Acta Cryst A FOUNDATIONS ADVANCES

ISSN 2053-2733

A fast two-stage algorithm for non-negative matrix factorization in smoothly varying data

Ran Gu,[a]* Simon J. L. Billinge[b,c] and Qiang Du[b,d]

[a]School of Statistics and Data Science, KLMDASR, LEBPS and LPMC, Nankai University, Tianjin 300071, People's Republic of China, [b]Department of Applied Physics and Applied Mathematics, Fu Foundation School of Engineering and Applied Sciences, Columbia University, New York, NY 10027, USA, [c]Condensed Matter Physics and Materials Science Department, Brookhaven National Laboratory, Upton, NY 11973, USA, and [d]The Data Science Institute, Columbia University, New York, NY 10027, USA. *Correspondence e-mail: rgu@nankai.edu.cn

Received 17 October 2022
Accepted 28 January 2023

Edited by A. Altomare, Institute of Crystallography - CNR, Bari, Italy

Keywords: non-negative matrix factorization; smoothly varying data; pair distribution function; interior point method.

This article reports the study of algorithms for non-negative matrix factorization (NMF) in various applications involving smoothly varying data such as time or temperature series diffraction data on a dense grid of points. Utilizing the continual nature of the data, a fast two-stage algorithm is developed for highly efficient and accurate NMF. In the first stage, an alternating non-negative least-squares framework is used in combination with the active set method with a warm-start strategy for the solution of subproblems. In the second stage, an interior point method is adopted to accelerate the local convergence. The convergence of the proposed algorithm is proved. The new algorithm is compared with some existing algorithms in benchmark tests using both real-world data and synthetic data. The results demonstrate the advantage of the algorithm in finding high-precision solutions.

1. Introduction

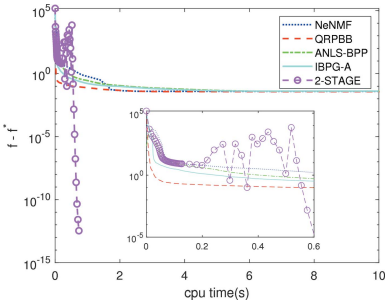
Non-negative matrix factorization (NMF) (Paatero & Tapper, 1994; Lee & Seung, 1999) refers to the factorization of a matrix approximately into the product of two non-negative matrices with low rank, $M \simeq XY$. It has become one of the most popular multi-dimensional data processing tools in various applications such as signal processing (Buciu, 2008), biomedical engineering (Sra & Dhillon, 2006), pattern recognition (Cichocki *et al.*, 2009), computer vision and image engineering (Buciu, 2008). More recently it has been applied in the realm of crystallography, for example, to *in situ* time-dependent diffraction measurements of synthesis (Liu *et al.*, 2021; Thatcher *et al.*, 2022) and spatially resolved electron diffraction maps (Rakita *et al.*, 2023).

Lee & Seung (1999) initiated the study of NMF and presented a method. Their method makes all decomposed components non-negative and achieves nonlinear dimension reduction at the same time. Developed by Lee & Seung (2001) for NMF, their multiplicative update rule has been a popular method due to the simplicity of its implementation.

A commonly used optimization formulation of $M \simeq XY$ is to use the square of Euclidean distance (SED) as the objective function, that is,

$$\begin{aligned} \min_{X\in\mathbb{R}^{n\times k},\,Y\in\mathbb{R}^{k\times m}} \quad & \tfrac{1}{2}\|XY - M\|_F^2, \\ \text{s.t.} \quad & X \geq 0, \\ & Y \geq 0. \end{aligned} \tag{1}$$

Many studies of NMF based on the above formulation have focused on the use of different optimization approaches like



the alternating non-negative least squares (ANLS) (Lin, 2007; Kim & Park, 2008; Guan *et al.*, 2012; Huang *et al.*, 2015), coordinate descent methods (Cichocki & Phan, 2009; Li & Zhang, 2009) and the alternating direction method of multipliers (ADMM) (Hajinezhad *et al.*, 2016). A comprehensive survey of various NMF models and many existing NMF algorithms can be found in the work of Wang & Zhang (2012). The NMF problem has been shown to be nonconvex and NP-hard (Vavasis, 2009). The algorithms studied in the literature can only guarantee the finding of a local minimum in general, rather than a global minimum of the cost function. Although Arora *et al.* (2016) presented a polynomial-time algorithm for constant k, the order of the complexity of the algorithm is too high to be applied in practice. Nevertheless, in many data mining applications, solving high-quality local minima is often desired with little time delay (Wang & Zhang, 2012; Gillis, 2020).

In this study, we are motivated by the application of NMF to problems involving continually generated smoothly varying data, which means data sampled progressively from smoothly varying processes. The data might be stored offline but they are assumed to have continuous distributions, such as those obtained from the real-time monitoring of reaction products in chemistry experiments and materials synthesis (Liu *et al.*, 2021; Todd *et al.*, 2020). For example, Zhao *et al.* (2011) measured X-ray diffraction data during the nucleation and growth of zeolite-supported Ag nanoparticles through reduction of Ag-exchanged mordenite (MOR), and then processed the data with pair distribution function (PDF) (Egami & Billinge, 2012) measurements. In the data, each PDF is approximately represented by a vector representation of n dimensions, which is recorded at m time instances in total. We should note the key features of these continuously distributed data: at any fixed time, the PDF is continuous in the distance variable; meanwhile, at any fixed distance in the PDF measurement, its value also has continuity in time; moreover, the spatially distributed data at later times are generated progressively following the data earlier in time. In the AgMOR data used by Zhao *et al.* (2011), the dimensions were $n = 3000$ and $m = 36$, where n is the length of each data point, and m is the number of measurements. It was anticipated that there are three materials present in the reaction, which means that $k = 3$. The focus of our study here is on smoothly varying data in the particular regime of $n \gg m \gg k$, with k being very small. This reflects the high dimensionality of data in an individual measurement (very large n) for systems with a relatively small number of components k.

Another difference of this work from the study of NMF in the field of machine learning is that in the applications considered above, we focus on high-precision solutions. Due to the nonconvexity of NMF, the algorithms mentioned above often converge to a local solution of the problem. Because the algorithm stops according to the stopping criterion, in general it does not get an exact local solution. Thus, a high-precision solution refers to the solution with an extremely small difference from the exact local solution. It usually has an extremely small deviation from the KKT (Karush–Kuhn–Tucker) optimization constraints and an objective function value that, when feasible, is barely higher than the objective function value of the exact local solution.

In the above example, the results obtained by NMF represent the PDF measurement data of several materials, which are often used for subsequent atomic structure reconstruction of materials (Juhás *et al.*, 2006; Jensen *et al.*, 2016; Gu *et al.*, 2019). In the process of atomic structure reconstruction, high-precision PDF data are needed, because each peak in the PDF represents the real atomic distance. Once there are errors in the data, they will cause difficulties in the subsequent scientific interpretation. Therefore, in this paper, we focus on methods to quickly solve for the high-precision solution of NMF when the data are smoothly varying.

Based on the motivation introduced above, our goal is to obtain high-precision local solutions for smoothly varying data with a relatively small scale, that is $n \leq 5000$, $m \leq 200$ and $k \leq 10$. Based on our observation, ANLS has a fast descent rate on the objective function at the first few iterations. Due to the continuity of smoothly varying data, we utilize the active set method in ANLS, so that the number of changes of the active set is very small and it takes a very small amount of CPU time to solve the subproblem. This allows us quickly to reach the vicinity of the local solution. However, the local convergence rate of ANLS is only proved to be sublinear in the nonconvex case (Beck & Tetruashvili, 2013), and in practice it cannot meet our needs for high-precision solutions. Therefore, we must use a second stage to accelerate the local convergence. For this point, we find the interior point method using Newton's method to solve the KKT system is a good choice. Previous researchers have never directly used the interior point method for (1) because the computational cost in each iteration is too large; however, in our application, the dimension of the problem has favorable properties such that the computational cost in each iteration is $O(nm^2k^3)$ in our algorithm design, which is completely acceptable. Thus, we propose to combine ANLS with the interior point method into a two-stage algorithm. In the first stage, an active set method is used for ANLS. In the second stage, a line search interior point method is adopted to reach a fast local convergence. It is worth mentioning that this paper is the first to utilize the second-order interior point method as an NMF algorithm and to apply it to practical problems. The proposed two-stage algorithm combines the advantages of the ANLS and interior point methods for the case of smoothly varying data.

The organization of this paper is as follows. Section 2 introduces the first stage of our proposed algorithm, which is based on the ANLS with an active set method for smoothly varying data. Section 3 proposes the second stage of the algorithm, which is a line search interior point method. Its convergence is also discussed. Section 4 gives the whole framework of our proposed two-stage algorithm to solve NMF in smoothly varying data. Section 5 shows the efficiency of our proposed algorithm by numerical tests. Section 6 concludes this paper with discussions on some future concerns.

2. ANLS framework and active set method

We first briefly review the ANLS framework for solving (1). In ANLS, variables are simply divided into two groups, which are then updated as outlined in Algorithm 1 (see Fig. 1).

Note that each subproblem can be split into a series of non-negative least-square problems:

$$\begin{array}{ll}\min\limits_{x} & \frac{1}{2}\|Cx-d\|_2^2, \\ \text{s.t.} & x \geq 0. \end{array} \tag{2}$$

For $C = X$, d corresponds to every column of M, while for $C = Y^{\mathrm{T}}$, d takes every row of M.

Although the original problem in (1) is nonconvex, the subproblems in Algorithm 1 are convex quadratic problems whose optimal solutions can be found in polynomial time. In addition, the convergence of Algorithm 1 to a stationary point of (1) has been proved (Grippo & Sciandrone, 2000).

On the basis of the ANLS framework, many algorithms for NMF have been developed, such as the active set method (Kim & Park, 2008), projection gradient method (Lin, 2007), projection Newton method (Gong & Zhang, 2012), projection quasi-Newton method (Kim *et al.*, 2007; Zdunek & Cichocki, 2006), Nesterov's gradient method (Guan *et al.*, 2012), and the method combined with Barzilai–Borwein step size (Huang *et al.*, 2015).

Different from others, the active set method solves the subproblem (2) exactly. The active set is defined as the set of indexes whose elements in the solution vector are zero. The active set method for solving the subproblem aims to find the correct active set $\mathcal{A}$ of the optimal solution x^*; then the value of the inactive set $\mathcal{I}$ of x^*, that is, the complement of $\mathcal{A}$, can be obtained from $x^*_{\mathcal{I}} = (C^{\mathrm{T}}_{\mathcal{I}}C_{\mathcal{I}})^{-1}C^{\mathrm{T}}_{\mathcal{I}}d$, and thereby x^* is solved. In the process of the active set method, the active set is constantly modified, and the above formula needs to be computed repeatedly. Considering that the subproblems need to solve for all d, and the active sets used in the computation are sometimes the same, multiple subproblems can be solved together in order to reduce computational cost. Kim & Park (2008) introduced an algorithm based on ANLS and the active set method. The constrained least-square problem in the matrix form is decomposed into several independent non-negative least-square problems with multiple right-hand side vectors. Later, Kim & Park (2011) proposed an active-set-like algorithm based on the ANLS framework. In their algorithm, the single least squares are solved by the block principal pivoting method, and the columns that have a common active set are grouped together to avoid redundant computation of Cholesky factorization in solving linear equations.

By noticing the continuity in smoothly varying data, when a single least-square problem (2) is solved, the same C and similar d may be used in the subsequent least square, resulting in similar solutions with likely the same active set. Therefore, we choose the active set method of Lawson & Hanson (1995) to solve the least-square problem (2), using the solution and its active set as the initial guess of the subsequent least square. This strategy is called the warm-start strategy or continuation technology, which is widely used in many algorithms and can reasonably improve the effectiveness (Liu *et al.*, 2016; Goldfarb & Ma, 2011). We do not need to do any regrouping, because the right-hand side vectors have been naturally grouped due to the continuity in the data, and we just need to solve them one by one sequentially. In fact, the active set of the solution in the subsequent least square usually has no or little change. When we solve the first set of equations in the new least-square subproblem, the Cholesky factorization performed in the previous step can be utilized to avoid redundant computation. Numerically, we see that the active set usually only changes on very few occasions or does not change at all for the smoothly varying data under consideration. Consequently, the first stage of our algorithm is chosen to be a combination of the ANLS framework, active set method and warm-start strategy, which builds upon the framework given by Algorithm 1, and uses the approximate solution already obtained previously as the initial value when solving (2) by the active set method.

Algorithm 1 Alternating nonnegative least squares (ANLS)
Repeat until stopping criteria are satisfied

$$\begin{array}{ll}\min\limits_{X\in\mathbb{R}^{n\times k}} & \frac{1}{2}\|XY-M\|_F^2, \\ s.t. & X \geq 0. \end{array}$$

$$\begin{array}{ll}\min\limits_{Y\in\mathbb{R}^{k\times m}} & \frac{1}{2}\|XY-M\|_F^2, \\ s.t. & Y \geq 0. \end{array}$$

end

Figure 1
Algorithm 1: alternating non-negative least squares (ANLS).

3. A line search interior point method for NMF

Interior point methods have proved to be successful for various nonlinear optimizations and for linear programming. In this section, we propose a line search interior point method for solving NMF. Its global convergence and computational cost are analyzed.

3.1. Algorithm

Given that the linear independence constraint qualification (LICQ) holds for the NMF problem, the KKT conditions (Kuhn & Tucker, 1951) for the problem can be written as

$$\begin{aligned} &(XY-M)Y^{\mathrm{T}} = R, \\ &X^{\mathrm{T}}(XY-M) = S, \\ &\langle R, X\rangle = 0, \\ &\langle S, Y\rangle = 0, \\ &X \geq 0, Y \geq 0, R \geq 0, S \geq 0. \end{aligned} \tag{3}$$

We denote $x = \mathrm{vec}(X^{\mathrm{T}})$, $y = \mathrm{vec}(Y)$, $r = \mathrm{vec}(R^{\mathrm{T}})$, $s = \mathrm{vec}(S)$, where vec transforms a matrix to a vector by expanding it by columns. Meanwhile, we define the inverse operations $\mathrm{mat}(x) = X^{\mathrm{T}}$, $\mathrm{mat}(y) = Y$, $\mathrm{mat}(r) = R^{\mathrm{T}}$, $\mathrm{mat}(s) = S$.

Applying Newton's method to the nonlinear system, in the variables x, y, r, s, we obtain

$$\begin{pmatrix} Q_1 & C^{\mathrm{T}} & -I & \\ C & Q_2 & & -I \\ \mathrm{Diag}(r) & & \mathrm{Diag}(x) & \\ & \mathrm{Diag}(s) & & \mathrm{Diag}(y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta r \\ \Delta s \end{pmatrix} = \begin{pmatrix} r - \mathrm{gra}X \\ s - \mathrm{gra}Y \\ \mu e - \mathrm{Diag}(r)x \\ \mu e - \mathrm{Diag}(s)y \end{pmatrix}, \tag{4}$$

with $\mu = 0$, where $\mathrm{gra}X = \mathrm{vec}(Y(XY - M)^{\mathrm{T}})$, $\mathrm{gra}Y = \mathrm{vec}(X^{\mathrm{T}}(XY - M))$, e is a vector of ones, $\mathrm{Diag}(x)$ constructs a diagonal matrix with the diagonal elements given by x,

$$Q_1 = \begin{pmatrix} YY^{\mathrm{T}} & & \\ & \ddots & \\ & & YY^{\mathrm{T}} \end{pmatrix},$$

$$Q_2 = \begin{pmatrix} X^TX & & \\ & \ddots & \\ & & X^TX \end{pmatrix},$$

$$C = \begin{pmatrix} X_{1,:}^{\mathrm{T}}Y_{:,1}^{\mathrm{T}} & \cdots & X_{n,:}^{\mathrm{T}}Y_{:,1}^{\mathrm{T}} \\ \vdots & \ddots & \vdots \\ X_{1,:}^{\mathrm{T}}Y_{:,m}^{\mathrm{T}} & \cdots & X_{n,:}^{\mathrm{T}}Y_{:,m}^{\mathrm{T}} \end{pmatrix} + \begin{pmatrix} (X_{1,:}Y_{:,1} - M_{1,1})I & \cdots & (X_{n,:}Y_{:,1} - M_{n,1})I \\ \vdots & \ddots & \vdots \\ (X_{1,:}Y_{:,m} - M_{1,m})I & \cdots & (X_{n,:}Y_{:,m} - M_{n,m})I \end{pmatrix}, \tag{5}$$

where $X_{i,:}$ is the ith row of X, and $Y_{:,j}$ is the jth column of Y.

Let μ be strictly positive, then the variables x, y, r and s are forced to take positive values. The trajectory $(x(\mu), y(\mu), r(\mu), s(\mu))$ is called the primal-dual central path. The variables are updated by

$$\begin{aligned} x &= x + \alpha_1 \Delta x, \\ y &= y + \alpha_1 \Delta y, \\ r &= r + \alpha_2 \Delta r, \\ s &= s + \alpha_2 \Delta s, \end{aligned} \tag{6}$$

where $\alpha_1 \in (0, \alpha_1^{\max}]$, $\alpha_2 \in (0, \alpha_2^{\max}]$, and

$$\begin{aligned} \alpha_1^{\max} &= \max\{\alpha \in (0, 1] : x + \alpha\Delta x \geq (1 - \tau)x, \\ &\quad y + \alpha\Delta y \geq (1 - \tau)y\}, \\ \alpha_2^{\max} &= \max\{\alpha \in (0, 1] : r + \alpha\Delta r \geq (1 - \tau)r, \\ &\quad s + \alpha\Delta s \geq (1 - \tau)s\}, \end{aligned} \tag{7}$$

with $\tau \in (0, 1)$. The condition (7) is called the fraction to the boundary rule (Nocedal & Wright, 2006), which is used to prevent the variables from approaching their lower bounds of 0 too quickly. In this work, we choose $\tau = 0.9$.

A predictor or probing strategy (Nocedal & Wright, 2006) can also be used to determine the parameter μ. We calculate a predictor (affine scaling) direction

$$(\Delta x^{\mathrm{aff}}, \Delta y^{\mathrm{aff}}, \Delta r^{\mathrm{aff}}, \Delta s^{\mathrm{aff}})$$

by setting $\mu = 0$. We probe this direction by letting $(\alpha_1^{\mathrm{aff}}, \alpha_2^{\mathrm{aff}})$ be the longest step lengths that can be taken along the affine scaling direction before violating the non-negativity conditions $(x, y, r, s) \geq 0$. Explicit formulas for these step lengths are given by (7) with $\tau = 1$. We then define μ^{aff} to be the value of complementarity along the (shortened) affine scaling step, that is,

$$\begin{aligned} \mu^{\mathrm{aff}} &= [(x + \alpha_1^{\mathrm{aff}}\Delta x)^{\mathrm{T}}(r + \alpha_2^{\mathrm{aff}}\Delta r) \\ &\quad + (y + \alpha_1^{\mathrm{aff}}\Delta y)^{\mathrm{T}}(s + \alpha_2^{\mathrm{aff}}\Delta s)]/(nk + mk), \end{aligned} \tag{8}$$

and a heuristic choice of μ is defined as follows: $\mu = \sigma\mu$ and

$$\sigma = \min\left\{\left(\frac{\mu^{\mathrm{aff}}}{(x^{\mathrm{T}}r + y^{\mathrm{T}}s)/(nk + mk)}\right)^3, 0.99\right\}. \tag{9}$$

We propose a two-level nested loop algorithm for the interior point search. In the inner loop, the parameter μ is fixed. In the outer loop, we gradually reduce μ to 0. We use the following error function to break the inner loop, which is based on the perturbed KKT system:

$$\begin{aligned} E(x, y, r, s; \mu) = \max\{&\|((\mathrm{gra}X - r)^{\mathrm{T}}, (\mathrm{gra}Y - s)^{\mathrm{T}})^{\mathrm{T}}\|, \\ &\|((r^{\mathrm{T}}\mathrm{Diag}(x), s^{\mathrm{T}}\mathrm{Diag}(y)) - \mu e^{\mathrm{T}})^{\mathrm{T}}\|\}. \end{aligned} \tag{10}$$

To guarantee the global convergence of the algorithm, we apply a line search approach. First, we consider the second derivation of the interior point method associated with the barrier problem:

$$\min_{X,Y} \frac{1}{2}\|XY - M\|_F^2 - \mu \sum_{i,h} \log(X_{i,h}) - \mu \sum_{j,h} \log(Y_{h,j}).$$

We use the exact merit function in the same way as the barrier function, which can be formed by

$$\begin{aligned} \phi(x, y) = &\frac{1}{2}\|\mathrm{mat}(x)^{\mathrm{T}}\mathrm{mat}(y) - M\|_F^2 \\ &- \mu \sum \log(x_i) - \mu \sum \log(y_j). \end{aligned}$$

In the algorithm, we utilize the Armijo line search to make the merit function decrease sufficiently.

Considering the convexity of the Hessian, we approximate

$$\begin{pmatrix} Q_1 & C^{\mathrm{T}} \\ C & Q_2 \end{pmatrix}$$

by a positive-definite matrix to guarantee the direction $(\Delta x, \Delta y)$ is always a descent direction of $\phi(x, y)$, so that such a line search can be implemented. Notice that the original NMF problem is a least-square problem. Thus we can utilize the Hessian in the traditional Gauss–Newton algorithm. The Hessian matrix is

$$\begin{pmatrix} Q_1 & \bar{C}^{\mathrm{T}} \\ \bar{C} & Q_2 \end{pmatrix}$$

and it is guaranteed to be positive semi-definite. Here

$$\bar{C} = \begin{pmatrix} X_{1,:}^{\mathrm{T}} Y_{:,1}^{\mathrm{T}} & \cdots & X_{n,:}^{\mathrm{T}} Y_{:,1}^{\mathrm{T}} \\ \vdots & \ddots & \vdots \\ X_{1,:}^{\mathrm{T}} Y_{:,m}^{\mathrm{T}} & \cdots & X_{n,:}^{\mathrm{T}} Y_{:,m}^{\mathrm{T}} \end{pmatrix},$$

which is the first item of C. For further safeguarding, we add a diagonal matrix ρI to this Hessian, where ρ is a small positive constant. Then we obtain the primal-dual direction by solving

$$\begin{pmatrix} Q_1 + \rho I & \bar{C}^{\mathrm{T}} & -I & \\ \bar{C} & Q_2 + \rho I & & -I \\ \mathrm{Diag}(r) & & \mathrm{Diag}(x) & \\ & \mathrm{Diag}(s) & & \mathrm{Diag}(y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta r \\ \Delta s \end{pmatrix} = \begin{pmatrix} r - \mathrm{gra}X \\ s - \mathrm{gra}Y \\ \mu e - \mathrm{Diag}(r)x \\ \mu e - \mathrm{Diag}(s)y \end{pmatrix}. \tag{11}$$

By eliminating Δr and Δs in (11), we have

$$\begin{pmatrix} R_1 & \bar{C}^{\mathrm{T}} \\ \bar{C} & R_2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = -\nabla\phi(x, y), \tag{12}$$

where

$$R_1 = Q_1 + \rho I + \mathrm{Diag}(x)^{-1}\mathrm{Diag}(r),$$
$$R_2 = Q_2 + \rho I + \mathrm{Diag}(y)^{-1}\mathrm{Diag}(s),$$
$$\nabla\phi(x, y) = (\mathrm{gra}X^{\mathrm{T}} - \mu(x^{-1})^{\mathrm{T}}, \mathrm{gra}Y^{\mathrm{T}} - \mu(y^{-1})^{\mathrm{T}})^{\mathrm{T}},$$

and $\mathrm{Diag}(x)^{-1}e$ is simply represented by x^{-1}. We simplify the above formula by

$$Bp = -\nabla\phi(x, y),$$

where B represents the coefficient matrix and p represents the direction $(\Delta x^{\mathrm{T}}, \Delta y^{\mathrm{T}})^{\mathrm{T}}$. Since B is positive-definite, the inner product $p^{\mathrm{T}}\nabla\phi(x, y) > 0$, which means that p is a descent direction.

Algorithm 2 A line search interior point method

Initialize: Choose $x_0, y_0, r_0, s_0 > 0$, Select an initial barrier parameter $\mu > 0$, parameters $\eta, \sigma \in (0, 1)$, and decreasing tolerances $\epsilon_\mu \downarrow 0$ and ϵ_{TOL}. Set $l = 0$.

Repeat until $E(x_l, y_l, r_l, s_l; 0) \le \epsilon_{TOL}$

  Repeat until $E(x_l, y_l, r_l, s_l; \mu) \le \epsilon_\mu$

    Compute the primal-dual direction by solving (11).

    Compute $\alpha_1^{\max}$ and $\alpha_2^{\max}$ using (7).

    Backtrack step lengths $\alpha_1 = \frac{1}{2^t}\alpha_1^{\max}, \alpha_2 = \alpha_2^{\max}$ to find the smallest integer $t \ge 0$ satisfying

$$\phi(x_l + \alpha_1\Delta x_l, y_l + \alpha_1\Delta y_l) \le \phi(x_l, y_l) + \eta\alpha_1(\Delta x_l^T, \Delta y_l^T)\nabla\phi(x_l, y_l). \tag{13}$$

    Compute $(x^{l+1}, y^{l+1}, r^{l+1}, s^{l+1})$ using (6).

    Set $l := l + 1$.

  end

  Compute parameter σ using (8) and (9) and update $\mu = \sigma\mu$.

end

Figure 2
Algorithm 2: a line search interior point method.

To sum up all the approaches above, we present the whole algorithm of the interior point method in Algorithm 2 (Fig. 2). It contains two loops. The parameter μ is fixed in the inner loop. Due to the Armijo line search (13) (Algorithm 2, see Fig. 2) in the inner loop, the stopping criterion of the inner loop can be satisfied in finite iterations. Further, the parameter μ and ϵ_μ are reduced gradually, and by the definition of error function E, the solutions of the two-loop algorithm satisfy the KKT system (3) within the error ϵ_{TOL}. In practice, the barrier stop tolerance can be defined as

$$\epsilon_\mu = \mu.$$

The complete convergence theorem and its proof are given in Theorem 3.1.

Theorem 3.1. Suppose that all the sequences $\{x_l\}$, $\{y_l\}$, $\{r_l\}$, $\{s_l\}$ generated by Algorithm 2 are bounded. Then Algorithm 2 stops in finite iterations.

Proof. We will consider the inner loop first and show that for a given $\mu > 0$, $E(x_l, y_l, r_l, s_l; \mu) \le \epsilon_\mu$ will be satisfied in finite iterations.

Based on the Armijo line search rule (13) (Algorithm 2), the value of $\phi(x_l, y_l)$ decreases monotonously. Then we have that the lower bound of x_l and y_k is greater than a strictly positive constant that depends on μ. Thus the smallest eigenvalue of the coefficient matrix of (12) is greater than a constant greater than 0. Furthermore, using the boundedness of (x_l, y_l), we obtain that $(\Delta x_l, \Delta y_l)$ is bounded. According to the lower bound of x_l, y_l, the boundedness of $(\Delta x_l, \Delta y_l)$ and (7), we have

$$\inf \alpha_1^{\max} > 0. \tag{14}$$

According to the step size rule (13) (Algorithm 2), we have

$$\alpha_1(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l) \to 0.$$

We aim to prove

$$(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l) \to 0 \tag{15}$$

next. To prove it by contradiction, we suppose that

$$(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l) \not\to 0. \tag{16}$$

This means that there exists a subsequence $\mathcal{T}$ and a constant $a > 0$ such that

$$-(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l) > a, \tag{17}$$

for $l \in \mathcal{T}$. Due to the boundedness of $\{(x_l, y_l)\}_{l\in\mathcal{T}}$, there exists a subsequence $\mathcal{T}_1 \in \mathcal{T}$ such that $\{(x_l, y_l)\}_{l\in\mathcal{T}_1}$ converges to $(\bar{x}, \bar{y})$. By (16) and (17), we have

$$\{\alpha_1^l\}_{l\in\mathcal{T}_1} \to 0.$$

According to (14),

$$\alpha_1^l < \inf \alpha_1^{\max},$$

when $l \in \mathcal{T}_1$ is large enough. For simplicity, we redefine the sequence satisfying the above condition as $\mathcal{T}_1$. From the step size rule, the condition (13) (Algorithm 2) is violated by $\alpha_1 = 2\alpha_1^l$. We have

$$(\phi(x_l + 2\alpha_1^l \Delta x_l, y_l + 2\alpha_1^l \Delta y_l) - \phi(x_l, y_l))/(2\alpha_1^l) > \eta(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l). \quad (18)$$

Taking the limit of the above inequality, we obtain

$$(\Delta\bar{x}_l^{\mathrm{T}}, \Delta\bar{y}_l^{\mathrm{T}})\nabla\phi(\bar{x}_l, \bar{y}_l) \geq \eta(\Delta\bar{x}_l^{\mathrm{T}}, \Delta\bar{y}_l^{\mathrm{T}})\nabla\phi(\bar{x}_l, \bar{y}_l).$$

Due to $0 < \eta < 1$, it follows that $(\Delta\bar{x}_l^{\mathrm{T}}, \Delta\bar{y}_l^{\mathrm{T}})\nabla\phi(\bar{x}_l, \bar{y}_l) \geq 0$. On the other hand, $(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l) < 0$. Therefore, $(\Delta\bar{x}_l^{\mathrm{T}}, \Delta\bar{y}_l^{\mathrm{T}})\nabla\phi(\bar{x}_l, \bar{y}_l) = 0$, which is in contradiction to (16). So (15) is established.

From (12) and (15), it follows that

$$\nabla\phi(x_l, y_l) \to 0$$

and

$$(\Delta x_l, \Delta y_l) \to 0.$$

Then, according to (11), we obtain

$$\Delta r_l \to \mu x_l^{-1} - r_l,$$
$$\Delta s_l \to \mu y_l^{-1} - s_l.$$

For an arbitrary cluster point $(\bar{x}, \bar{y})$, for any $\delta_\mu > 0$, there exists a constant κ such that

$$\|(x^\kappa, y^\kappa) - (\bar{x}, \bar{y})\| \leq \delta_\mu,$$

and

$$\|\Delta r_l + r_l - \mu x_l^{-1}\| \leq \delta_\mu,$$
$$\|\Delta s_l + s_l - \mu y_l^{-1}\| \leq \delta_\mu,$$
$$\|(\Delta x_l, \Delta y_l)\| \leq \delta_\mu,$$
$$\|\nabla\phi(x_l, y_l)\| \leq \delta_\mu$$

for all $l \geq \kappa$. Due to the boundedness of (x_l, y_l), α_2^l can reach 1 for some $l := \bar{l} \leq \kappa + T - 1$, where T is a constant. Then it follows from

$$\|(\bar{x}, \bar{y}) - (x_{\bar{l}}, y_{\bar{l}})\| \leq T\delta_\mu$$

that

$$\|r_{\bar{l}} - \mu x_{\bar{l}}^{-1}\| \leq c\delta_\mu,$$

$$\|s_{\bar{l}} - \mu y_{\bar{l}}^{-1}\| \leq c\delta_\mu,$$

where c is constant. Therefore, for a given $\epsilon_\mu > 0$, let δ_μ be sufficiently small; then there exists an l such that $E(x_l, y_l, r_l, s_l; \mu) \leq \epsilon_\mu$.

We denote the sequence satisfying the inner loop stopping criterion by

$$\{(x_l, y_l, r_l, s_l)\}_{l \in \mathcal{S}}.$$

To prove the theorem by contradiction, we suppose that there is no point satisfying the outer loop stopping criterion $E(x_l, y_l, r_l, s_l; 0) \leq \epsilon_{\mathrm{TOL}}$. Due to the boundedness, there exists a cluster point of $\{(x_l, y_l, r_l, s_l)\}_{l \in \mathcal{S}}$. For any cluster point $(\bar{x}, \bar{y}, \bar{r}, \bar{s})$, we consider the limits on both sides of

$$E(x_l, y_l, r_l, s_l; \mu)_{l \in \mathcal{T}_2} \leq \epsilon_\mu,$$

then we have that

$$E(\bar{x}, \bar{y}, \bar{r}, \bar{s}; 0) = 0.$$

Thus Algorithm 2 stops at some $l \in \mathcal{T}_2$, which is a contradiction. Therefore, Algorithm 2 stops in finite iterations. □

3.2. Computation

In general, the computational cost in each iteration of the interior point method is usually the cubic power of its size, which is impractical for large-scale problems. However, for the special NMF problem under consideration here, the amount of computation can be greatly reduced, so that the proposed method can be applied to practical problems involving smoothly varying data. In the following, we analyze the computational cost of Algorithm 2.

First of all, to compute the gradient in (1), $O(nmk)$ flops are needed.

The main computational cost of Algorithm 2 is computing the primal-dual direction (11). One can solve (12) to obtain $(\Delta x, \Delta y)$ first, and then compute $(\Delta r, \Delta s)$ within a low cost.

We rewrite (12) as

$$\begin{pmatrix} \bar{Q}_1 & \bar{C}^{\mathrm{T}} \\ \bar{C} & \bar{Q}_2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}. \quad (19)$$

In order to minimize the computational cost, we first decompose $\bar{Q}_1$. $\bar{Q}_1 = P^{\mathrm{T}}P$ can be obtained by Cholesky factorization or eigenvalue decomposition. Since the matrix $\bar{Q}_1$ is composed of n positive-definite diagonal blocks with the size of $k \times k$, one can obtain P and P^{-1} within $O(nk^3)$ flops. Equation (19) is equivalent to

$$\begin{pmatrix} I & P^{-\mathrm{T}}\bar{C}^{\mathrm{T}} \\ \bar{C}P^{-1} & \bar{Q}_2 \end{pmatrix} \begin{pmatrix} P\Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} P^{-\mathrm{T}}b_1 \\ b_2 \end{pmatrix}.$$

Then we solve Δy from

$$(\bar{Q}_2 - (\bar{C}P^{-1})(\bar{C}P^{-1})^{\mathrm{T}})\Delta y = b_2 - (\bar{C}P^{-1})(P^{-\mathrm{T}}b_1), \quad (20)$$

and compute Δx by

$$\Delta x = P^{-1}(P^{-\mathrm{T}}b_1 - (\bar{C}P^{-1})^{\mathrm{T}}\Delta y).$$

We need $O(nmk^2)$ flops for constructing $\bar{C}$ and $O(nmk^3)$ flops for computing $\bar{C}P^{-1}$. By considering that each block of $\bar{C}$ is rank one, we can compute $\bar{C}P^{-1}$ within $O(nmk^2)$ flops. The dominant computation is the computation of $(\bar{C}P^{-1})(\bar{C}P^{-1})^{\mathrm{T}}$ which costs $O(nm^2k^3)$ flops. If we consider that each block of $\bar{C}$ is rank one, it can be reduced to $O(nm^2k^2)$ flops. When solving (20) by Cholesky factorization, since the size of the coefficient matrix is mk by mk, the computational cost is $O(m^3k^3)$. Other computations, like computing the right side of (20) and computing Δx, are $O(nmk^2)$ flops.

To sum up, the computation cost of Algorithm 2 in each iteration is $O((n + mk)m^2k^2)$. Compared with the computation of gradient $O(nmk)$, the cost is no more than $O(mk^2)$ times.

Since $n \gg m \gg k$ and k is usually very small in our smoothly varying data, the computational complexity is completely acceptable.

4. A two-stage algorithm for NMF on smoothly varying data

In this section, we propose a practical algorithm with fast convergence for solving NMF in smoothly varying data. It combines both the ANLS framework with the active set method and the interior point method proposed in the previous sections.

In the early stage of the algorithm, we use the ANLS framework with the active set method. It can reduce the value of the objective function rapidly. We use the relative step tolerance, which is a relative lower bound on the size of a step, meaning

$$\|(x_l, y_l) - (x_{l+1}, y_{l+1})\| \leq \epsilon_{\mathrm{STOL}}(1 + \|(x_l, y_l)\|),$$

as the stopping criterion of this stage. If the algorithm attempts to take a step that is smaller than the step tolerance, the iterations end.

At the end of the first phase, the algorithm then enters the second phase of using the interior point method. However, the solutions from the active set method usually contain elements with a value zero, which is incompatible with the strict interior point required by the interior point method. Meanwhile, we also need to provide the initial dual variables (r, s) to the interior point method. To address these issues, we first change the primal variable smaller than ρ_0 to ρ_0 by

$$(x, y) := \max\{(x, y), \rho_0\}, \tag{21}$$

where ρ_0 is a small positive constant. We can choose

$$\rho_0 = 10^{-6} \max\{(x, y)\}, \tag{22}$$

before implementing (21).

Next, we give the initial value of the dual variable by

$$\begin{aligned} r &= \max\{|\mathrm{gra}X|\}e, \\ s &= \max\{|\mathrm{gra}Y|\}e. \end{aligned} \tag{23}$$

We set the parameters

$$\mu = \frac{x^{\mathrm{T}} r + y^{\mathrm{T}} s}{mk + nk} \tag{24}$$

and

$$\rho = \epsilon_{\mathrm{TOL}}. \tag{25}$$

After these preparations, the algorithm enters its second stage by implementing Algorithm 2. As the Hessian matrix is approximated in Algorithm 2 by the positive-definite matrix

$$\begin{pmatrix} Q_1 + \rho I & \bar{C} \\ \bar{C} & Q_2 + \rho I \end{pmatrix}, \tag{26}$$

in order to further speed up convergence, we can change $\bar{C}$ back to C at the right time. A heuristic way to switch to

$$\begin{pmatrix} Q_1 + \rho I & C \\ C & Q_2 + \rho I \end{pmatrix} \tag{27}$$

is by monitoring σ, which is given in (9). A small σ implies that the predictor step generates a point close to the boundary, and

Algorithm 3 A fast two-stage algorithm

Initialize: Choose initial X, Y

Stage 1: Implement Algorithm 1 with the active set method and warm start strategy

Update variables by (21), (22) and (23)

Stage 2: Set parameters by (24) and (25), $\eta = 0.5$, select the tolerance ϵ_{TOL}, and let $flag = 0$.

Repeat until $E(x_l, y_l, r_l, s_l; 0) \leq \epsilon_{TOL}$

  $\epsilon_\mu := \mu$.

  Repeat until $E(x_l, y_l, r_l, s_l; \mu) \leq \epsilon_\mu$

    If $flag = 1$,

      compute the primal-dual direction by solving (11) with Hessian (27).

      If $(\Delta x_l^T, \Delta y_l^T)\nabla\phi(x_l, y_l) \geq 0$,

        $flag = 0$, compute the primal-dual direction by solving (11).

    Else,

      compute the primal-dual direction by solving (11).

    Compute $\alpha_1^{\max}$ and $\alpha_2^{\max}$ using (7).

    Backtrack step lengths $\alpha_1 = \frac{1}{2^t}\alpha_1^{\max}, \alpha_2 = \alpha_2^{\max}$ to find the smallest integer $t \geq 0$ satisfying (13).

    Compute $(x^{l+1}, y^{l+1}, r^{l+1}, s^{l+1})$ using (6).

    Set $l := l + 1$.

  End

  Compute parameter σ using (8) and (9) and update $\mu = \sigma\mu$.

  If $\sigma \leq 0.01$, $flag = 1$, else $flag = 0$

End

Figure 3
Algorithm 3: a fast two-stage algorithm.

thus it is likely to be close to a local minimum. Therefore, when

$$\sigma \leq \sigma_c,$$

we switch to (27), where σ_c is a user-supplied constant, and we set $\sigma_c = 0.01$ in our test. The computation of the primal-dual direction is similar to that using $\bar{C}$. The difference is that each block of C is no longer rank one, and thus the computational cost is $O(nm^2k^3)$ flops. Since (27) is not guaranteed to be positive-definite, the primal-dual direction using (27) may not be a descent direction. We check the negativity of

$$(\Delta x_l^{\mathrm{T}}, \Delta y_l^{\mathrm{T}})\nabla\phi(x_l, y_l)$$

in (13) (Algorithm 2) before we implement the line search. If we fail to obtain a descent direction, we switch to the positive-definite Hessian (26).

All elements of the above approaches are presented in our two-stage algorithm, as outlined in Algorithm 3 (Fig. 3).

5. Numerical tests

We test our two-stage algorithm 2-STAGE and compare it with other ANLS-based methods, including NeNMF (Guan *et al.*, 2012), QRPBB (Huang *et al.*, 2015), ANLS-BPP (Kim & Park, 2011) and IBPG-A (Le *et al.*, 2020). All the tests are performed using *MATLAB* 2020a on a computer with a 64-bit system and 2.70 GHz CPU. Comparisons are done on both synthetic data sets and real-world problems.

5.1. Stopping criterion

The KKT conditions of (1) are given in (3). The definition of the error function $E(x, y, r, s; 0)$ (10) measures the violation of the KKT conditions. Therefore, we set

$$E(x, y, r, s; 0) \leq \epsilon_{\mathrm{TOL}}$$

to be the stopping criterion, where

$$\epsilon_{\mathrm{TOL}} = 10^{-6}.$$

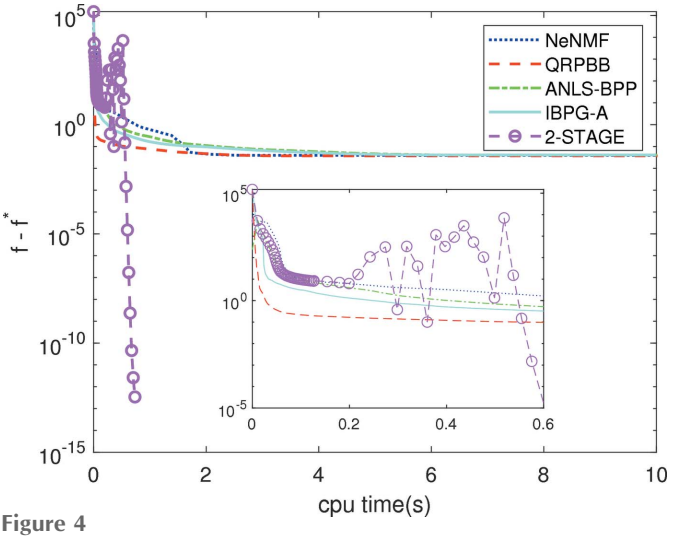


Figure 4
CPU time (s) versus tolerance values on AgMOR data.

The ANLS-based algorithms do not generate the dual variables r and s. Here we give a reasonable definition that

$$r = \max\{\mathrm{gra}X, 0\},$$
$$s = \max\{\mathrm{gra}Y, 0\}.$$

Since all the algorithms compared in this section belong to the class of feasible methods, the value of the objective function can also be used to assess the quality of the solution. Therefore, a stopping criterion can also be given by making the error between the objective function and the optimal solution less than 1e-6. When one of the above two criteria is satisfied, the algorithm can stop.

In some cases, we also limit the maximum CPU time, in cases where some algorithms cannot reach the given accuracy.

5.2. Streaming AgMOR PDF data

Zhao *et al.* (2011) measured the X-ray diffraction data during the nucleation and growth of zeolite-supported Ag nanoparticles through reduction of Ag-exchanged mordenite (MOR), and processed the data with atomic PDF measurements. In the field of chemistry, more and more people use mathematical tools to analyze their measured data. Chapman *et al.* (2015) used principal component analysis (PCA) and some post-processing to analyze the data given by Zhao *et al.* (2011), and obtained three principal components. Since PDF is a distribution-type function, NMF is intuitively more applicable as introduced by Liu *et al.* (2021). This is a representative instance that satisfies our definition of smoothly varying data and is also the data type we mainly want to apply our algorithm to. Then, we will test the effectiveness of our algorithm in smoothly varying data for this example.

We simply remove the negativity of the raw data by shifting each PDF up by the opposite of its original minimum value. Then we perform the NMF algorithms on the data. The size of

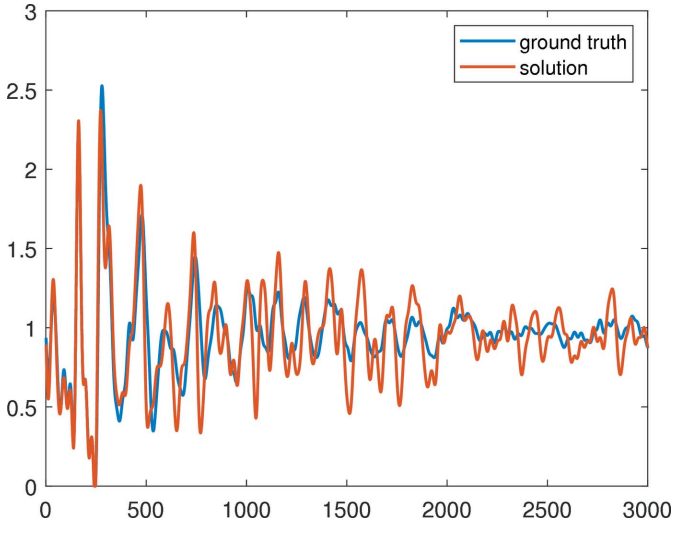


Figure 5
Comparison of true solution and approximate solution of AgMOR data on one component PDF.

Table 1
Experimental results on AgMOR data.

Algorithm	cpu(s):avrg(min,max)	*E*:avrg(min,max)	*f*:avrg(min,max)
NeNMF	20.00(20.00,20.00)	1.64(1.27,1.93)e-2	5.37246(5.37232,5.37262)e+2
QRPBB	20.00(20.00,20.00)	2.25(0.09,11.63)e-4	5.37223(5.37205,5.37246)e+2
ANLS-BPP	20.00(20.00,20.00)	1.23(0.05,1.67)e-2	5.37232(5.37205,5.37246)e+2
IBPG-A	20.00(20.00,20.00)	2.13(0.22,3.30)e-3	5.37232(5.37205,5.37245)e+2
2-STAGE	1.03(0.81,1.60)	5.04(1.36,9.98)e-7	5.37205(5.37205,5.37205)e+2

the data is $n = 3000$, $m = 36$, and in the algorithms k is set to be 3 based on the analysis of Chapman *et al.* (2015).

We use the same initial point for each algorithm. The relationship between the difference between the objective function $f = \frac{1}{2}\|XY - M\|_F^2$ and the optimal value f^* with the CPU time of the five algorithms is shown in Fig. 4. We can see that in the active set method part of 2-STAGE, the CPU time of each iteration of the algorithm decreases gradually. This is because the initial value is random and does not have continuity. With the gradual continuity of iterative variables, the active set method combined with the warm-start strategy we designed begins to play its role. It can be seen that the performances of NeNMF, ANLS-BPP and our 2-STAGE algorithm are similar in the early stage, IBPG-A is faster than them, and the decline of function value for QRPBB is the most obvious. After the active set method, there is an increase of error function value in 2-STAGE, because our algorithm begins to enter the second stage and the variables start to experience significant fluctuations later, which is caused by the instability of the early iterations of the interior point method before becoming stabilized on the central path. After several iterations, 2-STAGE becomes stable and converges quickly to meet the termination criterion. For this instance, 2-STAGE has a linear local convergence rate, while the others are sublinear as observed in Fig. 4.

In addition to the theoretically calculated differences given by the algorithm, the hidden operation of *MATLAB* also has serious effects on the CPU time. It is widely known that *MATLAB* optimizes the operation of the whole matrix, but different from QRPBB and IBPG-A, our active set method divides the matrix before the operation, which will produce a lot of additional CPU time. The latter could lead to disadvantages for us. The same trouble arises with the interior point method. As mentioned before, in theory, the dominant computation is $(\bar{C}P^{-1})(\bar{C}P^{-1})^{\mathrm{T}}$ with $O(nm^2k^3)$ flops, but in reality, the dominant computation is to construct $\bar{C}P^{-1}$ with a theoretically estimated $O(nmk^3)$ flops, and the actual CPU time is multiple times of $(\bar{C}P^{-1})(\bar{C}P^{-1})^{\mathrm{T}}$. This is because constructing $\bar{C}P^{-1}$ cannot be performed through the overall operation of the matrix in *MATLAB*. Even if 2-STAGE suffers losses in the underlying operation of *MATLAB*, the superiority of 2-STAGE is still shown in this benchmark of tests.

Next, we generate ten different random initial points. For each of them, all algorithms are implemented. The results are shown in Table 1. Because none of the three algorithms compared with the 2-STAGE algorithm can meet the stopping criterion in a short time, we set the maximum running CPU time as 20 s. Table 1 presents the average, minimum and maximum CPU time, the KKT violation E and the objective

Table 2
Experimental results on synthetic smoothly varying data.

Tol. = tolerance. Values in bold are the fastest for that group of data.

Tol.	Algorithm	cpu(s):avrg(min,max)	*E*:avrg(min,max)	*f*:avrg(min,max)
1e-3	NeNMF	25.41(2.06,60.00)	3.46(1.60,7.03)e-2	2.973909(2.973686,2.975458)
	QRPBB	3.05(0.47,13.93)	1.88(0.45,5.31)e-3	2.989887(2.973676,3.007711)
	ANLS-BPP	25.34(14.36,40.38)	1.79(1.56,2.01)e-2	2.973691(2.973691,2.973691)
	IBPG-A	14.76(1.32,33.16)	2.47(0.14,4.36)e-2	2.973681(2.973663,2.973691)
	2-STAGE	**1.03**(0.70,1.31)	8.71(4,45,17.12)e-1	2.972952(2.972724,2.973229)
1e-4	NeNMF	35.07(3.05,60.00)	1.26(0.30,2.96)e-2	2.973208(2.972791,2.975458)
	QRPBB	12.68(0.87,60.00)	1.01(0.06,2.43)e-3	2.975399(2.972790,2.993992)
	ANLS-BPP	50.87(38.83,60.00)	4.93(2.98,7.38)e-3	2.972877(2.972791,2.973043)
	IBPG-A	25.20(1.55,60.00)	7.31(0.32,14.89)e-3	2.972799(2.972790,2.972876)
	2-STAGE	**1.04**(0.70,1.33)	4.41(0.94,9.41)e-1	2.972724(2.972693,2.972757)
1e-5	NeNMF	39.46(4.04,60.00)	7.45(1.89,29.56)e-3	2.973161(2.972701,2.975458)
	QRPBB	22.15(1.68,60.00)	1.51(0.07,0.43)e-4	2.973254(2.972701,2.976984)
	ANLS-BPP	60.00(60.00,60.00)	4.00(1.51,7.38)e-3	2.972838(2.972709,2.973043)
	IBPG-A	27.25(1.79,60.00)	1.82(0.25,4.31)e-3	2.972732(2.972701,2.972876)
	2-STAGE	**1.06**(0.72,1.36)	7.74(3.54,13.13)e-2	2.972693(2.972692,2.972697)
1e-6	NeNMF	40.68(6.50,60.00)	6.19(0.68,29.56)e-3	2.973158(2.972692,2.975458)
	QRPBB	25.35(1.83,60.00)	1.25(0.19,2.69)e-4	2.973245(2.972692,2.976984)
	ANLS-BPP	60.00(60.00,60.00)	4.00(1.51,7.38)e-3	2.972838(2.972709,2.973043)
	IBPG-A	28.00(2.04,60.00)	6.19(2.49,13.70)e-4	2.972726(2.972692,2.972876)
	2-STAGE	**1.08**(0.74,1.36)	2.70(0.98,8.98)e-2	2.972691(2.972691,2.972691)

function value of each algorithm. It can be seen that our algorithm always has the minimum KKT violation and the minimum objective function value. It has a great advantage in finding a high-precision solution within a short amount of time.

In Table 1, the relative differences between the final value of the objective function obtained by 2-STAGE and those obtained by the other methods are less than 0.01%. One might wonder whether this is a truly significant difference for this application. In fact, the solution contains three PDFs as components, and one of them is very different from the real solution. We compare the solution with the function value of 5.37237e+2 solved by QRPBB in Fig. 4 with the true solution with the function value of 5.37205e+2, as shown in Fig. 5. It is obvious that many peaks are totally inconsistent with the true solution. This shows the necessity of solving high-precision solutions in this application. Therefore, it is of great significance to apply 2-STAGE to NMF of the smoothly varying data considered in our context.

5.3. Synthetic smoothly varying data

We artificially synthesize smoothly varying data to more comprehensively test the performance of the algorithms. The data are generated via the formula $M = XY$, where X is the concatenation of Gaussian mixture functions and Y is constructed by trigonometric functions. In detail, each column of X is the function values of a Gaussian mixture function on the grids from 0.01 to $ub = 0.01n$ with a separation of 0.01. The number of Gaussian functions is ub, and each is uniformly distributed between 1 and $ub - 1$. The standard deviation satisfies a uniform distribution of 0.2 to 0.4, and the intensity is a uniform distribution of 0 to 1. Each row of Y contains the function values of $\max\{A\sin(\omega y + \varphi), -0.1\}$ on the y grids from π/m to π with a separation of π/m. The frequency ω and the amplitude A are uniformly distributed over 1 to 2, and the phase φ is uniformly distributed over 0 to 2π. The threshold -0.1 indicates a noise level. Intuitively, the data generated in this way are similar to the PDF data we encountered in the

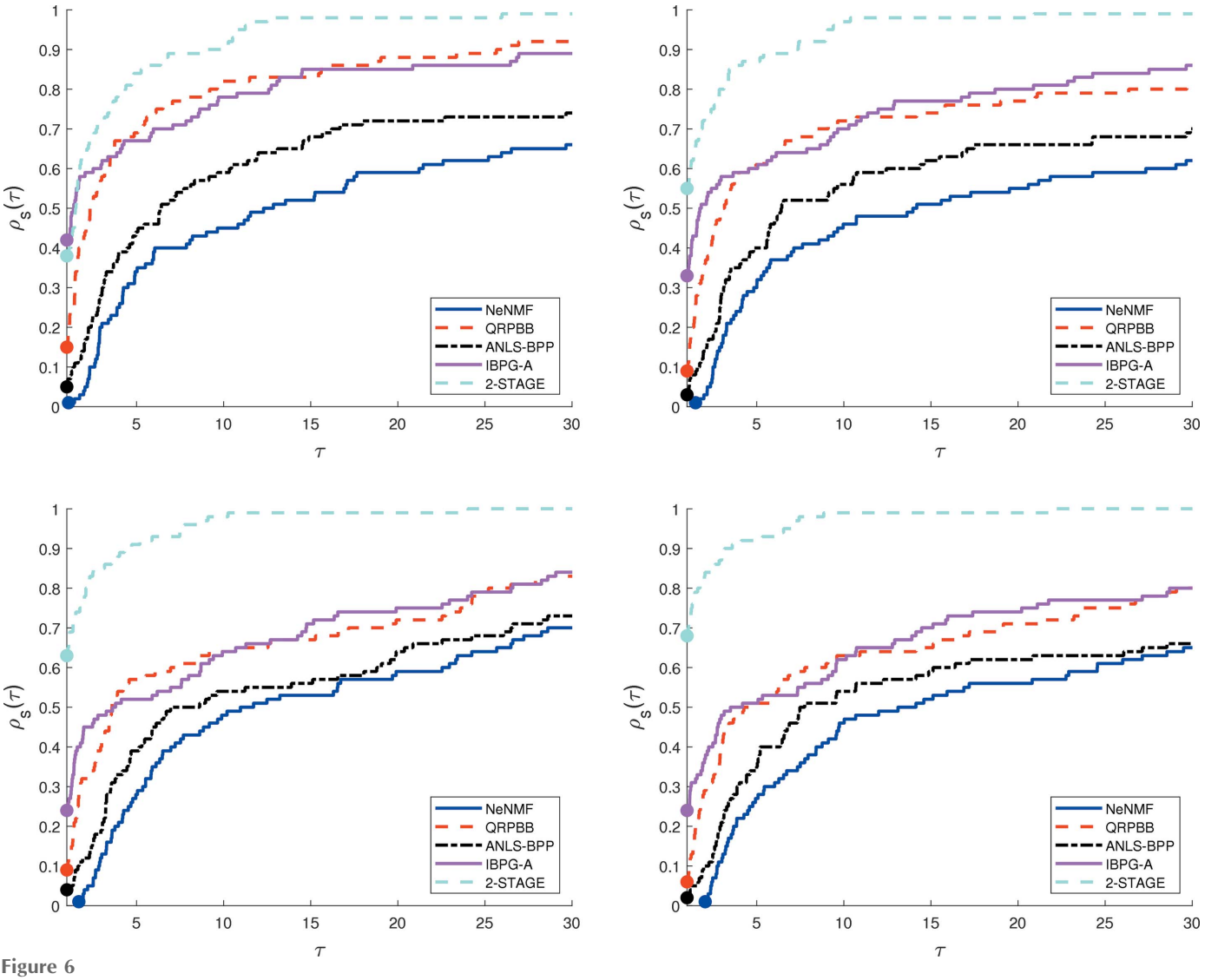


Figure 6
Performance profile on synthetic smoothly varying data with different tolerances, where the tolerance is 1e-3 at the top left, 1e-4 at the top right, 1e-5 at the bottom left and 1e-6 at the bottom right.

previous subsection, because X is the sum of Gaussian functions, which has some relationship with PDF (Gu *et al.*, 2019), and Y is to simulate the weight change of each component.

The problem size in this synthetic test is fixed as $(n = 3000, m = 36, k = 3)$, which is the same as for the AgMOR PDF data we used in the last subsection. Table 2 shows the results of a randomly generated instance with ten initial points. The tolerances in the stopping criterion are set as 1e-3, 1e-4, 1e-5 and 1e-6. At the tolerance level of 1e-3, NeNMF reaches the maximum time of 60 s. In terms of average time, 2-STAGE is the best. When the tolerance reaches 1e-4, all algorithms except 2-STAGE reach 60 s in a subset of instances. We can see that 2-STAGE can reach the tolerance from 1e-3 to 1e-6 in a very short amount of CPU time, thanks to its fast local convergence rate.

Next, 100 randomly generated problems are tested for each tolerance level. The performance profile is shown in Fig. 6. $\rho_s(\tau)$ means the proportion of problems in which the algorithm can stop within a certain CPU time in each problem, with the number being τ times the smallest CPU time in such a problem among all algorithms in the test. In the case of 1e-3, 2-STAGE is similar to IBPG-A in the low τ range ($\tau \leq 2$), but for a large τ range, the curve of 2-STAGE is above other algorithms, which means that 2-STAGE is more efficient and robust than the other algorithms. With the reduction of tolerance, that is, to solve the solution with higher precision, the advantage of 2-STAGE shown in Fig. 6 becomes more obvious.

5.4. Synthetic non-smoothly varying data

Theoretically, the 2-STAGE approach in finding high-precision solutions gains its advantage from the second stage interior point method and special data size. In this subsection, we artificially synthesize some random data for testing the theoretical predictions. These tests no longer use smoothly varying data, but the dimension of the data remains $n \gg m \gg k$.

We construct a set of artificial data by the following rule. First, we determine the problem size (n, m, k). Then, we generate a random matrix X whose size is $n \times k$, and each

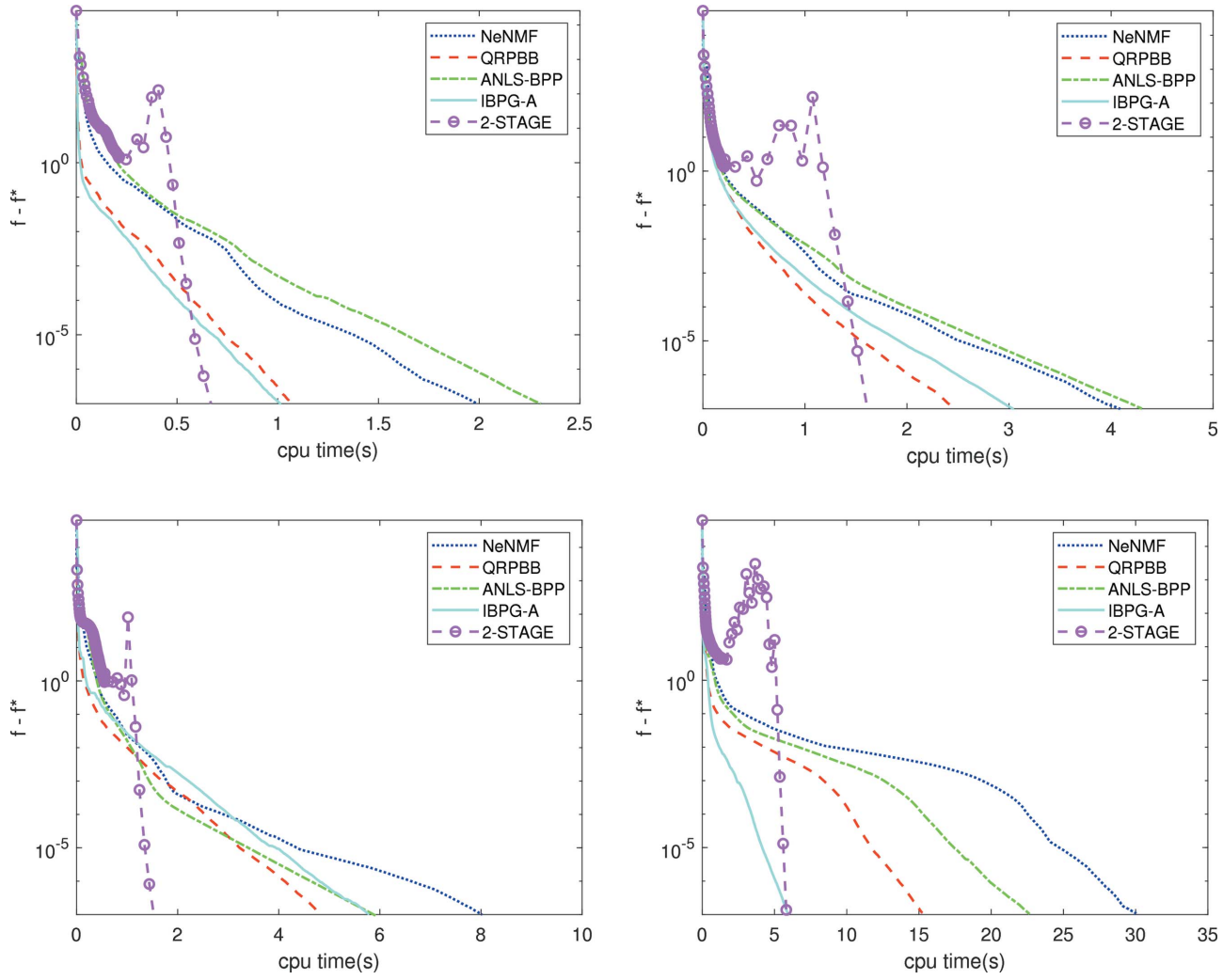


Figure 7
Experimental tests on synthetic data with different size, where (n, m, k) is (2000, 100, 3) at the top left, (2000, 200, 3) at the top right, (4000, 100, 3) at the bottom left and (2000, 100, 6) at the bottom right.

Table 3
Experimental results on synthetic non-smoothly varying data.

Values in bold are the fastest for that group of data.

Size(n,m,k)	Algorithm	cpu(s):avrg(min,max)	E:avrg(min,max)	f:avrg(min,max)
(2000,100,3)	NeNMF	2.02(1.70,2.49)	2.56(2.26,2.93)e-3	9.57337(9.57337,9.57337)e+2
	QRPBB	0.98(0.76,1.24)	1.72(0.02,6.05)e-4	9.57337(9.57337,9.57337)e+2
	ANLS-BPP	1.96(1.63,2.49)	2.57(2.33,2.87)e-3	9.57337(9.57337,9.57337)e+2
	IBPG-A	1.00(0.88,1.22)	5.99(5.29,6.89)e-4	9.57337(9.57337,9.57337)e+2
	2-STAGE	**0.69**(0.57,0.79)	3.97(0.47,15.94)e-3	9.57337(9.57337,9.57337)e+2
(2000,100,6)	NeNMF	20.74(13.32,28.35)	5.72(3.72,10.13)e-4	9.38620(9.38620,9.38620)e+2
	QRPBB	11.75(8.51,17.68)	9.11(5.17,15.86)e-5	9.38620(9.38620,9.38620)e+2
	ANLS-BPP	17.40(13.08,21.77)	7.76(5.82,8.72)e-4	9.38620(9.38620,9.38620)e+2
	IBPG-A	6.63(4.97,8.37)	2.42(1.94,3.05)e-4	9.38620(9.38620,9.38620)e+2
	2-STAGE	**4.38**(3.21,6.37)	9.80(3.30,27.70)e-3	9.38620(9.38620,9.38620)e+2
(2000,200,3)	NeNMF	5.92(3.97,11.64)	1.23(1.09,1.71)e-3	1.95720(1.95720,1.95720)e+3
	QRPBB	6.62(3.19,9.22)	2.38(0.40,5.81)e-5	1.95720(1.95720,1.95720)e+3
	ANLS-BPP	8.55(4.93,17.01)	1.24(1.08,1.75)e-3	1.95720(1.95720,1.95720)e+3
	IBPG-A	8.81(4.41,12.97)	3.46(2.92,4.06)e-4	1.95720(1.95720,1.95720)e+3
	2-STAGE	**1.78**(1.35,3.34)	3.78(0.31,8.51)e-3	1.95720(1.95720,1.95720)e+3
(2000,200,6)	NeNMF	15.02(12.47,22.12)	8.14(5.95,9.58)e-4	1.94330(1.94330,1.94330)e+3
	QRPBB	12.15(9.06,15.64)	5.38(3.79,12.54)e-5	1.94330(1.94330,1.94330)e+3
	ANLS-BPP	15.65(10.79,22.72)	9.73(7.56,10.48)e-4	1.94330(1.94330,1.94330)e+3
	IBPG-A	8.74(5.58,11.01)	3.54(3.18,3.82)e-4	1.94330(1.94330,1.94330)e+3
	2-STAGE	**8.55**(7.02,11.13)	1.90(0.31,5.32)e-2	1.94330(1.94330,1.94330)e+3
(4000,100,3)	NeNMF	6.62(5.12,8.19)	1.98(1.85,2.08)e-3	1.91702(1.91702,1.91702)e+3
	QRPBB	3.53(3.27,3.89)	6.94(1.66,1.23)e-5	1.91702(1.91702,1.91702)e+3
	ANLS-BPP	6.36(5.55,6.82)	1.96(1.83,2.11)e-3	1.91702(1.91702,1.91702)e+3
	IBPG-A	3.17(2.95,3.36)	3.36(3.08,3.63)e-4	1.91702(1.91702,1.91702)e+3
	2-STAGE	**1.47**(1.27,1.80)	6.14(0.74,19.00)e-3	1.91702(1.91702,1.91702)e+3
(4000,100,6)	NeNMF	22.32(15.98,29.65)	1.25(0.84,1.42)e-3	1.87715(1.87715,1.87715)e+3
	QRPBB	13.74(9.85,20.91)	8.66(1.44,1.96)e-5	1.87715(1.87715,1.87715)e+3
	ANLS-BPP	15.56(11.47,19.96)	1.38(1.11,1.59)e-3	1.87715(1.87715,1.87715)e+3
	IBPG-A	**6.34**(5.18,8.00)	3.19(2.75,4.41)e-4	1.87715(1.87715,1.87715)e+3
	2-STAGE	7.72(6.12,10.58)	2.54(0.47,5.77)e-2	1.87715(1.87715,1.87715)e+3
(4000,200,3)	NeNMF	12.84(11.06,14.71)	2.10(1.92,2.29)e-3	3.91802(3.91802,3.91802)e+3
	QRPBB	6.19(4.93,7.01)	9.62(6.74,14.00)e-5	3.91802(3.91802,3.91802)e+3
	ANLS-BPP	12.04(10.27,13.35)	2.05(1.92,2.27)e-3	3.91802(3.91802,3.91802)e+3
	IBPG-A	7.95(6.02,9.19)	4.74(4.39,5.45)e-4	3.91802(3.91802,3.91802)e+3
	2-STAGE	**3.03**(2.73,3.48)	8.07(1.94,24.68)e-3	3.91802(3.91802,3.91802)e+3
(4000,200,6)	NeNMF	22.39(18.36,24.59)	1.23(1.00,1.41)e-3	3.87687(3.87687,3.87687)e+3
	QRPBB	14.95(10.12,17.59)	2.04(0.17,3.98)e-4	3.87687(3.87687,3.87687)e+3
	ANLS-BPP	22.27(20.48,23.80)	1.58(1.35,1.90)e-3	3.87687(3.87687,3.87687)e+3
	IBPG-A	**10.16**(7.39,12.49)	4.76(4.41,5.26)e-4	3.87687(3.87687,3.87687)e+3
	2-STAGE	17.37(16.47,19.61)	1.26(0.41,4.52)e-2	3.87687(3.87687,3.87687)e+3

element is uniformly distributed from 0 to 1. In the same way, we generate a random matrix Y whose size is $k \times m$. Next, we compute $M = XY$ and add Gaussian noise, with expectation 0 and standard deviation 0.1, to each element of M. Finally, we change the negative elements in M to zeros.

The differences between the objective function and the optimal value of the five algorithms are plotted against the CPU time in Fig. 7. Due to the non-continuity of the randomly generated matrix, the active set method in the first stage cannot fully realize the benefits brought by its warm-start strategy; thus its speed is not as fast as that of the other algorithms.

One can observe from Fig. 7 that the local convergence rates of all algorithms appear to be linear. It can be easily seen that the convergence speed of 2-STAGE in the later second stage is the fastest since the slope is significantly steeper than the others. This means that to get higher-precision solutions, it is more appropriate to choose 2-STAGE.

From the comparison between 2-STAGE and other algorithms in Fig. 7, the influence of any changes with m and n on the local convergence speed is small. The influence of k is relatively large, and the increase of k is not favorable to 2-STAGE.

More results are shown in Table 3, where n is set to 2000 or 4000, m is set to 100 or 200, k is set to 3 or 6, tolerance is 1e-6, and each instance has ten initial points. It can be seen from the table that the final objective function f of all algorithms in each instance is equal, while the KKT error E does not meet the specified requirements. Therefore, they all stop because they meet the termination criterion of the objective function. Interestingly, in the tests, QRPBB usually has a smaller E, while E of 2-STAGE is usually large because at the beginning of the second stage, the initial value of the dual variable is far from the optimal solution. In the iterative process, the primal variable reaches the optimal faster than the dual variable. Therefore, E is still large when the objective function decreases to the termination criterion. However, we know that the interior point method has a fast local convergence rate, and E can achieve the accuracy of other algorithms in only one or two iterations.

Now we compare the CPU time of these algorithms in Table 3. QRPBB, IBPG-A and 2-STAGE generally take less time than NeNMF and ANLS-BPP. When $k = 3$, the performance of 2-STAGE stands out: the CPU time is always the shortest, and the relative gap is large compared with other algorithms. However, when $k = 6$, the CPU time increment of 2-STAGE is higher than that of other algorithms. Although the CPU time of 2-STAGE is still the lowest in the two instances when $n = 2000$, the trend no longer holds and is reversed by IBPG-A when $n = 4000$. In particular, when $n = 4000$ and $m = 200$, 2-STAGE only ranks third. This also confirms that 2-STAGE is greatly affected by the problem size, especially k.

Generally speaking, the performance of 2-STAGE in getting a high-precision solution truly stands out, when k is small and the size of the problem satisfies $n \gg m \gg k$. From the current test examples with the specially chosen problem size, the local convergence of 2-STAGE is the fastest, which shows that 2-STAGE is the strongest in the pursuit of solution precision.

6. Conclusions

In this paper, we focused on solutions to the NMF for smoothly varying data. We presented a fast two-stage algorithm, where the first stage is the ANLS framework with the active set method which benefits from the continuity of smoothly varying data, and the second stage is a line search interior point method which benefits from $n \gg m \gg k$. In addition, we have proved the global convergence of the proposed line search interior point method. The first stage reduces the value of the objective function rapidly, and the second stage converges to a local solution quickly due to the property of Newton-type direction. We tested the proposed algorithm on several real and synthetic data sets, and observed that, compared with other algorithms, our algorithm is more effective in solving high-precision local solutions.

The active set method in the first stage does not reach the expected speed, even if it is tested on continuous data. We think that this may be caused by the limitations of the underlying code implementation in *MATLAB*. On the other hand, we find that the transition part between the two stages may induce instability. This is because the solution of the active set method cannot be directly used as the initial guess of the interior point method, and its changes have an impact on stability. At present, the parameters used to generate starting points are selected carefully to avoid the instability. In the future, we will work to find a more stable transition technique.

Considering that, in addition to the basic NMF model, there are other variants of NMF, such as constrained NMFs and structured NMFs, our algorithm has the potential to be applicable to more problems through suitable extension. This will be further investigated in the future.

Funding information

Funding for this research was provided by: National Science Foundation, DMR (grant No. 1922234 to Ran Gu, Simon J. L. Billinge, Qiang Du); National Science Foundation, CCF (grant No. 1740833); National Natural Science Foundation of China (grant No. 12201318 to Ran Gu); Fundamental Research Funds for the Central Universities (grant No. 63223048); US Department of Energy ASCR DE (grant No. SC0022317 to Qiang Du).

References

Arora, S., Ge, R., Kannan, R. & Moitra, A. (2016). *SIAM J. Comput.* **45**, 1582–1611.
Beck, A. & Tetruashvili, L. (2013). *SIAM J. Optim.* **23**, 2037–2060.
Buciu, I. (2008). *Int. J. Comput. Commun. Contr.* **3**, 67–74.
Chapman, K. W., Lapidus, S. H. & Chupas, P. J. (2015). *J. Appl. Cryst.* **48**, 1619–1626.
Cichocki, A. & Phan, A.-H. (2009). *IEICE Trans. Fundamentals*, **E92-A**, 708–721.
Cichocki, A., Zdunek, R., Phan, A. H. & Amari, S.-I. (2009). *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation.* Hoboken: John Wiley & Sons.
Egami, T. & Billinge, S. J. L. (2012). *Underneath the Bragg Peaks: Structural Analysis of Complex Materials*, 2nd ed. Amsterdam: Elsevier.
Gillis, N. (2020). *Nonnegative Matrix Factorization.* Philadelphia: SIAM.
Goldfarb, D. & Ma, S. (2011). *Found. Comput. Math.* **11**, 183–210.
Gong, P. & Zhang, C. (2012). *Pattern Recognit.* **45**, 3557–3565.
Grippo, L. & Sciandrone, M. (2000). *Oper. Res. Lett.* **26**, 127–136.
Gu, R., Banerjee, S., Du, Q. & Billinge, S. J. L. (2019). *Acta Cryst.* A**75**, 658–668.
Guan, N., Tao, D., Luo, Z. & Yuan, B. (2012). *IEEE Trans. Signal Process.* **60**, 2882–2898.
Hajinezhad, D., Chang, T.-H., Wang, X., Shi, Q. & Hong, M. (2016). *2016 IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), Shanghai, China, pp. 4742–4746.
Huang, Y., Liu, H. & Zhou, S. (2015). *Data Min. Knowl. Disc*, **29**, 1665–1684.
Jensen, K. M., Juhas, P., Tofanelli, M. A., Heinecke, C. L., Vaughan, G., Ackerson, C. J. & Billinge, S. J. (2016). *Nat. Commun.* **7**, 11859.
Juhás, P., Cherba, D., Duxbury, P., Punch, W. & Billinge, S. (2006). *Nature*, **440**, 655–658.
Kim, D., Sra, S. & Dhillon, I. S. (2007). *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 343–354. Philadelphia: SIAM.
Kim, H. & Park, H. (2008). *SIAM J. Matrix Anal. Appl.* **30**, 713–730.
Kim, J. & Park, H. (2011). *SIAM J. Sci. Comput.* **33**, 3261–3281.
Kuhn, H. W. & Tucker, A. W. (1951). *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pp. 481–492. Berkeley, California: University of California Press. https://projecteuclid.org/euclid.bsmsp/1200500249.
Lawson, C. L. & Hanson, R. J. (1995). *Solving Least Squares Problems*, Vol. 15. Philadelphia: SIAM.
Le, H., Gillis, N. & Patrinos, P. (2020). *International Conference on Machine Learning*, pp. 5671–5681. PMLR.
Lee, D. D. & Seung, H. S. (1999). *Nature*, **401**, 788–791.
Lee, D. D. & Seung, H. S. (2001). *Adv. Neural Inf. Process. Syst.* pp. 556–562.
Li, L. (2009). *J. Electron. Imaging*, **18**, 033004.
Lin, C.-J. (2007). *Neural Comput.* **19**, 2756–2779.
Liu, C.-H., Wright, C. J., Gu, R., Bandi, S., Wustrow, A., Todd, P. K., O'Nolan, D., Beauvais, M. L., Neilson, J. R., Chupas, P. J., Chapman, K. W. & Billinge, S. J. L. (2021). *J. Appl. Cryst.* **54**, 768–775.
Liu, Y.-F., Hong, M. & Song, E. (2016). *IEEE Trans. Wireless Commun.* **15**, 4535–4547.

Nocedal, J. & Wright, S. (2006). *Numerical Optimization.* New York: Springer Science & Business Media.

Paatero, P. & Tapper, U. (1994). *Environmetrics*, **5**, 111–126.

Rakita, Y., Hart, J. L., Das, P. P., Shahrezaei, S., Foley, D. L., Mathaudhu, S. N., Nicolopoulos, S., Taheri, M. L. & Billinge, S. J. L. (2023). *Acta Mater.* **242**, 118426.

Sra, S. & Dhillon, I. S. (2006). *Nonnegative Matrix Approximation: Algorithms and Applications.* Computer Science Department, University of Texas at Austin.

Thatcher, Z., Liu, C.-H., Yang, L., McBride, B. C., Thinh Tran, G., Wustrow, A., Karlsen, M. A., Neilson, J. R., Ravnsbaek, D. B. & Billinge, S. J. L. (2022). *Acta Cryst.* A**78**, 242–248.

Todd, P. K., Wustrow, A., McAuliffe, R. D., McDermott, M. J., Tran, G. T., McBride, B. C., Boeding, E. D., O'Nolan, D., Liu, C.-H., Dwaraknath, S. S., Chapman, K. W., Billinge, S. J. L., Persson, K. A., Huq, A., Veith, G. M. & Neilson, J. R. (2020). *Inorg. Chem.* **59**, 13639–13650.

Vavasis, S. A. (2009). *SIAM J. Optim.* **20**, 1364–1377.

Wang, Y.-X. & Zhang, Y.-J. (2012). *IEEE Trans. Knowl. Data Eng.* **25**, 1336–1353.

Zdunek, R. & Cichocki, A. (2006). *International Conference on Artificial Intelligence and Soft Computing*, pp. 870–879. Zakopane: Springer.

Zhao, H., Nenoff, T. M., Jennings, G., Chupas, P. J. & Chapman, K. W. (2011). *J. Phys. Chem. Lett.* **2**, 2742–2746.