Boolean matrix factorisation for scRNA-seq

Ellen Visscher ELLEN.VISSCHER@BDI.OX.AC.UK
University of Oxford, United Kingdom

Michael Forbes M.FORBES@UQ.EDU.AU
University of Queensland, Australia

Christopher Yau CHRISTOPHER.YAU@WRH.OX.AC.UK
University of Oxford, United Kingdom

Abstract

We present `bfact`, a Python package for performing accurate low-rank Boolean or binary matrix factorisation (BMF). `bfact` uses a hybrid combinatorial optimisation/machine learning (ML) approach based on *a priori* candidate factors generated from clustering algorithms. It selects the best disjoint factors before optionally performing a second ML algorithm to recover the BMF. We verify `bfact` in simulated settings and show it achieves strong signal recovery with a much lower rank on single-cell RNA-sequencing datasets from the Human Lung Cell Atlas.

Keywords: Boolean matrix factorisation, combinatorial optimisation

Data and Code Availability

- Python package repository available here. Simulated data generation and code data are available here. Data from the Human Lung Cell Atlas is available here.

Institutional Review Board (IRB) IRB approval was not required for this work.

1. Introduction

Matrix factorisation methods decompose data into lower-dimensional forms that capture underlying patterns. For binary inputs, Boolean matrix factorisation (BMF) produces two binary factor matrices whose Boolean product approximates the original, preserving interpretability: one matrix encodes frequent feature combinations, the other links observations to them. BMF's binary factors enhance interpretability, with applications in market basket analysis, bioinformatics, and recommender systems.

We focus on BMF for biological data, particularly single-cell RNA sequencing (scRNAseq), where matrices are large (∼100k × 15k). Since scRNA-seq data are sparse and nearly binary, BMF is a natural fit, offering interpretable gene sets useful for downstream analysis Rukat et al. (2017); Liang, Zhu, and Lu (2020).

1.1. Theory

It is assumed some signal matrix $\mathbf{X} \in \{0,1\}^{M\times N}$ can be decomposed 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 rank $K \ll N$. 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 aim is to find the lowest rank (i.e where K is small) matrices $\hat{\mathbf{L}}$ and $\hat{\mathbf{R}}$ that best explain $\mathbf{Y}$ according to some objective. 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).

1.2. Related work and Contributions

Numerous heuristic and exact methods exist for BMF. ASSO, Miettinen et al. (2008), is a greedy itemset-mining algorithm that builds interpretable low-rank Boolean approximations. MDL4BMF, Miettinen and Vreeken (2014), extends this with a minimum description length (MDL) objective, automatically selecting the rank K. Panda+, Lucchese, Orlando, and Perego (2014), similarly uses a greedy search but

integrates complexity penalties directly into optimisation. MDL is a concept from information theory and encodes the complexity of a model as well as performance (i.e it quantifies the trade-off between increasing number of model parameters and increase in performance). PRIMP, Hess, Morik, and Piatkowski (2017), relaxes BMF into a continuous optimisation problem with a reconstruction error and regularisation term, solved via Proximal Alternating Linearised Minimisation (PALM), encouraging near-binary solutions and enabling MDL-based model selection. Finally, Kovacs, Gunluk, and Hauser (2021), uses a delayed column generation approach to solve the BMF problem by proposing rank-1 factors to underpin the data, leveraging the insight that a rank-K matrix factorisation can be decomposed as the sum of K rank-1 matrix factorisations. Here, the rank K must be fixed in advance, and the approach is limited to small and medium matrix sizes.

Most methods have not been evaluated in the context of scRNA-seq data, are either fully heuristic, use a continuous approximation, or if they are exact (i.e using combinatorial solving) are scale limited. Hence, we propose a hybrid approach that is partially combinatorial, partially heuristic (and optionally exact for certain modelling assumptions) and we evaluate models in the context of scRNA-seq.

2. Method

We have developed a novel BMF approach, called `bfact`, illustrated in Figure 1. First, candidate factors are generated through clustering on features. We then solve a warm-started restricted master problem (RMP-w) to approximate a disjoint binary matrix factorisation using up to K_c of these factors (K_c initialised to some $K_{\min}$). The method heuristically reassigns features and prunes factors based on minimising the description length (MDL) of the factorisation. The process iteratively increases the maximum number of factors, K_c, stopping if the MDL does not improve within s_i steps. Below, each of these steps is explained in detail.

We define a master problem (MP) that approximates a BMF by finding sets of (mostly) disjoint features that best explain the observations. Here, disjoint means that no two factors share the same feature. For our observed 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 $|\mathcal{A}| = 2^N - 1$. 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 first term of $C_{i,\alpha}$ is the intersection of features in an observation and a factor. While the right-hand side 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 $z_\alpha \in \{0,1\}\ \forall \alpha \in \mathcal{A}$. Equation 5 allows at most K profiles to be selected. Equation 6 ensures every feature is accounted for, either in at least one of the selected factors or by the variable u_j otherwise. If this constraint is an equality (i.e $\sum_{\alpha\in A} \delta_{\alpha,j} z_\alpha + u_j = 1$), then the above formulation gives the optimal disjoint factorisation, and the associated cost is the reconstruction error. The inequality relaxes this assumption, but note that the cost still implicitly favours disjointness, encouraging $z_{\alpha_1} + z_{\alpha_2} < 2$ if $\sum_{j=1}^N \delta_{\alpha_1,j}\delta_{\alpha_2,j} > 0$. Hence, the solution to the MP 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, this is not the same as a clustering approach on the features as u_j allows any number of features to be excluded from selected factors.

Solving the full MP is intractable due to the exponential number of factors. Instead, a restricted master problem (RMP) over a small subset of candidate factors ($\mathcal{A}'$, $|\mathcal{A}'| \ll |\mathcal{A}|$) can be solved efficiently. Delayed column generation iteratively adds factors that improve the objective of the linear relaxation of the MP. Branch-and-price can solve the integer MP to optimality without fully enumerating all factors. In practice, such problems are often warm-started with candidate factors before using the pricing problem of the linear relaxation of RMP. Hence, we warm-started

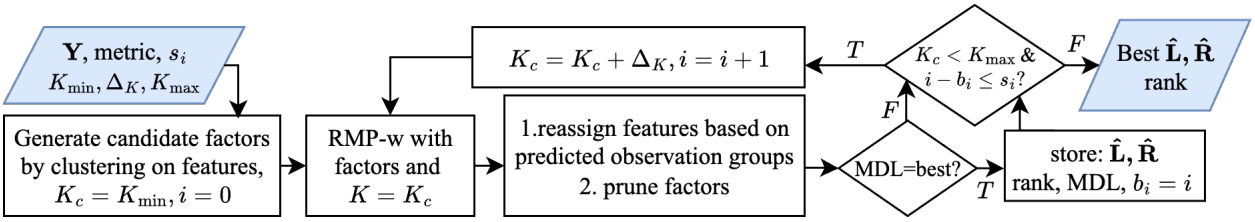


Figure 1: Overview of `bfact`

Algorithm 1: Reassign Features

Input: $Y, L, \hat{R}, \Delta t$
Output: R_b
$I_{kj} \leftarrow \sum_i L_{ik} Y_{ij}; \quad I \leftarrow \{I_{kj}\}_{K \times N}$
$N_k \leftarrow \sum_i L_{ik}; \quad N \leftarrow \{N_k\}_K$
$R_b \leftarrow \hat{R}$
$E \leftarrow \text{MDL}(Y, L, \hat{R})$
for $t \leftarrow 0$ **to** 1 **step** Δt **do**
  $R_t \leftarrow \left(I > \frac{1}{2} N(1+t)\right) \mid \hat{R}$
  $E_t \leftarrow \text{MDL}(Y, L, R_t)$
  if $E_t < E$ **then**
    $R_b \leftarrow R_t; \quad E \leftarrow E_t$
  end
end
return R_b

Algorithm 2: Prune factors

Input: Y, L, R
Output: L, R
$M, K \leftarrow \dim(L)$
$E \leftarrow \text{MDL}(Y, L, R)$
while $K > 0$ **do**
  $E_{\text{ls}} \leftarrow [\,]$
  for $r \leftarrow 1$ **to** K **do**
    $L_c \leftarrow L; \quad R_c \leftarrow R$
    $L_c[:, r] \leftarrow 0; \quad R_c[r, :] \leftarrow 0$
    $E_{\text{ls}}.\text{append}\big(\text{MDL}(Y, L_c, R_c)\big)$
  end
  $b_c \leftarrow \arg\min E_{\text{ls}}$
  if $E_{ls}[b_c] > E$ **then**
    break `// no better solution`
  end
  $L \leftarrow L.\text{delete}(\text{col} = b_c)$
  $R \leftarrow R.\text{delete}(\text{row} = b_c)$
  $K \leftarrow K - 1$
  $E \leftarrow E_{\text{ls}}[b_c]$
end
return L, R

the RMP with candidate factors (RMP-w) that were generated using hierarchical clustering, details Appendix A. Using this, the pricing problem was slow to offer any improvement to relaxed RMP-w, details Appendix B. Hence, RMP-w is used as the basis for our approach, `bfact`. This approach is far more computationally tractable than exact methods, including that in Kovacs, Gunluk, and Hauser (2021), scaling only with the size of the candidate set plus the number of features.

RMP-w does not directly produce a BMF, but can provide a good starting point. We use a two-step approach: first, RMP-w selects mostly disjoint feature sets to identify likely observation sets that share a given factor; second, these observation sets update the features assigned to this factor, allowing shared features and controlling model complexity for lower-rank approximations. 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, formalised in Algorithm 1. This uses the MDL loss defined in Hess, Morik, and Piatkowski (2017), (see Appendix C), which implicitly encodes the factorisation rank and sparsity.

Following the reassignment of features, redundant factors are greedily removed if the MDL cost is lower with the removal of a given factor. This is formalised in Algorithm 2.

Under the disjoint factorisation master problem, the model may achieve a lower overall objective when K is overspecified. 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 the RMP-w can be initialised once, and updated with different K values, after which it

Algorithm 3: Select best rank over multiple K

Input: $Y, K_{\min}, K_{\max}, \Delta K, s_i = 3$
Output: V_b
$A \leftarrow \text{GENCOLS}(Y)$
$V_b \leftarrow$ null; $E_b \leftarrow$ null
$b_i \leftarrow 0$; $i \leftarrow 0$
for $K_c \leftarrow K_{\min}$ **to** $K_{\max}$ **step** ΔK **do**
  $(L, R) \leftarrow \text{RMP}_w(Y, A, K_c)$
  $R \leftarrow \text{REASSIGN}(Y, L, R)$
  $(L, R) \leftarrow \text{PRUNE}(Y, L, R)$
  $E_k \leftarrow \text{MDL}(Y, L, R)$
  if $E_b =$ null **or** $E_k < E_b$ **then**
    $b_i \leftarrow i$; $E_b \leftarrow E_k$; $V_b \leftarrow (L, R, K_c)$
  end
  if $i - b_i > s_i$ **then**
    break `// stop early`
  end
  $i \leftarrow i + 1$
end
return V_b

is very 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. Appendix I investigates the MDL curve over all ranks in the context of scRNA-seq data, to show that a low stopping iteration is sufficient. `bfact` is hence the sequential pipeline of w-RMP followed by algorithms 1 to 3. Appendix J performs an ablation of the different components of `bfact` to demonstrate that the full pipeline is most performant.

3. Results

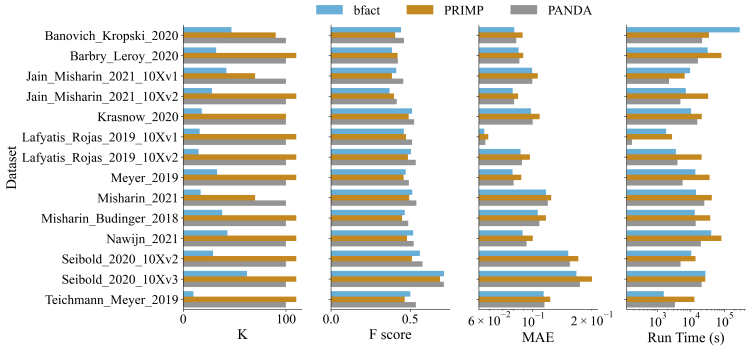
Here, `bfact` is compared to existing approaches PRIMP, PANDA+ and MDL4BMF. For implementation details, please see Appendix D. Each method is evaludated using a simulated framework, details and results in Appendix F. Here, the predicted rank is compared to the true underlying rank, and calculate the F_1 score of the predicted signal matrix $\hat{\mathbf{L}}\hat{\mathbf{R}}$ with respect to the known true signal matrix $\mathbf{X}$. Evaluating these in tandem is necessary as higher-rank predictions can overfit to noise.

Figure A2 and Figures A3, A4, show that `bfact` performs well at both F1 score and rank estimation across different simulated regimes. Panda consistently overestimates rank and achieves lower F1 scores but requires less computational time than other methods, Figure A5. PRIMP does particularly poorly when the sparsity of $\mathbf{R}$ is low. MDL4BMF does well at estimating rank but does worse in F1 scores. It is much slower than other methods to run despite being provided double the number of CPUs, Figure A5. All methods perform worse at higher density. On simulated data, `bfact` has a comparable run-time to PRIMP, Figure A5, with the algorithmic second step the time-limiting factor rather than the combinatorial RMP-w problem. Further, `bfact` and other methods struggle to identify true signal in high noise settings, as indicated in Figure A2

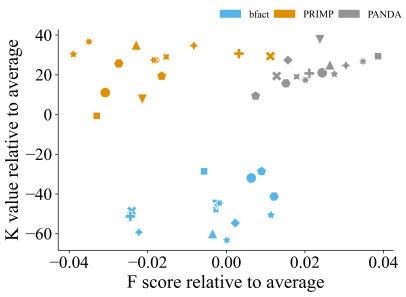
For real data evaluation, we use binarised single-cell RNA sequencing data from the Human Lung Cell Atlas (HLCA), Sikkema et al. (2023), which consists of 14 separate datasets on lung-derived cell types. Dataset size and density are detailed in Appendix G. For real data, the F_1 score is evaluated for $\mathbf{Y}$, and inspected in comparison to the predicted K value.

On the 14 single-cell RNA sequencing datasets, Figure 2 shows `bfact` achieves performant F-scores and MAE using significantly fewer factors and lower rank matrices. From an interpretability perspective, fewer factors are desirable for downstream analysis, such as identifying marker genes for biological processes. Despite performant F-scores, the high estimated rank for PANDA suggests it is overfitting to noise, given its poor performance in simulated settings. For example, in some simulated settings PANDA achieved the highest F score on observed data matrix $\mathbf{Y}$, despite having the lowest F score on signal matrix $\mathbf{X}$, Figure A4. Despite some promising results on lower-dimensional standard datasets, MDL4BMF took too long to run (>2 days with 24CPUs), so was excluded from the real-data evaluation. Overall, these results indicate `bfact` is a scalable and performant BMF approach in both simulated and real data scenarios.

To determine the biological relevance of identified factors for `bfact` compared to other BMF approaches, we perform an enrichment analysis, Table 1, of the identified gene sets (factors) in reference databases for known biologically relevant gene sets. The enrichment analysis uses enrichr, Chen et al. (2013), and reference databases of Cell Marker, CHip-Seq (CHea) and gene ontology (GO) biological process and cellular component sets. Results are also compared to traditional scRNA-seq approaches, including non-negative matrix factorisation, using the tool cNMF, Kotliar et al. (2019), and clustering results, according to the best practice guide in Luecken and Theis (2019). Appendix H further details the analysis procedure.



(*a*) Summary metrics across datasets; estimated rank (K), F score, mean absolute error and run time



(*b*) Mean shifted rank (K) and F score where shapes represent datasets.

Figure 2: Performance of different BMF methods on RNA-sequencing datasets from Human Lung Cell Atlas, please see Appendix E for descriptions of each metric. Average in (b) is taken across all method results, shifted values from (a) for easier comparison across datasets.

Reference Gene set	method	count	50%	75%	mean
CellMarker 2024	bfact MDL	252	14.43	36.24	31.44
	cNMF	269	26.83	45.62	33.37
	PANDA	202	3.98	9.09	12.74
	PRIMP	627	13.37	34.78	28.07
	clustering	215	22.59	45.71	29.22
ChEA 2022	bfact MDL	204	11.21	26.74	21.06
	cNMF	187	6.06	11.19	10.58
	PANDA	175	4.78	9.35	8.36
	PRIMP	677	8.29	19.03	16.5
	clustering	170	5.45	8.71	7.01
GO Biological Process 2025	bfact MDL	246	7.31	18.31	17.29
	cNMF	209	6.17	13.11	11.63
	PANDA	147	2.84	5.52	12.08
	PRIMP	650	5.32	10.86	11.96
	clustering	156	4.00	6.11	6.12
GO Cellular Component 2025	bfact MDL	222	10.92	29.83	19.57
	cNMF	197	8.34	18.94	12.95
	PANDA	132	3.28	6.98	9.73
	PRIMP	437	7.83	16.71	16.06
	clustering	116	4.99	12.56	11.77

Table 1: Enrichment results of method identified gene sets with reference database gene sets. Statistics taken across $-\log_{10} p$ for statistically significant adjusted p-values of unique factor genes. Higher values indicate higher enrichment.

Table 1 gives enrichment statistics for each approach across the reference gene sets, `bfact` (followed by PRIMP) has shifted higher enrichment in 3 out of 4 of these reference sets. In particular, the BMF approaches are shifted higher for the CHIP-seq datasets, suggesting that a binary approach in scRNA-seq may help to uncover gene sets modulated by the same transcription factor. `bfact` and PRIMP show slightly lower enrichment in the Cell Marker gene set, though their mean-to-median ratio suggests some gene sets have very high enrichment. This may suggest binary approaches identify fewer cell-type specific gene sets, however, as Cell Marker is computationally derived from scRNA-seq, there may be selection bias related to that pipeline and traditional appraoches (e.g conditioning on highly variable genes).

4. Conclusion

We introduce `bfact`, a Python package for precise low-rank Boolean and binary matrix factorisation (BMF). `bfact` uses a hybrid combinatorial-optimisation and machine-learning approach, starting from *a priori* candidate factors derived via clustering. It identifies the optimal disjoint factors and can optionally apply a second ML step to recover the BMF. We validate `bfact` on simulated data and demonstrate robust signal recovery at reduced rank on single-cell RNA-sequencing datasets from the Human Lung Cell Atlas. We show that `bfact` identified factors have promising biological relevance through higher enrichment in reference gene sets. `bfact` is available as a pip installable package.

References

Chen, E. Y.; Tan, C. M.; Kou, Y.; Duan, Q.; Wang, Z.; Meirelles, G. V.; Clark, N. R.; and Ma'ayan, A. 2013. Enrichr: interactive and collaborative HTML5 gene list enrichment analysis tool. *BMC Bioinformatics*, 14(1): 128.

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.

Kotliar, D.; Veres, A.; Nagy, M. A.; Tabrizi, S.; Hodis, E.; Melton, D. A.; and Sabeti, P. C. 2019. Identifying gene expression programs of cell-type identity and cellular activity with single-cell RNA-Seq. *eLife*, 8: e43803. Publisher: eLife Sciences Publications, Ltd.

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.

Luecken, M. D.; and Theis, F. J. 2019. Current best practices in single-cell RNA-seq analysis: a tutorial. *Molecular Systems Biology*, 15(6): e8746. Publisher: John Wiley & Sons, Ltd.

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.

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.

Appendix A. Candidate Factor Generation

To generate candidate factors for our RMP, we perform hierarchical clustering across features, based on their pairwise hamming distance, and cut the hierarchical tree at several different levels. Each resultant cluster from each level is taken as a candidate factor. We also perform Leiden community detection at different resolutions (a hyperparameter), which initially constructs a K-nearest-neighbour graph, again based on hamming distance. We take the union of clustered features as candidate factors to construct the RMP-w.

In the implemented code, candidate columns are generated by providing a target range of candidate

columns for RMP-w, with the idea that more candidate columns will likely be more informative. The code will automatically increase or decrease the number of hierarchical cuts to meet this target, although there is an upper limit on the number of candidate columns (i.e all cuts of the hierarchical tree). This is a hyperparameter, that if the model is underperforming, increasing this will likely improve results. Results are shown with an upper limit of 10000 columns for simulated settings and 20000 for real data settings (constant over each setting), indicating relative robustness to the number of columns provided.

Appendix B. Delayed Column Generation

The pricing problem (PP) for the restricted master problem is given by:

$$\min \sum_{i=1}^{M} t_i - \sum_{j=1}^{N} \pi_j^* x_j - \gamma^* \tag{7}$$

$$\text{s.t} \quad t_i = \min \left(\sum_{j|(i,j)\in E} x_j, \sum_{j|(i,j)\notin E} x_j \right) \tag{8}$$
$$x_j \in \{0,1\}, \qquad \forall j \in N$$
$$t_i \in \mathbb{R}^+, \qquad \forall i \in M$$

In practice, to model linearly, constraint 8 requires four constraints to implement, using a switch binary variable v_i for each M. Here π_j^* is the value of the dual variable of constraint 6 (main text) at the optimal solution of the restricted master problem and γ^* is the value of the dual of constraint 5 (main text).

Here, the number of (variables, constraints) of the PP scales with $(M+N,\ M)$. We found this to be slow. We demonstrate using the PP with the RMP, both warm-started or not, also comparing to RMP-w alone in Figure A1, where the delayed column generation process is capped at 30 minutes. Given the RMP-w alone takes approximately 10 minutes, this demonstrates the PP does not offer a significant/timely advantage to the RMP, when warm-started appropriately.

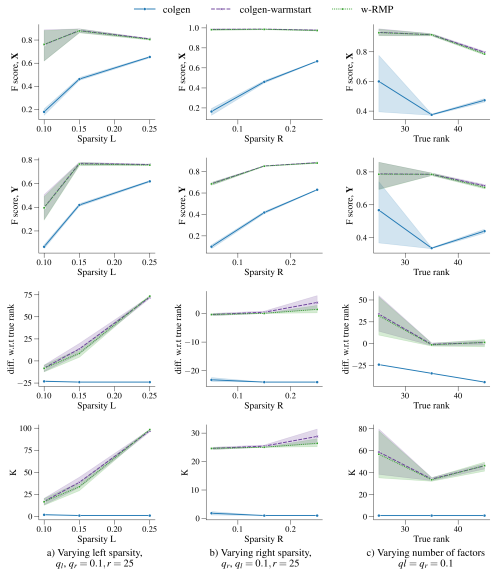


Figure A1: Comparison of delayed column generation used in tandem with bfact, simulations with $p^{\pm} = 0.1, M \times N = 1600 \times 400$. Each case is followed by the second algorithmic step of `bfact`, run only once for $K_{\min} = K_{\max} = 100$.

Appendix C. MDL cost

The cost measure in PRIMP, Hess, Morik, and Piatkowski (2017), is given by the code-table cost:

$$f_{CT}(\mathbf{L},\mathbf{R},\mathbf{Y}) = f^{D}_{CT}(\mathbf{L},\mathbf{R},\mathbf{Y}) + f^{\mathcal{M}}_{CT}(\mathbf{L},\mathbf{R},\mathbf{Y}) \tag{9}$$

$$f^{D}_{CT}(\mathbf{L},\mathbf{R},\mathbf{Y}) = -\sum_{k=1}^{K} |L_{\cdot k}| \log(p_k) - \sum_{j=1}^{N} |\epsilon_{\cdot j}| \log(p_{K+j})$$

$$f^{\mathcal{M}}_{CT}(\mathbf{L},\mathbf{R},\mathbf{Y}) = \sum_{k:|L_{\cdot k}|>0} (R_{k\cdot} c - \log(p_k)) + \sum_{j:|\epsilon_{\cdot j}|>0} (c_j - \log(p_{K+j})),$$

where ϵ is the difference between $\mathbf{Y}$ and the reconstructed matrix, and the probabilities p_k and p_{K+j} refer to the usage of non-singleton profiles $R_{k\cdot}$ and singleton profiles $\{j\}$ (i.e profiles containing only a single feature). These are given by:

$$p_k = \frac{|L_{\cdot k}|}{|\mathbf{L}| + |\epsilon|}, \quad p_{K+j} = \frac{\epsilon_{\cdot j}}{|\mathbf{L}| + |\epsilon|}$$

Further, $c : N \times 1$ is the vector of code lengths for each feature, given by, $c_j = -\log(|Y_j|/|\mathbf{Y}|)$.

Appendix D. Method Implementation Details

PRIMP We implemented PRIMP to run for 50000 steps, following what the authors did in Hess, Morik, and Piatkowski (2017), and used a $\Delta_k = 5$, from 5 to 100. PRIMP was implemented on simulations with access to 6 CPUs, and 1 NVIDIA GPU (of varying specifications, mostly Quadro RTX 8000 or P100 SXM2). For the real data, PRIMP was run for 50000 steps, used a $\Delta_k = 10$, from 10 to 100 (the method stops at $\max_K + \Delta_k$). Again it was run on machines with access to 6 CPUs and 1 NVIDIA GPU.

PANDA The PANDA documentation provides little guidance on what hyperparameters to use. We tested all and used the best combination, which was a frequency strategy, and a type 1 cost. It was run with a maximum K of 100 for both simulated and real scenarios.

MDL4BMF MDL4BMF takes longer than the other methods to run, we implemented simulations with 12 CPUs. We implement it with hyperparameters following the README example given in the code-with 10 threshold parameters and all error measures. For real data, we ran each dataset with access to 24CPUs, terminating if the model had not completed within 2 days.

bfact For matrices lower than a certain size, $M \times N < 5e6$, we transpose the matrix if $N < M$. We use a $\Delta_k = 10$, from 10 to 100.

Appendix E. Evaluation Metrics

The rank K is simply the number of factors output by each method. The F-score is the harmonic mean of the precision and recall of the 1s in the matrix $\hat{X} = \hat{L}\hat{R}$ compared with either the signal matrix X (when known, in simulated settings), or the observed data matrix Y. Given by:

$$p_X = \frac{\sum_{ij}(X \circ \hat{X})_{ij}}{\sum_{ij} \hat{X}_{ij}} \qquad p_Y = \frac{\sum_{ij}(Y \circ \hat{X})_{ij}}{\sum_{ij} \hat{X}_{ij}}$$

$$r_X = \frac{\sum_{ij}(X \circ \hat{X})_{ij}}{\sum_{ij} X_{ij}} \qquad r_Y = \frac{\sum_{ij}(Y \circ \hat{X})_{ij}}{\sum_{ij} Y_{ij}}$$

$$F_{1,X} = \frac{2 \times p_X \times r_X}{p_X + r_X} \qquad F_{1,Y} = \frac{2 \times p_Y \times r_Y}{p_Y + r_Y}$$

The reconstruction error is given by $R_X = \sum_{ij} |\hat{X} - X|$ or $R_Y = \sum_{ij} |\hat{X} - Y|$, and the MAE is simply this scaled by $M \times N$ (the number of entires).

Appendix F. Simulated benchmarks

Data generation 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:

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

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.

Results for various parameter settings are shown in Figures A2, A3 and A4, while time results across simulations are shown in Figure A5.

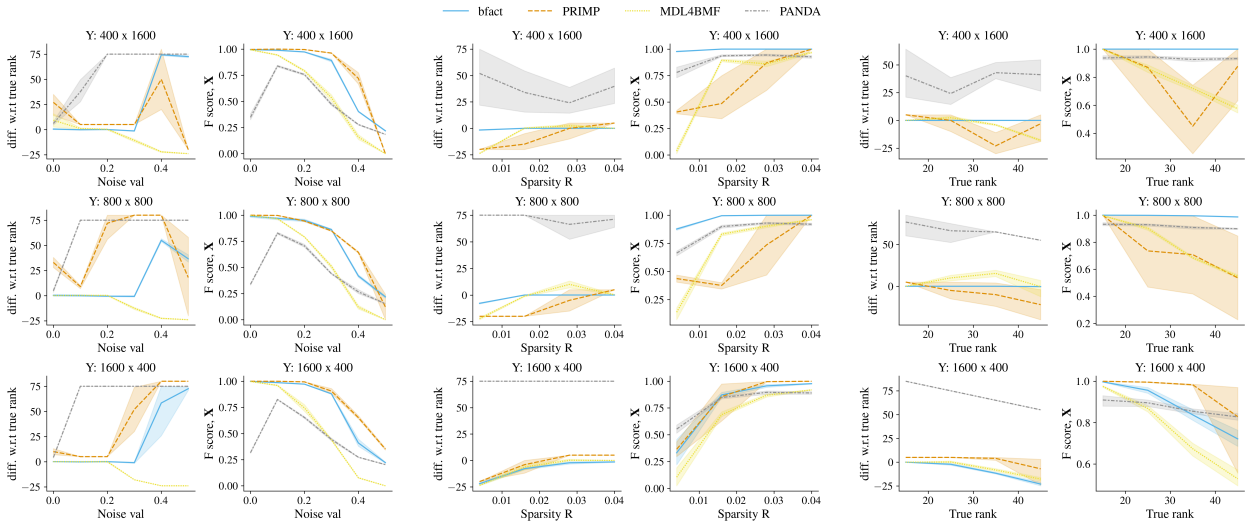


Figure A2: Simulation results, part A. Performance of different BMF methods under different simulated settings. Ideal factorisations ahve high F-score, and the same rank as the true signal matrix (i.e a difference of 0).

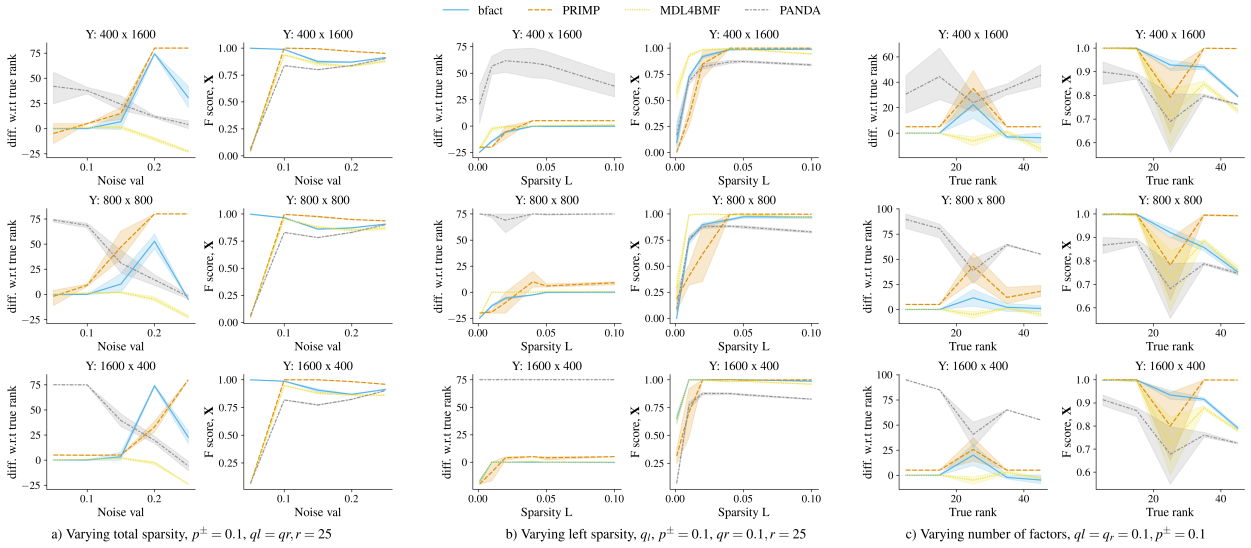


Figure A3: Simulation results, part B. Performance of different BMF methods under different simulated settings. Ideal factorisations have high F-score, and the same rank as the true signal matrix (i.e a difference of 0).

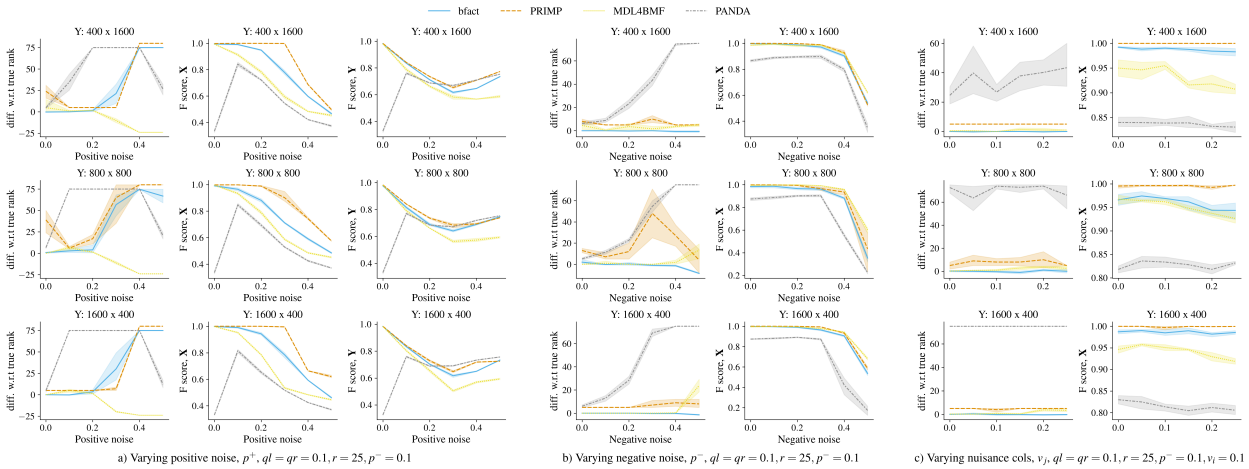


Figure A4: Simulation results, part C. Performance of different BMF methods under different simulated settings. For a) we also include the F-score on the data matrix, $\mathbf{Y}$ to show that a higher score here does not necessarily translate to a higher score for $\mathbf{X}$, due to overfitting to noise. Ideal factorisations have high F-score on X, and the same rank as the true signl matrix (i.e a difference of 0).

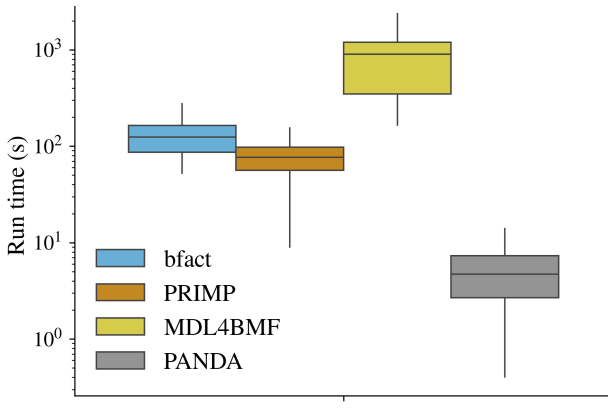


Figure A5: Run time across simulations

Origin	Dataset	M	N	Density
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 2: Dataset statistics

Appendix G. scRNA-seq data details

For each dataset, we used the clean raw UMI counts 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. Table 2 details the input data properties.

Appendix H. Enrichment Analysis Details

For cNMF, the factor gene sets were the most differentially expressed genes per factor, output by the cNMF

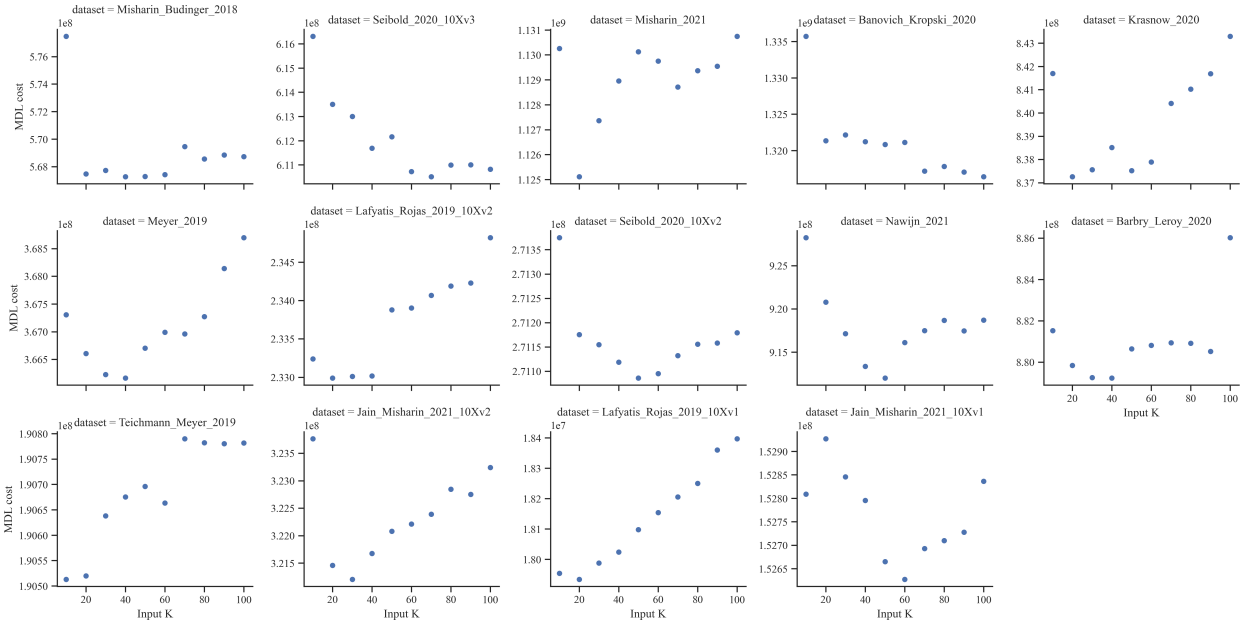


Figure A6: The MDL cost at different rank K input values on scRNA-seq datasets

tool, Kotliar et al. (2019). For cluster results, genes sets were the most uniquely differentially expressed per cluster (thresholded at least 20% expression within a cluster, max 20% expression in another cluster).

Results in Table 1 are taken across top-unique hits for each factor in each dataset. Results are Bonferroni corrected for the number of factors tested against each gene set. All tested terms (from databases) are thresholded on having an adjusted p-value lower than 0.05, ordered by p-value, and the top unique hit is taken for each factor. The large count for PRIMP reflects the fact that there are many more factors for this method; PANDA has many factors, but few of them have unique enrichment terms in the reference gene sets, hence the low count.

Appendix I. Stopping criteria evalutation

Figure A6 demonstrates that the MDL monotonically decreases/increases around the minimum MDL/rank combination with some local noise. This supports using a low stopping value criteria though greater than 1 to allow for local instability.

Appendix J. Ablation Study

Algorithmically, each stage can only ever improve on the MDL cost; hence, the final MDL cost (which is a trade-off between reconstruction error and F score), by definition, will be lower after the final stage. However, Table 3 also includes an ablation study on some of the stages, which shows that the multi-stage design consistently performs the best across the real datasets.

Ablation	K	F-score	MDL cost
RMP-w, max K = 50	50.0	0.448	5.66×10^8
RMP-w + algorithm 1 + 2, max K = 50	34.3	0.450	5.61×10^8
bfact (all stages)	31.0	0.454	5.60×10^8

Table 3: Ablation results, average results over all scRNA-seq datasets

Note the final stage for bfact iterates from K = 10 to K = 100, whereas the others are given a maximum K value and only ever solved for that. The results indicate increasing performance with each stage, hence justifying the multi-stage design.