# Binary Matrix Factorisation via Column Generation

**Réka Á. Kovács,**[1] **Oktay Günlük,** [2] **Raphael A. Hauser** [1]
[1] University of Oxford & The Alan Turing Institute
[2] Cornell University
reka.kovacs@maths.ox.ac.uk, ong5@cornell.edu, hauser@maths.ox.ac.uk

**Abstract**

Identifying discrete patterns in binary data is an important dimensionality reduction tool in machine learning and data mining. In this paper, we consider the problem of low-rank binary matrix factorisation (BMF) under Boolean arithmetic. Due to the hardness of this problem, most previous attempts rely on heuristic techniques. We formulate the problem as a mixed integer linear program and use a large scale optimisation technique of column generation to solve it without the need of heuristic pattern mining. Our approach focuses on accuracy and on the provision of optimality guarantees. Experimental results on real world datasets demonstrate that our proposed method is effective at producing highly accurate factorisations and improves on the previously available best known results for 15 out of 24 problem instances.

## 1 Introduction

Low-rank matrix approximation is an essential tool for dimensionality reduction in machine learning. For a given $n \times m$ data matrix $X$ whose rows correspond to $n$ observations or items, columns to $m$ features and a fixed positive integer $k$, computing an optimal rank-$k$ approximation consists of approximately factorising $X$ into two matrices $A$, $B$ of dimension $n \times k$ and $k \times m$ respectively, so that the discrepancy between $X$ and its rank-$k$ approximate $A \cdot B$ is minimum. The rank-$k$ matrix $A \cdot B$ describes $X$ using only $k$ derived features: the rows of $B$ specify how the original features relate to the $k$ derived features, while the rows of $A$ provide weights how each observation can be (approximately) expressed as a linear combination of the $k$ derived features.

Many practical datasets contain observations on categorical features and while classical methods such as singular value decomposition (SVD) [10] and non-negative matrix factorisation (NMF) [20] can be used to obtain low-rank approximates for real valued datasets, for a binary input matrix $X$ they cannot guarantee factor matrices $A, B$ and their product to be binary. Binary matrix factorisation (BMF) is an approach to compute low-rank matrix approximations of binary matrices ensuring that the factor matrices are binary as well [27]. More precisely, for a given binary matrix $X \in \{0,1\}^{n \times m}$ and a fixed positive integer $k$, the rank-$k$ binary matrix factorisation problem ($k$-BMF) asks to find two matrices $A \in \{0,1\}^{n \times k}$ and $B \in \{0,1\}^{k \times m}$ such that the product of $A$ and $B$ is a binary matrix denoted by $Z$, and the distance between $X$ and $Z$ is minimum in the squared Frobenius norm. Many variants of rank-$k$ BMF exist, depending on what arithmetic is used when the product of matrices $A$ and $B$ is computed. We focus on a variant where the Boolean arithmetic is used:

$$X = A \circ B \iff x_{ij} = \bigvee_{\ell=1}^{k} a_{i\ell} \wedge b_{\ell j},$$

so that 1s and 0s are interpreted as True and False, addition corresponds to logical disjunction ($\vee$) and multiplication to conjunction ($\wedge$). Apart from the arithmetic of the Boolean semi-ring, other choices include standard arithmetic over the integers or modulo 2 arithmetic over the binary field. We focus on the Boolean case, in which the property of Boolean non-linearity, 1 + 1 = 1 holds because many natural processes follow this rule. For instance, when diagnosing patients with a certain condition, it is only the presence or absence of a characteristic symptom which is important, and the frequency of the symptom does not change the diagnosis. As an example, consider the following data matrix (inspired by [29])

$$X = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

where rows correspond to patients and columns to symptoms, $x_{ij} = 1$ indicating patient $i$ presents symptom $j$. Let

$$X = A \circ B = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix} \circ \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

denote the rank-2 BMF of $X$. Factor $B$ reveals that 2 underlying diseases cause the observed symptoms, $\alpha$ causing symptoms 1 and 2, and $\beta$ causing 2 and 3. Factor $A$ reveals that patient 1 has disease $\alpha$, patient 3 has $\beta$ and patient 2 has both. In contrast, the best rank-2 real approximation

$$X \approx \begin{bmatrix} 1.21 & 0.71 \\ 1.21 & 0.00 \\ 1.21 & -0.71 \end{bmatrix} \begin{bmatrix} 0.00 & 0.71 & 0.50 \\ 0.71 & 0.00 & -0.71 \end{bmatrix}$$

fails to reveal a clear interpretation, and the best rank-2 NMF

$$X \approx \begin{bmatrix} 1.36 & 0.09 \\ 1.05 & 1.02 \\ 0.13 & 1.34 \end{bmatrix} \begin{bmatrix} 0.80 & 0.58 & 0.01 \\ 0.00 & 0.57 & 0.81 \end{bmatrix}$$

of $X$ suggests that symptom 2 presents with lower intensity in both $\alpha$ and $\beta$, an erroneous conclusion (caused by patient 2) that could not have been learned from data $X$ which is of "on/off" type. BMF-derived features are particularly natural to interpret in biclustering gene expression datasets [41], role based access control [22, 23] and market basket data clustering [21].

### 1.1 Complexity and related work

The Boolean rank of a binary matrix $X \in \{0,1\}^{n\times m}$ is defined to be the smallest integer $r$ for which there exist matrices $A \in \{0,1\}^{n\times r}$ and $B \in \{0,1\}^{r\times m}$ such that $X = A \circ B$, where $\circ$ denotes Boolean matrix multiplication defined as $x_{ij} = \bigvee_{\ell=1}^{r} a_{i\ell} \wedge b_{\ell j}$ for all $i \in \{1,\dots,n\} := [n]$, $j \in [m]$ [13]. This is equivalent to $x_{ij} = \min\{1, \sum_{\ell=1}^{r} a_{i\ell} b_{\ell j}\}$ using standard arithmetic. Equivalently, the Boolean rank of $X$ is the minimum value of $r$ for which it is possible to factor $X$ into the Boolean combination of $r$ rank-1 binary matrices $X = \bigvee_{\ell=1}^{r} \boldsymbol{a}_\ell \boldsymbol{b}_\ell^\top$ for $\boldsymbol{a}_\ell \in \{0,1\}^n, \boldsymbol{b}_\ell \in \{0,1\}^m$. Interpreting $X$ as the node-node incidence matrix of a bipartite graph $G$ with $n$ vertices on the left and $m$ vertices on the right, the problem of computing the Boolean rank of $X$ is in one-to-one correspondence with finding a minimum edge covering of $G$ by complete bipartite subgraphs (bicliques)[30]. Since the biclique cover problem is NP-hard [31, Theorem 8.1], [9, Problem GT18] and hard to approximate [39], computing the Boolean rank is hard as well. Finding an optimal $k$-BMF of $X$ has a graphic interpretation of minimizing the number of errors in an approximate covering of $G$ by $k$ bicliques. Even the computation of 1-BMF is hard because of its relation to the maximum edge biclique problem [33] which in the matrix setting is stated as finding the largest submatrix of $X$ of all 1s.

Many heuristic attempts have been made to approximately compute BMFs by focusing on recursively partitioning the given matrix $X \in \{0,1\}^{n\times m}$ and computing a rank-1 factorisation at each step. The first such recursive method called Proximus [18], [17] is used to compute BMF under standard arithmetic over the integers. For rank-1 factorisation Proximus uses an alternating iterative heuristic applied to a random starting point which is based on the observation that if $\boldsymbol{a} \in \{0,1\}^n$ is given, then a vector $\boldsymbol{b} \in \{0,1\}^m$ that minimizes the distance between $X$ and $\boldsymbol{a}\boldsymbol{b}^\top$ can be computed in $\mathcal{O}(nm)$ time. Since the introduction of Proximus, much research has been focused on computing efficient and accurate 1-BMF. [37] proposes an integer program (IP) for 1-BMF and several linear programming (LP) relaxations of it, one of which leads to a 2-approximation scheme. [38] provides a rounding based 2-approximation for 1-BMF by using an observation about the vertices of the polytope corresponding to the LP relaxation of an integer program. In [3], a modification of the Proximus framework is explored using the approach of [37] to compute 1-BMF at each step.

Rank-$k$ BMF under Boolean arithmetic is explicitly introduced in [28] [29], along with a heuristic algorithm called ASSO. The core of ASSO is based on an association rule-mining approach to create matrix $B \in \{0,1\}^{k\times m}$ and greedily fix $A \in \{0,1\}^{n\times k}$ with respect to $B$. The problem of finding an optimal $A$ with respect to fixed $B$ is NP-hard [26] but can be solved in $\mathcal{O}(2^k kmn)$ time [29]. The association rule-mining approach of [29] is further improved in [1] by a range of iterative heuristics employing this alternative fixing idea and local search. Another approach based on an alternating style heuristic is explored in [41] to solve a non-linear unconstrained formulation of $k$-BMF with penalty terms in the objective for non-binary entries. [40] proposes another iterative heuristic which at every iteration permutes the rows and columns of $X$ to create a dense submatrix in the upper right corner which is used as a rank-1 component in the $k$-BMF.

In [22, 23] a series of exponential size IPs for $k$-BMF and exact Boolean rank computation are introduced. They use an explicit enumeration of the $2^m$ possible rows for factor matrix $B$ and corresponding indicator variables. To tackle the exponential explosion of rows considered, a heuristic row generation using association rule mining and subset enumeration is developed. However, no non-heuristic method is proposed to overcome the exponential size of the model and hence the exact solution of the IPs is feasible up to only very limited sized datasets where complete enumeration of the $2^m$ rows is possible. An exact linear IP for $k$-BMF with polynomially many variables and constraints is presented in [16]. This model uses McCormick envelopes [25] to linearise quadratic terms.

### 1.2 Our contribution

In this paper, we present a novel IP formulation for $k$-BMF that overcomes several limitations of earlier approaches. In particular, our formulation does not suffer from permutation symmetry, it does not rely on heuristic pattern mining, and it has a stronger LP relaxation than that of [16]. On the other hand, our new formulation has an exponential number of variables which we tackle using a column generation approach that effectively searches over this exponential space without explicit enumeration, unlike the complete enumeration used for the exponential size model of [22, 23]. Our proposed solution method is able to prove optimality for smaller datasets, while for larger datasets it provides solutions with better accuracy than the state-of-the-art heuristic methods. In addition, due to the entry-wise modelling of $k$-BMF in our approach, we can handle matrices with missing entries and our solutions can be used for binary matrix completion.

The rest of the paper is organised as follows. In Section 2 we briefly discuss the model of [16] and its limitations. In Section 3 we introduce our integer linear programming formulation for $k$-BMF, detail a theoretical framework based on the large scale optimisation technique of column generation for its solution and discuss heuristics for the arising subproblems. Finally, in Section 4 we demonstrate the practical applicability of our approach on several real world datasets.

## 2 Problem formulation

Given a binary matrix $X \in \{0,1\}^{n\times m}$, and a fixed integer $k \ll \min(n,m)$ we wish to find two binary matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ to minimise $\|X - A \circ B\|_F^2$, where $\|\cdot\|_F$ denotes the Frobenius norm and $\circ$ stands for Boolean matrix multiplication as defined in Section 1.1. Since $X$ and $Z := A \circ B$ are binary matrices, the squared Frobenius and the entry-wise $\ell_1$ norm coincide and we can expand the objective function

$$\|X - Z\|_F^2 = \sum_{(i,j)\in E} (1 - z_{ij}) + \sum_{(i,j)\notin E} z_{ij}, \qquad (1)$$

where $E := \{(i,j) : x_{ij} = 1\}$ is the index set of the positive entries of $X$. [16] formulates the problem as an exact

integer linear program by introducing variables $y_{i\ell j}$ for the product of $a_{i\ell}$ and $b_{\ell j}$ ($(i,\ell,j) \in \mathcal{F} := [n] \times [k] \times [m]$), and using McCormick envelopes [25] to avoid the appearance of a quadratic constraint arising from the product. The McCormick envelopes represent the product of two binary variables $a$ and $b$ by a new variable $y$ and four linear inequalities given by $MC(a,b) = \{y \in \mathbb{R} : a + b - 1 \leq y, y \leq a, y \leq b, 0 \leq y\}$. The model of [16] reads as

$$(\text{IP}_{\text{exact}}) \quad \zeta_{IP} = \min_{a,b,y,z} \sum_{(i,j)\in E} (1 - z_{ij}) \quad + \sum_{(i,j)\notin E} z_{ij} \quad (2)$$

$$\text{s.t. } y_{i\ell j} \leq z_{ij} \leq \sum_{\ell=1}^{k} y_{i\ell j}, \qquad (i,\ell,j) \in \mathcal{F}, \quad (3)$$

$$y_{i\ell j} \in MC(a_{i\ell}, b_{\ell j}), \qquad (i,\ell,j) \in \mathcal{F}, \quad (4)$$

$$a_{i\ell}, b_{\ell j} \in \{0,1\}, \quad z_{ij} \leq 1, \qquad (i,\ell,j) \in \mathcal{F}, \quad (5)$$

The above model is exact in the sense that its optimal solutions correspond to optimal $k$-BMFs of $X$. Most general purpose IP solvers use an enumeration framework, which relies on bounds from the linear programming (LP) relaxation of the IP and consequently, it is easier to solve the IP when its LP bound is tighter. For $k = 1$, we have $y_{i1j} = z_{ij}$ for all $i, j$ and the LP relaxation of the model is simply the LP relaxation of the McCormick envelopes which has a rich and well-studied polyhedral structure [32]. However, for $k > 1$, $\text{IP}_{\text{exact}}$'s LP relaxation ($\text{LP}_{\text{exact}}$) only provides a trivial bound.

**Lemma 1.** *For $k > 1$, $\text{LP}_{\text{exact}}$ has optimal objective value 0 which is attained by at least $\binom{k}{2}$ solutions.*

For the proof of Lemma 1, see Appendix 5.1. Furthermore, for $k > 1$ the model is highly symmetric, since $AP \circ P^{-1}B$ is an equivalent solution for any permutation matrix $P$. These properties of the model make it unlikely to be solved to optimality in a reasonable amount of time for a large matrix $X$, though the symmetries can be partially broken by incorporating constraints $\sum_i a_{i\ell_1} \geq \sum_i a_{i\ell_2}$ for all $\ell_1 < \ell_2$.

Note that constraint (3) implies $\frac{1}{k}\sum_{\ell=1}^{k} y_{i\ell j} \leq z_{ij} \leq \sum_{\ell=1}^{k} y_{i\ell j}$ as a lower and upper bound on each variable $z_{ij}$. Due to these bounds, the objective function may be approximated by

$$\zeta_{IP}(\rho) = \sum_{(i,j)\in E} (1 - z_{ij}) + \rho \sum_{(i,j)\notin E} \sum_{\ell=1}^{k} y_{i\ell j}, \quad (6)$$

where $\rho$ is a parameter of the formulation. By setting $\rho = \frac{1}{k}$ we underestimate the original objective, while setting $\rho = 1$ we overestimate. Using (6) as the objective function reduces the number of variables and constraints in the model. Variables $z_{ij}$ need only be declared for $(i,j) \in E$, and constraint (3) simplifies to $z_{ij} \leq \sum_{\ell=1}^{k} y_{i\ell j}$ for $(i,j) \in E$.

## 3 A new formulation via column generation

The exact model presented in the previous section relies on polynomially many constraints and variables and constitutes the first approach towards obtaining rank-$k$ factorisations of binary matrices with optimality guarantees. However, such a compact IP formulation may be weak in the sense that its LP relaxation is a very coarse approximation to the convex hull of integer feasible points and an IP formulation with exponentially many variables or constraints can have the potential to provide a tighter relaxation[24]. Motivated by this fact, we introduce a new formulation with an exponential number of variables and detail a column generation framework for its solution.

Consider enumerating all possible rank-1 binary matrices of size $n \times m$ and let

$$\mathcal{R} = \{\boldsymbol{a}\boldsymbol{b}^\top : \boldsymbol{a} \in \{0,1\}^n, \boldsymbol{b} \in \{0,1\}^m, \boldsymbol{a}, \boldsymbol{b} \neq \boldsymbol{0}\}.$$

The size of $\mathcal{R}$ is $|\mathcal{R}| = (2^n - 1)(2^m - 1)$ as any pair of binary vectors $\boldsymbol{a}, \boldsymbol{b} \neq \boldsymbol{0}$ lead to a unique rank-1 matrix $Y = \boldsymbol{a}\boldsymbol{b}^\top$ with $Y_{ij} = 1$ for $\{(i,j) : a_i = 1, b_j = 1\}$. Define a binary decision variable $q_\ell$ to denote if the $\ell$-th rank-1 binary matrix in $\mathcal{R}$ is included in a rank-$k$ factorisation of $X$ ($q_\ell = 1$), or not ($q_\ell = 0$). Let $\boldsymbol{q} \in \{0,1\}^{|\mathcal{R}|}$ be a vector that has a component $q_\ell$ for each matrix in $\mathcal{R}$. We form a $\{0,1\}$-matrix $M$ of dimension $nm \times |\mathcal{R}|$ whose rows correspond to entries of an $n \times m$ matrix, columns to rank-1 binary matrices in $\mathcal{R}$ and $M_{(i,j),\ell} = 1$ if the $(i,j)$-th entry of the $\ell$-th rank-1 binary matrix in $\mathcal{R}$ is 1, $M_{(i,j),\ell} = 0$ otherwise. We split $M$ horizontally into two matrices $M_{\boldsymbol{0}}$ and $M_{\boldsymbol{1}}$, so that rows of $M$ corresponding to a positive entry of the given matrix $X$ are in $M_{\boldsymbol{1}}$ and the rest of rows of $M$ in $M_{\boldsymbol{0}}$,

$$M = \begin{bmatrix} M_{\boldsymbol{0}} \\ M_{\boldsymbol{1}} \end{bmatrix} \text{ where } \begin{matrix} M_{\boldsymbol{0}} \in \{0,1\}^{(nm-|E|)\times|\mathcal{R}|}, \\ M_{\boldsymbol{1}} \in \{0,1\}^{|E|\times|\mathcal{R}|}. \end{matrix} \quad (7)$$

The following Master Integer Program over an exponential number of variables is an exact model for $k$-BMF,

$$(\text{MIP}_{\text{exact}}) \ \zeta_{\text{MIP}} = \min \boldsymbol{1}^\top \boldsymbol{\xi} + \boldsymbol{1}^\top \boldsymbol{\pi} \quad (8)$$

$$\text{s.t. } M_{\boldsymbol{1}}\boldsymbol{q} + \boldsymbol{\xi} \geq \boldsymbol{1} \quad (9)$$

$$M_{\boldsymbol{0}}\boldsymbol{q} \leq k\boldsymbol{\pi} \quad (10)$$

$$\boldsymbol{1}^\top \boldsymbol{q} \leq k \quad (11)$$

$$\boldsymbol{\xi} \geq \boldsymbol{0}, \boldsymbol{\pi} \in \{0,1\}^{nm-|E|}, \quad (12)$$

$$\boldsymbol{q} \in \{0,1\}^{|\mathcal{R}|}. \quad (13)$$

Constraint (11) ensures that at most $k$ rank-1 matrices are active in a factorisation. Variables $\xi_{ij}$ correspond to positive entries of $X$, and are forced by constraint (9) to take value 1 and increase the objective if the $(i,j)$-th positive entry of $X$ is not covered. Similarly, variables $\pi_{ij}$ correspond to zero entries of $X$ and are forced to take value 1 by constraint (10) if the $(i,j)$-th zero entry of $X$ is erroneously covered in a factorisation. One of the imminent advantages of $\text{MIP}_{\text{exact}}$ is using indicator variables directly for rank-1 matrices instead of the entries of factor matrices $A, B$, hence no permutation symmetry arises. In addition, for all $k$ not exceeding a certain number that depends on $X$, the LP relaxation of $\text{MIP}_{\text{exact}}$ ($\text{MLP}_{\text{exact}}$ ) has strictly positive optimal objective value.

**Lemma 2.** *Let $i(X)$ be the isolation number of $X$. For all $k < i(X)$, we have $0 < \zeta_{MLP}$.*

For the definition of isolation number and the proof of Lemma 2 see Appendix 5.2. Similarly to the polynomial size exact model $\text{IP}_{\text{exact}}$ in the previous section, we consider a

modification of $\text{MIP}_{\text{exact}}$ with an objective that is analogous to the one in Equation (6),

$$(\text{MIP}(\rho))\ z_{\text{MIP}(\rho)} = \min\ \mathbf{1}^\top \boldsymbol{\xi} + \rho\, \mathbf{1}^\top M_{\mathbf{0}} \boldsymbol{q} \tag{14}$$

$$\text{s.t. (9), (11) hold and} \tag{15}$$

$$\boldsymbol{\xi} \geq \mathbf{0}, \quad \boldsymbol{q} \in \{0,1\}^{|\mathcal{R}|}.$$

The objective of MIP($\rho$) simply counts the number of positive entries of $X$ that are not covered by any of the $k$ rank-1 matrices chosen, plus the number of times zero valued entries of $X$ are erroneously covered weighted by parameter $\rho$. Depending on the selection of parameter $\rho$, MIP($\rho$) provides a lower or upper bound on $\text{MIP}_{\text{exact}}$. We denote the LP relaxation of MIP($\rho$) by MLP($\rho$).

**Lemma 3.** *For $\rho = \frac{1}{k}$, the optimal objective values of the LP relaxations $\text{MLP}_{exact}$ and MLP($\frac{1}{k}$) coincide.*

For a short proof of Lemma 3 see Appendix 5.3. Combining Lemmas 1, 2 and 3 we obtain the following relations between formulations $\text{IP}_{\text{exact}}$, $\text{MIP}_{\text{exact}}$, MIP($\rho$) and their LP relaxations for $k > 1$,

$$z_{\text{MIP}(\frac{1}{k})} \leq \zeta_{IP} = \zeta_{\text{MIP}} \leq z_{\text{MIP}(1)}, \tag{16}$$

$$0 = \zeta_{LP} \leq z_{\text{MLP}(\frac{1}{k})} = \zeta_{\text{MLP}} \leq z_{\text{MLP}(1)}. \tag{17}$$

Let $\boldsymbol{p}$ be the dual variable vector associated to constraints (9) and $\mu$ be the dual variable to constraint (11). Then the dual of MLP($\rho$) is given by

$$(\text{MDP}(\rho))\ z_{\text{MDP}(\rho)} = \max \mathbf{1}^\top \boldsymbol{p} - k\mu \tag{18}$$

$$\text{s.t. } M_{\mathbf{1}}^\top \boldsymbol{p} - \mu \mathbf{1} \leq \rho\, M_{\mathbf{0}}^\top \mathbf{1}, \tag{19}$$

$$\mu \geq 0,\ \boldsymbol{p} \in [0,1]^{|E|}. \tag{20}$$

Due to the number of variables in the formulation, it is not practical to solve MIP($\rho$) or its LP relaxation MLP($\rho$) explicitly. *Column generation* (CG) is a technique to solve large LPs by iteratively generating only the variables which have the potential to improve the objective function [2]. The CG procedure is initialised by explicitly solving a Restricted Master LP which has a small subset of the variables in MLP($\rho$). The next step is to identify a missing variable with a negative *reduced cost* to be added to this Restricted MLP($\rho$). To avoid explicitly considering all missing variables, a *pricing problem* is formulated and solved. The solution of the pricing problem either returns a variable with negative reduced cost and the procedure is iterated; or proves that no such variable exists and hence the solution of the Restricted MLP($\rho$) is optimal for the complete formulation MLP($\rho$).

We use CG to solve MLP($\rho$) by considering a sequence $(t = 1, 2, ...)$ of Restricted MLP($\rho$)'s with constraint matrix $M^{(t)}$ being a subset of columns of $M$, where each column $\boldsymbol{y} \in \{0,1\}^{nm}$ of $M$ corresponds to a flattened rank-1 binary matrix $\boldsymbol{a}\boldsymbol{b}^\top$ according to Equation (7). The constraint matrix of the first Restricted MLP($\rho$) may be left empty or can be warm started by identifying a few rank-1 matrices in $\mathcal{R}$, say from a heuristic solution. Upon successful solution of the $t$-th Restricted MLP($\rho$), we obtain a vector of dual variables $[\boldsymbol{p}^*, \mu^*] \geq \mathbf{0}$ optimal for the $t$-th Restricted MLP($\rho$). To identify a missing column of $M$ that has a negative reduced cost, we solve the following pricing problem (PP):

$$(\text{PP})\ \omega(\boldsymbol{p}^*) = \max_{a,b,y} \sum_{(i,j)\in E} p_{ij}^* y_{ij} - \rho \sum_{(i,j)\notin E} y_{ij}$$

$$\text{s.t. } y_{ij} \in MC(a_i, b_j),\ a_i, b_j \in \{0,1\},\ i \in [n], j \in [m].$$

The objective of PP depends on the current dual solution $[\boldsymbol{p}^*, \mu^*]$ and its optimal solution corresponds to a rank-1 binary matrix $\boldsymbol{a}\boldsymbol{b}^\top$ whose corresponding variable $q_\ell$ in MLP($\rho$) has the smallest reduced cost. If $\omega(\boldsymbol{p}^*) \leq \mu^*$, then the dual variables $[\boldsymbol{p}^*, \mu^*]$ of the Restricted MLP($\rho$) are feasible for MDP($\rho$) and hence the current solution of the Restricted MLP($\rho$) is optimal for MLP($\rho$). If $\omega(\boldsymbol{p}^*) > \mu^*$, then the variable $q_\ell$ associated with the rank-1 binary matrix $\boldsymbol{a}\boldsymbol{b}^\top$ is added to the Restricted MLP($\rho$) and the procedure is iterated. CG optimally terminates if at some iteration we have $\omega(\boldsymbol{p}^*) \leq \mu^*$.

To apply the CG approach above to $\text{MLP}_{\text{exact}}$ only a small modification needs to be made. The Restricted $\text{MLP}_{\text{exact}}$ provides dual variables for Constraints (10) which are used in the objective of PP for coefficients of $y_{ij}$ $(i,j) \notin E$. Note however, that CG cannot be used to solve a modification of $\text{MLP}_{\text{exact}}$ in which constraints (10) are replaced by exponentially many constraints $(M_{\mathbf{0}})_\ell\ q_\ell \leq \boldsymbol{\pi}$ for $\ell \in [|\mathcal{R}|]$ where $(M_{\mathbf{0}})_\ell$ denotes the $\ell$-th column of $M_{\mathbf{0}}$, see Appendix 5.4.

If the optimal solution of MLP($\rho$) is integral, then it is also optimal for MIP($\rho$). However, if it is fractional, then it only provides a lower bound on the optimal value of MIP($\rho$). In this case we obtain an integer feasible solution by simply adding integrality constraints on the variables of the final Restricted MLP($\rho$) and solving it as an integer program. If $\rho = 1$, the optimal solution of this integer program is optimal for MIP(1) if the objective of the Restricted MIP(1) and the ceiling of the Restricted MLP(1) agree. To solve the MIP($\rho$) to optimality in all cases, one needs to embed CG into branch-and-bound [24] which we do not do. However, note that even if the CG procedure is terminated prematurely, one can still obtain a lower bound on MLP($\rho$) and MIP($\rho$) as follows. Let the objective value of any of the Restricted MLP($\rho$)'s be

$$z_{\text{RMLP}} = \mathbf{1}^\top \boldsymbol{\xi}^* + \rho \mathbf{1}^\top M_0^* \boldsymbol{q}^* = \mathbf{1}^\top \boldsymbol{p}^* - k\mu^* \tag{21}$$

where $[\boldsymbol{\xi}^*, \boldsymbol{q}^*]$ is the solution of the Restricted MLP($\rho$), $[\boldsymbol{p}^*, \mu^*]$ is the solution of the dual of the Restricted MLP($\rho$) and $\mathbf{1}^\top M_0^*$ is the objective coefficient of columns in the Restricted MLP($\rho$). Assume that we solve PP to optimality and we obtain a column $\boldsymbol{y}$ for which the reduced cost is negative, $\omega(\boldsymbol{p}^*) > \mu^*$. In this case, we can construct a feasible solution to MDP($\rho$) by setting $\boldsymbol{p} := \boldsymbol{p}^*$ and $\mu := \omega(\boldsymbol{p}^*)$ and get the following bound on the optimal value $z_{\text{MLP}(\rho)}$ of MLP($\rho$),

$$z_{\text{MLP}(\rho)} \geq \mathbf{1}^\top \boldsymbol{p}^* - k\,\omega(\boldsymbol{p}^*) = z_{\text{RMLP}} - k(\omega(\boldsymbol{p}^*) - \mu^*). \tag{22}$$

If we do not have the optimal solution to PP but have an upper bound $\bar{\omega}(\boldsymbol{p}^*)$ on it, $\omega(\boldsymbol{p}^*)$ can be replaced by $\bar{\omega}(\boldsymbol{p}^*)$ in equation (22) and the bound on MLP($\rho$) still holds. Furthermore, this lower bound on MLP($\rho$) provides a valid lower bound on MIP($\rho$). Consequently, our approach always produces a bound on the optimality gap of the final solution which heuristic methods cannot do. There is, however, no a priory (theoretical) bound on this gap.

### 3.1 The pricing problem

The efficiency of the CG procedure described above greatly depends on solving PP efficiently. In standard form PP is a bipartite binary quadratic optimisation problem and can be written as

$$(\text{PP}) \quad \omega(\boldsymbol{p}^*) = \max_{\boldsymbol{a}\in\{0,1\}^n, \boldsymbol{b}\in\{0,1\}^m} \boldsymbol{a}^\top H \boldsymbol{b} \qquad (23)$$

for $H$ an $n \times m$ matrix with $h_{ij} = p^*_{ij} \in [0,1]$ for $(i,j) \in E$ and $h_{ij} = -\rho$ for $(i,j) \notin E$. Bipartite quadratic optimisation is NP-hard in general [33], hence for large $X$ it may take too long to solve PP to optimality at each iteration. To speed up computations, the formulation of PP may be improved by eliminating redundant constraints. The McCormick envelopes [25] set two lower and two upper bound constraints on $y_{ij}$. Due to the objective function it is possible to declare the lower (upper) bounds $y_{ij}$ for only $(i,j) \notin E$ ($(i,j) \in E$) without changing the optimum. In addition, it is enough to set integrality constraint on only $\boldsymbol{a}$ or $\boldsymbol{b}$ [34].

If a heuristic approach to PP provides a solution for which the reduced cost is negative, that is $\omega(p^*) > \mu^*$, then it is valid to add this heuristic solution as a column to the next Restricted MLP($\rho$) without making the solution of MLP($\rho$) heuristic. Most heuristic algorithms that are available for bipartite quadratic optimisation build on the idea that the optimal $\boldsymbol{a} \in \{0,1\}^n$ with respect to a fixed $\boldsymbol{b} \in \{0,1\}^m$ can be computed in $\mathcal{O}(nm)$ time and this procedure can be iterated by alternatively fixing $\boldsymbol{a}$ and $\boldsymbol{b}$. [12] presents several local search heuristics for PP based on this idea along with a simple greedy algorithm and [7] further develops these ideas. Below we detail the greedy algorithm of [12] and introduce some variants of it which we use in the next section to provide a warm start to PP at every iteration of the CG procedure.

The greedy algorithm of [12] aims to set entries of $\boldsymbol{a}$ and $\boldsymbol{b}$ to 1 which correspond to rows and columns of $H$ with the largest positive weights. In the first phase of the algorithm, the row indices $i$ of $H$ are put in decreasing order according to their sum of positive entries, so $\gamma_i^+ \geq \gamma_{i+1}^+$ where $\gamma_i^+ := \sum_{j=1}^m \max(0, h_{ij})$. Then sequentially according to this ordering, $a_i$ is set to 1 if $\sum_{j=1}^m \max(0, \sum_{\ell=1}^{i-1} a_\ell h_{\ell j}) < \sum_{j=1}^m \max(0, \sum_{\ell=1}^{i-1} a_\ell h_{\ell j} + h_{ij})$ and 0 otherwise. In the second phase, $b_j$ is set to 1 if $(\boldsymbol{a}^\top H)_j > 0$, 0 otherwise. The precise outline of the algorithm is given in Appendix 5.5.

There are many variants of the greedy algorithm one can explore. First, the solution greatly depends on the ordering of $i$'s in the first phase. If for some $i_1 \neq i_2$ we have $\gamma_{i_1}^+ = \gamma_{i_2}^+$, comparing the sum of negative entries of rows $i_1$ and $i_2$ can put more "influential" rows of $H$ ahead in the ordering. Let us call this ordering the *revised ordering* and the one which only compares the positive sums as the *original ordering*. Another option is to use a completely *random order* of $i$'s or to apply a small perturbation to sums $\gamma_i^+$ to get a *perturbed* version of the revised or original ordering. None of the above ordering strategies clearly dominates the others in all cases but they are fast to compute hence one can evaluate all five ordering strategies (original, revised, original perturbed, revised perturbed, random) and pick the best one. Second, the algorithm as presented above first fixes $\boldsymbol{a}$ and then $\boldsymbol{b}$. Changing the order of fixing $\boldsymbol{a}$ and $\boldsymbol{b}$ can yield a different result hence it is best to try for both $H$ and $H^\top$. In general, it is recommended to start the first phase on the smaller dimension. Third, the solution from the greedy algorithm may be improved by computing the optimal $\boldsymbol{a}$ with respect to fixed $\boldsymbol{b}$. This idea as previously mentioned then can be used to fix $\boldsymbol{a}$ and $\boldsymbol{b}$ in an alternating fashion and stop when no changes occur in either.

## 4 Experiments

The CG approach introduced in the previous section provides a theoretical framework for computing $k$-BMF with optimality guarantees. In this section we present some experimental results with CG to demonstrate the practical applicability of our approach on eight real world categorical datasets that were downloaded from online repositories [6], [19]. Table 1 shows a short summary of the eight datasets used, details on converting categorical columns into binary, missing value treatment and descriptions can be found in Appendix 5.7. Table 1 also shows the value of the isolation number $i(X)$ for each dataset, which provides a lower bound on the Boolean rank [30, Section 2.3].

Table 1: Summary of binary real world datasets

| | zoo | tumor | hepatitis | heart | lymp | audio | apb | votes |
|---|---|---|---|---|---|---|---|---|
| n | 101 | 339 | 155 | 242 | 148 | 226 | 105 | 434 |
| m | 17 | 24 | 38 | 22 | 44 | 94 | 105 | 32 |
| $i(X)$ | 16 | 24 | 30 | 22 | 42 | 90 | 104 | 30 |
| %1s | 44.3 | 24.3 | 47.2 | 34.4 | 29.0 | 11.3 | 8.0 | 47.3 |

Since the efficiency of CG greatly depends on the speed of generating columns, let us illustrate the speed-up gained by using heuristic pricing. At each iteration of CG, 30 variants of the greedy heuristic are computed to obtain an initial feasible solution to PP. The 30 variants of the greedy algorithm use the original and revised ordering, their transpose and perturbed version and 22 random orderings. All greedy solutions are improved by the alternating heuristic until no further improvement is found. Under *exact* pricing, the best heuristic solution is used as a warm start and PP is solved to optimality at each iteration using CPLEX [5]. In simple heuristic (*heur*) pricing, if the best heuristic solution to PP has negative reduced cost, $\omega_{\text{heur}}(\boldsymbol{p}^*) > \mu^*$, then the heuristic column is added to the next Restricted MLP($\rho$). If at some iteration, the best heuristic column does not have negative reduced cost, CPLEX is used to solve PP to optimality for that iteration. The multiple heuristic (*heur_multi*) pricing is a

Table 2: % optimality gaps after 20 mins under objective (6)

| | | zoo | | | tumor | |
|---|---|---|---|---|---|---|
| k | MIP(1) | [15] | [21] | MIP(1) | [15] | [21] |
| 2 | 0.0 | 0.0 | 100.0 | 0.9 | 40.8 | * |
| 5 | 0.0 | 59.2 | 100.0 | 9.3 | 98.0 | * |
| 10 | 3.0 | 95.8 | 100.0 | 28.4 | 100.0 | * |

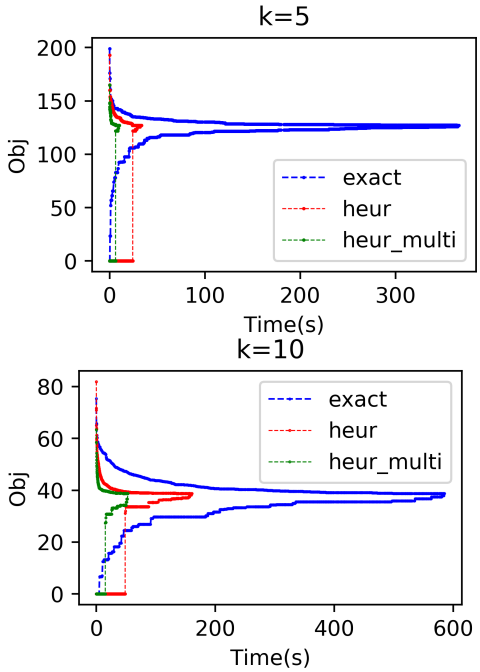
Table 3: Primal objectives of MLP(1), MLP($\frac{1}{k}$), MIP(1) after 20 mins of CG

| $k$ | | zoo | tumor | hepatitis | heart | lymp | audio | apb | votes |
|---|---|---|---|---|---|---|---|---|---|
| 2 | MLP($\frac{1}{k}$) | 206.5 | 1178.9 | 978.7 | 882.9 | 917.2 | 1256.5 | 709 | 1953 |
| | MLP(1) | 272 | 1409.8 | 1384 | 1185 | 1188.8 | 1499 | 776 | 2926 |
| | MIP(1) | 272(271) | 1411 | 1384(1382) | 1185 | 1197(1184) | 1499 | 776 | 2926 |
| 5 | MLP($\frac{1}{k}$) | 42.8 | 463.9 | 333.1 | 291.0 | 366.7 | 654.2 | 433.5 | 715.5 |
| | MLP(1) | 127 | 1019.3 | 1041.1 | 736 | 914.0 | 1159.3 | 683.0 | 2135.5 |
| | MIP(1) | 127(125) | 1029 | 1228 | 736 | 997(991) | 1176 | 684(683) | 2277(2274) |
| 10 | MLP($\frac{1}{k}$) | 4.8 | 192.8 | 142.5 | 102.3 | 165.1 | 351.4 | 166.8 | 307.9 |
| | MLP(1) | 38.8 | 575.5 | 734.8 | 419 | 653.2 | 867.2 | 574.2 | 1409.5 |
| | MIP(1) | 40 | 579 | 910 | 419 | 737(732) | 893 | 577(572) | 1566(1549) |

slight modification of the simple heuristic strategy, in which at each iteration all columns with negative reduced cost are added to the next Restricted MLP($\rho$).

Figure 1 indicates the differences between pricing strategies when solving MLP(1) via CG for $k = 5, 10$ on the zoo dataset. The objective of MLP(1) (decreasing curve) and the value of the dual bound (increasing curve) computed using the formula in Equation (22) are plotted against time. Sharp increases in the dual bound for heuristic pricing strategies correspond to iterations in which CPLEX was used to solve PP, as for the evaluation of the dual bound on MLP(1) we need a strong upper bound on $\omega(\boldsymbol{p}^*)$ which heuristic solutions do not provide. While we observe a tailing off effect [24] on all three curves, both heuristic pricing strategies provide a significant speed-up from exact pricing, with adding multiple columns at each iteration being the fastest.

Figure 1: Comparison of pricing strategies for solving MLP(1) on the zoo dataset




In Table 2 we present computational results comparing the optimality gap ($100 \times \frac{\text{best integer} - \text{best bound}}{\text{best integer}}$) of MIP(1), $\text{IP}_{\text{exact}}$ [16] and the exponential formulation in [22, 23] under objective (6) using a 20 mins time budget. The precise formulations that were used to obtain results for [16] and [22, 23] can be found in Appendix Section 5.6. Reading in the full exponential size model of [22, 23] using 16 GB memory is not possible for other datasets than 'zoo'. Table 2 shows that different formulations and algorithms to solve them make a difference in practice: our novel formulation MIP(1) combined with an effective computational optimization approach (CG) produces solutions with smaller optimality gap than other formulations as it scales well and it has a stronger LP relaxation.

In order for CG to terminate with a certificate of optimality, at least one pricing problem has to be solved to optimality. Unfortunately for the larger datasets this cannot be achieved in under 20 mins. Therefore, for datasets other than 'zoo', we change the multiple heuristic pricing strategy as follows: We impose an overall time limit of 20 mins on the CG process and use the barrier method in CPLEX as the LP solver for the Restricted MLP($\rho$) at each iteration. In order to maximise the diversity of columns added at each iteration, we choose at most two columns with negative reduced cost that are closest to being mutually orthogonal. If CPLEX has to be used to improve the heuristic pricing solution, we do not solve PP to optimality but abort CPLEX if a column with negative reduced cost has been found. While these modifications result in a speed-up, they reduce the chance of obtaining a strong dual bound. In case a strong dual bound is desired, we may continue applying CG iterations with exact pricing after the 20 mins of heuristic pricing have run their course.

In our next experiment, we explore the differences between formulations MLP($\frac{1}{k}$), MLP(1) and MIP(1). We warm start CG by identifying a few heuristic columns using the code of [1] and a new fast heuristic which works by sequentially computing $k$ rank-1 binary matrices via the greedy algorithm starting with the coefficient matrix $H = 2X - 1$ and then setting entries of $H$ to zero that correspond to entries of $X$ that are covered. The precise outline of this new *$k$-greedy* algorithm can be found in Appendix 5.9.

Table 4: Factorisation errors in $\|\cdot\|_F^2$ for eight methods for $k$-BMF

| | | zoo | tumor | hepatitis | heart | lymp | audio | apb | votes |
|---|---|---|---|---|---|---|---|---|---|
| k=2 | CG | **271** | 1411 | **1382** | **1185** | 1184 | **1499** | **776** | **2926** |
| | $IP_{exact}$ | **271** | **1408** | 1391 | 1187 | **1180** | **1499** | **776** | **2926** |
| | ASSO++ | 276 | 1437 | 1397 | 1187 | 1202 | 1503 | **776** | **2926** |
| | $k$-greedy | 325 | 1422 | 1483 | 1204 | 1201 | **1499** | **776** | 2929 |
| | pymf | 276 | 1472 | 1418 | 1241 | 1228 | 1510 | 794 | 2975 |
| | ASSO | 367 | 1465 | 1724 | 1251 | 1352 | 1505 | 778 | 2946 |
| | NMF | 291 | 1626 | 1596 | 1254 | 1366 | 2253 | 809 | 3069 |
| | MEBF | 348 | 1487 | 1599 | 1289 | 1401 | 1779 | 812 | 3268 |
| k=5 | CG | **125** | **1029** | **1228** | **736** | **991** | **1176** | **683** | **2272** |
| | $IP_{exact}$ | 133 | 1055 | **1228** | 738 | 1029 | 1211 | 690 | 2293 |
| | ASSO++ | 133 | 1055 | **1228** | 738 | 1039 | 1211 | 694 | 2302 |
| | $k$-greedy | 233 | 1055 | 1306 | 748 | 1063 | 1211 | 690 | 2310 |
| | pymf | 142 | 1126 | 1301 | 835 | 1062 | 1245 | 730 | 2517 |
| | ASSO | 354 | 1092 | 1724 | 887 | 1352 | 1505 | 719 | 2503 |
| | NMF | 163 | 1207 | 1337 | 995 | 1158 | 1565 | 762 | 2526 |
| | MEBF | 173 | 1245 | 1439 | 929 | 1245 | 1672 | 730 | 2832 |
| k=10 | CG | **40** | **579** | 910 | **419** | **730** | **893** | **572** | **1527** |
| | $IP_{exact}$ | 41 | 583 | **902** | **419** | 805 | 919 | 590 | 1573 |
| | ASSO++ | 55 | 583 | 910 | **419** | 812 | 922 | 591 | 1573 |
| | $k$-greedy | 184 | 675 | 1088 | 565 | 819 | 976 | 611 | 1897 |
| | pymf | 96 | 703 | 1186 | 581 | 987 | 1106 | 602 | 2389 |
| | ASSO | 354 | 587 | 1724 | 694 | 1352 | 1505 | 661 | 2503 |
| | NMF | 153 | 826 | 1337 | 995 | 1143 | 1407 | 689 | 2481 |
| | MEBF | 122 | 990 | 1328 | 777 | 1004 | 1450 | 662 | 2460 |

Table 3 shows the primal objective values of MLP(1) and MLP($\frac{1}{k}$) with heuristic pricing using a time limit of 20 mins, and the objective value of MIP(1) solved on the columns generated by MLP(1). If the error measured in $\|\cdot\|_F^2$ differs from the objective of MIP(1), the former is shown in parenthesis. It is interesting to observe that MLP(1) has a tendency to produce near integral solutions and that the objective value of MIP(1) often coincides with the error measured in $\|\cdot\|_F^2$. We note that once a master LP formulation is used to generate columns, any of the MIP models could be used to obtain an integer feasible solution. In experiments, we have found that formulation MIP($\rho$) is solved much faster than $MIP_{exact}$ and that setting $\rho$ to 1 or $0.95$ provides the best integer solutions.

We compare the CG approach against the most widely used $k$-BMF heuristics and the exact approach of [16]. The heuristic algorithms we evaluated include the ASSO algorithm [28], the alternating iterative local search algorithm of [1] (ASSO++) which uses ASSO [28] as a starting point, algorithm $k$-greedy detailed in Appendix 5.9, the penalty objective formulation (pymf) of [41] and the permutation-based heuristic MEBF [40]. We also evaluate $IP_{exact}$ with a time limit of 20 mins and provide the heuristic solutions of ASSO++ and $k$-greedy as a warm start to it. In addition, we compute rank-$k$ NMF and binarise it with a threshold of $0.5$. The exact details and parameters used in the computations can be found in Appendix 5.10. Our CG approach (CG) results are obtained by generating columns for 20 mins using formulation MLP(1) with a warm start of initial columns obtained from ASSO++ and $k$-greedy, then solving MIP($\rho$) for $\rho$ set to 1 and $0.95$ over the generated columns and picking the best. Table 4 shows the factorisation errors in $\|\cdot\|_F^2$ after evaluating the above described methods on all datasets for $k = 2, 5, 10$. The best result for each instance is indicated in boldface. We observe that CG provides the strictly smallest error for 15 out of 24 cases.

## 5 Conclusion

In this paper, we studied the rank-$k$ binary matrix factorisation problem under Boolean arithmetic. We introduced a new integer programming formulation and detailed a method using column generation for its solution. Our experiments indicate that our method using 20 mins time budget is producing significantly more accurate solutions than most heuristics available in the literature and is able to prove optimality for smaller datasets. In certain critical applications such as medicine, spending 20 minutes to obtain a higher accuracy factorisation with a bound on the optimality gap can be easily justified. In addition, solving BMF to near optimality via our proposed method paves the way to more robustly benchmark heuristics such as the ones of [29], [1], [41], [40]. Future directions that could be explored are related to designing more accurate heuristics and faster exact algorithms for the pricing problem. In addition, a full branch-and-price algorithm implementation would be beneficial once the pricing problems are solved more efficiently.

## References


[1] Barahona, F.; and Goncalves, J. 2019. Local Search Algorithms for Binary Matrix Factorization. *https://github.com/IBM/binary-matrix-factorization/blob/master/code* .

[2] Barnhart, C.; Johnson, E. L.; Nemhauser, G. L.; Savelsbergh, M. W. P.; and Vance, P. H. 1998. Branch-and-Price: Column Generation for Solving Huge Integer Programs. *Operations Research* 46(3): 316–329.

[3] Beckerleg, M.; and Thompson, A. 2020. A divide-and-conquer algorithm for binary matrix completion. *Linear Algebra and its Applications* 601: 113–133. ISSN 0024-3795.

[4] Cios, K. J.; and Kurgan, L. A. 2001. UCI Machine Learning Repository: SPECT Heart Data. URL https://archive.ics.uci.edu/ml/datasets/spect+heart.

[5] CPLEX Optimization. 2018. *Using the CPLEX Callable Library, V.12.8*. CPLEX Optimization, Inc., Incline Village, NV.

[6] Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository. URL http://archive.ics.uci.edu/ml.

[7] Duarte, A.; Laguna, M.; Martí, R.; and Sánchez-Oro, J. 2014. Optimization Procedures for the Bipartite Unconstrained 0-1 Quadratic Programming Problem. *Comput. Oper. Res.* 51: 123–129.

[8] Forsyth, R. 1990. UCI Machine Learning Repository: Zoo Data Set. URL http://archive.ics.uci.edu/ml/datasets/Zoo.

[9] Garey, M. R.; and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co.

[10] Golub, G.; and Van Loan, C. 1989. *Matrix Computations*. Baltimore: Johns Hopkins University Press, 2nd edition.

[11] Gong, G. 1988. UCI Machine Learning Repository: Hepatitis Data Set. URL https://archive.ics.uci.edu/ml/datasets/Hepatitis.

[12] Karapetyan, D.; and Punnen, A. P. 2012. Heuristic algorithms for the bipartite unconstrained 0-1 quadratic programming problem. *CoRR* abs/1210.3684. URL http://arxiv.org/abs/1210.3684.

[13] Kim, K. 1982. *Boolean Matrix Theory and Applications*. Monographs and textbooks in pure and applied mathematics. Dekker. ISBN 9780824717889.

[14] Kononenko, I.; and Cestnik, B. 1988. UCI Mach. Learn. Rep.: Primary Tumor Domain. URL https://archive.ics.uci.edu/ml/datasets/Primary+Tumor.

[15] Kononenko, I.; and Cestnik, B. 1988. UCI Machine Learning Repository: Lymphography Data Set. URL https://archive.ics.uci.edu/ml/datasets/Lymphography.

[16] Kovacs, R. A.; Gunluk, O.; and Hauser, R. A. 2017. Low-Rank Boolean Matrix Approximation by Integer Programming. NIPS Optimization for Machine Learning Workshop.

[17] Koyutürk, M.; and Grama, A. 2003. PROXIMUS: A Framework for Analyzing Very High Dimensional Discrete-attributed Datasets. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, 147–156. New York, NY, USA: ACM.

[18] Koyutürk, M.; Grama, A.; and Ramakrishnan, N. 2002. Algebraic Techniques for Analysis of Large Discrete-Valued Datasets. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '02, 311–324. London, UK, UK: Springer-Verlag.

[19] Krebs, V. 2008. Amazon Political Books. URL http://moreno.ss.uci.edu/data.html#books.

[20] Lee, D.; and Sebastian Seung, H. 1999. Learning the Parts of Objects by Non-Negative Matrix Factorization. *Nature* 401: 788–91.

[21] Li, T. 2005. A General Model for Clustering Binary Data. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '05. New York, NY, USA: Association for Computing Machinery.

[22] Lu, H.; Vaidya, J.; and Atluri, V. 2008. Optimal Boolean Matrix Decomposition: Application to Role Engineering. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, 297–306. Washington, DC, USA: IEEE Computer Society.

[23] Lu, H.; Vaidya, J.; and Atluri, V. 2014. An optimization framework for role mining. *Journal of Computer Security* 22(1): 1 – 31.

[24] Lübbecke, M. E.; and Desrosiers, J. 2005. Selected Topics in Column Generation. *Operations Research* 53(6): 1007–1023.

[25] McCormick, G. P. 1976. Computability of Global Solutions to Factorable Nonconvex Programs: Part I – Convex Underestimating Problems. *Math. Program.* 10(1): 147–175. ISSN 0025-5610.

[26] Miettinen, P. 2008. On the Positive–Negative Partial Set Cover Problem. *Inf. Process. Lett.* 108(4): 219–221.

[27] Miettinen, P. 2012. Binary Matrix Factorisations Tutorial @ ECML PKDD 2012.

[28] Miettinen, P.; Mielikäinen, T.; Gionis, A.; Das, G.; and Mannila, H. 2006. The Discrete Basis Problem. In Fürnkranz, J.; Scheffer, T.; and Spiliopoulou, M., eds., *Knowledge Discovery in Databases: PKDD 2006*, 335–346. Berlin, Heidelberg: Springer Berlin Heidelberg.

[29] Miettinen, P.; Mielikäinen, T.; Gionis, A.; Das, G.; and Mannila, H. 2008. The Discrete Basis Problem. *IEEE Trans. on Knowl. and Data Eng.* 20(10): 1348–1362.

[30] Monson, S. D.; Pullman, N. J.; and Rees, R. 1995. A Survey of Clique and Biclique Coverings and Factorizations of (0,1)–Matrices. *Bulletin – Institute of Combinatorics and its Applications* 14: 17–86. ISSN 1183-1278.

[31] Orlin, J. 1977. Contentment in graph theory: Covering graphs with cliques. *Indagationes Mathematicae (Proceedings)* 80(5): 406 – 424. ISSN 1385-7258. doi:10.1016/1385-7258(77)90055-5.

[32] Padberg, M. 1989. The Boolean Quadric Polytope: Some Characteristics, Facets and Relatives. *Math. Program.* 45(1): 139–172. ISSN 0025-5610.

[33] Peeters, R. 2003. The maximum edge biclique problem is NP-complete. *Discrete Applied Mathematics* 131(3): 651 – 654.

[34] Punnen, A. P.; and Stephen, T. 2019. The Bipartite Boolean Quadric Polytope.

[35] Quinlan, R. 1992. UCI Machine Learning Repository: Audiology (Standardized) Data Set. URL http://archive.ics.uci.edu/ml/datasets/audiology+(standardized).

[36] Schlimmer, J. 1987. UCI Machine Learning Repository: 1984 US Cong. Voting Records Database. URL https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records.

[37] Shen, B.-H.; Ji, S.; and Ye, J. 2009. Mining Discrete Patterns via Binary Matrix Factorization. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, 757–766. New York, NY, USA: ACM.

[38] Shi, Z.; Wang, L.; and Shi, L. 2014. Approximation method to rank-one binary matrix factorization. In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, 800–805.

[39] Simon, H. U. 1990. On Approximate Solutions for Combinatorial Optimization Problems. *SIAM J. Discrete Math.* 3: 294–310.

[40] Wan, C.; Chang, W.; Zhao, T.; Li, M.; Cao, S.; and Zhang, C. 2020. Fast and Efficient Boolean Matrix Factorization by Geometric Segmentation. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(04): 6086–6093.

[41] Zhang, Z.; Li, T.; Ding, C.; and Zhang, X. 2007. Binary Matrix Factorization with Applications. In *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, ICDM '07, 391–400. USA: IEEE Computer Society.

# Appendix

## 5.1 The strength of the LP relaxation of $\text{IP}_{\text{exact}}$

**Lemma 1.** *For $k > 1$, the LP relaxation of $\text{IP}_{exact}$ ($\text{LP}_{exact}$) has optimal objective value 0 which is attained by at least $\binom{k}{2}$ solutions.*

*Proof.* Observe that the objective function of $\text{LP}_{\text{exact}}$ satisfies $0 \leq \sum_{(i,j)\in E}(1 - z_{ij}) + \sum_{(i,j)\notin E} z_{ij}$ as constraints (3), the McCormick envelopes and $a_{i\ell}, b_{\ell j} \in [0, 1]$ imply $z_{ij} \in [0, 1]$. Let us construct a feasible solution to $\text{LP}_{\text{exact}}$ which attains this bound. Let $a_{i\ell} = b_{\ell j} = \frac{1}{2}$ for all $i \in [n], j \in [m], \ell \in [k]$. The McCormick envelopes then are equivalent to $MC(\frac{1}{2}, \frac{1}{2}) = \{y \in \mathbb{R} : \frac{1}{2} + \frac{1}{2} - 1 \leq y, y \leq \frac{1}{2}, y \leq \frac{1}{2}, 0 \leq y\} = [0, \frac{1}{2}]$ hence we may choose the value of $y_{i\ell j} \in MC(\frac{1}{2}, \frac{1}{2})$ depending on the objective coefficient of indices $(i, j)$. For $(i, j) \in E$ the objective function is maximising $z_{ij}$ hence we set $y_{i\ell j} = \frac{1}{2}$ so that the upper bound $\sum_{\ell=1}^{k} y_{i\ell j}$ on $z_{ij}$ becomes greater than equal to 1 and $z_{ij}$ can take value 1. For $(i, j) \notin E$ the objective function is minimising $z_{ij}$ hence we set $y_{i\ell j} = 0$ so that the lower bounds $y_{i\ell j} \leq z_{ij}$ evaluate to 0 and $z_{ij}$ can take value 0. Therefore, the following setting of the variables shows the lower bound of 0 on the objective function is attained,

$$a_{i\ell} = \frac{1}{2},\ i \in [n], \ell \in [k]; \qquad b_{\ell j} = \frac{1}{2},\ \ell \in [k], j \in [m];$$
$$y_{i\ell j} = \frac{1}{2},\ (i, j) \in E, \ell \in [k]; \quad y_{i\ell j} = 0,\ (i, j) \notin E, \ell \in [k];$$
$$z_{ij} = 1,\ (i, j) \in E; \qquad z_{ij} = 0,\ (i, j) \notin E.$$

Furthermore, for all $(i, j) \in E$ it is enough to set $y_{i\ell_1 j} = y_{i\ell_2 j} = \frac{1}{2}$ for only two indices $\ell_1, \ell_2 \in [k]$ since this already achieves the upper bound $z_{ij} \leq 1 = \sum_{\ell=1}^{k} y_{i\ell j}$. Hence, there is at least $\binom{k}{2}$ different solutions of $\text{LP}_{\text{exact}}$ with objective value 0. □

## 5.2 The strength of the LP relaxation of $\text{MIP}_{\text{exact}}$

For our proof we will need a definition from the theory of binary matrices.

**Definition 2.** *[30, Section 2.3] Let $X$ be a binary matrix. A set $S \subseteq E = \{(i, j) : x_{ij} = 1\}$ is said to be an isolated set of ones if whenever $(i_1, j_1), (i_2, j_2)$ are two distinct members of $S$ then*

1. *$i_1 \neq i_2$, $j_1 \neq j_2$ and*
2. *$x_{i_1,j_2} x_{i_2,j_1} = 0$.*

*The size of the largest cardinality isolated set of ones in $X$ is denoted by $i(X)$ and is called the isolation number of $X$.*

Observe that requirement (1.) implies that an isolated set of ones can contain the index corresponding to at most one entry in each column and row of $X$. Hence $i(X) \leq \min(n, m)$. Requirement (2.) implies that if $(i_1, j_1), (i_2, j_2)$ are members of an isolated set of ones then at least one of the entries $x_{i_1,j_2}$, $x_{i_2,j_1}$ is zero, hence members of an isolated set of ones cannot be contained in a common rank-1 submatrix of $X$. Therefore if the largest cardinality isolated set of ones has $i(X)$ elements, to cover all 1s of $X$ we need at least $i(X)$ many rank-1 binary matrices in a factorisation of $X$, so $i(X)$ provides a lower bound on the Boolean rank of $X$.

**Lemma 2.** *Let $X$ be a binary matrix with isolation number $i(X)$. Then for rank-$k$ binary matrix factorisation of $X$ with $k < i(X)$, the LP-relaxation of $\text{MIP}_{exact}$ has non-zero optimal objective value.*

*Proof.* Let $k$ be a fixed positive integer and let $X$ be a binary matrix with isolation number $i(X) > k$. For a contradiction, assume that $\text{MLP}_{\text{exact}}$ has objective value zero, $\zeta_{\text{MLP}} = 0$.
(1.) Now, if $\zeta_{\text{MLP}} = 0$ we must have $M_0 \boldsymbol{q} = \mathbf{0} = \boldsymbol{\pi}$ in constraint (10) which implies that none of the rank-1 binary matrices with $q_\ell > 0$ cover zero entries of $X$. In other words, all the rank-1 binary matrices active are submatrices of $X$.
(2.) Now, if $\zeta_{\text{MLP}} = 0$ we must also have $\boldsymbol{\xi} = \mathbf{0}$ which implies $M_{\mathbf{1}} \boldsymbol{q} \geq \mathbf{1}$ in constraint (9) for some $\boldsymbol{q}$ which may be fractional but satisfies $\mathbf{1}^\top \boldsymbol{q} \leq k$. Let $S := \{(i_1, j_1), \ldots, (i_r, j_r), \ldots, (i_{|S|}, j_{|S|})\} \subseteq E$ be an isolated set of ones in $X$ of cardinality $i(X) = |S|$. Since members of $S$ cannot be contained in a common rank-1 submatrix of $X$, all columns $M_\ell$ corresponding to rank-1 binary submatrices of $X$ for entries $(i_r, j_r) \in S$ satisfy

$$M_{(i_r,j_r),\ell} = 1 \Longrightarrow M_{(i,j),\ell} = 0 \ \forall\ (i, j) \neq (i_r, j_r), (i, j) \in S.$$

Therefore, we can partition the active rank-1 binary matrices into $i(X) + 1$ groups $G_r$,

$$G_r = \{\ell : q_\ell > 0 \text{ and } M_{(i_r,j_r),\ell} = 1\} \quad \text{for } r = 1, \ldots, i(X); \tag{24}$$
$$G_{i(X)+1} = \{\ell : q_\ell > 0 \text{ and } M_{(i_r,j_r),\ell} = 0\ \forall (i_r, j_r) \in S\}. \tag{25}$$

While $G_{i(X)+1}$ may be empty, $G_r$'s for $r = 1, \ldots, i(X)$ are not empty because we know that for all $(i, j) \in E$ we have $\sum_\ell M_{(i,j),\ell}\ q_\ell \geq 1$ and $S \subseteq E$. Hence for all $(i_r, j_r) \in S$ we have $\sum_{\ell \in G_r} q_\ell \geq 1$ which implies the contradiction $\mathbf{1}^\top \boldsymbol{q} \geq \sum_{r=1}^{i(X)} \sum_{\ell \in G_r} q_\ell \geq i(X) > k$ and therefore $\zeta_{\text{MLP}} > 0$. □

Could we replace the condition $k < i(X)$ in Lemma 2 by a requirement that $k$ has to be smaller than the Boolean rank of $X$? The following example shows that we cannot.

**Example 2.** *Let $X = J_4 - I_4$, where $J_4$ is the $4 \times 4$ matrix of all 1s and $I_4$ is the $4 \times 4$ identity matrix. One can verify that the Boolean rank of $X$ is 4 and its isolation number is 3.*

$$X = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{26}$$

*For $k = 3$, the optimal objective value of $\text{MLP}_{exact}$ is 0 which is attained by a fractional solution in which the following 6 rank-1 binary matrices are active.*

$$q_1 = \tfrac{1}{2}: \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad q_2 = \tfrac{1}{2}: \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad q_3 = \tfrac{1}{2}: \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$q_4 = \tfrac{1}{2} \quad \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad q_5 = \tfrac{1}{2} \quad \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad q_6 = \tfrac{1}{2} \quad \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

### 5.3 The relation between the LP relaxations of $\text{MIP}_{\text{exact}}$ and $\text{MIP}(\frac{1}{k})$

**Lemma 3.** *For $\rho = \frac{1}{k}$, the optimal objective values of the LP relaxations $\text{MLP}_{exact}$ and $\text{MLP}(\frac{1}{k})$ coincide.*

*Proof.* It is enough to observe that since $\text{MLP}_{\text{exact}}$ is a minimisation problem, $\boldsymbol{\pi}$ takes the minimal optimal value $\frac{1}{k} M_0 \boldsymbol{q}$ in $\text{MLP}_{\text{exact}}$ due to constraint (10) which equals the second term in the objective (14) of $\text{MLP}(\frac{1}{k})$. □

### 5.4 Column generation applied to the LP relaxation of the strong formulation of $\text{MIP}_{\text{exact}}$

We obtain a modification of $\text{MIP}_{\text{exact}}$ which we call the "strong formulation" by replacing constraints (10) by exponentially many constraints. The following is the LP relaxation of the strong formulation of $\text{MIP}_{\text{exact}}$,

$$\begin{aligned} (\text{MLP}_{\text{exact strong}}) \zeta_{\text{MLP}} = \min \; & \mathbf{1}^\top \boldsymbol{\xi} + \mathbf{1}^\top \boldsymbol{\pi} \\ \text{s.t. } & M_{\mathbf{1}} \boldsymbol{q} + \boldsymbol{\xi} \geq \mathbf{1} \\ & (M_{\mathbf{0}})_\ell \; q_\ell \leq \boldsymbol{\pi}, \quad \ell \in [(2^n - 1)(2^m - 1)] \\ & \mathbf{1}^\top \boldsymbol{q} \leq k \\ & \boldsymbol{\xi} \geq \mathbf{0}, \boldsymbol{\pi} \in [0,1]^{nm - |E|}, \boldsymbol{q} \in [0,1]^{|\mathcal{R}|}. \end{aligned}$$

**Lemma 4.** *Applying the CG approach to $\text{MLP}_{exact\ strong}$ cannot be used to generate sensible columns.*

*Proof.* Let us try applying column generation to solve $\text{MLP}_{\text{exact strong}}$ and add a column of all 1s as our first column $q_1$. Then at the 1st iteration, for $q_1 = 1$ the objective value of the Restricted MLP is $\zeta^{(1)}_{\text{RMLP}} = 0 + (nm - |E|)$ for solution vector $[\boldsymbol{\xi}^{(1)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{q}^{(1)}] = [\mathbf{0}, \mathbf{1}, 1]$. Adding the same column of all 1s at the next iteration and setting $[q_1, q_2] = [\frac{1}{2}, \frac{1}{2}]$, allows us to keep $\boldsymbol{\xi}^{(2)} = \mathbf{0}$ but set $\boldsymbol{\pi}^{(2)} = \frac{1}{2}\mathbf{1}$ to get $\zeta^{(2)}_{\text{RMLP}} = 0 + \frac{1}{2}(nm - |E|)$. Therefore continuing adding the same column of all 1s, after $t$ iterations we have $\zeta^{(t)}_{\text{RMLP}} = 0 + \frac{1}{t}(nm - |E|)$ for solution vector $[\boldsymbol{\xi}^{(t)}, \boldsymbol{\pi}^{(t)}, \boldsymbol{q}^{(t)}] = [\mathbf{0}, \frac{1}{t}\mathbf{1}, \frac{1}{t}\mathbf{1}]$. Therefore for $t \to \infty$ we have $\zeta^{(t)}_{\text{RMLP}} \to 0$ and we have not generated any other columns but the all 1s. □

### 5.5 The greedy algorithm for bipartite binary quadratic optimisation

For a given $n \times m$ coefficient matrix $H$, we aim to find $\boldsymbol{a} \in \{0,1\}^n$ and $\boldsymbol{b} \in \{0,1\}^m$ so that $\boldsymbol{a}^\top H \boldsymbol{b}$ is maximised. Let $\gamma_i^+$ be the sum of the positive entries of $H$ for each row $i \in [n]$, $\gamma_i^+ := \sum_{j=1}^m \max(0, h_{ij})$. Reorder the rows of $H$ according to decreasing values of $\gamma_i^+$. Algorithm 1 is the greedy heuristic of [12] which provides the optimal solution to the bipartite binary quadratic problem if $\min(n, m) \leq 2$ and has an approximation ratio of $1/(\min(n, m) - 1)$ otherwise.

**Algorithm 1:** Greedy Algorithm

```
Phase I. Order i ∈ [n] so that γ_i^+ ≥ γ_{i+1}^+.
Set a = 0_n, s = 0_m.
for i ∈ [n] do
    f_{i-1} = Σ_{j=1}^m max(0, s_j)
    f_i = Σ_{j=1}^m max(0, s_j + h_ij)
    if f_{i-1} < f_i then
        Set a_i = 1, s = s + h_i
    end
end
Phase II. Set b = 0_m.
for j ∈ [m] do
    if (a^⊤ H)_j > 0 then
        Set b_j = 1
    end
end
```

### 5.6 Formulations evaluated in Table 2

The formulation of [16] with objective (6) for $\rho = 1$ that was evaluated to get results in column [16] of Table 2 reads as

$$\begin{aligned} \min_{a,b,z} \; & \sum_{(i,j) \in E} (1 - z_{ij}) + \sum_{(i,j) \notin E} \sum_{\ell=1}^{k} y_{i\ell j} \\ \text{s.t. } & z_{ij} \leq \sum_{\ell=1}^{k} y_{i\ell j}, && (i,j) \in E, \\ & y_{i\ell j} \in MC(a_{i\ell}, b_{\ell j}), && i \in [n], \ell \in [k], j \in [m], \\ & z_{ij} \in [0,1], && (i,j) \in E, \\ & a_{i\ell}, b_{\ell j} \in \{0,1\}, && i \in [n], \ell \in [k], j \in [m], \end{aligned}$$

with $MC(a,b) = \{y \in \mathbb{R} : a + b - 1 \leq y, y \leq a, y \leq b, 0 \leq y\}$ denoting the McCormick envelopes as defined in Section 2. In the exponential formulation of [22, 23] all possible non-zero binary row vectors $\boldsymbol{\beta}_t \in \{0,1\}^{1 \times m}$ ($t \in [2^m - 1]$) for factor matrix $B \in \{0,1\}^{k \times m}$ are explicitly enumerated and treated as fixed input parameters to the formulation. The formulation with objective (6) for $\rho = 1$ that was evaluated to get results in column [22, 23] of Table 2 reads as

$$\begin{aligned} \min_{\alpha,\delta,z} \; & \sum_{(i,j) \in E} (1 - z_{ij}) + \sum_{(i,j) \notin E} \sum_{t=1}^{2^m - 1} \alpha_{it} \, \beta_{tj} \\ \text{s.t. } & z_{ij} \leq \sum_{t=1}^{2^m - 1} \alpha_{it} \, \beta_{tj}, && (i,j) \in E, \\ & \alpha_{it} \leq \delta_t, && i \in [n], t \in [2^m - 1], \\ & \sum_{t=1}^{2^m - 1} \delta_t \leq k, \\ & z_{ij} \in [0,1], && (i,j) \in E, \\ & \alpha_{it}, \delta_t \in \{0,1\}, && i \in [n], t \in [2^m - 1]. \end{aligned}$$

### 5.7 Datasets

In general if a dataset has a categorical feature $C$ with $N$ discrete options $v_j$, $(j \in [N])$, we convert feature $C$ into $N$ binary features $B_j$ $(j \in N)$ so that if the $i$-th sample takes value $v_j$ for $C$ that is $(C)_i = v_j$, then we have value $(B_j)_i = 1$ and $(B_\ell)_i = 0$ for all $\ell \neq j \in [N]$. This techinque of binarisation of categorical columns has been applied in [16] and [1]. The following datasets were used in the experiments:

- The Zoo dataset (*zoo*) [8] describes 101 animals with 16 characteristic features. All but one feature is binary. The categorical column which records the number of legs an animal has, is converted into two new binary columns indicating if the number of legs is *less than or equal* or *greater* than four. The size of the resulting fully binary dataset is $101 \times 17$.
- The Primary Tumor dataset (*tumor*) [14] contains observations on 17 tumour features detected in 339 patients. The features are represented by 13 binary variables and 4 categorical variables with discrete options. The 4 categorical variables are converted into 11 binary variables representing each discrete option. Two missing values in the binary columns are set to value 0. The final dimension of the dataset is $339 \times 24$.
- The Hepatitis dataset (*hepat*) [11] consists of 155 samples of medical data of patients with hepatitis. The 19 features of the dataset can be used to predict whether a patient with hepatitis will live or die. 6 of the 19 features take numerical values and are converted into 12 binary features corresponding to options: *less than or equal to the median value*, and *greater than the median value*. The column that stores the sex of patients is converted into two binary columns corresponding to labels man and female. The remaining 12 columns take values *yes* and *no* and are converted into 24 binary columns. The raw dataset contains 167 missing values, and according to the above binarisation if a sample has missing entry for an original feature it will have 0's in both columns binarised from that original feature. The final binary dataset has dimension $155 \times 38$.
- The SPECT Heart dataset (*heart*) [4] describes cardiac Single Proton Emission Computed Tomography images of 267 patients by 22 binary feature patterns. 25 patients' images contain none of the features and are dropped from the dataset, hence the final dimension of the dataset is $242 \times 22$.
- The Lymphography dataset (*lymp*) [15] contains data about lymphography examination of 148 patients. 8 features take categorical values and are expanded into 33 binary features representing each categorical value. One column is numerical and we convert it into two binary columns corresponding to options: *less than or equal to median value*, and *larger than median value*. The final binary dataset has dimension $148 \times 44$.
- The Audiology Standardized dataset (*audio*) [35] contains clinical audiology records on 226 patients. The 69 features include patient-reported symptoms, patient history information, and the results of routine tests which are needed for the evaluation and diagnosis of hearing disorders. 9 features that are categorical valued are binarised into 34 new binary variables indicating if a discrete option is selected. The final dimension of the dataset is $226 \times 94$.
- The Amazon Political Books dataset (*books*) [19] contains binary data about 105 US politics books sold by Amazon.com. Columns correspond to books and rows represent frequent co-purchasing of books by the same buyers. The dataset has dimension $105 \times 105$.
- The 1984 United States Congressional Voting Records dataset (*votes*)[36] includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. The 16 categorical variables taking values of "voted for", "voted against" or "did not vote", are converted into 32 binary variables. One congressman did not vote for any of the bills and its corresponding row of zero is dropped. The final binary dataset has dimension $434 \times 32$.

### 5.8 Data preprocessing

In practice, the input matrix $X \in \{0,1\}^{n \times m}$ may contain zero rows or columns. Deleting a zero row (column) leads to an equivalent problem whose solution $A$ and $B$ can easily be translated to a solution of the original dimension. In addition, if a row (column) of $X$ is repeated $\alpha_i$ $(\beta_j)$ times, it is sufficient to keep only one copy of it, solve the reduced problem and reinsert the relevant copies in the corresponding place. To ensure that the objective function of the reduced problem corresponds to the factorisation error of the original problem, the variable corresponding to the representative row (column) in the reduced problem is multiplied by $\alpha_i$ $(\beta_j)$.

**Algorithm 2:** Greedy algorithm for rank-$k$ binary matrix factorisation

Input: $X \in \{0,1\}^{n \times m}$, $k \in \mathbb{Z}_+$.
Set $H := 2X - 1$.
**for** $\ell \in [k]$ **do**
  $\boldsymbol{a}, \boldsymbol{b} = \text{Greedy}(H)$ `// Compute a rank-1 binary matrix via the greedy algorithm`
  $\boldsymbol{a}, \boldsymbol{b} = \text{Alt}(H, \boldsymbol{a}, \boldsymbol{b})$ `// Improve greedy solution with the alternating heuristic`
  $A_{:,\ell} = \boldsymbol{a}$
  $B_{\ell,:} = \boldsymbol{b}^\top$
  $H[\boldsymbol{a}\boldsymbol{b}^\top == 1] = 0$ `// Set entries of` $H$ `to zero that are covered`
**end**
Output: $A \in \{0,1\}^{n \times k}$, $B \in \{0,1\}^{k \times m}$

### 5.9 Rank-$k$ greedy heuristic

For a given $X \in \{0,1\}^{n \times m}$ and $k \in \mathbb{Z}_+$, according to Equation (1) we may write $\min \|X - Z\|_2^F$ as $|E| - \max \sum_{i \in [n], j \in [m]} h_{ij} z_{ij}$ were $h_{ij}$ are entries of $H := 2X -$

1. We propose the heuristic in Algorithm 2 to compute $k$-BMF by sequentially computing $k$ rank-1 binary matrices using the greedy Algorithm 1.

We remark that this rank-$k$ greedy algorithm can be used to obtain a heuristic solution to $k$-BMF under standard arithmetic as well: simply modify the last line to $H[\boldsymbol{a}\boldsymbol{b}^\top == 1] = -K$ for a large enough number $K$ (say $K := \sum_{ij} x_{ij}$) so that each entry of $X$ is covered at most once.

### 5.10 Comparison Methods

The following methods were evaluated for the comparison in Table 4.

- For the alternating iterative local search algorithm of [1] (ASSO++) we obtained the code from the author's github page. The code implements two variants of the algorithm and we report the smaller error solution from two variants of it.
- The greedy algorithm ($k$-greedy) detailed in Appendix 5.9 is evaluated with nine different orderings and the best result is chosen.
- For the method of [41], we used an implementation in the github pymf package by Chris Schinnerl and we ran it for 10000 iterations.
- We evaluated the heuristic method ASSO [28] which depends on a parameter and we report the best results across nine parameter settings ($\tau \in \{0.1, 0.2, \ldots, 0.9\}$). The code was obtained form the webpage of the author.
- In addition, we computed rank-$k$ non-negative matrix factorisation (NMF) and binarise it by a threshold of $0.5$: after an NMF is obtained, values greater than $0.5$ are set to 1, otherwise to $0$. For the computation of NMF we used `sklearn.decomposition` module in Python.
- For the MEBF method [40] we used the code from the author's github page. The raw code downloaded contained a bug and did not produce a solution for some instances while for others it produced factorisations whose error in $\|\cdot\|_F^2$ increased with the factorisation rank $k$. We fixed the code and the results shown in 4 correspond to the lowest error for each instance selected across 9 parameter settings $t \in \{0.1, \ldots, 0.9\}$.