Hybrid Restricted Master Problem for Boolean Matrix Factorisation

Ellen Visscher[1], **Michael Forbes**[2], **Christopher Yau**[1,3],

[1]University of Oxford
[2]University of Queensland
[3]Health Data Research UK
ellen.visscher@bdi.ox.ac.uk, m.forbes@uq.edu.au, christopher.yau@wrh.ox.ac.uk

Abstract

We present `bfact`, a Python package for performing accurate low-rank Boolean matrix factorisation (BMF). `bfact` uses a hybrid combinatorial optimisation approach based on *a priori* candidate factors generated from clustering algorithms. It selects the best disjoint factors before performing either a second combinatorial or heuristic algorithm to recover the BMF. We show that `bfact` does particularly well at estimating the true rank of matrices in simulated settings. In real benchmarks, using a collation of single-cell RNA-sequencing datasets from the Human Lung Cell Atlas, we show that `bfact` achieves strong signal recovery, with a much lower rank.

Extended Version — https://arxiv.org/abs/2509.06192
Package Code — https://github.com/e-vissch/bfact-core
Analysis Code — https://zenodo.org/records/17589218

Introduction

Matrix factorisation techniques are frequently used in machine learning to identify latent structure in data sets by decomposing them into lower-dimensional representations that capture common underlying patterns or concepts. Popular matrix factorisation techniques include Singular Value Decomposition (SVD), Principal Component Analysis (PCA), and Nonnegative Matrix Factorisation (NMF). SVD and PCA require orthogonal factors, while NMF constrains the target matrix and the factors to be nonnegative. In the special case where the input is a binary matrix, Boolean matrix factorisation (BMF) seeks to produce two low-rank Boolean factor matrices whose Boolean product is close to the original input. With binary outputs, BMF preserves a form of interpretability with one factor matrix composed of a limited set of binary basis vectors capturing combinations of frequently co-occurring raw features, while the other associates input samples to one or more basis vectors. This has seen its use in many data mining applications in market basket analysis, bioinformatics and recommender systems.

Motivation

In this work, we are particularly motivated by the use of BMF for applications in biology and specifically, that of single-cell RNA sequencing analysis (scRNAseq). Single-cell technologies now enable biologists to routinely screen hundreds of thousands of cells for the activity (or *expression*) of tens of thousands of genes. The data matrices are therefore significant (e.g $\sim 100\text{k} \times 15\text{k}$) and a common task is to perform a form of dimensionality reduction to interpret the data by projecting onto low-dimensional representations using techniques such as principal components analysis (PCA), non-negative matrix factorisation (NNMF) (Qi et al. 2020; Wang et al. 2022; Tsuyuzaki et al. 2020), visualisation methods (t-SNE, UMAP) and deep learning approaches (Wu and Zhang 2020; Wang, Zou, and Lin 2022). However, single-cell data can often be approximated as binary due to the relatively sparse number of sequencing reads per cell, making the use of BMF amenable (Rukat et al. 2017; Liang, Zhu, and Lu 2020). Further, the direct interpretation of factor feature (gene) sets is useful for downstream analysis.

Theory

We first introduce the formal mathematical background. We wish to decompose $\mathbf{X} \in \{0,1\}^{M\times N}$ into two other low-rank binary matrices, $\mathbf{L} \in \{0,1\}^{M\times K}$, $\mathbf{R} \in \{0,1\}^{K\times N}$ such that the original matrix can be recapitulated using Boolean logic according to $X_{ij} = \bigvee_{k=1}^{K} L_{ik} \wedge R_{kj}$. Here, (M, N) corresponds to the number of observations and features, respectively and typically $K \ll N$. In practice, observations are corrupted by noise and it is common to assume that the observations $\mathbf{Y} \in \{0,1\}^{M\times N} = \mathbf{X} + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon}$ is an additive noise matrix such that $\epsilon_{ij} \in \{-1, 0, 1\}$. The objective is to recover the signal matrix $\mathbf{X}$ through finding the lowest rank $\hat{\mathbf{L}}$ and $\hat{\mathbf{R}}$ that minimise reconstruction error on observed $\mathbf{Y}$. However, the exact problem of Boolean matrix factorisation is NP-complete, and its optimisation variant (i.e., minimising reconstruction error for a given rank) is NP-hard, necessitating heuristic or approximate methods in practice (Miettinen and Neumann 2020).

Related Work

Given the combinatorial complexity of the problem, numerous heuristic and exact methods have been developed, each with different assumptions and optimisation strategies. In this section, we review key contributions in the literature on Boolean matrix factorisation and related techniques.

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

ASSO (Miettinen et al. 2008) is a fast, greedy algorithm for Boolean Matrix Factorisation that constructs low-rank binary approximations by mining frequent itemsets from the input matrix. It identifies combinations of co-occurring features (called tiles) and selects a subset that best reconstructs the original matrix using Boolean OR and AND operations. ASSO iteratively selects itemsets that maximise coverage while minimising overlap and error, and solves a set cover problem to finalise the factor matrices, enabling interpretable and scalable BMF. MDL4BMF (Miettinen and Vreeken 2014) builds upon ASSO by integrating the Minimum Description Length (MDL) principle, which frames Boolean matrix factorisation as a model selection problem balancing complexity and goodness of fit. MDL4BMF includes a cost function that penalises the number of bits needed to encode the factor matrices $\mathbf{L}$ and $\mathbf{R}$. As a result, since the MDL objective balances model size and error, MDL4BMF can automatically select an appropriate number of factors K, avoiding the need to pre-specify K. PANDA+ (Lucchese, Orlando, and Perego 2014) is similar to MDL4BMF in that it generates candidate columns using a greedy algorithm; however, it directly optimises a complexity-based objective during the factorisation process, rather than applying complexity considerations post-hoc.

Using a continuous relaxation approach, PRIMP (Hess, Morik, and Piatkowski 2017) formulates Boolean matrix factorisation as an optimisation problem with a two-part objective function. The first component is a reconstruction error term measured using standard algebraic matrix multiplication and the Frobenius norm, rather than Boolean algebra. Specifically, given an observed binary matrix $\mathbf{Y} \in \{0,1\}^{M\times N}$, PRIMP seeks factor matrices $\mathbf{L} \in [0,1]^{M\times K}$ and $\mathbf{R} \in [0,1]^{K\times N}$ by minimising $\min_{\mathbf{L},\mathbf{R}} \|\mathbf{Y} - \mathbf{LR}\|_F^2 + \lambda\,\mathcal{R}(\mathbf{L},\mathbf{R})$, where $\|\cdot\|_F$ denotes the Frobenius norm, $\lambda > 0$ is a regularisation parameter, and $\mathcal{R}(\mathbf{L},\mathbf{R})$ is a regularisation term designed to promote binary-valued entries and low-rank structure in the factor matrices.

To solve this nonconvex and nonsmooth optimisation problem, PRIMP employs *Proximal Alternating Linearized Minimization (PALM)* (Bolte, Sabach, and Teboulle 2014), an iterative algorithm that generalises the Gauss-Seidel method to handle composite objective functions. PALM alternately updates $\mathbf{L}$ and $\mathbf{R}$ by performing proximal gradient steps that linearise the smooth part of the objective while incorporating proximal operators to handle nonsmooth regularisation terms. The proximal updates ensure convergence to a critical point despite the nonconvexity and nonsmoothness of the problem. By relaxing the binary constraints to continuous domains, PRIMP enables the use of efficient gradient-based optimisation methods, while the regularisation term encourages the learned matrices to be close to binary, facilitating interpretable factorisation. PRIMP evaluates the *Minimum Description Length* (MDL) cost across multiple candidate values of K for model selection.

Kovacs, Gunluk, and Hauser (2021) take a Mixed Integer Programming (MIP) approach for the BMF problem, leveraging the insight that a rank-K matrix factorisation can be decomposed as the sum of K rank-1 matrix factorisations. They construct a restricted master problem that iteratively selects the K best rank-1 matrices from a potentially large set of candidate matrices. To efficiently handle this, they employ delayed column generation, dynamically adding advantageous rank-1 matrices to the master problem during optimisation. This technique improves scalability by avoiding the need to enumerate all candidates upfront; however, the approach is still limited to medium-sized datasets. Another limitation of this approach is that the desired rank, K, must be prespecified before solving, requiring prior knowledge or additional model selection steps.

Of the limited more recent BMF methods that also estimate rank, few compare to the state of the art in literature, or do not significantly outperform them. Please see Extended Version Appendix A.1 for a detailed comparison.

Contributions

We introduce a novel BMF approach, called `bfact`, that solves a MIP related to BMF, selecting disjoint column sets that best explain the data. This can be combined with a second MIP-based or faster, yet effective, heuristic approach to perform standard BMF. The approach automatically selects the relevant rank using complexity measures or reconstruction error, and scales to large datasets. It performs well against existing BMF methods in simulated scenarios, standard real data benchmarks, and 14 scRNAseq datasets from the human cell lung atlas.

We provide the MIP formulation for finding an approximate BMF with disjoint column sets and provide the pricing problem to find the optimal disjoint BMF solution on smaller matrices, using delayed column generation. We provide a second MIP formulation that could be used for exact BMF on smaller datasets using delayed column generation. We show that complexity costs often used in BMF algorithms can favour sparser, higher-rank factorisations.

Method

Our approach, illustrated in Figure 1, begins by generating candidate factors through clustering on features. We then solve a warm-started restricted master problem (RMP-w) that approximates the BMF by selecting up to K_c of these factors (K_c initialised to some K_{min}). Depending on the selected metric, the method either heuristically reassigns features and prunes factors (`bfact`-recon or `bfact`-MDL) or performs a second combinatorial approach to refine the factorisation (`bfact`-MIP). The process iteratively increases the maximum number of factors, K_c, stopping to give the best factorisation solution if the metric error does not improve within s_i steps. This two-stage framework—starting from disjoint candidate factors and refining via heuristic or optimisation—echoes the ASSO methodology while incorporating modern optimisation techniques. Below, we explain each of these steps in detail.

Master Problem for Approximate BMF

Here we borrow ideas from the combinatorial optimisation world. We define a 'master problem' (MP) that approximates the BMF by finding sets of (mostly) disjoint features that best explain the observations. For our observed

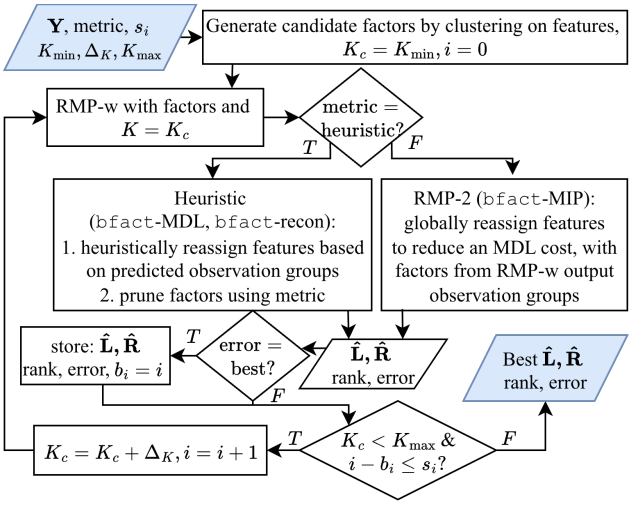

Figure 1: Overview of bfact

binary matrix $\mathbf{Y}$, consider the set $\mathcal{A}$ of all possible non-zero feature-sets, or factors, α, where $\delta_\alpha \in \{0,1\}^N$ and $|\mathcal{A}| = 2^N - 1$. We define also $c_{i,\alpha}$, the cost for observation i of choosing factor α, as:

$$C_{i,\alpha} = \left(\sum_{j|(i,j)\in E} \delta_{\alpha,j}, \sum_{j|(i,j)\notin E} \delta_{\alpha,j} \right) \tag{1}$$

$$c_{i,\alpha} = \min C_{i,\alpha} \tag{2}$$

$$l_{i,\alpha} = \arg\min C_{i,\alpha} \tag{3}$$

where $E = \{(i,j)|Y_{i,j} = 1\}$. The left term of $C_{i,\alpha}$ is the intersection of features in an observation and in a factor. While the right term can be thought of as the complement of an observation - that is, the features included in a factor that are not present in the observation.

The initial MP is thus defined as:

$$\min \sum_j u_j \sum_i Y_{ij} + \sum_\alpha z_\alpha \sum_i c_{i,\alpha} \tag{4}$$

$$\sum_{\alpha\in\mathcal{A}} z_\alpha \leq K \tag{5}$$

$$\sum_{\alpha\in A} \delta_{\alpha,j} z_\alpha + u_j \geq 1 \tag{6}$$

where $u_j \in \mathbb{R}^+, \forall j = 1,\ldots,N$, and $z_\alpha \in \{0,1\}\ \forall \alpha \in \mathcal{A}$.

To interpret this formulation, first consider if Equation 6 were an equality (i.e $\sum_{\alpha\in A} \delta_{\alpha,j} z_\alpha + u_j = 1$). This enforces that no factors may share overlapping features, making them disjoint, and here u_j represents features that are not included in any factor. In this case, Objective 4 is the reconstruction error, where the left term penalises false-negatives for any features that are omitted from the chosen factor representation. All other features are included in a factor, and hence, for a given chosen factor, each sample either:

1. Uses that factor - if the factor includes features not in the sample, then costs associated with false-positives should be incurred (right term of Equation 1).
2. Does not use that factor - any features included in that factor cannot be used in another factor (assuming equality of Equation 6), and overlapping features in the sample will not be covered, hence costs associated with false-negatives should be incurred (left term of Equation 1).

Although the inequality in Equation 6 relaxes the enforcement of disjoint factors, the precomputed cost still implicitly assumes disjointness in its calculation of false positives and negatives, meaning false negatives induced from one chosen factor across samples will still be penalised even if another factor covers those features. Similarly, false positives will be penalised according to standard (not Boolean) addition. Hence, the formulation is still mostly disjoint; the constraint relaxation makes it easier to solve (particularly in the RMP-w setting introduced in the next setting). Although a non-disjoint formulation can be proposed (and is proposed later in RMP-2), this formulation excels because costs are precomputed per factor instead of relying on costs dependent on a composition of chosen factors that cannot be precomputed at scale. Finally, Equation 5 allows at most K factors to be selected.

Hence, the solution to the MP above gives an approximate, (mostly) disjoint, BMF with $\hat{\mathbf{R}}_d = \{\delta_{\alpha,j}|z_\alpha = 1\}$, and $\hat{\mathbf{L}}_d = \{l_{i,\alpha}|z_\alpha = 1\}$. Note, the disjoint factorisation is similar to a clustering approach on the features, however, unlike in clustering, the term u_j allows any number of features to be excluded from selected factors. See the Extended Version Appendix A.2, Figure A1, for a brief example illustrating the formulation.

Warmstarted Restricted Master Problem (RMP-w)

Solving the MP above would require enumerating all possible factors $\mathcal{A}$, exponential in the number of variables and intractable. Delayed column generation is an optimisation technique that can combat this and involves solving the restricted master problem (RMP) (Dantzig and Wolfe 1960). The RMP is the master problem 'restricted' to a subset of possible factors, $\mathcal{A}' \subseteq \mathcal{A}$, where typically $|\mathcal{A}'| \ll |\mathcal{A}|$, making it tractable to solve.

In delayed column generation, factors that will reduce the objective are iteratively added to the RMP by solving the pricing problem, based on dual variables from the RMP linear relaxation (Vanderbeck and Savelsbergh 2006). Eventually, no more factors (i.e. columns) can be added that will reduce the objective, and the global optima of the linear relaxation is found, without having to realise the full MP linear relaxation. To solve the integer MP to optimality, a branch-and-price approach can be taken, again without having to realise the full MP.

Such an approach can work even when the RMP starts with an empty factor set. However, it is often faster to warm-start the RMP with *a priori* candidate factors. A benefit of the disjoint formulation is that clustering on the features provides a simple way to generate good candidate factors. Hence, we perform hierarchical and Leiden clustering across the features to generate a set of candidate factors, which, together with our RMP, we call RMP-w. For exact details on

the candidate factor generation, see Extended Version Appendix A.3.

We first attempted to solve the RMP-w using a delayed column generation approach, like Kovacs, Gunluk, and Hauser (2021). Here, we found that the pricing problem struggled to reduce the linear relaxation objective in a reasonable time frame (see Extended Version Appendix A.4), suggesting that the solution to RMP-w already produces good-quality disjoint factorisations. Given the formulation is an approximation to the BMF, we proceed using RMP-w alone (i.e. using only the cluster candidate factors, without solving to optimality).

Note, our RMP scales (variables, constraints) with $(|\mathcal{A}'|+ N,\ N)$. This makes it more computationally tractable than an exact approach such as Kovacs, Gunluk, and Hauser (2021), which scales with $(\rho MN + |\mathcal{D}'|,\ \rho MN)$, where $\mathcal{D}'$ is the restricted set of all rank-1 $M \times N$ matrices, and ρ is the density of ones in the data matrix.

Two-step BMF

RMP-w does not directly solve for a BMF; however, we hypothesised that it may provide a good basis for a BMF after adding some post-processing. Hence, we adopt a two-step approach. Intuitively, the first step, RMP-w, selects (mostly) disjoint pregenerated feature sets to find likely observation sets, and the second step uses these observation sets to update the feature sets, allowing them to share features. The second step also controls the model complexity to select lower-rank approximations, where appropriate.

Two-step MIP-BMF: Here, we formulate a second restricted master problem (RMP-2), similar to the above, but instead for (nearly) exact BMF. Consider the set of all factors $\mathcal{B}$, now with the set of observations, $\delta_\beta \in \{0,1\}^M$ associated with each factor, $\beta \in \mathcal{B}$. We define RMP-2 to be over a restricted factor set $\mathcal{B}' \subseteq \mathcal{B}$, given by:

$$\min \sum_{(i,j)\in E}(1-e_{i,j}) + \sum_{(i,j)\notin E}\sum_{\beta\in\mathcal{B}'}\delta_{\beta,i}R_{\beta,j} + \sum_{\beta\in\mathcal{B}'}\sum_{i=1}^{M} z_\beta\delta_{\beta,i} + \sum_{\beta\in\mathcal{B}'}\sum_{j=1}^{N} R_{\beta,j} \tag{7}$$

subject to

$$\sum_\beta \delta_{\beta,i}R_{\beta,j} \geq e_{i,j}, \quad \forall\, i,j \in E \tag{8}$$

$$\sum_j R_{\beta,j} \leq N z_\beta, \quad \forall\, \beta \in \mathcal{B}' \tag{9}$$

$$\sum_\beta z_\beta \leq K \tag{10}$$

where $z_\beta \in \{0,1\}, \forall\ \beta \in \mathcal{B}'$, $R_{\beta,j} \in \{0,1\}, \forall\ \beta \in \mathcal{B}',\ j = 1 \dots N$ and $e_{i,j} \in \{0,1\}, \forall\ i,j \in E$. Here again $E = \{(i,j)|Y_{ij} = 1\}$.

The first two terms in objective 7 capture false negatives and positives, respectively. The third and fourth terms are proxies for the complexity of the model and correspond to $|\hat{\mathbf{L}}|$ and $|\hat{\mathbf{R}}|$, these regularise the model to select fewer factors, and sparser $\hat{\mathbf{L}}$ and $\hat{\mathbf{R}}$ (derived from chosen factors). This is similar to the regularisation taken in PANDA+ (Lucchese, Orlando, and Perego 2014). The third and fourth terms could be dropped to give a BMF for a specific K.

Equation 8 uses the term $e_{i,j}$ to allow for boolean logic, restricted to non-zero elements of Y to reduce the number of variables in the model. Hence, Boolean logic is encoded only for true positives, so overlaps at false positives are penalised more than in exact BMF. This could easily be remedied by using $e_{i,j}$ for all elements in the matrix, at the expense of scalability.

RMP-2 scales (variables, constraints) with $(N(\rho M + |\mathcal{B}'|),\ \rho MN + |\mathcal{B}'|)$ where ρ is the density of the matrix. A column generation approach could be used for RMP-2, but it is likely to be intractable for larger problems, given its scaling and that of its associated pricing problem. It could also be solved similarly to RMP-w, using candidate factors. However, using candidate factors generated by a clustering approach would not capitalise on the added expressiveness of this formulation, given that such candidates are disjointly derived (though their composition may not be).

Instead, we opt to use RMP-2 as a second step after finding a solution to RMP-w, to refine the feature sets of factors (i.e. $\hat{\mathbf{R}}_d$) and chosen factors. Again, we use candidate factors by taking $\mathcal{B}'$ as the unique set of observation memberships across the selected factors of the solved RMP-w, with $\delta_\beta,\ \beta \in \mathcal{B}'$ given by the unique columns of $\hat{\mathbf{L}}_d$. Hence, RMP-2 selects known observation factor groups found with RMP-w, and reassigns features to them based on these groups, removing any redundant factors through the regularisation terms.

Given $|\mathcal{B}'| \leq K$ in our RMP-2 approach, we found it was still computationally tractable for larger matrices (order 100k x 20k, $\rho \approx 0.1$), but with heavy memory requirements and longer runtimes. Adapting the above formulation to be exact (given the factors) would require $e_{i,j}$ to cover all locations, intractable given an approximate 10-fold increase in the number of variables. Given these, we explored a less memory-intensive, heuristic approach.

Two-step Heuristic-BMF By grouping observations containing the same factor(s), we can recover additional features present in the grouping that should be added to that factor's feature set. RMP-2 does this globally based on the defined observation sets and using a basic complexity cost.

To allow features to be allocated to multiple factors, a coarse grid search is performed to identify at which global proportion of representation in an observation group a feature should be added to a factor. This optionally uses either the reconstruction error or MDL loss as defined in Hess, Morik, and Piatkowski (2017), see Extended Version Appendix A.5, which accounts for the sparsity, hence implicitly the rank, of the factorisation. The approach is formalised in Algorithm 1.

Following the reassignment of features, we greedily remove redundant factors, or factors that result in marginal improvement of the loss. To remove a factor with reconstruction error, the error without the factor must be some minimum percentage f of the reconstruction error with the

Algorithm 1: Reassign Features

function REASSIGN($Y, L, \hat{R}$, metric, Δt)
  $I_{kj} = \sum_i L_{ik} Y_{ij}$; $I \leftarrow \{I_{kj}\}_{K \times N}$
  $N_k = \sum_i L_{ik}$; $N \leftarrow \{N_k\}_K$
  $R_b \leftarrow \hat{R}$
  $E \leftarrow$ ERROR($Y, L, \hat{R}$, metric)
  for $t = 0$ **to** 1 **step** Δt **do**
    $R_t = (I > \frac{1}{2} N(1+t)) \vee \hat{R}$
    $E_t =$ ERROR(Y, L, R_t, metric)
    if $E_t < E$ **then**
      $R_b \leftarrow R_t$; $E \leftarrow E_t$
  return R_b

Algorithm 2: Iteratively remove factors

function PRUNE(Y, L, R, metric, $f = 1$)
  $M, K \leftarrow \dim(L)$
  $E \leftarrow$ ERROR(Y, L, R)
  while $K > 0$ **do**
    $E_{\text{ls}} = []$
    for $r = 1 \dots K$ **do**
      $L_c \leftarrow L, R_c \leftarrow R$
      $L_c[:, r] \leftarrow 0, R_c[r, :] \leftarrow 0$
      E_{ls}.append(ERROR(Y, L_c, R_c, metric))
    $b_c \leftarrow \arg\min E_{\text{ls}}$
    if not $E_{\text{ls}}[b_c] \leq fE$ **then**
      break // no better sln
    $L \leftarrow L$.delete(col $= b_c$)
    $R \leftarrow R$.delete(row $= b_c$)
    $K \leftarrow K - 1, E \leftarrow E_{ls}[b_c]$
  return L, R

factor. The MDL loss already accounts for removing a factor through reduced complexity (so $f = 1$ with MDL loss). This is formalised in Algorithm 2.

Finding Optimal Rank

Under the disjoint factorisation master problem, the model may achieve a lower overall objective when K is overspecified, see Figure A1, Extended Version Appendix A.2. Although the postprocessing should combine or remove redundant features, the initial rank specification will affect downstream reconstruction, reassignment and feature pruning. Hence, we perform the process over multiple initial ranks and select the best result (noting that the RMP-w can be initialised once, and updated with different K values, after which it is fast to solve). If increasing the rank does not result in a better solution for s_i iterations, the algorithm is stopped early. This is formalised in Algorithm 3.

We term `bfact` as the sequential pipeline of w-RMP followed by the second step (RMP-2 or heuristic) followed by algorithm 3. `bfact`-MIP is this pipeline where the second step is RMP-2 (as w-RMP and RMP-2 are both MIPs). `bfact`-recon is the pipeline where the second step is heuristic (algorithms 1 and 2) with reconstruction error, while `bfact`-MDL is the same using the MDL cost.

Algorithm 3: Select best rank over multiple K

function PIPELINE($Y, K_{\min}, K_{\max}, \Delta K$, metric, $f = 1, s_i = 2$)
  $A =$ GENCOLS(Y)
  $V_b =$ null, $E_b =$ null, $K_b =$ null
  $b_i = 0, i = 0$
  for $K_c = K_{\min}$ **to** $K_{\max}$ **step** ΔK **do**
    $L, R = \text{RMP}_1(Y, A, K_c)$
    if metric **is** mip **then**
      $L, R \leftarrow \text{RMP}_2(Y, L, K_c)$
    else
      $R \leftarrow$ REASSIGN(Y, L, R, metric)
      $L, R \leftarrow$ PRUNE(Y, L, R, metric, f)
    $E_k \leftarrow$ ERROR(Y, L, R, metric)
    if E_b **is** null **or** $E_k \leq f^{K_c - K_b} E_b$ **then**
      $b_i \leftarrow i, E_b \leftarrow E_k, V_b \leftarrow (L, R), K_b \leftarrow K_c$
    if $i - b_i > s_i$ **then**
      break // stop early
    $i += 1$
  return V_b

Experiments

Here, we compare `bfact` to existing approaches PRIMP, PANDA+ and MDL4BMF. For implementation details, please see Extended Version Appendix A.8.

Data

Simulation. We considered several simulation setups to benchmark our method. We generated a variety of scenarios with different matrix sizes, underlying ranks, noise levels and data density scenarios. Some rows and columns were also generated to be 'nuisance' variables, imitating realistic datasets where some features or observations are not relevant to the factorisation.

Formally, the main simulation set-up is as follows:

$$\begin{aligned}
\text{inputs:}\quad & M, N, k, q_l, q_r, v_i, v_j, p^+, p^- \\
\mathbf{L} \sim \{\text{Ber}(q_l)\}_{M \times k} \qquad & \mathbf{R} \sim \{\text{Ber}(q_r)\}_{k \times N} \\
n \sim \{\text{Ber}(v_i)\}_M \qquad & m \sim \{\text{Ber}(v_j)\}_N \\
\mathbf{L}[n > 0, .] = 0 \qquad & \mathbf{R}[., m > 0] = 0 \\
\mathbf{X} = \mathbf{LR} \qquad & \boldsymbol{\epsilon}^{\pm} \sim \{\text{Ber}(p^{\pm})\}_{M \times N} \\
\mathbf{Y} = \mathbf{X} + \boldsymbol{\epsilon}^+[\mathbf{X} = 0] - \boldsymbol{\epsilon}^-[\mathbf{X} > 0]
\end{aligned}$$

where $\text{Ber}(\alpha)$ represents the Bernoulli distribution with probability α. For each simulated experiment, we take 5 replicates.

Note, it is possible that for extremely sparse sampling, the true rank is lower than specified; however, we assume the effect of this is negligible and would likely be captured by measured algorithms.

Real-World Datasets. We also consider real-world data sets. This includes binarised versions of the Chess and

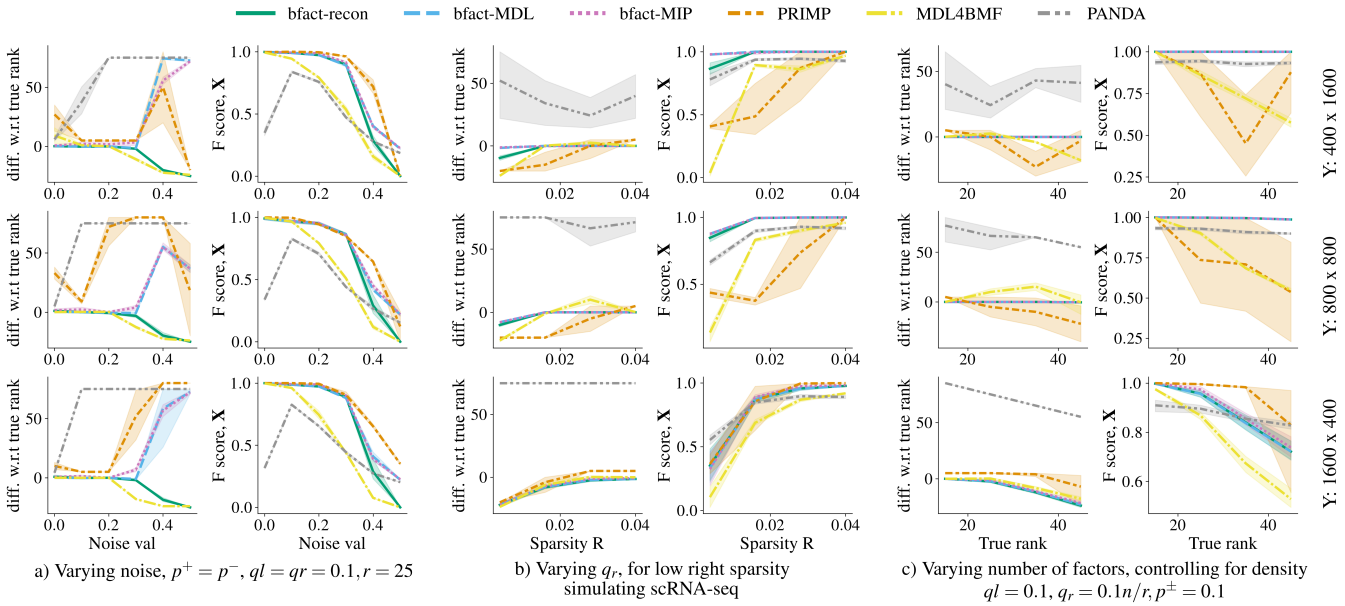

Figure 2: Simulation results.

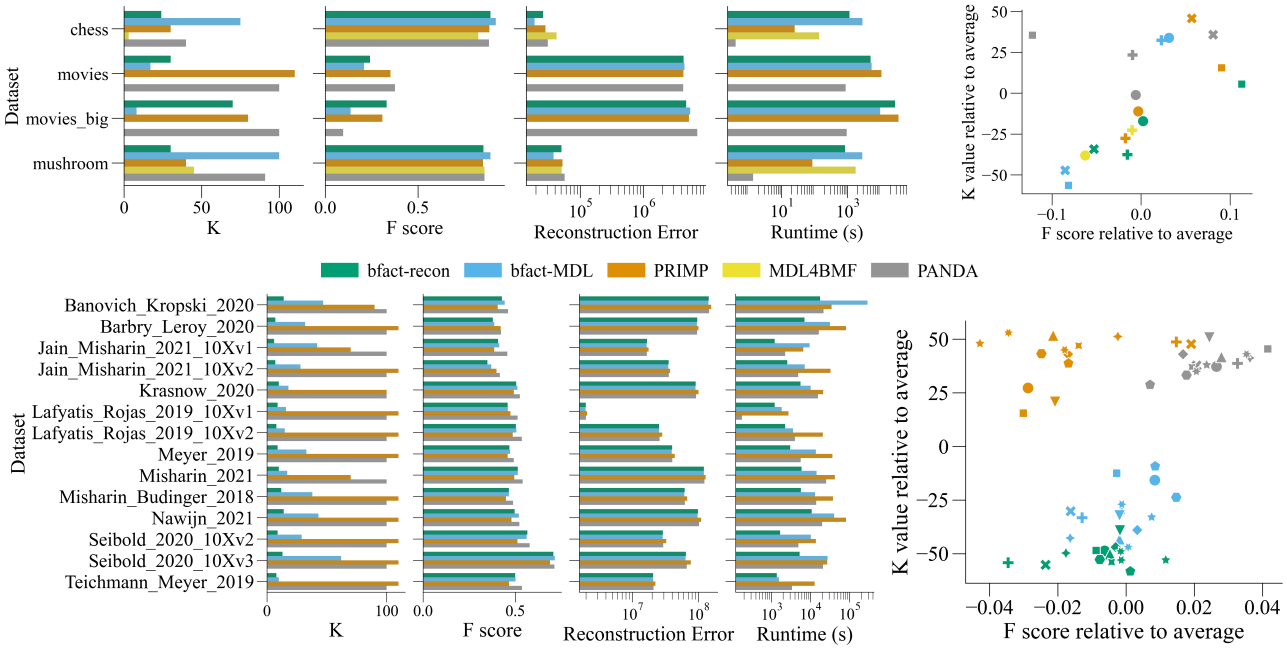

Figure 3: Real benchmarking results, standard benchmarks (top), and HLCA RNA-sequencing results (bottom). Shapes on right-hand plots correspond to different datasets.

Mushroom UCI datasets (Markelle Kelly, Rachel Longjohn, and Kolby Nottingham 2025) and two versions of the MovieLens 10M dataset, (Harper and Konstan 2015), where rows are users and columns are movies, with entries the star rating given by a user to a movie. Following Hess, Morik, and Piatkowski (2017), we set $Y_{ij} = 1$, if a user rates a movie with more than 3 stars. This constitutes the larger dataset, we then also take a smaller dataset filtered to select users who recommend more than 50 movies and movies that receive at least 5 recommendations.[1]

We also use single-cell RNA sequencing data from the

[1]While we follow precedent in previous publications, in some of the real data examples, the data has been binarised from discrete, categorical data using one-hot encoding. While such transformations enable the use of binary matrix factorisation, it is unclear whether they truly reflect the nature of the data.

Origin	Dataset	M	N	Density
UCI	Chess	3196	75	0.493
UCI	Mushroom	8124	119	0.193
MovieLens	Movies	29 980	9144	0.018
MovieLens	Movies Big	69 878	10 677	0.008
HLCA	Banovich Kropski 2020	121894	14495	0.101
HLCA	Barbry Leroy 2020	74484	15047	0.102
HLCA	Jain Misharin 2021 10Xv1	12422	13423	0.124
HLCA	Jain Misharin 2021 10Xv2	33135	13392	0.094
HLCA	Krasnow 2020	60982	15139	0.133
HLCA	Lafyatis Rojas 2019 10Xv1	2921	11943	0.073
HLCA	Lafyatis Rojas 2019 10Xv2	21258	13818	0.117
HLCA	Meyer 2019	35554	14153	0.103
HLCA	Misharin 2021	64842	15938	0.157
HLCA	Misharin Budinger 2018	41219	14057	0.136
HLCA	Nawijn 2021	70395	15579	0.119
HLCA	Seibold 2020 10Xv2	12127	15718	0.215
HLCA	Seibold 2020 10Xv3	21466	17825	0.310
HLCA	Teichmann Meyer 2019	12231	14855	0.150

Table 1: Dataset statistics

Human Lung Cell Atlas (HLCA), (Sikkema et al. 2023), which consists of 14 separate datasets on lung-derived cell types. We used the clean raw counts for each dataset and binarised them based on zero/non-zero values in the data. We also removed genes that were not expressed in at least 0.5% of cells, and cells that expressed fewer than 200 genes and more than 10,000. The size and density of each example are included in Table 1.

Evaluation Metrics

To evaluate each approach we compared the predicted rank to the true underlying rank, and the F1 score of the predicted signal matrix $\hat{\mathbf{L}}\hat{\mathbf{R}}$ with respect to the true signal matrix $\mathbf{X}$ in simulated settings, or the observed data matrix, $\mathbf{Y}$ in real data. Please see Extended Version Appendix A.7 for details of the F1 score. For real data, we also present the reconstruction error, given by $\sum_{ij} |\hat{X}_{ij} - Y_{ij}|$.

Methods with higher-rank predictions can overfit to noise, which could give a higher F1 score on $\mathbf{Y}$ but a low F1 score on the true signal matrix $\mathbf{X}$. Similarly, some methods may split factors into smaller ones, obscuring the true decomposition to give a high F1 score (even on $\mathbf{X}$) but also a higher predicted rank than necessary. Hence, the predicted rank should be considered in tandem with the reconstruction and $F1$ score. Other approaches compare MDL terms, which aim to balance model complexity and reconstruction error implicitly. However, as we show in the Extended Version Appendix A.6, MDL costs favour sparse decomposition, which do not necessarily align with a BMF, particularly for lower-rank, higher-complexity factorisations.

Results

Simulations. Figure 2 and Figures A5, A6 (Extended Version Appendix A.9) show that the three variations of `bfact` perform similarly in both F1 score and rank estimation across different simulated regimes. PANDA+ consistently overestimates rank and achieves lower F1 scores, but it should be noted that it requires less computational time than other methods (Figure A4). PRIMP does particularly poorly when the sparsity of $\mathbf{R}$ is low. MDL4BMF does well at estimating rank but has lower F1 scores. It is much slower than other methods to run despite being provided double the number of CPUs (Figure A4). All methods perform worse at higher density. On simulated data, `bfact` has a comparable run-time to PRIMP (Figure A4). Interestingly, `bfact`-MIP does slightly worse at recovering the true rank (Figures A5, A6). Likely, this is due to the regularisation approach taken, $|\hat{\mathbf{L}}| + |\hat{\mathbf{R}}|$, which encourages sparse, not necessarily low-rank reconstruction. This is supported by the fact that it has as high F-scores as the other `bfact` approaches. Given `bfact`-MIP is also slower and more memory-intensive, we do not explore `bfact`-MIP further. For `bfact`-recon and `bfact`-MDL, the time-limiting factor is the heuristic post-processing, rather than the combinatorial RMP-w problem.

Real-World Datasets. On real-world benchmark data, Figure 3 (top) shows that both `bfact` variants achieve comparable F-scores and reconstruction errors to other methods, and in particular, `bfact`-recon does this with fewer factors (K). A potential reason for this is that `bfact`-recon is the only method not to use a complexity-based score for rank approximation, and such scores do not always favour the lowest rank factorisation (see Extended Version Appendix A.6). Furthermore, some of these datasets have been one-hot encoded from categorical data; hence, it is unclear whether a BMF is the correct model for such data. On the 14 single-cell RNA sequencing datasets, for which there is more precedent for a BMF, Figure 3 (bottom) shows the `bfact` variants achieve both performant F-scores and reconstruction errors but use significantly fewer factors than other methods. In scRNA-seq, PANDA+ is likely overfitting to noise rather than signal, given its high predicted rank. To support this, consider simulated results in Figure A6, where PANDA+ achieves the highest F score on the observed data matrix $\mathbf{Y}$, despite having the lowest F score on the signal matrix $\mathbf{X}$, also with a very high rank. Despite some promising results on lower-dimensional standard datasets, MDL4BMF often took too long to run (more than 2 days with 24 CPUs), so it is omitted for the corresponding datasets.

Conclusion

We present a new binary matrix factorisation approach, including several sub-variants, called `bfact`, which uses a hybrid combinatorial optimisation approach based on *a priori* factors generated from existing clustering algorithms. We demonstrate that it performs well in simulations, particularly when applied to single-cell RNA sequencing data from the Human Lung Cell Atlas. We demonstrate that it is scalable to data of this size. We also include formulations and the pricing problem for finding the optimal disjoint BMF on smaller datasets. `bfact` is available as a pip installable package.

Acknowledgements

CY is supported by an UKRI Turing AI Acceleration Fellowship (Ref: EP/V023233/1) and EPSRC grant (Ref: EP/Y018192/1). EV is supported by the Oxford EPSRC Centre for Doctoral Training in Health Data Science (Ref: EP/S02428X/1).

References

Bolte, J.; Sabach, S.; and Teboulle, M. 2014. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1): 459–494.

Dantzig, G. B.; and Wolfe, P. 1960. Decomposition principle for linear programs. *Operations research*, 8(1): 101–111.

Harper, F. M.; and Konstan, J. A. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*, 5(4): 19:1–19:19.

Hess, S.; Morik, K.; and Piatkowski, N. 2017. The PRIMPING routine—Tiling through proximal alternating linearized minimization. *Data Mining and Knowledge Discovery*, 31(4): 1090–1131.

Kovacs, R. A.; Gunluk, O.; and Hauser, R. A. 2021. Binary Matrix Factorisation via Column Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5): 3823–3831. Number: 5.

Liang, L.; Zhu, K.; and Lu, S. 2020. BEM: Mining Coregulation Patterns in Transcriptomics via Boolean Matrix Factorization. *Bioinformatics*, 36(13): 4030–4037.

Lucchese, C.; Orlando, S.; and Perego, R. 2014. A Unifying Framework for Mining Approximate Top- k Binary Patterns. *IEEE Transactions on Knowledge and Data Engineering*, 26(12): 2900–2913.

Markelle Kelly; Rachel Longjohn; and Kolby Nottingham. 2025. The UCI Machine Learning Repository.

Miettinen, P.; Mielikäinen, T.; Gionis, A.; Das, G.; and Mannila, H. 2008. The Discrete Basis Problem. *IEEE Transactions on Knowledge and Data Engineering*, 20(10): 1348–1362. Conference Name: IEEE Transactions on Knowledge and Data Engineering.

Miettinen, P.; and Neumann, S. 2020. Recent Developments in Boolean Matrix Factorization. ArXiv:2012.03127 [cs].

Miettinen, P.; and Vreeken, J. 2014. MDL4BMF: Minimum Description Length for Boolean Matrix Factorization. *ACM Trans. Knowl. Discov. Data*, 8(4): 18:1–18:31.

Qi, R.; Ma, A.; Ma, Q.; and Zou, Q. 2020. Clustering and classification methods for single-cell RNA-sequencing data. *Briefings in Bioinformatics*, 21(4): 1196–1208.

Rukat, T.; Holmes, C. C.; Titsias, M. K.; and Yau, C. 2017. Bayesian Boolean Matrix Factorisation. In *Proceedings of the 34th International Conference on Machine Learning*, 2969–2978. PMLR. ISSN: 2640-3498.

Sikkema, L.; Ramírez-Suástegui, C.; Strobl, D. C.; Gillett, T. E.; Zappia, L.; Madissoon, E.; Markov, N. S.; Zaragosi, L.-E.; Ji, Y.; Ansari, M.; Arguel, M.-J.; Apperloo, L.; Banchero, M.; Bécavin, C.; Berg, M.; Chichelnitskiy, E.; Chung, M.-i.; Collin, A.; Gay, A. C. A.; Gote-Schniering, J.; Hooshiar Kashani, B.; Inecik, K.; Jain, M.; Kapellos, T. S.; Kole, T. M.; Leroy, S.; Mayr, C. H.; Oliver, A. J.; von Papen, M.; Peter, L.; Taylor, C. J.; Walzthoeni, T.; Xu, C.; Bui, L. T.; De Donno, C.; Dony, L.; Faiz, A.; Guo, M.; Gutierrez, A. J.; Heumos, L.; Huang, N.; Ibarra, I. L.; Jackson, N. D.; Kadur Lakshminarasimha Murthy, P.; Lotfollahi, M.; Tabib, T.; Talavera-López, C.; Travaglini, K. J.; Wilbrey-Clark, A.; Worlock, K. B.; Yoshida, M.; van den Berge, M.; Bossé, Y.; Desai, T. J.; Eickelberg, O.; Kaminski, N.; Krasnow, M. A.; Lafyatis, R.; Nikolic, M. Z.; Powell, J. E.; Rajagopal, J.; Rojas, M.; Rozenblatt-Rosen, O.; Seibold, M. A.; Sheppard, D.; Shepherd, D. P.; Sin, D. D.; Timens, W.; Tsankov, A. M.; Whitsett, J.; Xu, Y.; Banovich, N. E.; Barbry, P.; Duong, T. E.; Falk, C. S.; Meyer, K. B.; Kropski, J. A.; Pe'er, D.; Schiller, H. B.; Tata, P. R.; Schultze, J. L.; Teichmann, S. A.; Misharin, A. V.; Nawijn, M. C.; Luecken, M. D.; and Theis, F. J. 2023. An integrated cell atlas of the lung in health and disease. *Nature Medicine*, 29(6): 1563–1577. Publisher: Nature Publishing Group.

Tsuyuzaki, K.; Sato, H.; Sato, K.; and Nikaido, I. 2020. Benchmarking principal component analysis for large-scale single-cell RNA-sequencing. *Genome Biology*, 21(1): 9.

Vanderbeck, F.; and Savelsbergh, M. W. P. 2006. A generic view of Dantzig–Wolfe decomposition in mixed integer programming. *Operations Research Letters*, 34(3): 296–306.

Wang, C.-Y.; Gao, Y.-L.; Kong, X.-Z.; Liu, J.-X.; and Zheng, C.-H. 2022. Unsupervised Cluster Analysis and Gene Marker Extraction of scRNA-seq Data Based On Non-Negative Matrix Factorization. *IEEE Journal of Biomedical and Health Informatics*, 26(1): 458–467.

Wang, J.; Zou, Q.; and Lin, C. 2022. A comparison of deep learning-based pre-processing and clustering approaches for single-cell RNA sequencing data. *Briefings in Bioinformatics*, 23(1): bbab345.

Wu, Y.; and Zhang, K. 2020. Tools for the analysis of high-dimensional single-cell RNA sequencing data. *Nature Reviews Nephrology*, 16(7): 408–421. Publisher: Nature Publishing Group.