Bi-level algorithm for optimizing hyperparameters in penalized nonnegative matrix factorization

Nicoletta Del Buono[a], Flavia Esposito[a], Laura Selicato[a,*], Rafał Zdunek[b]

[a]*Department of Mathematics, University of Bari Aldo Moro, Via Orabona 4, Bari, 70125, Italy*
[b]*Faculty of Electronics, Photonics, and Microsystems, Wroclaw University of Science and Technology, 27 Wybrzeze Wyspianskiego st., Wrocław, 50370, Poland*

Abstract

Learning approaches rely on hyperparameters that impact the algorithm's performance and affect the knowledge extraction process from data. Recently, Nonnegative Matrix Factorization (NMF) has attracted a growing interest as a learning algorithm. This technique captures the latent information embedded in large datasets while preserving feature properties. NMF can be formalized as a penalized optimization task in which tuning the penalty hyperparameters is an open issue. The current literature does not provide any general framework addressing this task. This study proposes to express the penalty hyperparameters problem in NMF in terms of a bi-level optimization. We design a novel algorithm, named Alternating Bi-level (AltBi), which incorporates the hyperparameters tuning procedure into the updates of NMF factors. Results of the existence and convergence of numerical solutions, under appropriate assumptions, are studied, and numerical experiments are provided.

Keywords: Nonnegative Matrix Factorization, Hyperparameter Optimization, Penalty coefficient, Low-rank approximation
2010 MSC: 15A23, 65K10, 65F55, 68Q32, 68V20, 90C46, 46N10

1. Introduction

All learning models require setting some hyperparameters (HPs)– variables governing the learning approach – before starting the learning process from data. HPs tuning requires a substantial effort, depending on the user, and affects the learner's performance [1]. Automatic Hyperparameter Optimization (HPO) would bring a solution to these problems [2].

---

*Corresponding author
Email address: laura.selicato@uniba.it (Laura Selicato)
Abbreviations - HP: Hyperparameter - HPO: Hyperparameter Optimization - GB: Gradient-based - MU: Multiplicative Updates - RMD: Reverse-Mode Differentiation - FMD: Forward-Mode Differentiation - P-MU: Penalized Multiplicative Update.

HPO strategies commonly used in the literature range from simple methods, such as the grid or random search, to more complex ones, such as the Bayesian optimization or the Genetic Algorithms (GAs) [3–8]. Grid search explores a prescribed set of HPs in a given search space, while random search defines a random sampling of HPs without any assumption on the search space. Both these strategies are time-consuming since they are driven by some performance metrics, commonly measured by cross-validation. Moreover, they require domain experts to justify a search space that is meaningful for the application domain. Bayesian optimization attempts to predict how unseen combinations of HPs will perform based on a so-called surrogate model that approximates the HPO problem. GAs are based on stochastic optimization and are inspired by the biological phenomena of natural evolution. Recently, some works proved that Gradient-Based (GB) approaches can obtain great results in HPO for large-scale problems, using only local information and at least one HP (learning rate) [9, 10]. GB methods reduce the validation error, computing or approximating the gradient with respect to HPs [11–13]. One of the ways to go through GB methods for HPO is to formalize the problem as a bi-level task [14–16]. Bi-level programming solves an outer optimization problem subject to the optimality of an inner optimization one [17].

Formally, let $\mathscr{A}$ be a learner with hyperparameter vector $\boldsymbol{\lambda} \in \mathbb{R}^p$, parameter vector[2] $\mathbf{w} \in \mathbb{R}^q$, with $p, q \in \mathbb{N}$, and $\mathbf{X} \in \mathbb{R}^{n \times m}$ with $n, m \in \mathbb{N}$ an assigned data matrix. For learning model $\mathscr{A}$, the HPO can be written as:

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda} \in \Lambda} \mathscr{F}(\mathscr{A}(\mathbf{w}(\boldsymbol{\lambda}), \boldsymbol{\lambda}), \mathbf{X}) \quad \text{s. t.} \quad \mathbf{w}(\boldsymbol{\lambda}) = \arg\min_{\mathbf{w}} \mathscr{L}(\boldsymbol{\lambda}, \mathbf{X}), \quad (1)$$

where $\mathscr{F}$ evaluates how good is $\mathbf{w}$ gained by learner $\mathscr{A}$ tuned with hyperparameter $\boldsymbol{\lambda}$ on $\mathbf{X}$, and $\mathscr{L}$ is an empirical loss. Typically, the inner problem aims to minimize empirical loss $\mathscr{L}$; the outer problem is related to HPs. Because of the implicit dependence of the outer problem on $\boldsymbol{\lambda}$, equation (1) is challenging to solve. Recently, first order bi-level optimization techniques based on estimating Jacobian $\frac{d\mathbf{w}(\boldsymbol{\lambda})}{d\boldsymbol{\lambda}}$ via implicit or iterative differentiation have been proposed to solve (1) [13, 15, 18].

However, there are still no effective results of using GB methods for HPO in the unsupervised field. This study aims to use these techniques to revise problem (1) in an unsupervised learning context, to automatically achieve HPs. We consider Nonnegative Matrix Factorization (NMF) and its constrained variants (in particular sparseness constraint) [19–27]. We regard these problems as penalized optimization tasks in which penalty coefficients are HPs, focusing on their proper choice via HPO. Taking advantage of the bi-level HPO problem formulation, we construct an alternating bi-level approach that includes the HPs choices as a part of the algorithm that computes the factors in the NMF data approximation task under study. The rest of this section reviews prelimi-

---

[2]$\mathbf{w}$ can be a scalar, a vector or a matrix.

nary concepts on NMF and its sparsity constraints with the importance of the penalty HPs for sparse NMF. Section 2 describes the novel bi-level formulation of the penalized NMF and its treatment via an alternating methodology. We prove the existence of the solution to this problem, and we use convergence results to design a new algorithm, named Alternating Bi-level (AltBi), which is described in Section 3. It is our numerical proposal to solve the HPO issue in NMF models with additional sparsity constraints. Section 4 illustrates the numerical results obtained using the AltBi algorithm on synthetic and real signal datasets. Section 5 sketches some conclusive remarks and future works.

1.1. Preliminaries

NMF groups some methodologies aiming to approximate nonnegative data matrix $\mathbf{X} \in \mathbb{R}_+^{n \times m}$ as $\mathbf{X} \approx \mathbf{W}\mathbf{H}$, where $\mathbf{W} \in \mathbb{R}_+^{n \times r}$ is the *basis* matrix, and $\mathbf{H} \in \mathbb{R}_+^{r \times m}$ is the *encoding* (or *coefficient*) matrix. The choice of parameter r, which determines the number of rows of $\mathbf{H}$ (respectively, columns of $\mathbf{W}$) and $r << \min(n, m)$, is problem-dependent and user-specified; and it represents an example of HP connected with NMF. A general NMF problem can be formulated as an optimization task

$$\min_{\mathbf{W} \geq 0, \mathbf{H} \geq 0} D_\beta(\mathbf{X}, \mathbf{W}\mathbf{H}) = \min_{\mathbf{W} \geq 0, \mathbf{H} \geq 0} \sum_{i=1}^{n} \sum_{j=1}^{m} d_\beta(x_{ij}, \sum_{k=1}^{r} w_{ik}h_{kj}), \qquad (2)$$

where the objective function $D_\beta(\cdot, \cdot)$ is a β-divergence assessing how well its reconstruction $\mathbf{W}\mathbf{H}$ fits $\mathbf{X}$, where d_β is generally defined for each $x, y \in \mathbb{R}$ as

$$d_\beta(x, y) = \begin{cases} \frac{1}{\beta(\beta-1)}(x^\beta + (\beta-1)y^\beta - \beta xy^{\beta-1}) & \beta \in \mathbb{R} \setminus \{0, 1\}; \\ x\log(\frac{x}{y}) - x + y & \beta = 1; \\ \frac{x}{y} - \log(\frac{x}{y}) - 1 & \beta = 0. \end{cases}$$

Either data properties and specific application domain influence the particular choice of D_β (popular measures are for $\beta = 2, 1, 0$, i.e., the Frobenius norm, the generalized Kullback-Leibler (KL) and the Itakura-Saito (IS) divergences, respectively).

The NMF model in (2) can also be enriched with additional constraints by introducing penalty terms

$$\min_{\mathbf{W} \geq 0, \mathbf{H} \geq 0} D_\beta(\mathbf{X}, \mathbf{W}\mathbf{H}) + \lambda_{\mathbf{W}} \mathscr{R}_1(\mathbf{W}) + \lambda_{\mathbf{H}} \mathscr{R}_2(\mathbf{H}), \qquad (3)$$

where $\mathscr{R}_1 : \mathbb{R}^{n \times r} \to \mathbb{R}$ and $\mathscr{R}_2 : \mathbb{R}^{r \times m} \to \mathbb{R}$ are some penalty functions enforcing specific properties on the factor matrices; $\lambda_{\mathbf{W}}, \lambda_{\mathbf{H}} \in \mathbb{R}_+$ are the penalty coefficients (i.e., HPs), that balance the bias-variance trade-off in approximating $\mathbf{X}$ and preserving the additional constraints. It is assumed that at least one of the two HPs is non-null for the penalty to make sense, and (3) allows to penalize simultaneously one or both factors. The problem of properly selecting the penalty HPs is still an unsolved issue in constrained NMF.

One example of a suitable constraint to impose on NMF factors is sparsity. Sparseness leads to several advantages; it allows obtaining some form of compression, improves the computational cost and gives us better interpretability when many features (the columns in $\mathbf{X}$) are present, and the model becomes very large. Several zeros avoid over-fitting, allow a way for feature extraction, and elude modeling the noise implicitly embedded in the data. Nonnegativity in the NMF algorithms naturally produces sparse factors. Nonetheless, because the factor sparseness degree is uncontrollable, it is preferable to use direct constraints that can enforce this property [28]. Various penalty terms enforce sparsity in NMF: an example is to apply ℓ_0 "norm" on $\mathbf{W}$ and $\mathbf{H}$ [29]. However, this penalization makes the associated objective function non-smooth, globally non-differentiable, and non-convex, resulting in an NP-hard optimization problem (3). Conversely, due to their analytical proprieties, ℓ_1 and ℓ_2 norms are valid alternatives to ℓ_0 [30]. In particular, ℓ_1 norm originates from the Lasso problem [31] and addresses several computational issues in machine learning and pattern recognition. Sparsity can also be imposed via $\ell_{1,2}$ norm which is used either as a penalty function or as an objective function [32–34]. The Hoyer's sparse NMF optimization task uses the normalized ratio of ℓ_1 and ℓ_2 norm computed on the columns of $\mathbf{W}$ and rows of $\mathbf{H}$ [35]. Section 3 illustrates our algorithm proposal to tune HPs using an objective function based on the KL-divergence and ℓ_1 norm.

1.2. The penalty HP in NMF

Usually, static optimization mechanisms, such as the grid or random search, perform HPs tuning in constrained NMF (3). These approaches solve several variants of the same problem associated with a predefined discrete set of HPs and then choose the best one according to empirical criteria, (an example can be found in the context of gene expression analysis [36]). Other approaches are based on the Discrepancy Principle (DP) and the L-curve criterion which are empirical methods used to tune the penalty value in Tikhonov regularization [37, 38]. Active-set approaches for the NMF model, which are based on the Frobenius norm and the Tikhonov regularization on $\mathbf{W}$, are other sophisticated strategies for tuning penalty HPs, and they usually choose the best penalty HP according to clustering performance [39]. Bayesian optimization methodologies are exploited to solve the problem

$$\min_{\mathbf{H} \geq 0} \frac{1}{2} ||\mathbf{X} - \mathbf{WH}||_F^2 + \frac{\lambda}{2} \mathscr{R}(\mathbf{H}), \tag{4}$$

where $\mathscr{R}(\mathbf{H}) = \mathrm{Tr}\left(\mathbf{H}^\top \mathbf{EH}\right) = \sum_i ||\mathbf{h}_i||_1^2 = ||\mathbf{H}||_{2,1}^2$, $\mathbf{h}_i$ is the i-th row of $\mathbf{H}$, $\mathbf{E} \in \mathbb{R}^{r \times r}$ is the all-ones matrix that enforces sparsity on $\mathbf{H}$'s columns using the squared norm $\ell_{2,1}$, and $\lambda \in \mathbb{R}$ is the penalty HP [40]. In the associated minimization problem, the choice of λ is made according to the following exponential rule

$$\lambda^{(k)} = \lambda_0 \exp\left(-\tau k\right), \tag{5}$$

4

where k is the number of iteration in the algorithm, λ_0 is the initial value of the HP and τ is a parameter controlling the results.

In this study, we want to automate the choice of HPs through GB methods and bi-level approach in order to free the HPs tuning from the domain expert and any empirical or cross-validation related techniques.

2. New Formulation

Several approaches can tackle HPO in model (3), even though a uniform theory applicable to general objectives and penalty functions is still lacking. The results reported in this study aim to fill this void.

This section presents the main contribution of the work. We reformulate the model (3) as:

$$\min_{\mathbf{H}\geq 0,\mathbf{W}\geq 0} D_\beta(\mathbf{X},\mathbf{WH}) + \mathscr{R}_1(\mathbf{L_W W}) + \mathscr{R}_2(\mathbf{H L_H}), \tag{6}$$

where $\mathbf{L_W} \in \mathbb{R}^{n\times n}$ and $\mathbf{L_H} \in \mathbb{R}^{m\times m}$ are diagonal matrices of HPs associated with each row of $\mathbf{W}$ and each column of $\mathbf{H}$, respectively, and $\mathscr{R}_1 : \mathbb{R}^{n\times r} \to \mathbb{R}$ and $\mathscr{R}_2 : \mathbb{R}^{r\times m} \to \mathbb{R}$ are the penalty functions being continuous and such that $\mathscr{R}_i(\mathbb{0}) = 0$ for each $i = 1,2$, where $\mathbb{0}$ is the zero element in $\mathbb{R}^{n\times r}$ and $\mathbb{R}^{r\times m}$, respectively. In this way, each row and each column are penalized independently. Although the problem can be written for both factors $\mathbf{W}$ and $\mathbf{H}$, we, for now, focus on the case where $\mathbf{L_H} = \mathbb{0}_{\mathbb{R}^{m\times m}}$ and $\mathbf{L_W} = \mathbf{L} \in \mathbb{R}^{n\times n}$, diagonal and non-null matrix (because the penalty makes sense) so that (6) is reduced to

$$\min_{\mathbf{H}\geq 0,\mathbf{W}\geq 0} D_\beta(\mathbf{X},\mathbf{WH}) + \mathscr{R}(\mathbf{LW}). \tag{7}$$

A symmetric extension can be easily derived for $\mathbf{L_H} \neq \mathbb{0}_{\mathbb{R}^{m\times m}}$ and $\mathbf{L_W} = \mathbb{0}_{\mathbb{R}^{n\times n}}$. On the other hand, simultaneous optimization on both factors (with respect to (6)) requires some supplement theory related to the bi-level formulation of NMF for columns, which will be the subject of future works. Problem (7) is convex in each variable separately[3]. Alternating optimization techniques are helpful to incorporate into the minimization process the updates of each NMF factor separately. Firstly, fixing $\mathbf{W}$, one estimates $\mathbf{H}$; subsequently, $\mathbf{H}$ is fixed to estimate $\mathbf{W}$. To tune the penalty HP matrix $\mathbf{L}$, we incorporate it simultaneously into the process of updating factor $\mathbf{W}$, introducing a bi-level strategy on each row of $\mathbf{W}$.

Let $\mathbf{w}_i \in \mathbb{R}^r$ be the i-th column of $\mathbf{W}^\top$ and from now on, let $\boldsymbol{\lambda} \in \mathbb{R}^n$ indicate the vector of diagonal elements of $\mathbf{L}$ and $\lambda_i \in \Lambda \subset \mathbb{R}$ the i-th diagonal element of $\mathbf{L}$. We first consider the simple minimization problem in $\mathbf{H}$ (for fixed $\mathbf{W}$):

$$\min_{\mathbf{H}\geq 0} D_\beta(\mathbf{X},\mathbf{WH}). \tag{8}$$

---

[3]for particular values of β and specific penalty functions.

To obtain the update for **W** and achieve an optimal solution for matrix **L**, we use the bi-level task applied to each $\mathbf{w}_i$, i.e the i-th row of **W**, which for each $i = 1, \ldots, n$ reads:

$$\min_{\lambda_i \in \Lambda} f(\lambda_i), \qquad f(\lambda_i) = \inf\{\mathscr{E}(\mathbf{w}_i(\lambda_i), \lambda_i) : \mathbf{w}_i(\lambda_i) \in \arg\min_{\mathbf{u} \in \mathbb{R}^r} \mathscr{L}_{\lambda_i}(\mathbf{u})\}, \qquad (9)$$

where $f : \Lambda \to \mathbb{R}$ is the so-called *Response Function* (RF) of the outer problem related to the i-th row of **W** (according to the bi-level notation). Namely, first we fix an outer level hyperparameter λ_i, then we solve the inner level problem finding $\mathbf{w}_i$ as argmin of a loss function. Finally, the feasible solution of $\min_{\lambda_i \in \Lambda} f(\lambda_i)$ is evaluated. Note that the RF associated with the entire matrix problem is $F(\boldsymbol{\lambda}) = \sum_{i=1}^{n} f(\lambda_i)$. *Error Function* (EF) $\mathscr{E}$ is the outer objective such that

$$\mathscr{E} : \mathbb{R}^r \times \Lambda \to \mathbb{R} : (\mathbf{w}_i, \lambda_i) \mapsto \sum_{j=1}^{m} d_\beta(\mathbf{x}_j, \sum_{k=1}^{r} w_{ik}(\lambda_i)h_{kj}), \qquad (10)$$

where for every $\lambda_i \in \Lambda$; whereas *Loss Function* (LF) $\mathscr{L}_{\lambda_i}$ is the inner objective

$$\mathscr{L}_{\lambda_i} : \mathbb{R}^r \to \mathbb{R} : \mathbf{w}_i \mapsto \sum_{j=1}^{m} d_\beta(\mathbf{x}_j, \sum_{k=1}^{r} w_{ik}h_{kj}) + \lambda_i \varkappa(\mathbf{w}_i), \qquad (11)$$

where $\varkappa : \mathbb{R}^r \to \mathbb{R}$ is a linear function, closely related to the enforcement of the constraint, such that $\sum_{i=1}^{n} \lambda_i \varkappa(\mathbf{w}_i) = \mathscr{R}(\mathbf{LW})$.

In the following section, we clarify how to handle each part of optimization problems (8) and (9).

2.1. Finding the unpenalized factor

To solve (8), different update rules satisfying diverse requirements exist (fast convergence or easy implementation mechanisms); they range from multiplicative to additive update rules [41, 42]. In this study, we focus on the standard NMF Multiplicative Updates (MU) [43] due to their ease of implementation and monotonic convergence. From initial matrices, MU uses scaling rules from the minimization of an auxiliary function (derived from Richardson-Lucy or Expectation-Maximization (EM) approaches [44–48]). Any approach based on an auxiliary function is often used to solve NMF problems because it ensures the nonnegativity of the computed factors without further handling, notwithstanding that it converges slowly [43, 49].

We briefly review the update rule for the general β divergence giving the particular result for the KL divergence.

Considering the update rule

$$\mathbf{H} \leftarrow \mathbf{H}. * \frac{\mathbf{W}^\top ((\mathbf{WH})^{\cdot[\beta-2]} \cdot * \mathbf{X})}{\mathbf{W}^\top (\mathbf{WH})^{\cdot[\beta-1]}}, \tag{12}$$

being $.*$ the Hadamard product (exponential and ratio operators are computed element-wise), it is known that the general β-divergence $D_\beta(\cdot, \cdot)$, is non-increasing using rule (12) for $0 \leq \beta \leq 2$. In particular, the paper [50] shows this result for $\beta = 2$ and $\beta = 1$. In [51], it is generalized to the case $1 \leq \beta \leq 2$. In practice, we observe that the criterion is still non-increasing under update (12) for $\beta < 1$ and $\beta > 2$ (and in particular for $\beta = 0$, that corresponds to the IS divergence). More details on theoretical results and proofs can be found in [43, 49, 52].

Specifically for the KL divergence, (12) becomes:

$$\mathbf{H} \leftarrow \mathbf{H}. * \frac{\mathbf{W}^\top (\mathbf{X}./(\mathbf{WH}))}{(\sum\limits_{i=1}^{n} \mathbf{w}_i) \cdot \mathbb{1}_\mathbb{m}^\top}, \tag{13}$$

where $\mathbb{1}_m$ is the ones-vector of dimension m.

2.2. Finding the penalized factor and solving the HPO

To obtain the update for $\mathbf{W}$ and determine an optimal solution for penalty matrix $\mathbf{L}$, we use bi-level approach (9) applied on each row of $\mathbf{W}$. To simplify the notation, from now until the end of subsection 2.2, subscript i for $\mathbf{w}_i$ and λ_i is omitted. For the sake of simplicity, we suppose the existence of a unique minimizer $\mathbf{w}_{(\lambda)}$ for the inner objective. Nevertheless, problem (9) generally has no closed expression for $\mathbf{w}_{(\lambda)}$, so it does not allow to optimize the outer objective function directly.

A reliable approach is to replace the inner problem with a dynamical system [13, 18, 53]. This point allows us to compute an exact gradient of an approximation of (9). It also enables optimization of the HPs that define the learning dynamics. As mentioned before, depending on how the gradient with respect to HPs is calculated, two main approaches can be used: the implicit differentiation, based on the implicit function theorem, and the iterative differentiation approach. In this work, we will focus on the latter.

Therefore, the solution of the inner object minimization as a dynamical system with state $\mathbf{w}^{(t)} \in \mathbb{R}^r$ can be written as:

$$\mathbf{w}^{(t)} = \Phi_t(\mathbf{w}^{(t-1)}, \lambda) \quad t = 1, ., T; \tag{14}$$

with initial condition $\mathbf{w}^{(0)} = \Phi_0(\lambda)$, where $\Phi_t : (\mathbb{R}^r \times \mathbb{R}) \to \mathbb{R}^r$ is a smooth map, and it is the row-wise update for $\mathbf{W}$, for $t = 1, \ldots, T$. Note that $\mathbf{w}^{(t)}$ for all $i = 1, \ldots, n$ depend on λ, implicitly.

Bi-level problem (9) can be approximated (for each $i = 1, \ldots, n$) using the constrained procedure:

$$\min_{\lambda} f(\lambda) \quad \text{s. t.} \quad \mathbf{w}^{(t)} = \Phi_t(\mathbf{w}^{(t-1)}, \lambda) \quad \text{for} \quad t = 1, \dots, T. \tag{15}$$

In general, procedure (15) might not be the best approximation for bi-level problem (9) since the minimizer of $\mathscr{L}_\lambda$, to which the optimization dynamic converges, does not necessarily minimize $\mathscr{E}$. This problem is overcome by assuming the uniqueness of the minimizer of $\mathscr{L}_\lambda$, for any $\lambda \in \Lambda \subset \mathbb{R}$, as we will see in detail in the following subsection 2.2.1. Moreover, we note that for $1 \leq \beta \leq 2$, thanks to the convexity of the β divergence function, and consequently, of $\mathscr{L}_\lambda{}^4$, the associated problems $\arg\min f^{(T)}(\lambda)$, $\arg\min f(\lambda)$, and $\arg\min \mathscr{L}_\lambda$ are singleton, where $f^{(T)}$ is the response function at time T.

2.2.1. Existence and Convergence Results

We provide results on the existence of solutions to problem (9) and the (variational) convergence for approximate problem (15) related to it.

Hypothesis 1. *Considering the following assumptions:*

1. $\Lambda \subset \mathbb{R}$ *is compact;*
2. *Error Function* (10) *is jointly continuous[5];*
3. *application* $(\mathbf{w}, \lambda) \to \mathscr{L}_\lambda(\mathbf{w})$ *is jointly continuous, and problem* $\arg\min \mathscr{L}_\lambda$ *is a singleton for every* $\lambda \in \Lambda$;
4. $\forall \lambda \in \Lambda$, $\mathbf{w}_{(\lambda)} = \arg\min \mathscr{L}_\lambda$ *is bounded.*

Therefore, bi-level problem (9) can be reformulated as follows:

$$\min_{\lambda \in \Lambda} f(\lambda) = \mathscr{E}(\mathbf{w}_{(\lambda^*)}, \lambda^*), \qquad \mathbf{w}_{(\lambda)} = \arg\min_{\mathbf{u}} \mathscr{L}_\lambda(\mathbf{u}), \tag{16}$$

where $(\mathbf{w}_{(\lambda^*)}, \lambda^*)$ is the optimal solution.

Theorem 2.1 (Existence). *Problem* (16) *admits solutions under the assumptions* $1 - 4$.

Proof. From the compactness of Λ, the continuity of f ensures minimizers exist. Consider $\hat{\lambda} \in \Lambda$ and sequence $(\lambda_n)_{n \in \mathbb{N}}$ in Λ converging to $\hat{\lambda}$. Due to the boundness of associate sequence $(\mathbf{w}_{(\lambda_n)})_{n \in \mathbb{N}}$, there is a converging subsequence $(\mathbf{w}_{(\lambda_{k_n})})_{n \in \mathbb{N}}$ such that $\lim_{\lambda_{k_n} \to \hat{\lambda}} \mathbf{w}_{(\lambda_{k_n})} = \hat{\mathbf{w}} \in \mathbb{R}^r$.

For point 3 in Hypothesis 1, since λ_{k_n} converges to $\hat{\lambda}$, it results:

$$\forall \mathbf{w} \in \mathbb{R}^r \quad \mathscr{L}_{\hat{\lambda}}(\hat{\mathbf{w}}) = \lim_n \mathscr{L}_{\lambda_{k_n}}(\mathbf{w}_{(\lambda_{k_n})}) \leq \lim_n \mathscr{L}_{\lambda_{k_n}}(\mathbf{w}) = \mathscr{L}_{\hat{\lambda}}(\mathbf{w}). \tag{17}$$

Thus, $\hat{\mathbf{w}}$ is a minimizer of $\mathscr{L}_{\hat{\lambda}}$ and consequently $\hat{\mathbf{w}} = \hat{\mathbf{w}}_{(\lambda)}$. This proves that sequence $(\mathbf{w}_{(\lambda_n)})_{n \in \mathbb{N}}$ is bounded and has a unique accumulation point.

---

[4]$\mathscr{L}_\lambda$ is convex as a sum of convex functions.
[5]The function is continuous with respect to each variable separately.

Consequently $(\mathbf{w}_{(\lambda_n)})_{n\in\mathbb{N}}$ converges to $\mathbf{w}_{(\hat{\lambda})}$ (i.e. its unique accumulation point). Lastly, for point 2 of Hypothesis 1 and since $(\mathbf{w}_{(\lambda_n)}, \lambda_n) \to (\mathbf{w}_{(\hat{\lambda})}, \hat{\lambda})$, it follows $f(\lambda_n) = \mathscr{E}(\mathbf{w}_{(\lambda_n)}, \lambda_n) \to \mathscr{E}(\mathbf{w}_{(\hat{\lambda})}, \hat{\lambda}) = f(\hat{\lambda})$, that concludes the proof. $\qquad\square$

Theorem 2.2 (Convergence). *In addition to Hypothesis 1, suppose that:*

    5. $\mathscr{E}(\cdot, \lambda)$ is uniformly Lipschitz continuous;

    6. $(\mathbf{w}_{(\lambda)}^{(T)})_{T\in\mathbb{N}} \to \mathbf{w}_{(\lambda)}$ uniformly on Λ for $T \to +\infty$.

Then

    (a) $\inf f^{(T)}(\lambda) \to \inf f(\lambda)$,

    (b) $\arg\min f^{(T)}(\lambda) \to \arg\min f(\lambda)$.

To prove Theorem 2.2, the following preliminary result concerning the stability of minima and minimizers in optimization problems is helpful (the complete proof of this result can be found in [54]).

Theorem 2.3. *Let g_T and g be lower semi-continuous functions defined on a compact set Λ. If $g_T \to g$ uniformly on Λ for $T \to +\infty$, then*

    (a) $\inf g_T \to \inf g$

    (b) $\arg\min g_T \to \arg\min g$.

Thanks to these results, Theorem 2.2 can be proved.

Proof of Theorem 2.2. The uniform Lipschitz continuity of $\mathscr{E}(\cdot, \lambda)$ ensures that there exists $\nu > 0$ such that:

$$|f^{(T)}(\lambda) - f(\lambda)| = |\mathscr{E}(\mathbf{w}_{(\lambda)}^{(T)}, \lambda) - \mathscr{E}(\mathbf{w}_{(\lambda)}, \lambda)| \leq \nu \|\mathbf{w}_{(\lambda)}^{(T)} - \mathbf{w}_{(\lambda)}\|,$$

for every $T \in \mathbb{N}$, and $\lambda \in \Lambda$.
Since $\mathscr{E}(\cdot, \lambda)$ is uniformly Lipschitz continuous, it results that $f^{(T)}(\lambda) \to f(\lambda)$ uniformly on Λ as $T \to +\infty$. The thesis follows by Theorem 2.3. $\qquad\square$

Hypotheses $1 - 6$ are satisfied by many problems of practical interest, in particular when $1 \leq \beta \leq 2$. Results for other values of β could be obtained, losing the hypothesis of convexity of f and $\mathscr{L}_\lambda$.

2.2.2. Solving the bi-level problem

Bi-level problem (16) or approximate problem (15) satisfy existence and convergence theorems, respectively, so we can focus on finding penalty HPs matrix $\mathbf{L}$ in practice. We now reintroduce subscript i for $\mathbf{w}_i$ and λ_i. Applying a gradient type approach on each diagonal element of $\mathbf{L} = \text{diag}(\boldsymbol{\lambda})$, the optimization of $\boldsymbol{\lambda}$ depends on the approximation of hypergradient $\nabla_{\boldsymbol{\lambda}} F$. Using the chain rule, it results:

$$\frac{\partial F}{\partial \lambda_i} = \frac{\partial f}{\partial \lambda_i} + \frac{\partial f}{\partial \mathbf{w}_i^{(T)}} \cdot \frac{d\mathbf{w}_i^{(T)}}{d\lambda_i}, \quad \forall i = 1, \ldots, n, \tag{18}$$

where $\frac{\partial f}{\partial \lambda_i} \in \mathbb{R}$ and $\frac{\partial f}{\partial \mathbf{w}_i^{(T)}} \in \mathbb{R}^r$ are available.

Following the iterative differentiation approach, the computation of the hypergradient can be done using the Reverse-Mode Differentiation (RMD) or Forward-Mode Differentiation (FMD). RMD computes the hypergradient by back-propagation; instead, FMD works with forwarding propagation. In our algorithm, we use only the second mode; for completeness, we report both.

Reverse Mode. The reverse strategy to compute the hypergradient is based on the Lagrangian perspective calculated for (15), that is $\mathfrak{L} : \mathbb{R}^r \times \Lambda \times \mathbb{R}^r \to \mathbb{R}$ which is defined as $\mathfrak{L}(\mathbf{w}_i, \lambda_i, \boldsymbol{\alpha}) = \mathscr{E}(\mathbf{w}_i^{(T)}, \lambda_i) + \sum_{t=1}^{T} \boldsymbol{\alpha}_t^\top (\Phi_t(\mathbf{w}_i^{(t-1)}, \lambda_i) - \mathbf{w}_i^{(t)})$ for $i = 1, \ldots, n$, where, for each $t = 1, \ldots, T$, $\boldsymbol{\alpha}_t \in \mathbb{R}^r$ are the Lagrange multipliers associated with the t-th step of the dynamics. The partial derivatives of Lagrangian $\mathfrak{L}$ are

$$\frac{\partial \mathfrak{L}}{\partial \boldsymbol{\alpha}_t} = \Phi_t(\mathbf{w}_i^{(t-1)}, \lambda_i) - \mathbf{w}_i^{(t)}, \quad \frac{\partial \mathfrak{L}}{\partial \mathbf{w}_i^t} = \boldsymbol{\alpha}_{t+1}^\top \mathbf{A}_{t+1} - \boldsymbol{\alpha}_t^\top,$$

$$\frac{\partial \mathfrak{L}}{\partial \mathbf{w}_i^{(T)}} = \nabla \mathscr{E}(\mathbf{w}_i^{(T)}, \lambda_i) - \boldsymbol{\alpha}_T^\top, \quad \frac{\partial \mathfrak{L}}{\partial \lambda_i} = \sum_{t=1}^{T} \boldsymbol{\alpha}_t^\top \mathbf{b}_t,$$

where

$$\mathbf{A}_t = \frac{\partial \Phi_t(\mathbf{w}_i^{(t-1)}, \lambda_i)}{\partial \mathbf{w}_i^{(t-1)}} \in \mathbb{R}^{r \times r} \quad \text{and} \quad \mathbf{b}_t = \frac{\partial \Phi_t(\mathbf{w}_i^{(t-1)}, \lambda_i)}{\partial \lambda_i} \in \mathbb{R}^{r \times 1}. \tag{19}$$

Therefore the optimality conditions give the iterative rules of RMD:

$$\begin{cases} \boldsymbol{\alpha}_T^\top = \nabla \mathscr{E}(\mathbf{w}_i^{(T)}, \lambda_i), \\ h_T = \frac{\partial f}{\partial \lambda_i}, \\ h_{t-1} = h_t + \mathbf{b}_t \boldsymbol{\alpha}_t^\top, \\ \boldsymbol{\alpha}_{t-1}^\top = \mathbf{A}_t \boldsymbol{\alpha}_t^\top, \end{cases} \tag{20}$$

for $t = T, \ldots, 1$ and $i = 1, \ldots, n$. Then the i-th component of the hypergradient can be computed as $\frac{\partial f}{\partial \lambda_i} = h_0$.

Forward-Mode. FMD computes the derivative of (18) by the chain rule. Each Φ_t depends on λ_i directly, and on $\mathbf{w}_i^{(t-1)}$ indirectly, for $t = 1, \ldots, T$. Hence:

$$\frac{d\mathbf{w}_i^{(t)}}{d\lambda_i} = \frac{\partial \Phi_t(\mathbf{w}_i^{(t-1)}, \lambda_i)}{\partial \mathbf{w}_i^{(t-1)}} \frac{d\mathbf{w}_i^{(t-1)}}{d\lambda_i} + \frac{\partial \Phi_t(\mathbf{w}_i^{(t-1)}, \lambda_i)}{\partial \lambda_i}. \tag{21}$$

Defining $\mathbf{s}_t = \frac{d\mathbf{w}_i^{(t)}}{d\lambda_i} \in \mathbb{R}^r$, each FMD iterate is:

$$\begin{cases} \mathbf{s}_0 = \mathbf{b}_0; \\ \mathbf{s}_t = \mathbf{A}_t \mathbf{s}_{t-1} + \mathbf{b}_t \quad t = 1, \ldots, T; \end{cases} \tag{22}$$

where $\mathbf{A}_t$ and $\mathbf{b}_t$ are defined as above, and the i-th component of the hyper-gradient is

$$\frac{\partial F}{\partial \lambda_i} = \mathbf{g}_T^\top \cdot \mathbf{s}_T \in \mathbb{R}, \quad \text{being} \quad \mathbf{g}_T = \frac{\partial f}{\partial \mathbf{w}_i^{(T)}} \in \mathbb{R}^r. \tag{23}$$

Letting $\mathbf{s}_0 = 0$, the solution of (22) solves:

$$\frac{\partial F(\lambda_i)}{\partial \lambda_i} = \frac{\partial f^{(T)}(\lambda_i)}{\partial \mathbf{w}_i^{(T)}} \Big(\mathbf{b}_T + \sum_{t=0}^{T-1} (\prod_{s=t+1}^{T} \mathbf{A}_s) \mathbf{b}_t \Big). \tag{24}$$

Computational considerations. Opting between RMD and FMD depends on balancing the trade-off based on the size of $\mathbf{w}_i$ and λ_i. The RMD approach requires that $\mathbf{w}_i^{(t)}$ for all $i = 1, \ldots, n$ and all $t = 1, \ldots, T$ are stored in memory to compute $\mathbf{A}_t$ and $\mathbf{b}_t$ in the backward pass, and therefore it is suitable when the quantity rT is small. As we will see later, our approach uses the FMD strategy that requires time $O(rT)$ and space $O(r)$ for every row and iteration.

3. Alternating Bi-level Algorithm - AltBi

In this section, we present our Alternating Bi-level (AltBi) algorithm for the particular case of $\beta = 1$ and ℓ_1 as penalty function in (7):

$$\min_{\mathbf{H} \geq 0, \mathbf{W} \geq 0} D_1(\mathbf{X}, \mathbf{W}\mathbf{H}) + ||\mathbf{L}\mathbf{W}||_1. \tag{25}$$

It implements the procedures described in the previous sections, performing NMF updating, including the automatic setting of the HPs. As its name suggests, AltBi optimizes $\mathbf{H}$ and $\mathbf{W}$ alternately through the bi-level approach. Sub-interval of arbitrary length T, called *bunch*, is considered to perform the bi-level procedure on $\mathbf{W}$. It ensures the extraction of a convergent sub-sequence from any bounded sequence[6]. Even if this is not unique, it is enough to consider its sub-sequence to have the same limit.

---

[6]This holds for the Bolzano-Weierstrass Theorem.

Algorithm 1 shows the pseudo-code for AltBi. It receives as input data matrix $\mathbf{X}$, the rank of factorization r, initial matrices $\mathbf{W}$, $\mathbf{H}$, and vector $\boldsymbol{\lambda}$ of the diagonal elements of $\mathbf{L}$. We initialize the number of iterates $MaxIter$, tolerance tol, and length T of the bunch. The outer while-loop repeats the alternating algorithm until one of the two conditions $err > tol$ or $iter < MaxIter$ is false. Error is defined as the absolute value of the difference in the divergence calculated between two successive iterates divided by the divergence at the initial step. The inner for-loop performs the bi-level procedure for every bunch, calculating the hypergradient to update $\boldsymbol{\lambda}$ with a gradient method. The algorithm returns the optimal matrices $\mathbf{W}^*$, $\mathbf{H}^*$, and $\mathbf{L}^* = \text{diag}(\boldsymbol{\lambda}^*)$.

Algorithm 1: Alternating Bi-level Algorithm - AltBi

Data: $\mathbf{X} \in \mathbb{R}_+^{n \times m}$, $r < \min(n, m)$.
Result: $\mathbf{W}^* \in \mathbb{R}_+^{n \times r}$, $\mathbf{H}^* \in \mathbb{R}_+^{r \times m}$, $\mathbf{L}^* = \text{diag}(\boldsymbol{\lambda}^*) \in \mathbb{R}_+^{n \times n}$.
Initializations: $\mathbf{W} \in \mathbb{R}_+^{n \times r}$, $\mathbf{H} \in \mathbb{R}_+^{r \times m}$, $\mathbf{L} = \text{diag}(\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n))$, T, $MaxIter$, tol, err, and $iter$.
while *(err > tol) & (iter < MaxIter)* **do**
    update $\mathbf{H}$ as in (13);
    for $t \in \{1, \ldots, T\}$ **do**
        for $i \in \{1, \ldots, n\}$ **do**
            update $\mathbf{w}_i^{(t)}$ as in (28);
            compute $\mathbf{A}_t$ and $\mathbf{b}_t$ as in (19);
            compute $\frac{\partial F}{\partial \lambda_i^{(t)}}$ as in (24);
        end
    end
    rearrange $\mathbf{w}_i$ for all $i = 1, \ldots, n$ to construct $\mathbf{W}$;
    rearrange $\frac{\partial F}{\partial \lambda_i^{(t)}}$ for all $i = 1, \ldots, n$ to construct $\nabla_{\boldsymbol{\lambda}} F$;
    update $\boldsymbol{\lambda}$ as in (29);
    $iter += 1$;
end

Referring to (25), we use the MU rule specified in (13) for updating $\mathbf{H}$. For the bi-level formulation, we keep the KL divergence with the ℓ_1 norm as a loss function:

$$\mathscr{L}_{\lambda_i} : \mathbb{R}^r \to \mathbb{R} : \mathbf{w}_i \mapsto \sum_{j=1}^{m} d_1\left(\mathbf{x}_j, \sum_{k=1}^{r} w_{ik} h_{kj}\right) + \lambda_i ||\mathbf{w}_i||_1, \qquad (26)$$

whereas the KL divergence is the error function of the outer problem:

$$\mathscr{E} : \mathbb{R}^r \times \Lambda \to \mathbb{R} : (\mathbf{w}_i, \lambda_i) \mapsto \sum_{j=1}^{m} d_1\left(\mathbf{x}_j, \sum_{k=1}^{r} w_{ik}(\lambda_i) h_{kj}\right). \qquad (27)$$

To assess the theoretical results for the previous functions, Hypothesis 1 needs to be verified. Observe that $\mathscr{E}$ is jointly continuous with respect to $\mathbf{w}_i$ and λ_i. Similarly, d_1 and the map $(\mathbf{w}_i, \lambda_i) \mapsto \mathscr{L}_i(\mathbf{w}_i)$. From the convexity and the compactness of Λ, $\arg\min \mathscr{L}_{\lambda_i}$ is a singleton for any λ_i. Finally, $\mathbf{w}_{i(\lambda_i)} = \arg\min \mathscr{L}_{\lambda_i}$ remains bounded as λ_i varies in Λ, in fact:

$$||\mathbf{w}_i(\lambda_i)|| \leq M \quad \forall \lambda_i \in \Lambda \quad \text{with} \quad M > 0, \quad M \leq M^*,$$

being $M^* = \max\{||\mathbf{w}_i(\lambda_i)||_2^2, \quad \lambda_i \in \Lambda\}$.

Matrix $\mathbf{W}$ is updated using the following novel rule by rows $\Phi : \mathbb{R}^r \times \Lambda \to \mathbb{R}^r$ s.t. $(\mathbf{w}_i^{(t-1)}, \lambda_i) \mapsto \mathbf{w}_i^{(t)}$ (this update can be similarly derived as in [43, 55]), then for $k = 1, \ldots, r$ and $i = 1, \ldots, n$

$$w_{ik}^{(t)} = w_{ik}^{(t-1)} \frac{\sum\limits_{j=1}^{m} h_{kj}(x_{ij} / \sum\limits_{a=1}^{r} w_{ia} h_{aj})}{\sum\limits_{j=1}^{m} h_{kj} + \lambda_i}. \tag{28}$$

Its proof is detailed in Appendix A.

Vector $\boldsymbol{\lambda}$ is also updated by the steepest descent procedure:

$$\boldsymbol{\lambda} = \boldsymbol{\lambda} - c\nabla_{\boldsymbol{\lambda}} F(\boldsymbol{\lambda}), \tag{29}$$

with stepsize $c = \frac{1}{iter}$[7].

Each component of the hypergradient in (23) can be expressed with

$$(\mathbf{g}_T{}^\top)_k = -\sum_{j=1}^{m} \left(\frac{x_{ij}}{\sum\limits_{a=1}^{r} w_{ia}^{(T)} h_{aj}} h_{kj} + h_{kj} \right) \quad \text{for } k = 1, \ldots, r,$$

while $\mathbf{A}_t$ and $\mathbf{b}_t$ for $t = 1, \ldots, T$ are given by:

$$(A_{kl})_t = \begin{cases} \dfrac{\sum\limits_{j=1}^{m} h_{kj} \cdot (x_{ij} / \sum\limits_{a=1}^{r} w_{ia}^{(t-1)} \cdot h_{aj}) - w_{ik}^{(t-1)} \cdot \sum\limits_{j=1}^{m} h_{kj}^2 \cdot (x_{ij} / (\sum\limits_{a=1}^{r} w_{ia}^{(t-1)} \cdot h_{aj})^2)}{\sum\limits_{j=1}^{m} h_{kj} + \lambda_i} & \text{if } l = k, \\[4mm] -w_{ik}^{(t-1)} \cdot \dfrac{\sum\limits_{j=1}^{m} h_{kj} \cdot (x_{ij} / (\sum\limits_{a=1}^{r} w_{ia}^{(t-1)} \cdot h_{aj})^2) \cdot h_{lj}}{\sum\limits_{j=1}^{m} h_{kj} + \lambda_i} & \text{if } l \neq k; \end{cases}$$

$$(b_k)_t = -w_{ik}^{(t-1)} \cdot \frac{\sum\limits_{j=1}^{m} h_{kj} \cdot (x_{ij} / (\sum\limits_{a=1}^{r} w_{ia}^{(t-1)} \cdot h_{aj}))}{(\sum\limits_{j=1}^{m} h_{kj} + \lambda_i)^2} \quad \text{for } k = 1, \ldots, r.$$

---

[7]The usual conditions on the stepsize are fulfilled: $\sum\limits_{s=1}^{MaxIter} c_s = \infty$ and $\sum\limits_{s=1}^{MaxIter} c_s^2 < \infty$.

Although we focused on a specific objective function and its associated update rules, AltBi can be generalized for any β-divergence and penalty functions $\mathscr{R}$, respecting the assumptions in Section 2.

Remark 1. *The computational complexity of rule (13) amounts to $\mathcal{O}(Kmnr)$, where K is the number of iterations. Update rules (28) and (19) are more expensive due to the use of the bunch and require $\mathcal{O}(KTmnr)$. The complexity of other rules in Algorithm 1 is lower, which implies $\mathcal{O}(KTmnr)$ for the whole algorithm. Note that the complexity of the proposed algorithm is larger with respect to the standard multiplicative update rules in NMF only by factor T.*

4. Numerical Experiments

This section illustrates the numerical results obtained using the AltBi algorithm on two synthetic and two real datasets. It was implemented in MATLAB 2021a environment, and numerical experiments were executed on the i7 octa-core, 16GB RAM machine. The benchmarks[8] used in the experiments are generated according to the model[9] $\mathbf{X} \approx \mathbf{Y} = \mathbf{WH}$.

The datasets used are described in the following:

A) Factor matrices were generated randomly as full rank uniformly distributed matrices. Matrix $\mathbf{H} \in \mathbb{R}_+^{r \times m}$ was generated using the MATLAB command `rand`, while $\mathbf{W} \in \mathbb{R}_+^{n \times r}$ was generated using the command `randn` to obtain sparse columns. Negative entries were replaced with a zero-value.

B) Each column in $\mathbf{W}$ is expressed as a sinusoidal wave signal with the frequency and the phase set individually for each component/column. The example of this signal waveform is plotted in Figure 1a. The negative entries are replaced with a zero-value. Factor matrix $\mathbf{H}$ was randomly generated as a full rank sparse matrix with sparseness level α_H adjusted by the user.

C) The source signals from the file `AC10_art_spectr_noi` of MATLAB toolbox NMFLAB for Signal Processing [56] have been used. These signals form matrix $\mathbf{W} \in \mathbb{R}_+^{n \times r}$. Exemplary five signals for $n = 1000$ are plotted in Figure 1b. Also, in this case, $\mathbf{H}$ was generated as a sparse matrix with α_H fixed sparsity level.

D) Real reflectance signals taken from the U.S. Geological Survey (USGS) database have been used as endmembers to generate the mixtures modelling real hyperspectral imaging data. Using the NMF model, the aim is to perform hyperspectral unmixing to obtain spectral components and their corresponding proportion maps called abundances. In our approach,

---

[8]https://github.com/flaespo/Dataset_signal_HPO

[9]Noiseless dataset $\mathbf{Y} \in \mathbb{R}^{n \times m}$ was constructed. Since our goal is to solve the identification problem, it is unnecessary to perturb matrix $\mathbf{Y}$. In this way, we preserve initial sparsity.

the column vectors of $\mathbf{W}$ contain the spectral signatures (endmembers) (Figure 1d), and $\mathbf{H}$ represents the mixing matrix or vectorized abundance maps (Figure 1c). The spectral signals are divided into 224 bands covering the range of wavelengths from 400 nm to 2.5 μm. The angle between any pair of the signals is greater than 15 degrees. These signals form matrix $\mathbf{W} \in \mathbb{R}_+^{n \times r}$, where $n = 224$. The rank of factorization r determines the number of endmembers.
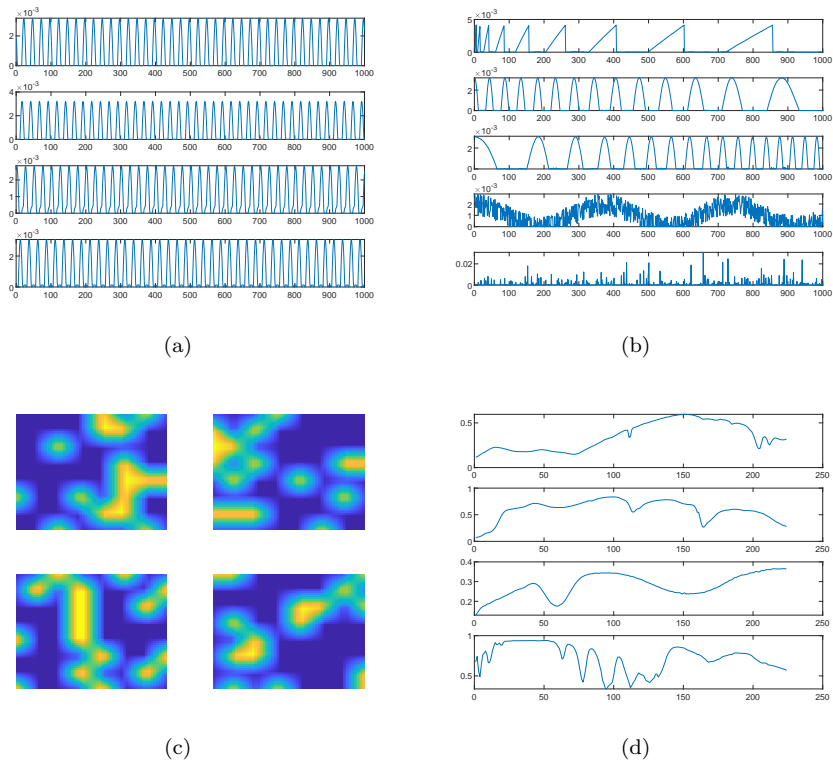

Figure 1: Waveform of signals in benchmark B (a), and benchmark C (b), Abundance maps (c) and Spectral signatures (d) of benchmark D.

Three NMF algorithms were used and compared: AltBi, the standard unpenalized MU in [50], and the standard penalized that alternates rule (13) and the modified version of (A.2) in which $\lambda_i = 0.5 \quad \forall i = 1, \ldots, n$ (referred to P-MU). The same random initializer generated from a uniform distribution starts all the algorithms [57]. The efficiency of the methods is analyzed by performing 30 Monte Carlo (MC) runs for the NMF algorithms, where for each run, initial matrices $\mathbf{W}$ and $\mathbf{H}$ are different. At the beginning of the process, initial $\boldsymbol{\lambda}$ is chosen to have homogeneity between the terms characterizing the objective function, according to:

$$\lambda_i = \frac{\sum\limits_{j=1}^{m} d_1\left(\mathbf{x}_j, \sum\limits_{k=1}^{r} w_{ik}h_{kj}\right)}{10 \cdot \digamma(\mathbf{w}_i)} \qquad \text{for} \quad i = 1, \ldots, n;$$

where $\digamma$ is the ℓ_1 penalty norm in this particular experimental case. The maximum number of iterations for all the algorithms was set to 1000, the tolerance for early termination to 10^{-6}, and the number of inner iterations (length of the bunch) to 4, i.e., $T = 4$. The following tests were performed:

1) Benchmark A was used with $n = 1000$, $m = 50$, $r = 4$.

2) Benchmark B was used with $n = 1000$, $m = 50$, $r = 4$, $\alpha_H = 0.1$.

3) Benchmark C was used with $n = 1000$, $m = 50$, $r = 5$, $\alpha_H = 0.1$.

4) Benchmark D was used with $n = 224$, $m = 3025$, $r = 5$.

In all the tests, no noisy perturbations were used.

To evaluate the goodness of the approximation and the effectiveness of the minimization process, we report the relative error[10] and the evolution of the objective function with respect to iterations for benchmark A in Figure 2. All other benchmarks present similar results as reported in Section 4.1.

Figure 2: (*a*) Relative error, (*b*) evolution of objective function with respect to iterations (Benchmarks A).

The performance of the NMF algorithms was evaluated with the Signal-to-Interference Ratio (SIR) measure [58] between the estimated signals and the true ones. Figure 3 shows the SIR statistics (in dB) for assessing the columns in $\mathbf{W}$ and the rows in $\mathbf{H}$ for benchmark A. Table 1 reports the numerical results of Mean-SIR in estimating $\mathbf{W}$ and $\mathbf{H}$ for benchmark A.

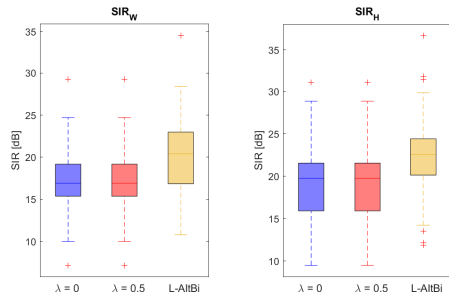[10]In this case, we compute the relative error as $D_1(\mathbf{X}, \mathbf{WH})/\sum_{i,j} x_{ij}log(x_{ij})$ [21].

16

Figure 3: SIR statistics for estimating columns of **W** and rows **H** (Benchmark A).

Table 1: Mean-SIR [dB] for estimating matrices **W** and **H**.

	MU	P-MU	AltBi
SIR for **W**	16.7325	16.7325	21.3388
SIR for **H**	19.2147	19.2147	23.3308

The general structure of the optimized $\boldsymbol{\lambda}$ has also been inspected. Figure 4 compares final and initial HPs for benchmark A: pointwise and distribution of vector $\boldsymbol{\lambda}$, in Figures 4a and 4b, respectively. The peak of the distribution of initial HPs shifts its location from a positive towards the zero value. Thus, the optimized $\boldsymbol{\lambda}$ is a sparse vector, suggesting the algorithm prefers to penalize the selected rows of **W** rather than all. Finally, the numerical results are also compared to evaluate the sparsity of **W** and **H** by $\text{Sp}(\mathbf{A})^{11} = 100 \cdot \frac{(1 - \#(\mathbf{A} > \tau))}{\#\mathbf{A}}$, for $\mathbf{A} \in \mathbb{R}^{n \times m}$. The sparsity constraint was added only on **W**, and for benchmarks C and D, the user provided the sparsity on **H**. As shown in Figure 5, the proposed method enforces the sparsity on **W** and does not affect the sparsity profile in **H**, as expected.

Please observe that optimal $\boldsymbol{\lambda}$ obtained from AltBi gives the best results either for identification and fitting problems and its choice is automatic. Figure 6 depicts the behavior of the response function for fixed values of λ in the P-MU algorithm compared with the non-penalized MU and AltBi. AltBi shows the best performance.

4.1. Results for benchmarks B, C, and D

All the experiments confirmed the expected behavior of AltBi in terms of the identification problem. Figures 7c, 8c, and 9a show that the SIR values obtained with AltBi are better than those obtained with MU and P-MU. Moreover,

---

[11]$\text{Sp}(\mathbf{A})$ represents the ratio between the complement of the number of elements greater than a certain threshold and the total number of elements in matrix **A**.
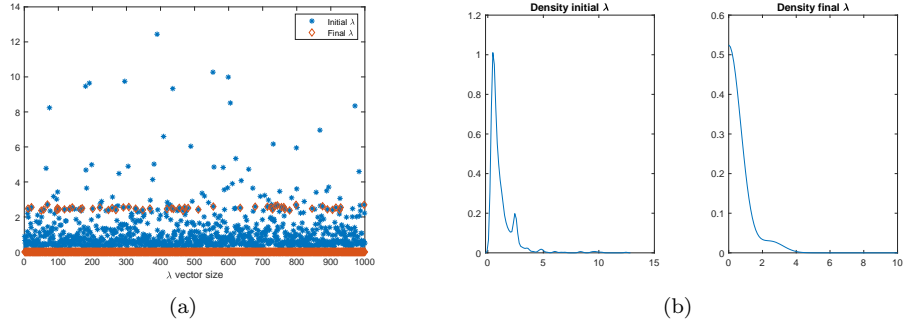
(a)   (b)

Figure 4: Initial $\boldsymbol{\lambda}$ compared with final $\boldsymbol{\lambda}$: vector components (a); density plot of $\boldsymbol{\lambda}$ vector (b) (Benchmark A).
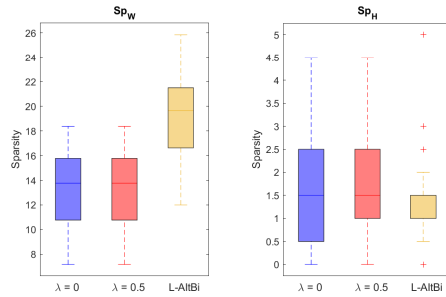

Figure 5: Statistics of the sparseness measure for $\mathbf{W}$ and $\mathbf{H}$ (Benchmark A).

for benchmark D, we show the original abundance maps (10a) and spectral signatures (10c) compared to the estimated abundance maps (10b) and spectral signatures (10d). Similar results are obtained for the relative error and the objective function in benchmark D, which we omit for brevity. The abundance maps are estimated with lower SIR performance than the spectral signatures (matrix $\mathbf{W}$). This result is not surprising: no penalty is imposed on $\mathbf{H}$. The sparsity-enforcing term was considered only for estimating matrix $\mathbf{W}$.

5. Conclusions

We proposed the alternating HPO procedure for NMF problems which incorporates the penalty HPs into the optimization problem with the bi-level mode. We proved the existence and convergence results for the solution of the considered task and provided promising numerical experiments and comparisons.

HPO in an unsupervised scenario of data matrix factorization represents an evolving topic. However, when the size of the problem increases, the computational cost required by AltBi could not make this algorithm very competitive. To improve the computational efficiency, a column-wise version of AltBi is under
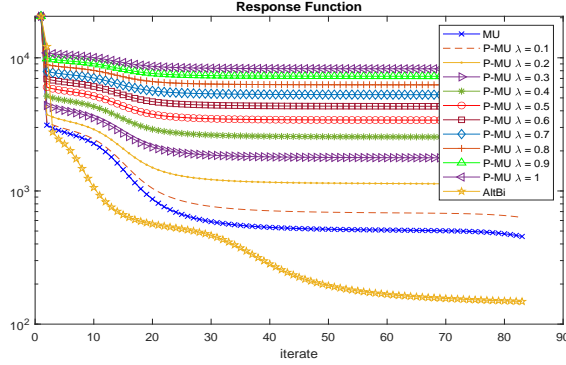
Figure 6: Response functions obtained through the P-MU algorithm with different λ values in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ compared with the unpenalized MU case and AltBi.

study with the possibility of speeding up the algorithm by varying the length of the bunch to make the local truncation error approximately constant.

The extension of the theoretical results under hypotheses $(1) - (6)$ with no convex error and loss functions could also be considered. These aspects could accomplish this evolving topic together with the analysis of the effects made by different choices of the penalty functions on performance and computational issues for large dataset applications (such as gene expression analysis [21, 59], blind spectral unmixing [39, 60], and text mining).

Appendix A. Convergence and Correctness for the W update in (28)

Without loss of generality, the function in (25) can be rewritten neglecting constants which are not relevant to the minimization process. Thus:

$$\sum_{i,j} \left(-x_{ij} log \left(\sum_{k=1}^{r} w_{ik} h_{kj} \right) + \sum_{k=1}^{r} w_{ik} h_{kj} \right) + \sum_{i,j} \lambda_i w_{ij}. \qquad \text{(A.1)}$$

In particular, we theorize its element-wise update rules as:

$$w_{ia} \leftarrow w_{ia} \frac{\sum_{j=1}^{m} \left(h_{aj} x_{ij} / \sum_{k=1}^{r} w_{ik} h_{kj} \right)}{\sum_{j=1}^{m} h_{aj} + \lambda_i}, \quad \text{for } i = 1, \ldots, n \text{ and } a = 1, \ldots, r. \quad \text{(A.2)}$$

Fixing the i-th row, let $\mathbf{w}_i \in \mathbb{R}^r$ and $\mathbf{x}_i \in \mathbb{R}^m$ be the i-th rows of $\mathbf{W}$ and $\mathbf{X}$, respectively, the function in (A.1) can be rewritten with respect to unknown $\mathbf{w}_i$ as

19

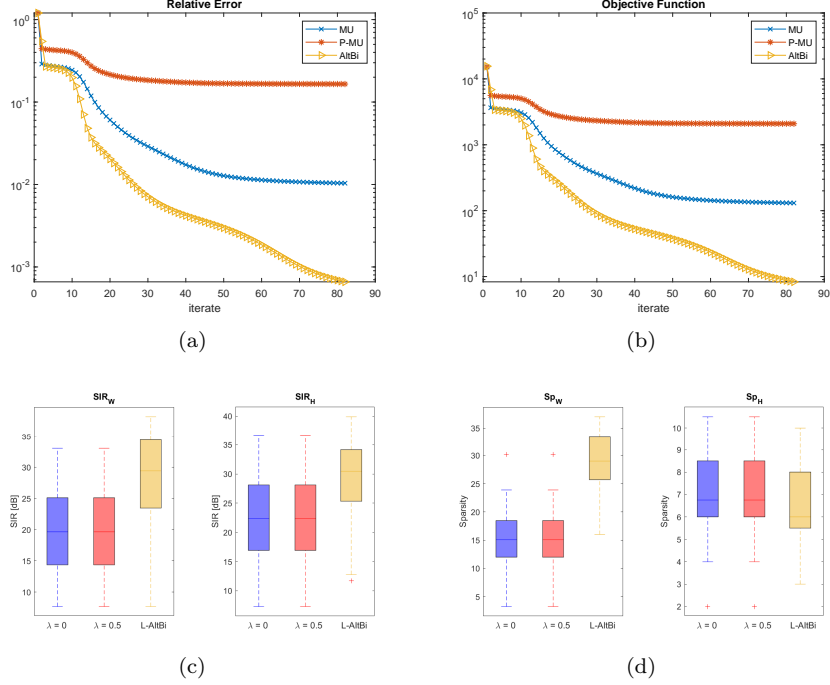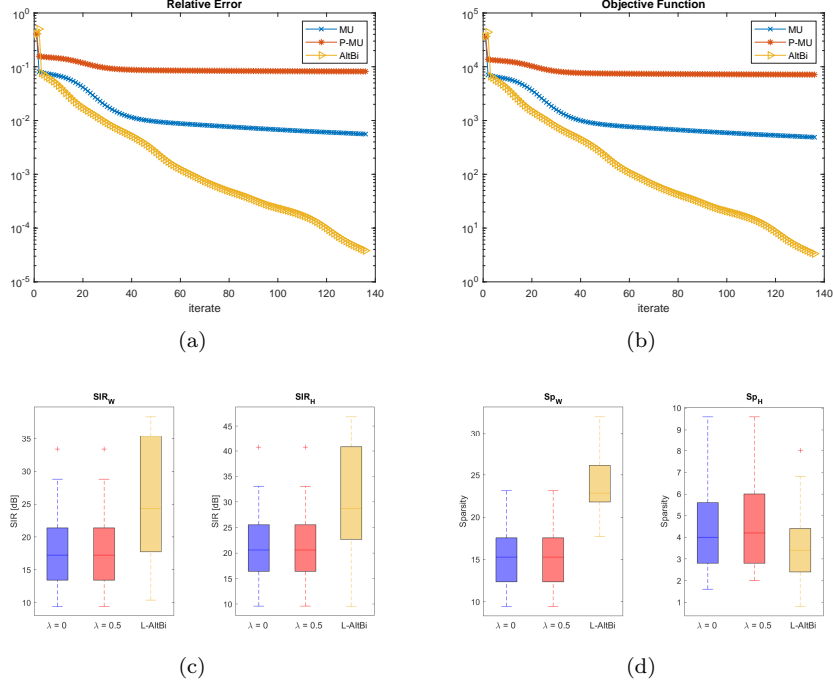Figure 7: (a) Relative error and (b) evolution of objective function with respect to iterations; (c) SIR statistics for estimating the columns of **W** and the rows of **H**; (d) Statistics of the sparseness measure in Benchmark B.

$$\mathscr{F}(\mathbf{w}_i) = \sum_{j=1}^{m} -x_{ij} \log \left(\sum_{a=1}^{r} w_{ia} h_{aj} \right) + \sum_{j=1}^{m} \sum_{a=1}^{r} w_{ia} h_{aj} + \lambda_i \sum_{a=1}^{r} w_{ia}, \qquad \text{(A.3)}$$

then the updates for unknown $\mathbf{w}_i$ follow from Theorem Appendix A.1.

Theorem Appendix A.1. *The divergence in (A.3) is non-increasing under update rules (A.2). The divergence is invariant under these updates if and only if $\mathbf{w}_i$ is a stationary point of the divergence.*

The following proof proceeds the demonstration scheme proposed by Lee and Seung [61] and Liu et al [55], but it adopts a different and more general formulation of the auxiliary function for objective function (A.3).

Lemma Appendix A.2. $\mathscr{G}(\mathbf{w}_i, \mathbf{w}_i^t) = \sum\limits_{j=1}^{m} \sum\limits_{a=1}^{r} w_{ia} h_{aj}$

20

Figure 8: (*a*) Relative error and (*b*) evolution of objective function with respect to iterations; (*c*) SIR statistics for estimating the columns of **W** and the rows of **H**; (*d*) Statistics of the sparseness measure in Benchmark C.

$$-\sum_{j=1}^{m}\sum_{a=1}^{r} x_{ij} \frac{w_{ia}^{t} h_{aj}}{\sum_{b=1}^{r} w_{ib}^{t} h_{bj}} \left(log\left(w_{ia}h_{aj}\right) - log\left(\frac{w_{ia}^{t} h_{aj}}{\sum_{b=1}^{r} w_{ib}^{t} h_{bj}} \right) \right) + \lambda_{i}\sum_{a=1}^{r} w_{ia}$$

is an auxiliary function for $\mathscr{F}(\mathbf{w}_i)$.

Proof. We prove that $\mathscr{G}(\mathbf{w}_i, \mathbf{w}_i^t)$ is an auxiliary function for $\mathscr{F}(\mathbf{w}_i)$. Due to the basic proprieties of the logarithmic function, the condition $\mathscr{G}(\mathbf{w}_i, \mathbf{w}_i) = \mathscr{F}(\mathbf{w}_i)$ is straightforward. To prove that $\mathscr{G}(\mathbf{w}_i, \mathbf{w}_i^t) \geq \mathscr{F}(\mathbf{w}_i)$, we consider the quantity

$$\alpha_{aj} = \frac{w_{ia}^{t} h_{aj}}{\sum_{b} w_{ib}^{t} h_{bj}} \quad with \quad \sum_{j}\sum_{a}\alpha_{aj} = 1. \tag{A.4}$$

Due to the convexity of the logarithmic function, the inequality

$$\sum_{j} x_{ij} \log \sum_{a} w_{ia}h_{aj} - \sum_{j}\sum_{a} x_{ij}\alpha_{aj} \log\left(\frac{w_{ia}H_{aj}}{\alpha_{aj}} \right) \geq 0 \tag{A.5}$$
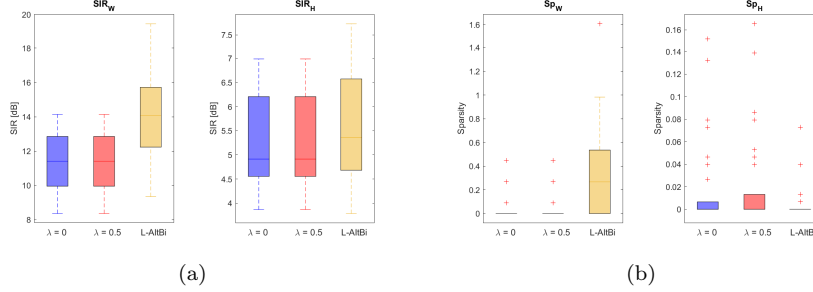
21

Figure 9: SIR statistics for estimating the columns of matrix $\mathbf{W}$ (spectral signatures) and the rows of matrix $\mathbf{H}$ (a); Statistics of the sparseness measure (b) in Benchmark D.

holds, so that the proof follows. $\qquad\square$

Lemma Appendix A.3. *Objective function $\mathscr{F}$ is non-increasing when its auxiliary function is minimized.*

Proof. The minimum value of $\mathscr{G}(\mathbf{w}_i, \mathbf{w}_i^t)$ with respect to $\mathbf{w}_i$ satisfies

$$\frac{d\mathscr{G}(\mathbf{w}_i, \mathbf{w}_i^t)}{dw_{ia}} = \sum_j h_{aj} - \sum_j x_{ij} \frac{w_{ia}^t h_{aj}}{\sum_b w_{ib}^t h_{bj}} \left(\frac{1}{h_{aj}} \right) + \lambda_i = 0. \qquad (A.6)$$

Thus, the update rule is (A.2). $\qquad\square$

According to this new update, the KKT conditions with respect to the non-negative constraints are:

$$\begin{cases} \mathbf{W}. * \nabla_{\mathbf{W}}\mathscr{F}(\mathbf{W}, \mathbf{H}) = 0, \\ \nabla_{\mathbf{W}}\mathscr{F}(\mathbf{W}, \mathbf{H}) \geq 0, \\ \mathbf{W} \geq 0, \end{cases} \qquad (A.7)$$

where $.*$ is the Hadamard pointwise product and $\nabla_{\mathbf{W}}$ is the gradient of (A.1). This formulation allows to prove that update (A.2) satisfies KKT conditions (A.7) at the convergence, then its correctness is ensured.

(a) Original abundance maps   (b) Estimated abundance maps

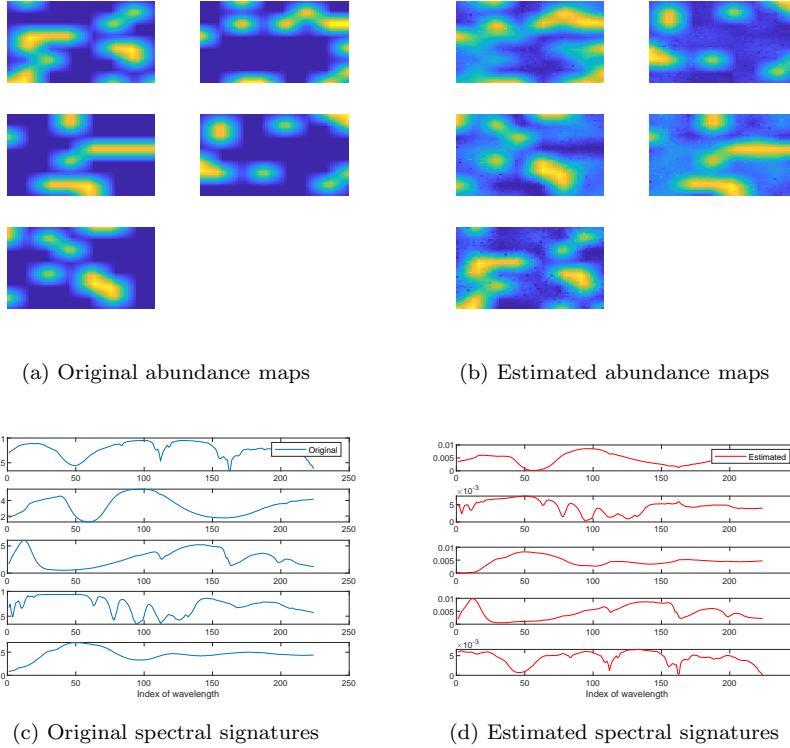(c) Original spectral signatures   (d) Estimated spectral signatures

Figure 10: Abundance maps: (*a*) original, (*b*) estimated with AltBi. Spectral signatures: (*c*) original, (*d*) estimated (in Benchmark D)

References

[1] S. Falkner, A. Klein, F. Hutter, Bohb: Robust and efficient hyperparameter optimization at scale, in: ICML, PMLR, 2018, pp. 1437–1446.

[2] J. Bergstra, D. Yamins, D. Cox, Making a science of model search: Hpo in hundreds of dimensions for vision architectures, in: ICML, 2013, pp. 115–123.

[3] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J Mac. Learn. Res. 13 (10) (2012) 281–305.

[4] C. D. Francescomarino, M. Dumas, M. Federici, C. Ghidini, F. M. Maggi, W. Rizzi, L. Simonetto, Genetic algorithms for hyperparameter optimization in predictive business process monitoring, Inf. Syst. 74 (2018) 67–83.

[5] J. S. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyperparameter optimization, in: Advances in neural information processing systems, 2011, pp. 2546–2554.

[6] D. Marinov, D. Karapetyan, Hyperparameter optimisation with early termination of poor performers, 2019 11th Computer Science and Electronic Engineering (CEEC) (2019) 160–163.

[7] H. Alibrahim, S. A. Ludwig, Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization, in: 2021 IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 1551–1559. `doi:10.1109/CEC45853.2021.9504761`.

[8] G. Sui, Y. Yu, Bayesian contextual bandits for hyper parameter optimization, IEEE Access 8 (2020) 42971–42979. `doi:10.1109/ACCESS.2020.2977129`.

[9] N. Del Buono, F. Esposito, L. Selicato, Methods for hyperparameters optimization in learning approaches: An overview, in: International Conference on Machine Learning, Optimization, and Data Science, Springer, 2020, pp. 100–112.

[10] Y. Bengio, Gradient-based optimization of hyperparameters, Neural Computation 12 (8) (2000) 1889–1900. `doi:10.1162/089976600300015187`.

[11] L. Bottou, Online algorithms and stochastic approximations, Online learn. neur. net. (1998).

[12] L. Bottou, Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT'2010, Springer, 2010, pp. 177–186.

[13] D. Maclaurin, D. Duvenaud, R. Adams, Gradient-based hyperparameter optimization through reversible learning, in: Proc. of ICML, 2015, pp. 2113–2122.

[14] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, M. Pontil, Bilevel programming for hyperparameter optimization and meta-learning, in: ICML, PMLR, 2018, pp. 1568–1577.

[15] F. Pedregosa, Hyperparameter optimization with approximate gradient, in: ICML, PMLR, 2016, pp. 737–746.

[16] N. Del Buono, F. Esposito, L. Selicato, Toward a new approach for tuning regularization hyperparameter in nmf, in: International Conference on Machine Learning, Optimization, and Data Science, Springer, 2021, pp. 500–511.

[17] J. F. Bard, Practical bilevel optimization: algorithms and applications, Vol. 30, Springer Science & Business Media, 2013.

[18] L. Franceschi, M. Donini, P. Frasconi, M. Pontil, Forward and reverse gradient-based hyperparameter optimization, in: ICML, PMLR, 2017, pp. 1165–1173.

[19] A. Cichocki, R. Zdunek, Multilayer nonnegative matrix factorization using projected gradient approaches, Int J Neu Sys 17 (06) (2007) 431–446.

[20] D. Chu, W. Shi, S. Eswar, H. Park, An alternating rank-k nonnegative least squares framework (arknls) for nonnegative matrix factorization, SIAM Journal on Matrix Analysis and Applications 42 (4) (2021) 1451–1479.

[21] F. Esposito, N. Gillis, N. Del Buono, Orthogonal joint sparse NMF for microarray data analysis, J. Math. Biol. 79 (1) (2019) 223–247.

[22] N. Gillis, Nonnegative Matrix Factorization, SIAM, 2020.

[23] H. Kim, H. Park, Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis, Bioinformatics 23 (12) (2007) 1495–1502.

[24] H. Kim, H. Park, Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method, SIAM journal on matrix analysis and applications 30 (2) (2008) 713–730.

[25] C.-J. Lin, Projected gradient methods for nonnegative matrix factorization, Neural computation 19 (10) (2007) 2756–2779.

[26] J.-X. Liu, D. Wang, Y.-L. Gao, C.-H. Zheng, Y. Xu, J. Yu, Regularized non-negative matrix factorization for identifying differentially expressed genes and clustering samples: a survey, IEEE/ACM Trans Comp Biol Bioinfor 15 (3) (2017) 974–987.

[27] M. Merritt, Y. Zhang, Interior-point gradient method for large-scale totally nonnegative least squares problems, J Opt Th Appl 126 (1) (2005) 191–202.

[28] C.-H. Zheng, D.-S. Huang, L. Zhang, X.-Z. Kong, Tumor clustering using nonnegative matrix factorization with gene selection, IEEE Trans. Inf. Technol. Biomed 13 (4) (2009) 599–607.

[29] T. Gao, Y. Guo, C. Deng, S. Wang, Q. Yu, Hyperspectral Unmixing based on Constrained Nonnegative Matrix Factorization via Approximate L0, in: Proc. IEEE Int. Geoscience Remote Sens. Symp, 2015, pp. 2156–2159.

[30] Z. Zhang, Y. Xu, J. Yang, X. Li, D. Zhang, A survey of sparse representation: algorithms and applications, IEEE access 3 (2015) 490–530.

[31] R. Tibshirani, Regression shrinkage and selection via the lasso, J Roy. Stat. Soc. B (1996) 267–288.

[32] D. Kong, C. Ding, H. Huang, Robust nonnegative matrix factorization using l21-norm, in: Proc 20th ACM-CIKM, 2011, pp. 673–682.

[33] Z. Li, Z. Tang, S. Ding, Dictionary learning by nonnegative matrix factorization with 1/2-norm sparsity constraint, in: Proc. IEEE-CYBCON, IEEE, 2013, pp. 63–67.

[34] F. Nie, H. Huang, X. Cai, C. H. Ding, Efficient and robust feature selection via joint ℓ 2, 1-norms minimization, in: Adv. Neural Inf. Process. Syst., 2010, pp. 1813–1821.

[35] P. O. Hoyer, Non-negative matrix factorization with sparseness constraints, J. Mach. Learn. Research 5 (Nov) (2004) 1457–1469.

[36] H. Kim, H. Park, Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis, Bioinformatics 23 (12) (2007) 1495–1502.

[37] P. C. Hansen, Analysis of discrete ill-posed problems by means of the l-curve, SIAM review 34 (4) (1992) 561–580.

[38] P. C. Hansen, D. P. O'Leary, The use of the l-curve in the regularization of discrete ill-posed problems, SIAM SISC 14 (6) (1993) 1487–1503.

[39] R. Zdunek, Regularized nonnegative matrix factorization: Geometrical interpretation and application to spectral unmixing, Int. J. Appl. Math. Comp. Science 24 (2) (2014) 233–247.

[40] R. Zdunek, A. Cichocki, Nonnegative matrix factorization with constrained second-order optimization, Signal Process 87 (8) (2007) 1904–1916.

[41] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, R. J. Plemmons, Algorithms and applications for approximate nonnegative matrix factorization, Computational statistics & data analysis 52 (1) (2007) 155–173.

[42] Y.-X. Wang, Y.-J. Zhang, Nonnegative matrix factorization: A comprehensive review, IEEE Tran. Knowl. Data Eng. 25 (6) (2013) 1336–1353.

[43] D. Lee, H. S. Seung, Algorithms for non-negative matrix factorization, in: Proc. 13th NIPS, 2000, pp. 100–112.

[44] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the em algorithm, J. Royal Stat. Soc. B (1977) 1–38.

[45] K. Lange, R. Carson, et al., Em reconstruction algorithms for emission and transmission tomography, J Comput Assist Tomogr 8 (2) (1984) 306–16.

[46] L. B. Lucy, An iterative technique for the rectification of observed distributions, The astronomical journal 79 (1974) 745.

[47] W. H. Richardson, Bayesian-based iterative method of image restoration, JoSA 62 (1) (1972) 55–59.

[48] L. Saul, F. Pereira, Aggregate and mixed-order markov models for statistical language processing, arXiv preprint cmp-lg/9706007 (1997).

[49] C. Févotte, J. Idier, Algorithms for nmf with the β-divergence, Neur. Comput. 23 (9) (2011) 2421–2456.

[50] D. D. Lee, H. S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788.

[51] R. Kompass, A generalized divergence measure for nonnegative matrix factorization, Neural Comp 19 (3) (2007) 780–791.

[52] C. Févotte, N. Bertin, J.-L. Durrieu, Nmf with the itakura-saito divergence: With application to music analysis, Neur. Comput. 21 (3) (2009) 793–830.

[53] L. Franceschi, A unified framework for gradient-based hyperparameter optimization and meta-learning, Ph.D. thesis, UCL (University College London) (2021).

[54] A. L. Dontchev, T. Zolezzi, Well-posed optimization problems, Springer, 2006.

[55] W. Liu, N. Zheng, X. Lu, Non-negative matrix factorization for visual coding, in: Proc of ICASSP'03, Vol. 3, IEEE, 2003, pp. III–293.

[56] A. Cichocki, R. Zdunek, Nmflab for signal processing toolbox for (01 2006).

[57] F. Esposito, A review on initialization methods for nonnegative matrix factorization: towards omics data experiments, Mathematics 9 (9) (2021) 1006.

[58] A. Cichocki, R. Zdunek, A. H. Phan, S.-i. Amari, Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation, John Wiley & Sons, 2009.

[59] L. Taslaman, B. Nilsson, A framework for regularized non-negative matrix factorization, with application to the analysis of gene expression data, PloS one 7 (11) (2012) e46331.

[60] V. Leplat, N. Gillis, C. Févotte, Multi-resolution beta-divergence nmf for blind spectral unmixing, arXiv preprint arXiv:2007.03893 (2020).

[61] D. Seung, L. Lee, Algorithms for non-negative matrix factorization, Adv. Neural Inf. Process. Syst. 13 (2001) 556–562.