Covariate-moderated Empirical Bayes Matrix Factorization

William R. P. Denault, Karl Tayeb, Peter Carbonetto & Matthew Stephens
Departments of Statistics and Human Genetics
University of Chicago
Chicago, IL 60637, USA
{wdenault,ktayeb,pcarbo,mstephens}@uchicago.edu

Jason Willwerscheid
Mathematics and Computer Science
Providence College
Providence, RI 02918, USA
jwillwer@providence.edu

Abstract

Matrix factorization is a fundamental method in statistics and machine learning for inferring and summarizing structure in multivariate data. Modern data sets often come with "side information" of various forms (images, text, graphs) that can be leveraged to improve estimation of the underlying structure. However, existing methods that leverage side information are limited in the types of data they can incorporate, and they assume specific parametric models. Here, we introduce a novel method for this problem, *covariate-moderated empirical Bayes matrix factorization* (cEBMF). cEBMF is a modular framework that accepts any type of side information that is processable by a probabilistic model or a neural network. The cEBMF framework can accommodate different assumptions and constraints on the factors through the use of different priors, and it adapts these priors to the data. We demonstrate the benefits of cEBMF in simulations and in analyses of spatial transcriptomics and collaborative filtering data. A PyTorch-based implementation of cEBMF with flexible priors is available at `https://github.com/william-denault/cebmf_torch`.

1 Introduction

Matrix factorization methods, which include principal component analysis (PCA), factor analysis, and non-negative matrix factorization (NMF) [1–3], are very widely used methods for inferring latent structure from data, performing exploratory data analyses, and visualizing large data sets (e.g., [4–6]). Matrix factorization methods are also instrumental in other statistical analyses such as adjusting for unobserved confounding [7]. When factorizing a matrix, say $\mathbf{Z}$, the matrix may be accompanied by additional row or column data—"side information"—that may be able to "guide" the matrix factorization algorithm toward a more accurate or interpretable factorization. A recent prominent example of this in genomics research is spatial transcriptomics data [8], which is expression profiled in many genes at many spatial locations ("pixels") [9]. For a variety of reasons, one typically seeks to factorize $\mathbf{Z}$, the matrix of gene expression profiles. But the 2-d coordinates of the pixels also provide important information about the biological context of the cells; for example, we might expect nearby pixels to belong to the same cell type or tissue region. Therefore, "spatially aware" matrix factorization methods have recently been proposed for spatial transcriptomics data [10–12].

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

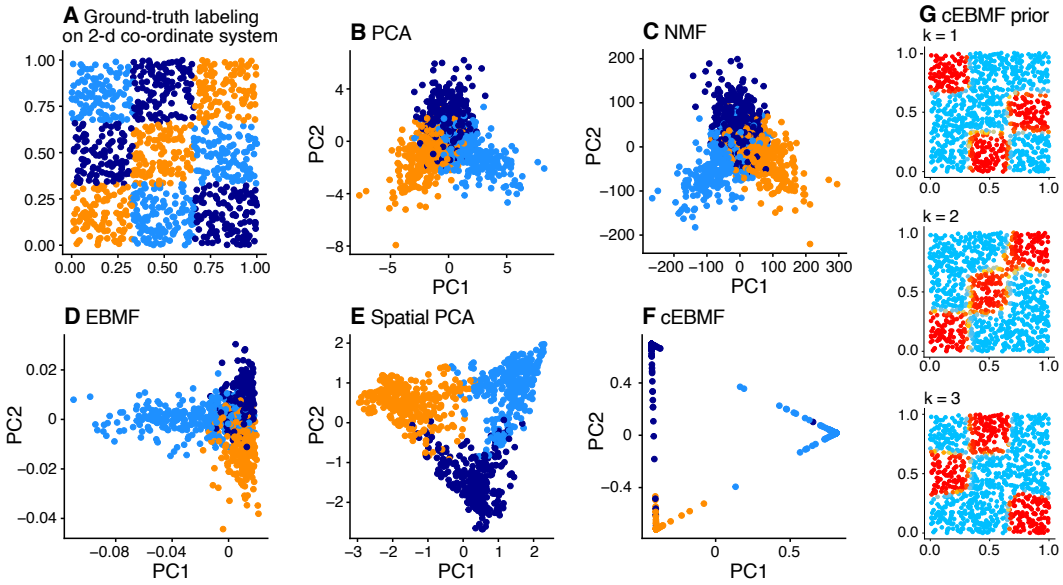

Figure 1: Toy simulation illustrating cEBMF for learning a matrix factorization, $\mathbf{Z} \approx \mathbf{L}\mathbf{F}^T$. In this example, $\mathbf{Z}$ is a 1,000 $\times$ 200 matrix. Each of the $n = 1{,}000$ data points is assigned to one of three clusters (**orange**, **light blue**, **dark blue**). Points near each other tend to be assigned to the same cluster, except near the boundaries (A). Without the side information (the 2-d coordinates in A), PCA, NMF and EBMF with $K = 3$ factors cluster some points accurately, but many other points are not clustered accurately (B–D). By contrast, Spatial PCA [11] and our new method, cEBMF, which both incorporate the side information into the prior, more accurately cluster the points (E, F). (For consistency of visualization, the top 2 PCs of the $\mathbf{L}$ matrices from NMF, EBMF and cEBMF are shown.) Spatial PCA assumes the data points are spatial, whereas cEBMF does make this assumption; instead, it has a flexible prior that is adapted to the data. This learned prior is shown in G: the color of the points shows the prior probability that row i, column k of $\mathbf{L}$ is nonzero (**blue** = low prior probability, **red** = high prior probability). See Sections 3 and 4 for definitions and additional details.

In this paper, we describe a novel matrix factorization framework that allows high-dimensional row and column data to guide the matrix factorizations without having to make specific assumptions about how these data inform the factorization. For example, although our framework can be applied to data that exhibit spatial properties, it does not assume or require that the data be spatial. Our framework is also flexible in that it includes many existing approaches as special cases, including unconstrained matrix factorization [13, 14], non-negative matrix factorization [15], semi-non-negative matrix factorization [16], and more recent methods that incorporate side information [17]. These features are achieved by taking an empirical Bayes approach, building on the recent empirical Bayes matrix factorization (EBMF) framework [13, 14]. In particular, we extend the EBMF approach of [13] with adaptive priors that are modified by the side information. We call this approach "covariate-moderated empirical Bayes matrix factorization," or "cEBMF" for short. See Fig. 1 for a toy example that illustrates the key features of cEBMF.

2 Related work

The literature on matrix factorization methods that incorporate side information is quite extensive. The different methods make different modeling assumptions, and are typically motivated by certain types of data. Although it is not possible to review all relevant literature here, we discuss a few of the most important or related methods.

Several variants of the topic model—which can be viewed as matrix factorizations with "sum-to-one" constraints on $\mathbf{L}$ and $\mathbf{F}$ [18]—incorporate side information in different ways; for example, the correlated topic model [19] and the structural topic model [20] incorporate document-level side information into the priors on $\mathbf{L}$. Collective matrix factorization (CMF) [21–23] has gained

considerable interest, but CMF is based on ideas that are quite different from cEBMF: like cEBMF, CMF assumes that the side information is the form of a matrix, but unlike cEBMF, CMF assumes that the side information factorizes in a similar way to $\mathbf{Z}$. Clearly, this assumption will not make sense for some types of data. Another prominent theme in matrix factorization with side information is incorporating group-level or categorical information, including ontological data. Among the methods in this area are CTPF [24] and the method of [25]. Another important class of methods related to cEBMF from the deep learning literature are variational autoencoders (VAE) [26], conditional variational autoencoders (cVAE) [27] and neural collaborative filtering (NCF) [28]. These methods generalize the concept of matrix factorization to nonlinear embeddings.

The method that is most closely related to cEBMF is MFAI [17] (see also [29] for related ideas). MFAI is in fact a special case of cEBMF in which the priors on $\mathbf{F}$ are normal and the prior means are informed by the covariates. Like cEBMF, MFAI allows these priors to be adapted separately for each dimension k. However, MFAI is not nearly as general as cEBMF; it implements only a single model, a single prior family with a specific parametric form, a specific procedure for fitting these priors (using gradient boosted tree methods [30]), and it only accommodates row-wise side information.

Several matrix factorization methods have been developed specifically for spatial transcriptomics data. Spatial PCA [11] models the spatial similarity among rows of $\mathbf{L}$ using Gaussian process prior. (Spatial PCA is similar to GP-LVM [31]. See also [32].) An NMF version of this approach generates "parts-based representations" guided by the spatial context of the data points [12]. More recently, IRIS [33] regularizes the matrix factors through a penalty function that encodes the spatial information in a graph (see also [34]).

3 Covariate-moderated empirical Bayes matrix factorization

3.1 Background: empirical Bayes matrix factorization

Empirical Bayes matrix factorization (EBMF) [13] is a flexible framework for matrix factorization: it approximates a matrix $\mathbf{Z} \in \mathbb{R}^{n\times p}$ as the product of two low-rank matrices,

$$\mathbf{Z} \approx \mathbf{L}\mathbf{F}^T, \tag{1}$$

where $\mathbf{L} \in \mathbb{R}^{n\times K}$, $\mathbf{F} \in \mathbb{R}^{p\times K}$, and $K \geq 1$. (In our applications, $K \ll n$, p.) EBMF assumes a normal model of the data,

$$\mathbf{Z} = \mathbf{L}\mathbf{F}^T + \mathbf{E}, \quad e_{ij} \sim N(0, \tau_{ij}^{-1}), \tag{2}$$

in which $N(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2, and the residual variances τ_{ij}^{-1} may vary by row (i) or by column (j) or both. (EBMF, and by extension cEBMF, also allows $\mathbf{Z}$ to contain missing values [13], which is important in many applications of matrix factorization, including collaborative filtering; see Sec. 4.2.) EBMF assumes prior distributions for elements of $\mathbf{L}$ and $\mathbf{F}$, which are themselves estimated among pre-specified prior families $\mathcal{G}_{\ell,k}$ and $\mathcal{G}_{f,k}$, respectively:

$$\begin{aligned} \ell_{ik} &\sim g_k^{(\ell)}, \quad g_k^{(\ell)} \in \mathcal{G}_{\ell,k}, \quad k = 1, \ldots, K \\ f_{jk} &\sim g_k^{(f)}, \quad g_k^{(f)} \in \mathcal{G}_{f,k}, \quad k = 1, \ldots, K. \end{aligned} \tag{3}$$

The flexibility of EBMF comes from the wide range of possible prior families (including non-parametric families) [35]. Different choices of prior family correspond to different existing matrix factorization methods. For example, if all families $\mathcal{G}_{\ell,k}$ and $\mathcal{G}_{f,k}$ are the family of zero-mean normal priors, then $\mathbf{L}\mathbf{F}^T$ is similar to a truncated singular value decomposition (SVD) [36, 37]. When the prior families are all point-normal (mixture of a point mass at zero and a zero-centered normal), one obtains empirical Bayes versions of sparse SVD or sparse factor analysis [38–40]. The prior families can also constrain $\mathbf{L}$ and $\mathbf{F}$; for example, families that only contain distributions with non-negative support result in empirical Bayes versions of NMF. In summary, EBMF (2–3) is a highly flexible modeling framework for matrix factorization that includes important previous methods as special cases, but also many new combinations (e.g., [41]).

3.2 The cEBMF modeling framework

In covariate-moderated EBMF (cEBMF), we assume that we have some "side information" (covariates) for rows and/or columns of $\mathbf{Z}$ [42, 43]. Let $\boldsymbol{x}_i$ denote the available information for the ith row

of $\mathbf{Z}$, and let $\boldsymbol{y}_j$ denote the available information for the jth column of $\mathbf{Z}$. In principle, $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ can be any information processable by a neural net (text, graph, image, other structured data), but for simplicity we assume that this information is stored as a matrix. Therefore, let $\mathbf{X} \in \mathbb{R}^{n \times n_x}$ be a matrix containing information on the rows of $\mathbf{Z}$, with $\boldsymbol{x}_i$ corresponding to the ith row of $\mathbf{X}$ (e.g., $\boldsymbol{x}_i$ might contain the 2-d coordinate of cell i). Similarly, let $\mathbf{Y} \in \mathbb{R}^{p \times n_y}$ contain information on the columns of $\mathbf{Z}$, with $\boldsymbol{y}_j$ corresponding to the jth row of $\mathbf{Y}$. In cEBMF, we incorporate this side information into the model through *parameterized priors*,

$$\begin{aligned} \ell_{ik} &\sim g_k^{(\ell)}(\boldsymbol{x}_i), \quad g_k^{(\ell)}(\boldsymbol{x}_i) \in \mathcal{G}_{\ell,k}, \quad k = 1, \ldots, K \\ f_{jk} &\sim g_k^{(f)}(\boldsymbol{y}_j), \quad g_k^{(f)}(\boldsymbol{y}_i) \in \mathcal{G}_{f,k}, \quad k = 1, \ldots, K, \end{aligned} \tag{4}$$

where $g_k^{(\ell)}(\boldsymbol{x}_i)$ is a probability distribution within the family $\mathcal{G}_{\ell,k}$, parameterized by $\boldsymbol{x}_i$, and $g_k^{(f)}(\boldsymbol{y}_j)$ is a probability distribution within the family $\mathcal{G}_{f,k}$ parameterized by $\boldsymbol{y}_i$.

A key limitation of many existing approaches is that they integrate the side information using restrictive parametric models that may or may not be appropriate for the particular application. Another limitation is that the priors chosen for these methods may make strong or perhaps unrealistic assumptions about the structure underlying the data; for example, Gaussian process priors, which have been used in matrix factorization (e.g., [11, 31, 44]), typically assume that the factors vary smoothly in space, which makes it difficult to capture sharp changes at boundaries [45]. Existing methods also typically rely on hyperparameters that need to be tuned or selected (e.g., using cross-validation).

To address these issues, we propose cEBMF, a method that:

1. Can leverage a large variety of models (e.g., multinomial regression, multilayer perceptron, graphical neural nets, convolutional neural nets) to integrate the side information into the prior.
2. Can use families of priors that are flexible in form and thus do not make strong assumptions.
3. Allows automatic selection of the hyperparameters in (4) via an empirical Bayes approach.

More formally, we fit a prior for each column k of $\mathbf{L}$, which maps each vector of covariates $\boldsymbol{x}_i$ to a given element $g_k^{(\ell)}(\boldsymbol{x}_i) \in \mathcal{G}_{\ell,k}$, and similarly for each column k of $\mathbf{F}$. In Sec. 3.3, we describe a simple yet general algorithm that simultaneously learns the factors $\mathbf{L}, \mathbf{F}$ and the priors $g_k^{(\ell)}(\boldsymbol{x}_i), g_k^{(f)}(\boldsymbol{y}_j)$. A PyTorch-based [46] implementation of cEBMF with several different parameterized prior families is available at `https://github.com/william-denault/cebmf_torch`.

3.2.1 An illustration: cEBMF with side information on factor sparsity

Here we illustrate the implementation of the cEBMF framework using a simple yet broadly applicable prior family. This prior family assumes that the covariates $\mathbf{X}, \mathbf{Y}$ *only inform the pattern of sparsity*—that is, the placement of zeros—in $\mathbf{L}$ and $\mathbf{F}$. This type of prior is of particular interest for matrix factorization because matrix factorizations are typically invariant to rescaling, and therefore priors that inform the magnitudes of ℓ_{ik} and f_{jk} are difficult to design. (By "invariant to rescaling," we mean that the likelihood or objective does not change if we replace $\mathbf{L}\mathbf{F}^T$ by $\tilde{\mathbf{L}}\tilde{\mathbf{F}}^T$, where $\tilde{\mathbf{L}} = \mathbf{L}\mathbf{D}, \tilde{\mathbf{F}} = \mathbf{F}\mathbf{D}^{-1}$, and $\mathbf{D}$ is an invertible diagonal matrix.) We define this prior family as

$$\mathcal{G}_{\text{ss}} := \{g : g(u) = (1 - \pi(\boldsymbol{x}, \boldsymbol{\theta}))\delta_0(u) + \pi(\boldsymbol{x}, \boldsymbol{\theta}) g_1(u; \boldsymbol{\omega})\}, \tag{5}$$

in which $\delta_0(u)$ denotes the point-mass at zero, $g_1(u; \boldsymbol{\omega})$ denotes the density of some probability distribution $g_1(\boldsymbol{\omega})$ on $u \in \mathbb{R}$, and $\boldsymbol{x} \in \mathbb{R}^m$ denotes the covariate. For example, when g_1 is the normal distribution and $\boldsymbol{\omega}$ specifies the mean and variance, (5) is a family of parameterized "spike-and-slab" priors [47], and cEBMF with $\mathcal{G}_{\ell,k} = \mathcal{G}_{\text{ss}}, \mathcal{G}_{f,k} = \mathcal{G}_{\text{ss}}$ implements a version of sparse factor analysis [38–40] in which the sparsity of the factors is informed by the covariates. (Note that the "ss" in $\mathcal{G}_{\text{ss}}$ is short for "spike-and-slab.") Alternatively, if g_1 is a distribution with support only on non-negative numbers, such as an exponential distribution, then cEBMF implements a version of sparse NMF. The free parameters are $\boldsymbol{\theta}$, which control the weight on the "spike", δ_0, and $\boldsymbol{\omega}$, which control the shape of the "slab", g_1. One simple parameterization of $\pi(\boldsymbol{x}, \boldsymbol{\theta})$ uses a logistic regression model,

$$\pi(\boldsymbol{x}, \boldsymbol{\theta}) = \phi\big(\theta_0 + \textstyle\sum_{t=1}^m x_t \theta_t\big), \tag{6}$$

where $\phi(x) := 1/(1 + e^{-x})$ denotes the sigmoid function, and $\boldsymbol{\theta} \in \mathbb{R}^{m+1}$. Most of the parameterized prior families used in this paper and in the cEBMF software are variants or elaborations on $\mathcal{G}_{\text{ss}}$.

3.3 The cEBMF learning algorithm

A key feature of the cEBMF modeling framework is that the algorithm for fitting the priors and estimating the factorization is simple to describe and often straightforward to implement. In brief, the cEBMF learning algorithm reduces a complex model-fitting task to a series of simpler subproblems. Each of these subproblems involves fitting a covariate-moderated variant of an empirical Bayes normal means (EBNM) model [35]. This also has the advantage of making the cEBMF framework and software modular so that a method that solves a covariate-moderated EBNM problem can be "plugged in" to the generic cEBMF algorithm.

3.3.1 Background: empirical Bayes normal means

Given observations $\hat{\beta}_i \in \mathbb{R}$ with known standard deviations $s_i > 0$, $i = 1, \ldots, n$, the normal means model [48–50] is

$$\hat{\beta}_i \overset{\text{ind.}}{\sim} N(\beta_i, s_i^2), \tag{7}$$

in which the "true" means $\beta_i \in \mathbb{R}$ are unknown. It is further assumed that the unknown means are

$$\beta_i \overset{\text{i.i.d.}}{\sim} g \in \mathcal{G}, \tag{8}$$

where $\mathcal{G}$ is some pre-specified family of probability distributions.

The empirical Bayes (EB) approach to fitting the normal means model (7–8) exploits the fact that the noisy observations $\hat{\beta}_i$ contain information not only about the underlying means β_i but also how the means are collectively distributed. EB "borrows information" across the observations to estimate g; typically this is done by maximizing the marginal log-likelihood of (7–8). The unknown means β_i are typically estimated by their posterior means (given the estimate of g).

To adapt the EBNM model (7–8) to cEBMF, we allow the prior for the ith unknown mean to depend on additional data $\boldsymbol{d}_i$ and parameters $\boldsymbol{\theta}$,

$$\beta_i \overset{\text{ind.}}{\sim} g(\boldsymbol{d}_i, \boldsymbol{\theta}) \in \mathcal{G}, \tag{9}$$

so that each combination of $\boldsymbol{\theta}$ and $\boldsymbol{d}_i$ maps to an element of $\mathcal{G}$. We refer to this as "covariate-moderated EBNM" (cEBNM). Solving the cEBNM problem therefore involves two key computations:

1. Estimate the model parameters. Compute

$$\hat{\boldsymbol{\theta}} := \underset{\boldsymbol{\theta} \in \mathbf{R}^m}{\operatorname{argmax}} \mathcal{L}(\boldsymbol{\theta}), \tag{10}$$

where $\mathcal{L}(\boldsymbol{\theta})$ denotes the marginal likelihood,

$$\mathcal{L}(\boldsymbol{\theta}) := p(\hat{\boldsymbol{\beta}} \mid \boldsymbol{s}, \boldsymbol{\theta}, \mathbf{D}) = \prod_{i=1}^{n} \int N(\hat{\beta}_i; \beta_i, s_i^2)\, g(\beta_i; \boldsymbol{d}_i, \boldsymbol{\theta})\, d\beta_i, \tag{11}$$

in which $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \ldots, \hat{\beta}_n)$, $\boldsymbol{s} = (s_1, \ldots, s_n)$, $\mathbf{D}$ is a matrix storing $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_n$, $N(\hat{\beta}_i; \beta_i, s_i^2)$ denotes the density of $N(\beta_i, s_i^2)$ at $\hat{\beta}_i$, and $g(\beta_i; \boldsymbol{d}_i, \boldsymbol{\theta})$ denotes the density of $g(\boldsymbol{d}_i, \boldsymbol{\theta})$ at β_i.

2. Compute posterior summaries. Compute summaries from the posterior distributions given the estimated prior,

$$p(\beta_i \mid \hat{\beta}_i, s_i, \hat{\boldsymbol{\theta}}, \mathbf{D}) = \frac{N(\hat{\beta}_i; \beta_i, s_i^2)\, g(\beta_i; \boldsymbol{d}_i, \hat{\boldsymbol{\theta}})}{\int N(\hat{\beta}_i; t, s_i^2)\, g(t; \boldsymbol{d}_i, \hat{\boldsymbol{\theta}})\, dt}. \tag{12}$$

For many classical prior families, such as the spike-and-slab family in Sec. 3.2.1, the integrals in (11) and (12) can be computed analytically. More generally, standard numerical techniques such as Gauss-Hermite quadrature may provide reasonably fast and accurate solutions for prior families that do not result in closed-form integrals since the integrals in (11) and (12) are one-dimensional. As a result, $\hat{\boldsymbol{\theta}}$ can often be obtained efficiently using off-the-shelf optimization algorithms even when the chosen priors do not result in analytical integrals.

In summary, solving the cEBNM problem consists of finding a mapping from the known quantities $(\hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D})$ to a tuple $(\hat{\boldsymbol{\theta}}, \hat{q})$, where each $(\boldsymbol{d}_i, \hat{\boldsymbol{\theta}})$ maps to an element $g(\boldsymbol{d}_i, \hat{\boldsymbol{\theta}}) \in \mathcal{G}$, and $\hat{q}$ is the posterior

distribution of the unknown means, $\hat{q}(\boldsymbol{\beta}) := \prod_{i=1}^{n} p(\beta_i \mid \hat{\beta}_i, s_i, \hat{\boldsymbol{\theta}}, \mathbf{D})$. To facilitate the description of the cEBMF algorithm below, we denote this mapping as

$$\text{cEBNM}(\hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}, \mathcal{G}) = (\hat{\boldsymbol{\theta}}, \hat{q}). \tag{13}$$

Note that in practice the full posterior $\hat{q}(\boldsymbol{\beta})$ is not needed; the first and second posterior moments are sufficient (see Sec. 3.3.2). Any prior family is admissible under the cEBMF framework so long as the mapping (13) is computable (either numerically or analytically).

3.3.2 Algorithm

Given a method for solving the cEBNM problem (Sec. 3.3.1), the cEBMF model can be fitted using a simple coordinate ascent algorithm. In brief, the cEBMF algorithm maximizes an objective function—the evidence lower bound (ELBO) [51] under a variational approximation with conditional independence assumptions on $\mathbf{L}$ and $\mathbf{F}$ (see the Appendix)—by iterating over the following updates for each factor $k = 1, \ldots, K$ until some stopping criterion is met:

1. Disregard the kth factor in $\bar{\mathbf{R}}$, the $n \times p$ matrix expected residuals, $\bar{\mathbf{R}}^k = \bar{\mathbf{R}} + \bar{\boldsymbol{\ell}}_k \bar{\boldsymbol{f}}_k^T$.
2. For each $i = 1, \ldots, n$, compute the least-squares estimates of ℓ_{ik}, denoted $\hat{\ell}_{ik}$, and the standard deviations s_{ik}^{ℓ} of these estimates,

$$\hat{\ell}_{ik} = (s_{ik}^{\ell})^2 \textstyle\sum_{j=1}^{p} \tau_{ij} \bar{r}_{ij}^k \bar{f}_{jk} \tag{14}$$

$$s_{ik}^{\ell} = (\textstyle\sum_{j=1}^{p} \tau_{ij} \bar{f}_{jk}^2)^{-1/2}, \tag{15}$$

   where $\bar{f}_{jk}$ and $\bar{f}_{jk}^2$ denote, respectively, the first and second posterior moments of f_{jk}.
3. Update $g_k^{(\ell)} \in \mathcal{G}_{\ell,k}$ by solving (10), in which we make the following substitutions in (10): $\hat{\beta}_i \leftarrow \hat{\ell}_{ik}, s_i \leftarrow s_{ik}^{\ell}, i = 1, \ldots, n, \mathbf{D} \leftarrow \mathbf{X}, \mathcal{G} \leftarrow \mathcal{G}_{\ell,k}$.
4. Making the same substitutions in (12), update the posterior means $\bar{\boldsymbol{\ell}}_k = (\bar{\ell}_{1k}, \ldots, \bar{\ell}_{nk})^T$ and posterior second moments $\bar{\boldsymbol{\ell}}_k^2 = (\bar{\ell}_{1k}^2, \ldots, \bar{\ell}_{nk}^2)^T$.
5. Perform updates similar to those in Steps 2–4 to update $\bar{\boldsymbol{f}}_k$, $\bar{\boldsymbol{f}}_k^2$ and $g_k^{(f)} \in \mathcal{G}_{f,k}$.
6. Update the matrix of expected residuals, $\bar{\mathbf{R}} = \bar{\mathbf{R}}^k - \bar{\boldsymbol{\ell}}_k \bar{\boldsymbol{f}}_k^T$.

These steps are iterated until some stopping criterion is met. The algorithm must be initialized with initial estimates of $\bar{\mathbf{L}}, \bar{\mathbf{F}}$. The expected residuals are then initialized as $\bar{\mathbf{R}} = \mathbf{Z} - \bar{\mathbf{L}}\bar{\mathbf{F}}^T$. Note that to simplify presentation we have omitted some details such as how to update the residual variances τ_{ij}^{-1}. These and other details are provided in the Appendix.

4 Experiments

4.1 Simulations

To assess the benefits of cEBMF, we compared cEBMF with other matrix factorization methods in simulated data sets. We compared with several methods that do not use side information, including EBMF (flashier R package [13, 35]), penalized matrix decomposition ("PMD"; PMA R package [39]), and a variational autoencoder (VAE) [26] implemented in PyTorch [46]. We also compared with other methods that use side information, including MFAI (mfair R package [17]), Spatial PCA [11], conditional VAE (cVAE) [27], and neural collaborative filtering (NCF) [28]. cVAE and NCF were also implemented in PyTorch. Note that Spatial PCA accepts only a specific type of side information, the 2-d coordinates of the data points, so was not included in all the simulations.

We compared the methods in four simulation scenarios designed to capture a range of settings where one might perform a matrix factorization analysis, with or without side information: (1) a "sparsity-driven covariate" setting in which the covariates only informed the sparsity pattern of $\mathbf{L}$ and $\mathbf{F}$; (2) an "uninformative covariate" setting in which the covariates provided no information about the true matrix factorization; (3) a "tiled-clustering" setting in which $\mathbf{L}$ depended on the 2-d location of the data points; and (4) a "shifted tiled-clustering" setting in which the cEBMF priors were unable recover the true data generating process. The latter scenario was used to assess cEBMF under model

misspecification. We simulated 100 data sets in each setting, and we assessed the ability of each method to recover the true matrix factorization as measured by root mean squared error (RMSE) between the true matrix factorization $\mathbf{LF}^T$ and estimated matrix factorization $\hat{\mathbf{L}}\hat{\mathbf{F}}^T$. More detailed descriptions of the simulations and the methods compared are given in the Appendix.

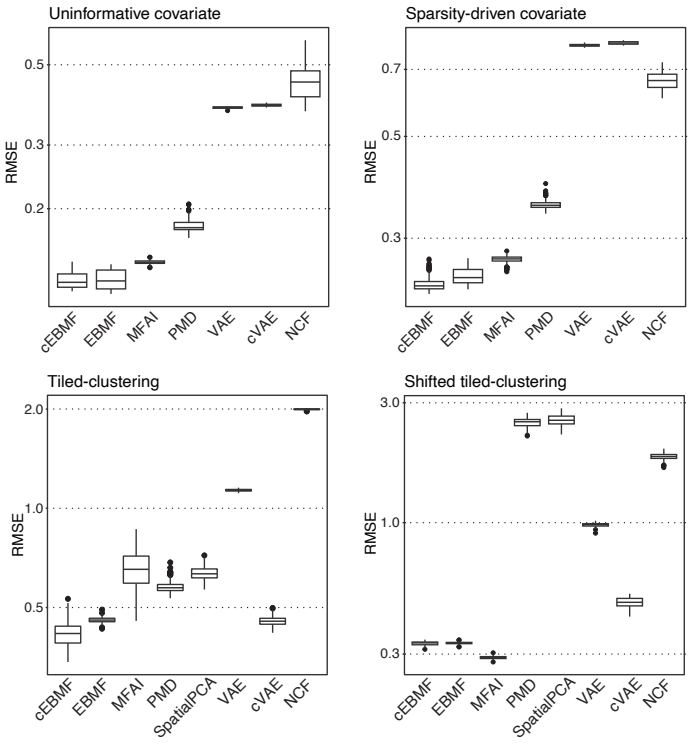

Figure 2: Performance of the different matrix factorization methods in the simulated data sets. Each boxplot summarizes the root mean squared errors (RMSEs) across 100 simulations in that scenario. (Lower RMSEs are better.) See Figures 5–8 in the Appendix for additional results from the simulations. Note Fig. 1 shows results from one of the tiled-clustering simulations in detail.

The results are summarized in Fig. 2. cEBMF was generally more accurate than the other methods, particularly when the covariates were informative; cEBMF achieved the greatest gains over EBMF in the tiled-clustering setting where the covariates were also the most informative. Reassuringly, cEBMF did not perform worse than EBMF in settings with an uninformative covariate or a prior that was misspecified ("shifted tiled-clustering"). The deep learning approaches (VAE, cVAE, NCF) generally performed worse than the other methods. cVAE sometimes outperformed VAE when the side information was highly informative, such as in the tiled-clustering scenario, but did not provide improvements over VAE in the more challenging sparsity-driven scenario. We also ran Spatial PCA on the tiled-clustering and shifted tiled-clustering data sets where the factors were partly driven by the 2-d locations of the data points. Despite the fact that Spatial PCA can exploit the side information, it had worse accuracy than EBMF which did not use the side information. This may be because Spatial PCA makes assumptions (e.g., orthogonal factors) that were not met by our simulations. MFAI generally performed worse than EBMF and cEBMF except in the shifted tiled-clustering setting; MFAI is a much less flexible model than cEBMF and therefore its performance was sensitive to the appropriateness of its modeling assumptions. (All models were misspecified in the shifted tiled-clustering setting, but perhaps MFAI was the least misspecified.) Additional results including comparisons with other methods (PCA/SVD, Sparse SVD [38], CMF [21]) are in the Appendix.

4.2 Collaborative filtering

To provide a quantitative assessment of the matrix factorization methods in real data, we ran the same methods on the MovieLens 100K data [52], a standard collaborative filtering benchmark in which the goal is to predict the unobserved elements of the matrix. Here, $\mathbf{Z}$ was a $1{,}682 \times 943$ matrix containing integer-valued movie ratings, with rows corresponding to movies and columns corresponding to users. Since most (93%) of the movie ratings were missing, this example highlights the ability of these methods to handle missing data (unlike most NMF methods). The side information $\mathbf{X}$ was a $1{,}682 \times 19$ binary matrix containing information about the movie's genre (comedy, adventure, etc). We held out some of the moving ratings at random, and used these held-out ratings as a test set.

We ran EBMF and cEBMF so as to produce non-negative matrix factorizations, which is common in collaborative filtering (e.g., [21]). Therefore, in the results we labeled these methods as "EBNMF" and "cEBNMF". (Note that MFAI cannot produce non-negative matrix factorizations.) To enforce non-

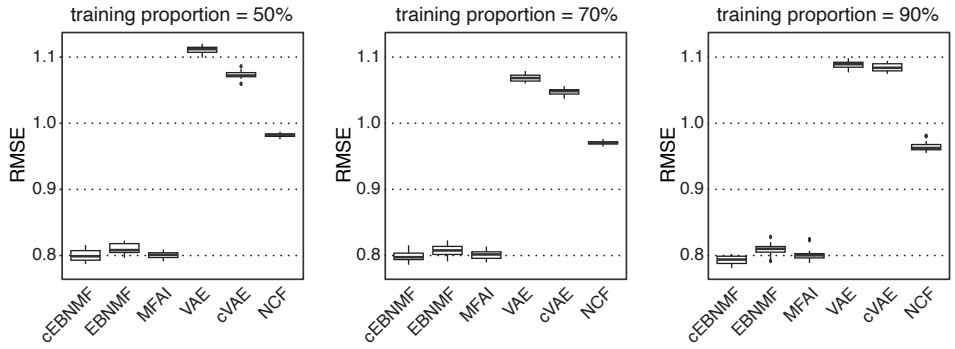

Figure 3: Prediction performance of different matrix factorization and deep learning methods in the MovieLens 100K data [52]. Training proportion = $X\%$ means that $X\%$ of the movie ratings were used in training, and the remaining $(100 - X)\%$ were used to evaluate accuracy (measured using RMSE). The results at each training proportion are from 10 random training-test splits.

negativity in $\mathbf{L}$ and $\mathbf{F}$, we used mixture-of-exponentials priors. For cEBNMF, the side information was incorporated into the priors on $\mathbf{L}$ using a multi-layer perceptron. (See the Appendix for further details.) Since we didn't know the true number of factors, K was chosen adaptively in EBMF, cEBMF and MFAI. (We set an upper limit of 7 for EBMF and cEBMF, and 12 for MFAI.)

The results are summarized in Fig. 3. Both MFAI and cEBNMF were able to use the side information (the movie genres) to improve over EBNMF, and all three matrix factorization methods were more accurate than the deep learning methods. We conjecture that the deep learning methods would have performed better with more data (such as the more recently released MovieLens data sets that are much larger). On the MovieLens 100K data, cEBNMF yielded overall the best prediction accuracy across the different training-set splits.

4.3 Spatial transcriptomics

Although cEBMF was not specifically designed for spatial data, here we show that cEBMF also yields compelling results from spatial transcriptomics data [8] by exploiting the side information, the spatial locations of the data points. We illustrate this using a data set [53, 54] that has been annotated by domain experts and has been used in several papers to benchmark methods for spatial transcriptomics (e.g., [11, 55–57]). The data were collected from 12 slices of the human dorsolateral prefrontal cortex (DLPFC). After data preprocessing, each slice contained about 4,000 pixels and expression measured in about 5,000 genes ($n \approx 4000$, $p \approx 5{,}000$).[1]

In this application, our aim was to generate a "parts-based" representation of the data, with the hopes that the "parts" would resolve to biologically interpretable units (e.g., cell types, tissue regions, gene programs) [12, 58, 59]. This is a fundamentally different aim from the previous examples: in the previous examples, the goal was to generate accurate low-dimensional representations, but we did not ask whether the *individual dimensions* were accurate or interpretable. With this aim in mind, we ran cEBMF so as to produce non-negative matrix factorizations ("cEBNMF") using the same priors that were used for the MovieLens data. We compared to two other non-negative matrix factorizations that did not leverage the side information: NMF implemented in the R package NNLM [15], and EBMF with point-exponential priors ("EBNMF"). (The point-exponential prior is a simplification of the mixture-of-exponentials prior with a single exponential component in which the rate parameter is also learned.). We also compared to several of the methods that were considered in the previous experiments, including methods such as Spatial PCA and cVAE that make use of the side information, and others that do not.

Spatial PCA deserves special mention because it was specifically designed for spatial transcriptomics data [11]. Although Spatial PCA does not produce a parts-based decomposition, the Spatial PCA software automatically clusters the data points after projection onto the principal components (PCs), and this clustering can be compared to the non-negative matrix factorizations. Following [11], we

[1]We used the data prepared by the authors of the SpatialLIBD package [53] which were made available for download at `https://research.libd.org/spatialLIBD/`.

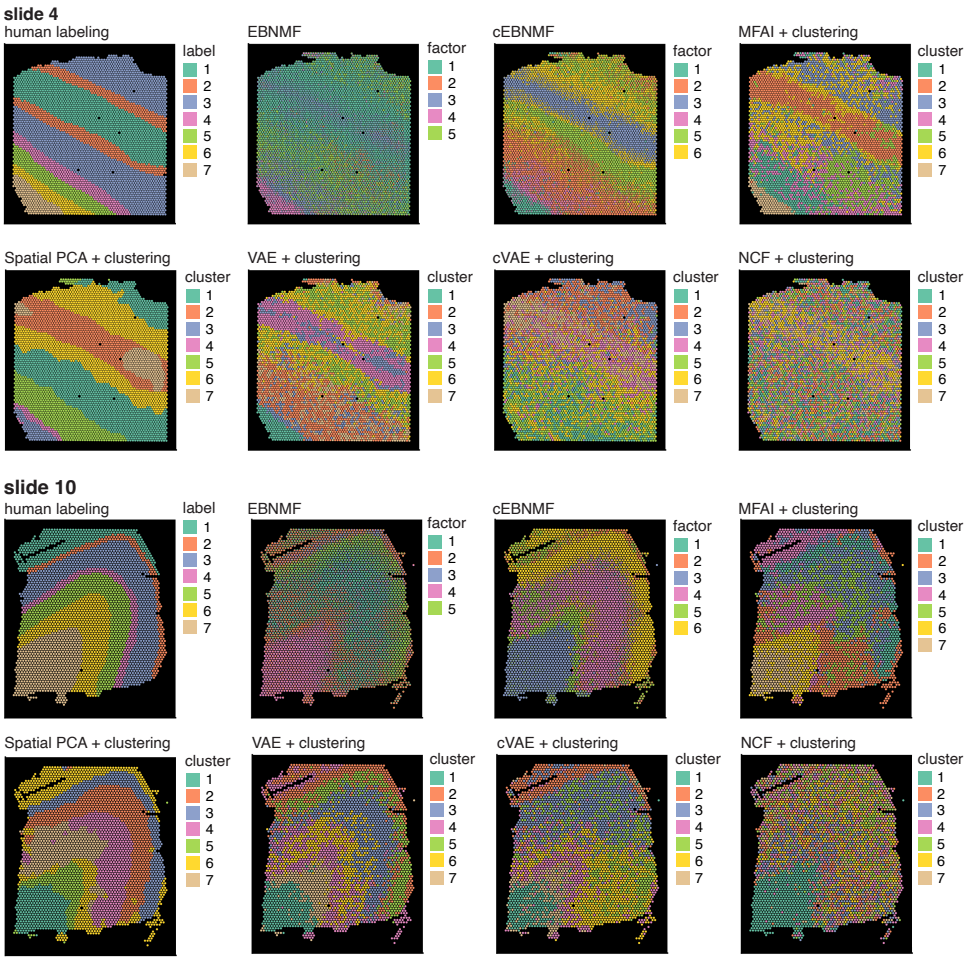

Figure 4: Results on slides 4 (top row) and 10 (bottom row) of the DLPFC spatial transcriptomics data [53, 54]. For the NMF, EBMF and cEBNMF results, each data point ("pixel") i is shown as a pie chart using the relative values of ith row of $\mathbf{L}$ (after performing an "LDA-style" post-processing of $\mathbf{L}, \mathbf{F}$ [12]). (Higher-resolution versions of these images are available online at `https://github.com/william-denault/cEBMF_experiment`.) Since MFAI and other methods did not produce a non-negative matrix factorization, we clustered the low-dimensional embeddings using the same approach that was used in Spatial PCA [11]. CMF results for these two slices and additional results on all 12 slices are given in the Appendix.

computed the top 20 PCs, then we ran the walk-trap clustering algorithm [60] on the PCs. Additional details of the Spatial PCA analysis and the other methods are given in the Appendix.

Figure 4 shows results on two of the slices, with additional results on all 12 slices provided in the Appendix. The manual annotations on the left-hand side should be viewed as a useful reference point, but not necessarily the "ground truth". (Consider that the data-driven annotations might identify previously unknown cellular structures.) EBNMF, cEBNMF and MFAI adapted the number of factors K to the data (with upper limits of 50, 20 and 9, respectively). For the other methods, the number of clusters was set to match the manual annotation. Qualitatively, some of the factors from NMF and EBMF seem to correspond to the expert-labeled regions, but several other factors appear to be capturing other substructures that have no obvious spatial quality. Comparatively, the cEBNMF results in slices 4 and 9 capture the expert labeling much more closely, with most factors showing a clear spatial quality. The clusters from Spatial PCA, MFAI and VAE also capture spatial structure and expert labeling well, but with some notable exceptions, e.g., Spatial PCA cluster 7 in slices 4 and 9. (The Spatial PCA software performed additional post-processing on the clusters which is why these clusters look less "noisy" than the others.) cVAE, despite using the side information, did not seem to

		number of rows (n)			
method	software	10^3	10^4	10^5	10^6
EBMF	flashier [13, 35]	0.8	2.5	36.9	165.1
cEBMF	`cebmf_torch`*	5.2	35.5	416.3	3,403.6
MFAI	mfair [17]	45.4	251.3	11,293.2	–
Spatial PCA	SpatialPCA [11]	234.8	8,213.7	–	–

Table 1: Running times of matrix factorization methods on data sets in which we varied n, the number of rows in $\mathbf{X}$ and $\mathbf{Z}$. The numbers in the table are the average running times (in seconds) from 10 simulated data sets. *Available at `https://github.com/william-denault/cebmf_torch`.

improve over VAE. The CMF results were comparatively poor (Fig. 9 in the Appendix), reflecting the inappropriateness of the CMF assumptions in this setting. Note that the NMF methods can capture continuous variation in expression within and across cell types or regions—as well as the expectation that some pixels might reflect combinations of cellular structures—whereas the clustering cannot.

4.4 Scalability benchmark

cEBMF can also handle much larger data sets than the MovieLens and DLPFC data sets considered above. (One reason we did not use larger data sets was to allow for comparison with methods that do not scale well to large data sets.) To illustrate this, we ran EBMF and cEBMF on "tiled-clustering" data sets (using the same priors described Sec. 4.1), in which the data sets were simulated with different numbers of rows, n. We compared the running times with two other matrix factorization methods that make use of the side information, MFAI and Spatial PCA (Table 1). While cEBMF had considerably higher running times than EBMF on the same data, cEBMF completed in much less time on average than MFAI and Spatial PCA. Further, while cEBMF was able to handle data sets with 1 million rows, Spatial PCA struggled to analyze data sets with 100,000 or more rows due to its high memory usage; for example, Spatial PCA needed approximately 300 GB of memory for $n = 100{,}000$. MFAI crashed frequently in data sets with $n \geq 100{,}000$ rows (only 2 of 10 of the runs completed at $n = 100{,}000$). Note this benchmark was performed on a computer with 32 GB memory, an NVIDIA GeForce RTX™ 4070 GPU and an AMD Ryzen™ 9 7940HS CPU (8 cores, 16 threads). The EBMF and cEBMF algorithms were run for at most 20 iterations. See also the Appendix where we describe some of the computational properties of the cEBMF algorithmic framework.

5 Conclusions

We have introduced cEBMF, a general and flexible framework for matrix factorization that (i) incorporates side information through flexible covariate-dependent priors and (ii) learns these priors from the data using empirical Bayes ideas. Considerable effort has gone into optimizing the software implementation building on our previous work on this topic [13, 35]. As a result, cEBMF scales well to large data sets with, say, hundreds of thousands or millions of rows and/or columns. Our experiments highlight the importance of using matrix factorization models that make appropriate assumptions about the data or are sufficiently flexible to adapt to the data. In our experiments, cEBMF performed competitively against other matrix factorization methods and deep learning approaches that make use of the side information. Because the priors in cEBMF can take the form of virtually any probabilistic model optimized via equations (10–11), our framework opens the door to incorporating other types of side information, including images and graphs.

Note: R and Python code implementing the experiments is available at `https://github.com/william-denault/cEBMF_experiment`, and a PyTorch-based implementation of cEBMF is available at at `https://github.com/william-denault/cebmf_torch`.

Acknowledgments

We thank the staff at the Research Computing Center at the University of Chicago for providing the high-performance computing resources used to implement the numerical experiments. We also thank

Deaglan Bartlett and Augustin Marignier for their help in developing the PyTorch implementation. This work was supported in part by NIH grant R01HG002585 (to M.S.) and Eric and Wendy Schmidt AI in Science Postdoctoral Fellowship, a Schmidt Sciences, LLC program (to W.R.P.D.). Additional support came from the University of Chicago Data Science Institute through the 2024 AI+Science Research Initiative. We also thank the reviewers for their feedback.

References

[1] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[2] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, volume 13, pages 556–562, 2001.

[3] N. Gillis. *Nonnegative matrix factorization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2021.

[4] T. Sainburg, M. Thielk, and T. Q. Gentner. Finding, visualizing, and quantifying latent structure across diverse animal vocal repertoires. *PLoS Computational Biology*, 16(10):e1008228, 2020.

[5] T. A. Alexander, R. A. Irizarry, and H. C. Bravo. Capturing discrete latent structures: choose LDs over PCs. *Biostatistics*, 24(1):1–16, 2023.

[6] J. Novembre and M. Stephens. Interpreting principal component analyses of spatial population genetic variation. *Nature Genetics*, 40(5):646–649, 2008.

[7] J. T. Leek and J. D. Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3(9):e161, 2007.

[8] V. Marx. Method of the year: spatially resolved transcriptomics. *Nature Methods*, 18:9–14, 2021.

[9] K. Vandereyken, A. Sifrim, B. Thienpont, and T. Voet. Methods and applications for single-cell and spatial multi-omics. *Nature Reviews Genetics*, 24(8):494–515, 2023.

[10] B. Velten, J. M. Braunger, R. Argelaguet, D. Arnol, J. Wirbel, D. Bredikhin, G. Zeller, and O. Stegle. Identifying temporal and spatial patterns of variation from multimodal data using MEFISTO. *Nature Methods*, 19:179–186, 2022.

[11] L. Shang and X. Zhou. Spatially aware dimension reduction for spatial transcriptomics. *Nature Communications*, 13:7203, 2022.

[12] F. W. Townes and B. E. Engelhardt. Nonnegative spatial factorization applied to spatial genomics. *Nature Methods*, 20:229–238, 2023.

[13] W. Wang and M. Stephens. Empirical Bayes matrix factorization. *Journal of Machine Learning Research*, 22(120):1–40, 2021.

[14] X. Zhong, C. Su, and Z. Fan. Empirical Bayes PCA in high dimensions. *Journal of the Royal Statistical Society, Series B*, 84(3):853–878, 2022.

[15] X. Lin and P. C. Boutros. Optimization and expansion of non-negative matrix factorization. *BMC Bioinformatics*, 21:7, 2020.

[16] C. H. Ding, T. Li, and M. I. Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):45–55, 2010.

[17] Z. Wang, F. Zhang, C. Zheng, X. Hu, M. Cai, and C. Yang. MFAI: a scalable Bayesian matrix factorization approach to leveraging auxiliary information. *Journal of Computational and Graphical Statistics*, 33(4):1339–1349, 2024.

[18] P. Carbonetto, A. Sarkar, Z. Wang, and M. Stephens. Non-negative matrix factorization algorithms generally improve topic model fits. *arXiv*, 2105.13440, 2025.

[19] J. Lafferty and D. Blei. Correlated topic models. In *Advances in Neural Information Processing Systems*, volume 18, pages 147–154, 2005.

[20] M. E. Roberts, B. M. Stewart, and E. M. Airoldi. A model of text for experimentation in the social sciences. *Journal of the American Statistical Association*, 111(515):988–1003, 2016.

[21] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 650–658, 2008.

[22] H. Lee and S. Choi. Group nonnegative matrix factorization for EEG classification. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 320–327, 2009.

[23] G. Bouchard, D. Yin, and S. Guo. Convex collective matrix factorization. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, pages 144–152, 2013.

[24] P. K. Gopalan, L. Charlin, and D. Blei. Content-based recommendations with Poisson factorization. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

[25] C. Hu, P. Rai, and L. Carin. Non-negative matrix factorization for discrete data with hierarchical side-information. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1124–1132, 2016.

[26] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations*, 2014.

[27] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

[28] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

[29] I. Porteous, A. Asuncion, and M. Welling. Bayesian matrix factorization with side information and Dirichlet process mixtures. *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 563–568, 2010.

[30] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

[31] N. D. Lawrence and A. J. Moore. Hierarchical Gaussian process latent variable models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 481–488, 2007.

[32] T. Zhou, H. Shan, A. Banerjee, and G. Sapiro. Kernelized probabilistic matrix factorization: exploiting graphs and side information. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 403–414, 2012.

[33] Y. Ma and X. Zhou. Accurate and efficient integrative reference-informed spatial domain detection for spatial transcriptomics. *Nature Methods*, 21:1231–1244, 2024.

[34] D. Cai, X. He, J. Han, and T. S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8): 1548–1560, 2011.

[35] J. Willwerscheid, P. Carbonetto, and M. Stephens. ebnm: an R package for solving the empirical Bayes normal means problem using a variety of prior families. *Journal of Statistical Software*, 114(3):1–32, 2025.

[36] S. Nakajima and M. Sugiyama. Theoretical analysis of Bayesian matrix factorization. *Journal of Machine Learning Research*, 12(79):2583–2648, 2011.

[37] S. Nakajima, M. Sugiyama, S. D. Babacan, and R. Tomioka. Global analytic solution of fully-observed variational Bayesian matrix factorization. *Journal of Machine Learning Research*, 14: 1–37, 2013.

[38] D. Yang, Z. Ma, and A. Buja. A sparse singular value decomposition method for high-dimensional data. *Journal of Computational and Graphical Statistics*, 23(4):923–942, 2014.

[39] D. M. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534, 2009.

[40] B. E. Engelhardt and M. Stephens. Analysis of population structure: a unifying framework and novel methods based on sparse factor analysis. *PLoS Genetics*, 6(9):e1001117, 2010.

[41] Y. Liu, P. Carbonetto, J. Willwerscheid, S. A. Oakes, K. F. Macleod, and M. Stephens. Dissecting tumor transcriptional heterogeneity from single-cell RNA-seq data by generalized binary covariance decomposition. *Nature Genetics*, 57:263–273, 2025.

[42] I. Virshup, S. Rybakov, F. J. Theis, P. Angerer, and F. A. Wolf. anndata: access and store annotated data matrices. *Journal of Open Source Software*, 9(101):4371, 2024.

[43] R. P. Adams, G. E. Dahl, and I. Murray. Incorporating side information in probabilistic matrix factorization with Gaussian processes. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 2010.

[44] N. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6(60):1783–1816, 2005.

[45] H.-M. Kim, B. K. Mallick, and C. C. Holmes. Analyzing nonstationary spatial data using piecewise Gaussian processes. *Journal of the American Statistical Association*, 100(470): 653–668, 2005.

[46] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: an imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, 2019.

[47] T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

[48] H. Robbins. Asymptotically subminimax solutions of compound statistical decision problems. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1951, vol. II*, pages 131–149. University of California Press, Berkeley and Los Angeles, CA, 1951.

[49] B. Efron and C. Morris. Limiting the risk of Bayes and empirical Bayes estimators—Part II: the empirical Bayes case. *Journal of the American Statistical Association*, 67(337):130–139, 1972.

[50] M. Stephens. False discovery rates: a new deal. *Biostatistics*, 18(2):275–294, 2017.

[51] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: a review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[52] F. M. Harper and J. A. Konstan. The MovieLens datasets: history and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19, 2015.

[53] B. Pardo, A. Spangler, L. M. Weber, S. C. Page, S. C. Hicks, A. E. Jaffe, K. Martinowich, K. R. Maynard, and L. Collado-Torres. spatialLIBD: an R/Bioconductor package to visualize spatially-resolved transcriptomics data. *BMC Genomics*, 23:434, 2022.

[54] K. R. Maynard, L. Collado-Torres, L. M. Weber, C. Uytingco, B. K. Barry, S. R. Williams, J. L. Catallini, M. N. Tran, Z. Besich, M. Tippani, J. Chew, Y. Yin, J. E. Kleinman, T. M. Hyde, N. Rao, S. C. Hicks, K. Martinowich, and A. E. Jaffe. Transcriptome-scale spatial gene expression in the human dorsolateral prefrontal cortex. *Nature Neuroscience*, 24:425–436, 2021.

[55] M. Varrone, D. Tavernari, A. Santamaria-Martínez, L. A. Walsh, and G. Ciriello. CellCharter reveals spatial cell niches associated with tissue remodeling and cell plasticity. *Nature Genetics*, 56:74–84, 2024.

[56] E. Zhao, M. R. Stone, X. Ren, J. Guenthoer, K. S. Smythe, T. Pulliam, S. R. Williams, C. R. Uytingco, S. E. B. Taylor, P. Nghiem, J. H. Bielas, and R. Gottardo. Spatial transcriptomics at subspot resolution with BayesSpace. *Nature Biotechnology*, 39:1375–1384, 2021.

[57] J. Zhu, L. Shang, and X. Zhou. SRTsim: spatial pattern preserving simulations for spatially resolved transcriptomics. *Genome Biology*, 24:39, 2023.

[58] P. Carbonetto, K. Luo, A. Sarkar, A. Hung, K. Tayeb, S. Pott, and M. Stephens. GoM DE: interpreting structure in sequence count data with differential expression analysis allowing for grades of membership. *Genome Biology*, 24:236, 2023.

[59] K. K. Dey, C. J. Hsiao, and M. Stephens. Visualizing the structure of RNA-seq expression data using grade of membership models. *PLoS Genetics*, 13(3):e1006599, 2017.

[60] P. Pons and M. Latapy. Computing communities in large networks using random walks. In *Proceedings of the 20th International Symposium on Computer and Information Sciences*, pages 284–293, 2005.

[61] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[62] L. K. Saul and M. I. Jordan. Exploiting tractable substructures in intractable netw orks. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 486–492. MIT Press, 1996.

[63] Z. Ghahramani and G. E. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12(4):831–864, 2000.

[64] M. A. van de Wiel, D. E. Te Beest, and M. M. Münch. Learning from a lot: empirical Bayes for high-dimensional model-based prediction. *Scandinavian Journal of Statistics*, 46(1):2–25, 2019.

[65] Y. Kim, W. Wang, P. Carbonetto, and M. Stephens. A flexible empirical Bayes approach to multiple linear regression and connections with penalized regression. *Journal of Machine Learning Research*, 25(185):1–59, 2024.

[66] F. Morgante, P. Carbonetto, G. Wang, Y. Zou, A. Sarkar, and M. Stephens. A flexible empirical Bayes approach to multivariate multiple regression, and its improved accuracy in predicting multi-tissue gene expression from genotypes. *PLoS Genetics*, 19(7):e1010539, 2023.

[67] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[68] G. Wang, A. Sarkar, P. Carbonetto, and M. Stephens. A simple new approach to variable selection in regression, with application to genetic fine mapping. *Journal of the Royal Statistical Society, Series B*, 82(5):1273–1300, 2020.

[69] S. J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151:3–34, 2015.

[70] Y. Kim, P. Carbonetto, M. Stephens, and M. Anitescu. A fast algorithm for maximum likelihood estimation of mixture proportions using sequential quadratic programming. *Journal of Computational and Graphical Statistics*, 29(2):261–273, 2020.

[71] F. Chollet and others. Keras, 2015. URL `https://keras.io`.

[72] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction are supported by (i) a qualitative comparison of cEBMF to related work and (ii) empirical assessments in a variety of data sets.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We empirically assessed the performance of cEBMF in the situation where the prior was misspecified.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The key theoretical results are found in the Appendix. We have provided proofs of Proposition 1 and Lemma 1, and we have clearly stated the assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have provided detailed descriptions of the experiments, including the software used. We have also provided the code that was used to generate all the results in the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have provided the R and Python code implementing the methods and experiments. The data sets were either included or instructions for accessing the data sets were given.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Many of these details are given in the Appendix. Additional implementation details can

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We summarize some results using boxplots with the conventional definitions for whiskers, box bounds, center line and outliers.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details of the computing hardware used in the scability benchmark are given in Sec. 4.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and our work abides by it to the best of our knowledge.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We don't believe that our work has any obvious direct social impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This work did not release data or models that would be considered to have a strong potential for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All the original data sources are credited/cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the source code for our new methods, and it is accompanied with detailed documentation.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: this work does not involve research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs were not were not used in this work.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendices

A Derivations and additional method details

The cEBMF algorithm (Sec. 3.3.2) is a *variational empirical Bayes* algorithm [61–64] that is formulated as solving the following optimization problem:

$$\underset{q,\, g,\, \boldsymbol{\tau}}{\operatorname{argmax}} \operatorname{ELBO}(q, g, \boldsymbol{\tau}), \tag{16}$$

where g is shorthand for the priors $g_1^{(\ell)}, \ldots, g_K^{(\ell)}, g_1^{(f)}, \ldots, g_K^{(f)}$, q is a distribution on $(\mathbf{L}, \mathbf{F})$, and $\operatorname{ELBO}(q, g, \boldsymbol{\tau})$ is the "Evidence Lower BOund" (ELBO) [51], a lower bound to the "evidence", $\log p(\mathbf{Z} \mid g, \boldsymbol{\tau})$:

$$\operatorname{ELBO}(q, g, \boldsymbol{\tau}) := \mathbb{E}_q[\log p(\mathbf{Z} \mid \mathbf{L}, \mathbf{F}, \boldsymbol{\tau})] + \mathbb{E}_q\left[\log\left\{\frac{p(\mathbf{L}, \mathbf{F} \mid \mathbf{X}, \mathbf{Y})}{q(\mathbf{L}, \mathbf{F})}\right\}\right]. \tag{17}$$

See [13, 65, 66] for other variational empirical Bayes algorithms derived in a similar way.

To achieve tractable update expressions for the model parameters, we approximate the posterior $q(\mathbf{L}, \mathbf{F})$ so that it factorizes over all elements of $\mathbf{L}$ and all elements of $\mathbf{F}$ (sometimes called a "mean field" approximation):

$$\begin{aligned} q(\mathbf{L}, \mathbf{F}) &= q^{\ell}(\mathbf{L})\, q^{f}(\mathbf{F}) \\ q^{\ell}(\mathbf{L}) &= \prod_{i=1}^{n} \prod_{k=1}^{K} q_{ik}^{\ell}(\ell_{ik}) \\ q^{f}(\mathbf{F}) &= \prod_{j=1}^{p} \prod_{k=1}^{K} q_{jk}^{f}(f_{jk}). \end{aligned} \tag{18}$$

With this factorization (or conditional independence) qconstraint on q, the right-hand part of the ELBO can be immediately decomposed into a sum of expectations over the individual elements of $\mathbf{L}$ and $\mathbf{F}$, so we have

$$\begin{aligned} \operatorname{ELBO}(q, g, \boldsymbol{\tau}) = \mathbb{E}_q[&\log(p(\mathbf{Z} \mid \mathbf{L}, \mathbf{F}, \boldsymbol{\tau})] \\ &+ \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{E}_q\left[\log\left\{\frac{g_k^{(\ell)}(l_{ik}; \boldsymbol{x}_i)}{q_{ik}^{\ell}(\ell_{ik})}\right\}\right] \\ &+ \sum_{j=1}^{p} \sum_{k=1}^{K} \mathbb{E}_q\left[\log\left\{\frac{g_k^{(f)}(f_{jk}; \boldsymbol{y}_j)}{q_{jk}^{f}(f_{jk})}\right\}\right], \end{aligned} \tag{19}$$

where $g_k^{(\ell)}(\ell; \boldsymbol{x}_i)$ denotes the density of $g_k^{(\ell)}(\boldsymbol{x}_i)$ at ℓ, and $g_k^{(f)}(f; \boldsymbol{y}_j)$ denotes the density of $g_k^{(f)}(\boldsymbol{x}_i)$ at f.

A.1 Updating the factors

The following proposition formally connects the updates of the individual factors $k = 1, \ldots, K$ (Step 2–4 of the algorithm in Sec. 3.3.2) to learning a covariate-moderated EBNM model (Sec. 3.3.1).

Proposition 1. Let $\boldsymbol{\ell}_k = (\ell_{1k}, \ldots, \ell_{nk})^T$ denote the kth column of $\mathbf{L}$, let $\boldsymbol{f}_k = (f_{1k}, \ldots, f_{pk})^T$ denote the kth column of $\mathbf{F}$, let $\bar{\boldsymbol{\ell}}_k = \mathbb{E}_q[\boldsymbol{\ell}_k]$, $\bar{\boldsymbol{f}}_k = \mathbb{E}_q[\boldsymbol{f}_k]$, $\bar{\boldsymbol{\ell}}_k^2 = \mathbb{E}_q[\boldsymbol{\ell}_k^2]$ and $\bar{\boldsymbol{f}}_k^2 = \mathbb{E}_q[\boldsymbol{f}_k^2]$, and we further define

$$q_k^{\ell}(\boldsymbol{\ell}_k) := \prod_{i=1}^{n} q_{ik}^{\ell}(\ell_{ik}) \tag{20}$$

$$q_k^{f}(\boldsymbol{f}_k) := \prod_{j=1}^{p} q_{jk}^{f}(f_{jk}). \tag{21}$$

Let $\bar{\mathbf{R}}^k$ denote the $n \times p$ matrix of expected residuals (with elements $\bar{r}_{ij}^k$) that ignores the contribution of the kth factor,

$$\bar{r}_{ij}^k := z_{ij} - \sum_{k' \neq k} \bar{l}_{ik'} \bar{f}_{jk'}. \tag{22}$$

Also define $\hat{\boldsymbol{\ell}}(\mathbf{Z}, \boldsymbol{t}, \boldsymbol{w}, \boldsymbol{\tau})$, $\hat{\boldsymbol{f}}(\mathbf{Z}, \boldsymbol{t}, \boldsymbol{w}, \boldsymbol{\tau})$, $\boldsymbol{s}^\ell(\boldsymbol{w}, \boldsymbol{\tau})$ and $\boldsymbol{s}^f(\boldsymbol{w}, \boldsymbol{\tau})$ as vector-valued functions in which the individual vector elements given by

$$\hat{\ell}_i(\mathbf{Z}, \boldsymbol{t}, \boldsymbol{w}, \boldsymbol{\tau}) = \frac{\sum_{j=1}^p \tau_{ij} z_{ij} t_j}{[s_i^\ell(\boldsymbol{w}, \boldsymbol{\tau})]^2} \tag{23}$$

$$\hat{f}_j(\mathbf{Z}, \boldsymbol{t}, \boldsymbol{w}, \boldsymbol{\tau}) = \frac{\sum_{i=1}^n \tau_{ij} z_{ij} t_i}{[s_j^f(\boldsymbol{w}, \boldsymbol{\tau})]^2} \tag{24}$$

$$s_i^\ell(\boldsymbol{w}, \boldsymbol{\tau}) = (\textstyle\sum_{j=1}^p \tau_{ij} w_j)^{-1/2} \tag{25}$$

$$s_j^f(\boldsymbol{w}, \boldsymbol{\tau}) = (\textstyle\sum_{i=1}^n \tau_{ij} w_i)^{-1/2}. \tag{26}$$

Then using the definition of the ELBO in (17) and the cEBNM mapping defined in (13), we have that

$$\operatorname{argmax}_{q_k^\ell,\, g_k^{(\ell)} \in \mathcal{G}_{\ell,k}} \operatorname{ELBO}(q, g, \boldsymbol{\tau}) = \operatorname{cEBNM}(\hat{\boldsymbol{\ell}}(\bar{\mathbf{R}}^k, \bar{\boldsymbol{f}}_k, \bar{\boldsymbol{f}}_k^2, \boldsymbol{\tau}), \boldsymbol{s}_{\boldsymbol{\ell}}(\bar{\boldsymbol{f}}_k^2, \boldsymbol{\tau}), \mathbf{X}, \mathcal{G}_{\ell,k}) \tag{27}$$

$$\operatorname{argmax}_{q_k^f,\, g_k^{(f)} \in \mathcal{G}_{f,k}} \operatorname{ELBO}(q, g, \boldsymbol{\tau}) = \operatorname{cEBNM}(\hat{\boldsymbol{f}}(\bar{\mathbf{R}}^k, \bar{\boldsymbol{\ell}}_k, \bar{\boldsymbol{\ell}}_k^2, \boldsymbol{\tau}), \boldsymbol{s}_{\boldsymbol{f}}(\bar{\boldsymbol{\ell}}_k^2, \boldsymbol{\tau}), \mathbf{Y}, \mathcal{G}_{f,k}). \tag{28}$$

Note that this identity requires a slight change to the definition of the cEBNM mapping (13) as returning the priors $g(\boldsymbol{d}_i, \hat{\boldsymbol{\theta}})$ at $\hat{\boldsymbol{\theta}}$ rather than the parameter estimates $\hat{\boldsymbol{\theta}}$ themselves.

Proof. Starting from (19), we expand on the parts of ELBO that involve q_k^ℓ or $g_k^{(\ell)}$ or both:

$$\operatorname{ELBO}(q, g, \boldsymbol{\tau}) = -\frac{1}{2} \sum_{i=1}^n \mathbb{E}_q[a_{ik} l_{ik}^2 - 2 b_{ik} l_{ik}] + \sum_{i=1}^n \mathbb{E}_q \left[\log \left\{ \frac{g_k^{(\ell)}(\ell_{ik}; \boldsymbol{x}_i)}{q_{ik}^{(\ell)}(\ell_{ik})} \right\} \right] + \text{const}, \tag{29}$$

where "const" is a placeholder for the terms in the ELBO that do not depend on the kth factor, and we define

$$a_{ik} := \sum_{j=1}^p \tau_{ij} (\bar{f}_{jk})^2 \tag{30}$$

$$b_{ik} := \sum_{j=1}^p \tau_{ij} \bar{r}_{ij}^k \bar{f}_{jk}. \tag{31}$$

The identity (27) then follows from Lemma 1 (given below). The other identity (28) is proved similarly. □

A.2 Updating the residual variances

Focussing on the part of the ELBO depends on $\boldsymbol{\tau}$, we have

$$\operatorname{ELBO}(q, g, \boldsymbol{\tau}) = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^p (\log \tau_{ij} - \tau_{ij} \bar{r}_{ij}^2) + \text{const}, \tag{32}$$

in which "const" is a placeholder for the other terms in the ELBO that do not involve $\boldsymbol{\tau}$, and $\bar{r}_{ij}^2$ is the expected squared difference between the observation z_{ij} and the value predicted by the matrix factorization:

$$\bar{r}_{ij}^2 := \mathbb{E}_q[(z_{ij} - \hat{z}_{ij})^2], \tag{33}$$

where

$$\hat{z}_{ij} = \sum_{k=1}^K l_{ik} f_{jk}. \tag{34}$$

If one makes the modeling assumption that all the residual variances are the same, i.e., $\tau_{ij} = \tau$, then from (32) the update for τ works out to

$$\tau = \frac{n \times p}{\sum_{i=1}^{n} \sum_{j=1}^{p} \bar{r}_{ij}^2}. \tag{35}$$

If instead one makes the weaker modeling assumption that the residual variances are the same in each column, i.e., $\tau_{ij} = \tau_j, j = 1, \ldots, p$, then the updates work out to

$$\tau_j = \frac{n}{\sum_{i=1}^{n} \bar{r}_{ij}^2}. \tag{36}$$

Similarly, for row-specific residual variances the updates are

$$\tau_i = \frac{p}{\sum_{j=1}^{p} \bar{r}_{ij}^2}. \tag{37}$$

For all these expressions, the squared differences $\bar{r}_{ij}^2$ are easily computed given the conditional independence assumptions of the fully-factorized approximation (18):

$$\bar{r}_{ij}^2 = \left(z_{ij} - \sum_{k=1}^{K} \bar{l}_{ik} \bar{f}_{jk} \right)^2 + \sum_{k=1}^{K} (\bar{l}_{ik}^2)(\bar{f}_{jk}^2) - \sum_{k=1}^{K} (\bar{l}_{ik} \bar{f}_{jk})^2, \tag{38}$$

in which we have defined $\bar{l}_{ik} := \mathbb{E}_q[l_{ik}]$, $\bar{f}_{jk} := \mathbb{E}_q[f_{jk}]$, $\bar{l}_{ik}^2 := \mathbb{E}_q[l_{ik}^2]$ and $\bar{f}_{jk}^2 := \mathbb{E}_q[f_{jk}^2]$.

A.3 Covariate-moderated EBNM

To complete the proof of Proposition 1, it remains to show that the identity (27) is satisfied at the objective function given by (29). (And similarly for the identity (28).) This connection is made in the following lemma.

Lemma 1. Consider the cEBNM mapping defined in (13). An equivalent definition of this mapping is

$$(\hat{\boldsymbol{\theta}}, \hat{q}) = \operatorname{argmax}_{\boldsymbol{\theta},\, q} F(\boldsymbol{\theta}, q; \hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}), \tag{39}$$

where

$$F(\boldsymbol{\theta}, q; \hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}) = -\frac{1}{2} \sum_{i=1}^{n} \mathbb{E}_q[a_i \beta_i^2 - 2 b_i \beta_i] + \sum_{i=1}^{n} \mathbb{E}_q \left[\log \left\{ \frac{g(\beta_i; \boldsymbol{d}_i, \boldsymbol{\theta})}{q_i(\beta_i)} \right\} \right], \tag{40}$$

and $g(\beta; \boldsymbol{d}_i, \boldsymbol{\theta})$ denotes the density of $g(\boldsymbol{d}_i, \boldsymbol{\theta})$ at β, and we further define

$$q(\boldsymbol{\beta}) = \textstyle\prod_{i=1}^{n} q_i(\beta_i) \tag{41}$$

$$a_i = 1/s_i^2 \tag{42}$$

$$b_i = \hat{\beta}_i / s_i^2. \tag{43}$$

Proof. We begin with the ELBO for the cEBNM model (7, 9):

$$\mathrm{ELBO}(\boldsymbol{\theta}, q; \hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}) = \log \mathcal{L}(\boldsymbol{\theta}) - D_{\mathrm{KL}}(q \,\|\, p_{\mathsf{post}}). \tag{44}$$

where $D_{\mathrm{KL}}(q \,\|\, p)$ denotes the Kullback-Leibler (K-L) divergence from a distribution p to a distribution q [67], $\mathcal{L}(\boldsymbol{\theta})$ is the marginal likelihood defined in (11), and $p_{\mathsf{post}}(\boldsymbol{\beta})$ is the (exact) posterior distribution, $p_{\mathsf{post}}(\boldsymbol{\beta}) := \prod_{i=1}^{n} p(\beta_i \mid \hat{\beta}_i, s_i, \hat{\boldsymbol{\theta}}, \mathbf{D})$ (see eq. 12). Since $D_{\mathrm{KL}}(q \,\|\, p)$ is always zero or greater, and is exactly zero when $p = q$, we have that $\operatorname{argmax}_q \mathrm{ELBO}(\boldsymbol{\theta}, g; \hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}) = p_{\mathsf{post}}$ and $\max_q \mathrm{ELBO}(\boldsymbol{\theta}, g; \hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}) = \log \mathcal{L}(\boldsymbol{\theta})$. Next, a basic identity of the ELBO (see for example Appendix B of [68]) is that the ELBO can be rewritten as

$$mathrmELBO(\boldsymbol{\theta}, q; \hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}) = \mathbb{E}_q[\log p(\hat{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \boldsymbol{s})] + \sum_{i=1}^{n} \mathbb{E}_q \left[\log \left\{ \frac{g(\beta_i; \boldsymbol{d}_i, \boldsymbol{\theta})}{q_i(\beta_i)} \right\} \right]. \tag{45}$$

To complete the proof, we expand terms in the log-likelihood in (45):

$$\log p(\hat{\boldsymbol{\beta}} \mid \boldsymbol{\beta}, \boldsymbol{s}) = -\frac{1}{2} \sum_{i=1}^{n} \frac{(\beta_i - \hat{\beta}_i)^2}{s_i^2} + \text{const}, \tag{46}$$

where "const" is a placeholder for terms that do not involve q (or g). Plugging this identity into (45), and with a bit of additional algebraic manipulation, we recover (40). □

Algorithm 1 cEBMF algorithm

Require: $n \times p$ data matrix, $\mathbf{Z}$; covariate or "side information" matrices, $\mathbf{X}$ $(n \times n_x)$ and $\mathbf{Y}$ $(p \times n_y)$; K, the number of factors; the prior families $\mathcal{G}_{\ell,k}$ and $\mathcal{G}_{f,k}$, $k = 1, \ldots, K$; and initial estimates of the first and second moments of $\mathbf{L}$ $(n \times K)$, $\mathbf{F}$ $(p \times K)$, which are denoted by $\bar{\mathbf{L}}$, $\bar{\mathbf{F}}$, $\bar{\mathbf{L}}^2$, $\bar{\mathbf{F}}^2$.
Compute the expected residuals, $\bar{\mathbf{R}} = \mathbf{Z} - \bar{\mathbf{L}}\bar{\mathbf{F}}^T$.
repeat
  Update the residual variances $\boldsymbol{\tau}$ using (35), (36) or (37).
  for $k = 1, \ldots, K$ **do**
    Remove the effect of the kth factor from the expected residuals, $\bar{\mathbf{R}}^k = \bar{\mathbf{R}} + \bar{\boldsymbol{\ell}}_k \bar{\boldsymbol{f}}_k^T$.
    Perform a single-factor update for factor k (Algorithm 2).
    Update the expected residuals, $\bar{\mathbf{R}} = \bar{\mathbf{R}}^k - \bar{\boldsymbol{\ell}}_k \bar{\boldsymbol{f}}_k^T$.
  end for
until some convergence criterion is met
return $\bar{\mathbf{L}}, \bar{\mathbf{F}}, \boldsymbol{\tau}, g_1^{(\ell)}, \ldots, g_K^{(\ell)}, g_1^{(f)}, \ldots, g_K^{(f)}$

Algorithm 2 cEBMF single-factor update

Require: covariate or "side information" matrices, $\mathbf{X}$ $(n \times n_x)$ and $\mathbf{Y}$ $(p \times n_y)$; $k \in \{1, \ldots, K\}$, the dimension to update; the prior families $\mathcal{G}_{\ell,k}$ and $\mathcal{G}_{f,k}$; an implementation of $\text{cEBNM}(\hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{D}, \mathcal{G}) \to (\hat{\boldsymbol{\theta}}, \hat{q})$ (eq. 13) for prior families $\mathcal{G} = \mathcal{G}_{\ell,k}$ and $\mathcal{G} = \mathcal{G}_{f,k}$; the expected residuals, $\bar{\mathbf{R}}^k$; estimates of the second moments, $\bar{\mathbf{L}}^2$ $(n \times K)$, $\bar{\mathbf{F}}^2$ $(p \times K)$; and the residuals variances, $\boldsymbol{\tau}$.

1. $\hat{\boldsymbol{\beta}} \leftarrow \hat{\boldsymbol{\ell}}(\bar{\mathbf{R}}^k, \bar{\boldsymbol{f}}_k, \bar{\boldsymbol{f}}_k^2, \boldsymbol{\tau})$
2. $\boldsymbol{s} \leftarrow \boldsymbol{s}_\ell(\bar{\boldsymbol{f}}_k^2, \boldsymbol{\tau})$
3. $(g_k^{(\ell)}, q_k^\ell) \leftarrow \text{cEBNM}(\hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{X}, \mathcal{G}_{\ell,k})$
4. Compute posterior moments $\bar{\ell}_{ik} := E_q[\ell_{ik}]$ and $\bar{\ell}_{ik}^2 := E_q[\ell_{ik}^2]$, $i = 1, \ldots, n$.
5. $\hat{\boldsymbol{\beta}} \leftarrow \hat{\boldsymbol{f}}(\bar{\mathbf{R}}^k, \bar{\boldsymbol{\ell}}_k, \bar{\boldsymbol{\ell}}_k^2, \boldsymbol{\tau})$
6. $\boldsymbol{s} \leftarrow \boldsymbol{s}_f(\bar{\boldsymbol{\ell}}_k^2, \boldsymbol{\tau})$
7. $(g_k^{(f)}, q_k^f) \leftarrow \text{cEBNM}(\hat{\boldsymbol{\beta}}, \boldsymbol{s}, \mathbf{Y}, \mathcal{G}_{f,k})$
8. Compute posterior moments $\bar{f}_{jk} := E_q[f_{jk}]$ and $\bar{f}_{jk}^2 := E_q[f_{jk}^2]$, $j = 1, \ldots, p$.
9. **return** $\bar{\boldsymbol{\ell}}_k, \bar{\boldsymbol{\ell}}_k^2, \bar{\boldsymbol{f}}_k, \bar{\boldsymbol{f}}_k^2, g_k^{(\ell)}, g_k^{(f)}$

A.4 Detailed algorithms

In summary, the cEBMF algorithm is a block co-ordinate ascent algorithm [69] for finding a local maximum of the ELBO (17), in which the "blocks"—i.e., the subsets of parameters to be updated—are the individual factors $k = 1, \ldots, K$ (Sec. A.1) and the residual variances $\boldsymbol{\tau}$ (Sec. A.2). This co-ordinate ascent algorithm is described in Algorithm 1, and the single-factor update is described in Algorithm 2. (And it is described informally in Sec. 3.3.2.) In practice, we run Algortithm 1 until the increase in the ELBO across two successive iterations is smaller than some specified tolerance, or until we have reached an upper bound on the number of iterations.

Two features of the empirical Bayes approach to matrix factorization discussed in [13] are worth highlighting here. First, there is a simple stepwise procedure for obtaining good initial estimates of $\mathbf{L}$ and $\mathbf{F}$ by introducing the factors sequentially. This was called a "greedy initialization" in [13]. Second, instead of fixing the number of factors, the EBMF approach can also select K automatically by adapting the priors $g_k^{(\ell)}, g_k^{(f)}$ separately for each factor k. The idea is that factors that are not useful for explaining variation in the data should produce priors that are concentrated near zero (this feature of course requires that the chosen prior families $\mathcal{G}_{\ell,k}, \mathcal{G}_{f,k}$ include distributions that are concentrated near zero). Therefore, K can initially be set to a large value, and the cEBMF algorithm will automatically determine an appropriate number of factors by "shrinking" the unneeded factors.

A.4.1 Computational complexity

Since cEBMF is a modeling and algorithmic framework, and not a specific method or algorithm, we cannot give the exact computational complexity of Algorithm 1. However, we can provide some rules

of thumb. Steps 1, 2, 5 and 6 in Algorithm 2 (also, Steps 1 and 2 in Sec. 3.3.2) involve preparing the inputs for the cEBNM solver (13). Since these steps do not depend on the prior families $\mathcal{G}_{\ell,k}, \mathcal{G}_{f,k}$, we can give their computational complexity: when $\mathbf{Z}$ is a "dense" (non-sparse) matrix, the time complexity for updating a single factor k is $O(np)$; when $\mathbf{Z}$ is sparse, the complexity is $O(S)$, where S is the number of nonzero entries in $\mathbf{Z}$. (Note this requires careful implementation that avoids directly storing $\bar{\mathbf{R}}$). Steps 3 and 7 in Algorithm 2 (or Steps 3 and 4 in Sec. 3.3.2) will depend on the details of the cEBNM solver and the type of side information. However, when the priors on $\mathbf{L}, \mathbf{F}$ are simple and involve low-dimensional covariates, the other steps are expected to dominate, in which case the complexity of Algorithm 1 is expected to be $O(npK)$ or $O(SK)$.

B Details of the experiments

B.1 Simulations

We simulated data sets from different cEBMF models. In all cases, the data were generated with homoskedastic noise, $\tau_{ij} = \tau$.

Sparsity-driven covariate simulations. This simulation was intended to illustrate the behaviour of cEBMF when provided with simple row and column-covariates that inform only the sparsity of $\mathbf{L}$ and $\mathbf{F}$ (and not the magnitudes of their elements). The side information was stored in 1,000 $\times$ 10 and 200×10 matrices $\mathbf{X}$ and $\mathbf{Y}$, and the 1,000 $\times$ 200 matrix $\mathbf{Z}$ was simulated using a simple cEBMF model with $K = 2$ and with spike-and-slab priors chosen to ensure that 90% of the elements of $\mathbf{LF}^T$ were zero. Specifically, we used the following priors:

$$\begin{aligned} \ell_{ik} &\sim \pi_{ik}\delta_0 + (1 - \pi_{ik})N(0, 1) \\ f_{jk} &\sim \alpha_{jk}\delta_0 + (1 - \alpha_{jk})N(0, 1) \\ \pi_{ik} &:= \phi(\boldsymbol{\theta}_k^T \boldsymbol{x}_i) \\ \alpha_{jk} &:= \phi(\boldsymbol{\omega}_k^T \boldsymbol{y}_j), \end{aligned} \tag{47}$$

in which $\boldsymbol{\theta}_k$ and $\boldsymbol{\omega}_k$ were chosen to achieve 90% zeros in $\mathbf{LF}^T$.

Uninformative covariate simulations. To verify that cEBMF was robust to situations in which the side information was not helpful, we considered an "uninformative covariate" setting in which the covariates were just noise. The data sets were simulated in the same way as in the sparsity-driven covariate simulations except that the true factors were simulated as $\ell_{ik} \sim \pi\delta_0 + (1 - \pi)N(0, 1)$, $f_{jk} \sim \alpha\delta_0 + (1 - \alpha)N(0, 1)$, with π, α chosen to achieve a target sparsity of 90% zeros in $\mathbf{LF}^T$.

Tiled-clustering simulations. In this setting, we simulated rank-3 matrix factorizations in which $\mathbf{L}$—but not $\mathbf{F}$—depended on the 2-d locations of the data points. (One of these simulations is shown in Fig. 1.) This was accomplished as follows. First, we generated a periodic tiling of $[0, 1] \times [0, 1]$, randomly labeling each tile 1, 2 or 3. For each data point $i = 1, \ldots,$ 1,000, we sampled its 2-d location uniformly from $[0, 1] \times [0, 1]$, then we set $\ell_{ik} = 1$ if the data point fell in the tile with label k, otherwise $\ell_{ik} = 0$. The 200×3 matrix $\mathbf{F}$ was simulated from a scale mixture of zero-centered normals that did not depend on tile membership.

Shifted tiled-clustering simulations. To assess robustness to model misspecification, we simulated data using a prior that could not be recovered by the prior family we used in cEBMF. These simulations were the same as the tiled-clustering simulations except that we generated the ith row of $\mathbf{L}$ as follows: $(1, 2, 3)$ if data point was i in the tile with label 1; $(3, 1, 2)$ if data point was in the tile with label 2, and $(2, 3, 1)$ if data point was in the tile with label 3.

B.2 Additional details on the methods compared

We first describe how the methods were run on the simulated data sets. Modifications to the methods for the MovieLens and spatial transcriptomics data are given in the main text, with additional technical details below. For all methods, when possible to do so, we set the rank, K, to match the rank of the simulated matrix factorization.

For EBMF and cEBMF, we assumed homoskedastic noise ($\tau_{ij} = \tau$), and the prior families were chosen to align with how the data were simulated (except for the shifted-tiled clustering simulations, which were intended to illustrate the methods' behaviour when the priors were misspecified). For EBMF, the prior families were all elaborations of the "spike and slab" priors (Sec. 3.2.1). For cEBMF, the priors were of the same form as EBMF in which the mixture weights were parameterized using either a multinomial regression (i.e., a single-layer neural network with a softmax link function) or a multilayer perceptron.

In the sparsity-driven covariate and uninformative covariate simulations, the priors for $\mathbf{L}$ and $\mathbf{F}$ in EBMF were all scale mixtures of normals with a fixed grid of scales [50, 70]. For cEBMF, we used priors of the same form, except that side information was incorporated into the prior mixture weights as follows using priors of the following form:

$$g(\boldsymbol{d}_i, \boldsymbol{\theta}) = \pi_0(\boldsymbol{d}_i, \boldsymbol{\theta})\delta_0 + \sum_{m=1}^{M} \pi_m(\boldsymbol{d}_i, \boldsymbol{\theta})N(0, \sigma_m^2). \tag{48}$$

The mixture weights $\pi_0, \ldots, \pi_M$ were implemented using a standard multinomial regression model with the softmax link function.

In the tiled-clustering and shifted tiled-clustering simulations, the true $\mathbf{L}$ was always non-negative. Therefore, we chose the prior families in EBMF and cEBMF to produce *semi-non-negative matrix factorizations* [16] with a non-negative $\mathbf{L}$. Specifically, we assigned mixture-of-exponentials priors to $\mathbf{L}$, similar to the scale-mixture-of-normals priors, except with support for non-negative numbers only [35]. And we assigned the scale-mixture-of-normal priors, same as above, to $\mathbf{F}$. In cEBMF, side information was incorporated into the mixture weights in the prior in a manner similar to above:

$$g(\boldsymbol{d}_i, \boldsymbol{\theta}) = \pi_0(\boldsymbol{d}_i, \boldsymbol{\theta})\delta_0 + \sum_{m=1}^{M} \pi_m(\boldsymbol{d}_i, \boldsymbol{\theta}) \exp(\lambda_m), \tag{49}$$

in which $\exp(\lambda)$ denotes the exponential distribution with scale parameter λ, and $\lambda_{m-1} < \lambda_m$, $m = 2, \ldots, M$. As before, the mixture weights $\pi_0, \ldots, \pi_M$ were implemented using a standard multinomial regression model with the softmax link function.

The deep learning methods (VAE, cVAE, NCF) were all implemented in PyTorch. All the models were trained for 50 epochs using the Adam optimizer with learning rate 0.001 and batch size 64. VAE had three hidden layers (of width 128, 64 and 30) in both the encoder and decoder (20 hidden dimensions). ReLU activations were used throughout. We use the ELBO from [26] to train the model. We proceeded similarly for the cVAE, conditioning both the encoder and decoder on the available covariate data $\mathbf{X}, \mathbf{Y}$. For cVAE, we used the training objective from [27]. NCF models $\mathbf{Z}$ using two separate multilayer perceptrons for the row and column covariates [28]. The multilayer perceptrons were implemented in a similar way to the VAE encoders and decoders; that is, three hidden layers (of width 128, 64 and 30) with RELU activations. The penalty parameters in PMD were tuned via cross-validation as recommended by the authors. SSVD (R package "ssvd") was run with its default values.

For the spatial transcriptomics data, we fit cEBMF and EBMF using gene-specific residual variances, $\sigma_{ij}^2 = \sigma_j^2$. We used mixture-of-exponential priors for $\mathbf{F}$, and the parameterized mixture-of-exponential priors (49) for $\mathbf{L}$ in which the mixture weights were learned using a multilayer perceptron instead of a multinomial regression. The multilayer perceptions were defined as sequential models with a dense layer with 64 units and ReLU activations. We use two subsequent dense layers, each with 64 units, and ReLU activations using an L2 regularization coefficient of 0.001 to prevent overfitting. These regularized layers were followed by a dropout layer (with a dropout rate of 0.5). The subsequent layers were four dense layers each with 64 units, ReLU activations and L2 regularization coefficient of 0.001. The final layer was a dense layer with a softmax activation. These models were trained during each single-factor update using 300 epochs and a batch size of 1,500.

In the simulations, cEBMF was implemented in R, in which learning the parameterized priors was performed using the Keras R interface [71] to TensorFlow [72]. For the MovieLens and spatial transcriptomics data sets, we used the PyTorch-based implementation of cEBMF which we have made available as a Python package on GitHub.

C Additional results

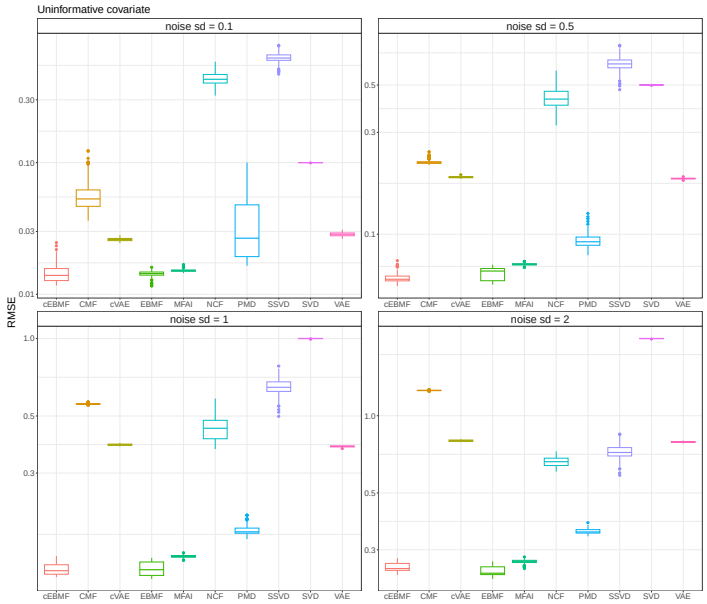

Figure 5: Simulation results from the "uninformative covariate" setting in which the data were simulated under different noise levels, τ. Note that for improved visualization the RMSE is shown on the log-scale.

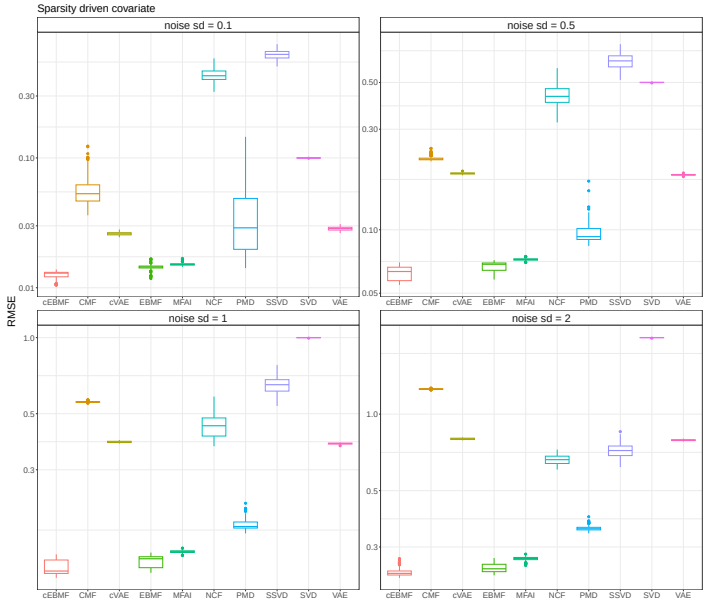

Figure 6: Simulation results from the "sparsity-driven covariate" setting in which the data were simulated under different noise levels, τ. Note that for improved visualization the RMSE is shown on the log-scale.

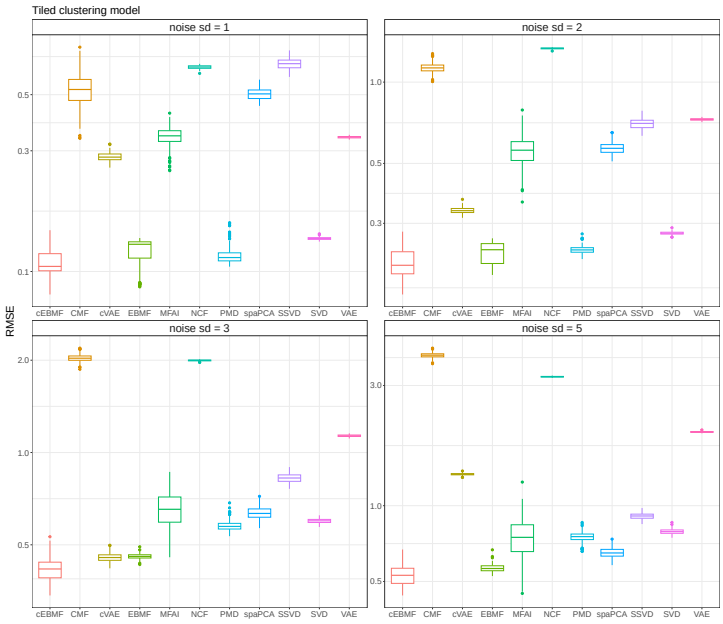

Figure 7: Simulation results from the "tiled-clustering" setting in which the data were simulated under different noise levels, τ. Note that for improved visualization the RMSE is shown on the log-scale. (spaPCA = Spatial PCA)

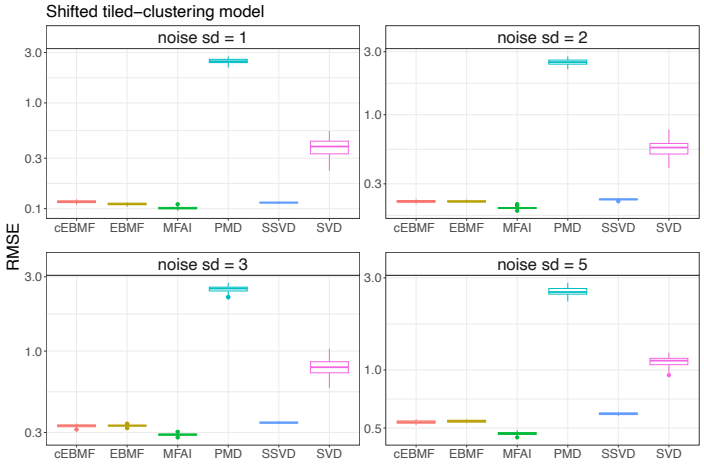

Figure 8: Simulation results from the "shifted tiled-clustering" setting in which the data were simulated under different noise levels, τ. Note that for improved visualization the RMSE is shown on the log-scale.

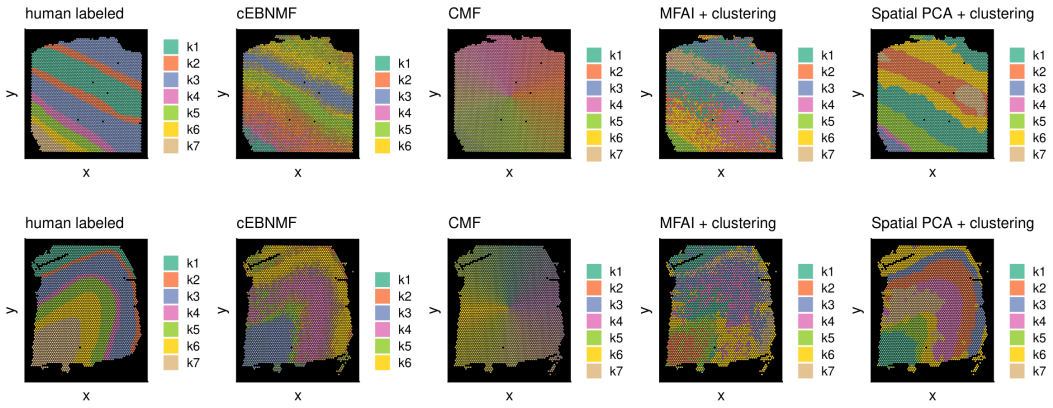

Figure 9: Additional results on slides 4 (top) and 10 (bottom) of the DLPFC spatial transcriptomics data. See Fig. 4 for additional information about these results.

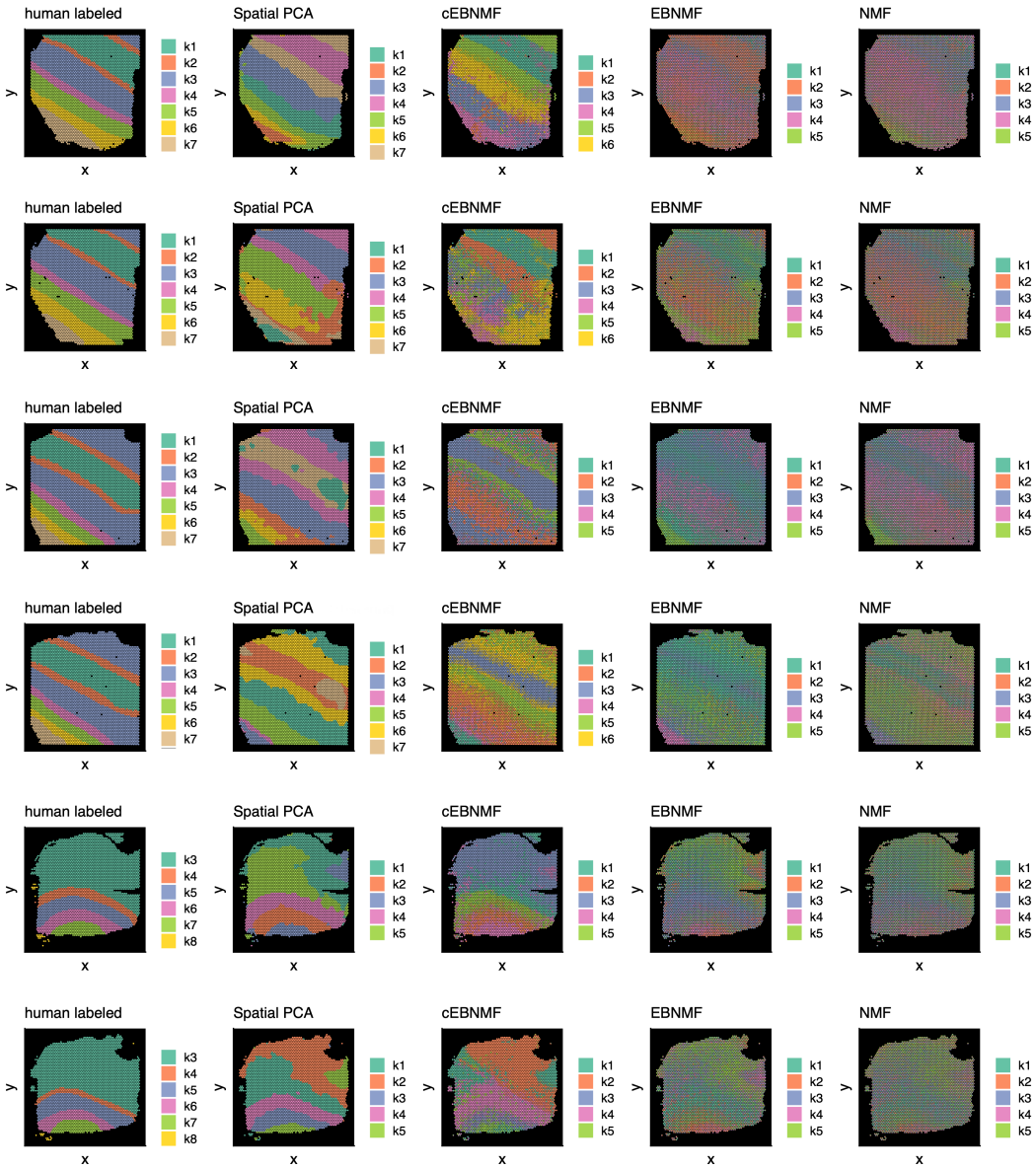

Figure 10: Selected results on slices 1 (top row) through 6 (bottom row) of the DLPFC spatial transcriptomics data.

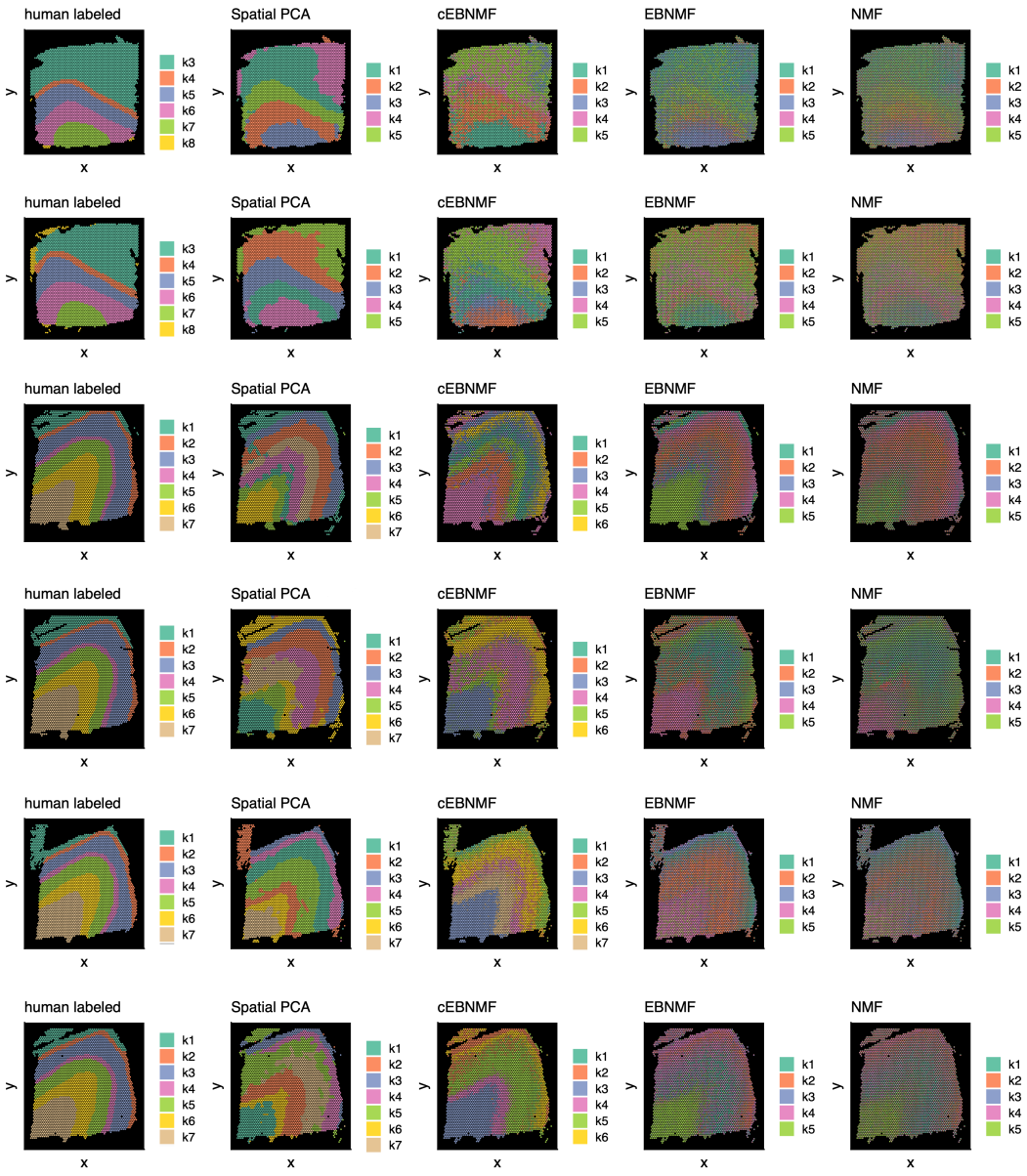

Figure 11: Selected results on slices 7 (top row) through 12 (bottom row) of the DLPFC spatial transcriptomics data.