–381.

Fazel, M. 2002. *Matrix rank minimization with applications*. Ph.D. Dissertation, Stanford University.

Fomin, F. V.; Golovach, P. A.; Lokshtanov, D.; Panolan, F.; and Saurabh, S. 2019. Approximation schemes for low-rank binary matrix approximation problems. *ACM Trans. Algorithms* 16(1):12:1–12:39.

Fomin, F. V.; Golovach, P. A.; and Panolan, F. 2018. Parameterized low-rank binary matrix approximation. In *ICALP 2018*, 53:1–53:16.

Ganian, R.; Kanj, I.; Ordyniak, S.; and Szeider, S. 2018. Parameterized algorithms for the matrix completion problem. In *ICML 2018*, 1642–1651.

Garey, M. R., and Johnson, D. S. 1990. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co.

Garey, M. R.; Johnson, D. S.; and Stockmeyer, L. J. 1976. Some simplified NP-complete graph problems. *Theor. Comput. Sci.* 1(3):237–267.

Hardt, M.; Meka, R.; Raghavendra, P.; and Weitz, B. 2014. Computational limits for matrix completion. In *The 27th Conference on Learning Theory*, volume 35, 703–725.

van Hoeve, W.-J., and Katriel, I. 2006. Global constraints. In Rossi, F.; van Beek, P.; and Walsh, T., eds., *Handbook of Constraint Programming*. Elsevier. chapter 6.

Holyer, I. 1981. The NP-completeness of some edge-partition problems. *SIAM J. Comput.* 10(4):713–717.

Horn, G. B., and Kschischang, F. R. 1996. On the intractability of permuting a block code to minimize trellis complexity. *IEEE Transactions on Information Theory* 42(6):2042–2048.

Jain, K.; Mandoiu, I. I.; and Vazirani, V. V. 1998. The 'art of trellis decoding' is computationally hard for large fields. *IEEE Transactions Information Theory* 44(3):1211–1214.

Kashyap, N. 2008. Matroid pathwidth and code trellis complexity. *SIAM J. Discrete Math.* 22(1):256–272.

Keshavan, R. H.; Montanari, A.; and Oh, S. 2010a. Matrix completion from a few entries. *IEEE Trans. Information Theory* 56(6):2980–2998.

Keshavan, R. H.; Montanari, A.; and Oh, S. 2010b. Matrix completion from noisy entries. *JMLR* 11:2057–2078.

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

Li, C., and Vidal, R. 2016. A structured sparse plus structured low-rank framework for subspace clustering and completion. *IEEE Transactions on Signal Processing* 64(24):6557–6570.

Peeters, R. 1996. Orthogonal representations over finite fields and the chromatic number of graphs. *Combinatorica* 16(3):417–431.

Pimentel-Alarcón, D.; Balzano, L.; Marcia, R.; Nowak, R.; and Willett, R. 2016. Group-sparse subspace clustering with missing data. In *2016 IEEE Statistical Signal Processing Workshop (SSP)*, 1–5.

Recht, B. 2011. A simpler approach to matrix completion. *JMLR* 12:3413–3430.

Vardy, A. 1997a. Algorithmic complexity in coding theory and the minimum distance problem. In *STOC 1997*, 92–109. ACM.

Vardy, A. 1997b. The intractability of computing the minimum distance of a code. *IEEE Transactions on Information Theory* 43(6):1757–1766.

Yao, T.; Choi, A.; and Darwiche, A. 2017. Learning bayesian network parameters under equivalence constraints. *Artif. Intell.* 244:239–257.