RICE UNIVERSITY

Block Coordinate Descent for Regularized Multi-convex Optimization

by

Yangyang Xu

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Master of Arts

Approved, Thesis Committee:

Wotao Yin, Chair
Associate Professor of Computational and Applied Mathematics

Richard Tapia
University Professor
Maxfield-Oshman Professor in Engineering
Computational and Applied Mathematics

Yin Zhang
Professor of Computational and Applied Mathematics

Richard Baraniuk
Victor E. Cameron Professor of Electrical and Computer Engineering

Houston, Texas

November, 2012

ABSTRACT

Block Coordinate Descent for Regularized Multi-convex Optimization

by

Yangyang Xu

This thesis considers regularized block multi-convex optimization, where the feasible set and objective function are generally non-convex but convex in each block of variables. I review some of its interesting examples and propose a generalized block coordinate descent (BCD) method. The generalized BCD uses three different block-update schemes. Based on the property of one block subproblem, one can freely choose one of the three schemes to update the corresponding block of variables. Appropriate choices of block-update schemes can often speed up the algorithm and greatly save computing time. Under certain conditions, I show that any limit point satisfies the Nash equilibrium conditions. Furthermore, I establish its global convergence and estimate its asymptotic convergence rate by assuming a property based on the Kurdyka-Łojasiewicz inequality. As a consequence, this thesis gives a global linear convergence result of cyclic block coordinate descent for strongly convex optimization. The proposed algorithms are adapted for factorizing nonnegative matrices and tensors, as well as completing them from their incomplete observations. The algorithms were tested on synthetic data, hyperspectral data, as well as image sets from the CBCL, ORL and Swimmer databases. Compared to the existing state-of-the-art algorithms, the proposed algorithms demonstrate superior performance in both speed and solution quality.

Acknowledgments

First, I want to express my deepest appreciation to my advisor, Professor Wotao Yin, who is my committee chair. He continuously and convincingly conveys to me a spirit of research and gradually leads me to the exciting academic research way. Without his guidance, precious advice and persistent encouragement, this thesis would not be possible.

Secondly, I would like to thank my committee members, Professor Richard Tapia, Professor Yin Zhang, and Professor Richard Baraniuk. They spend a lot of precious time reviewing my thesis and make great advice, which are very helpful to improve the overall quality of my thesis.

In addition, I want to thank Professor Zaiwen Wen, who carefully and patiently helped me solve some critical issues encountered during my research. Also, I want to thank the staff in our department. They provide generous help to my research.

Finally, I would like to particularly thank my lovely wife, Jun. She accompanies me, encourages me, and gives me the best care. She helps me in both of my everyday and academic life. Without her persistent help and encouragement, it is impossible to conquer every difficulty and make every achievement.

Contents

Illustrations

Chapter 1

Introduction

One type of optimization problem that arises in many applications has the following two properties: (i) its objective can be written as a sum of a smooth function and a (non-smooth) separable function; (ii) its variables can be partitioned into a few disjoint blocks, and the objective can be jointly non-convex but convex with respect to each block of variables while all the others are fixed. One particular example is dictionary learning with ℓ_1-regularizer [60]

$$\min_{\mathbf{D},\mathbf{\Theta}} \frac{1}{2}\|\mathbf{D\Theta} - \mathbf{X}\|_F^2 + \lambda\|\mathbf{\Theta}\|_1, \text{ subject to } \mathbf{D} \in \mathcal{D}, \tag{1.1}$$

where $\mathcal{D} = \{\mathbf{D} : \|\mathbf{d}_j\| \leq 1,\ j = 1, \ldots, K\}$ is used to control the scale of $\mathbf{D}$, $\mathbf{d}_j$ denotes the jth column of $\mathbf{D}$ and $\|\mathbf{\Theta}\|_1 = \sum_{i,j} |\theta_{ij}|$. Using indicator function

$$\delta_{\mathcal{D}}(\mathbf{D}) = \begin{cases} 0 & \text{if } \mathbf{D} \in \mathcal{D} \\ \infty & \text{otherwise} \end{cases}$$

one can write (1.1) as

$$\min_{\mathbf{D},\mathbf{\Theta}} \frac{1}{2}\|\mathbf{D\Theta} - \mathbf{X}\|_F^2 + \lambda\|\mathbf{\Theta}\|_1 + \delta_{\mathcal{D}}(\mathbf{D}), \tag{1.2}$$

whose objective is a sum of the smooth term $\frac{1}{2}\|\mathbf{D\Theta} - \mathbf{X}\|_F^2$ and non-smooth separable function $\lambda\|\mathbf{\Theta}\|_1 + \delta_{\mathcal{D}}(\mathbf{D})$. The variables of (1.2) can be automatically partitioned into

two blocks: $\mathbf{D}$ and $\mathbf{\Theta}$. The objective is jointly non-convex since the product $\mathbf{D\Theta}$ couples $\mathbf{D}$ and $\mathbf{\Theta}$ together. However, it is convex with respect to each one of the two blocks while the other is fixed.

For some of these problems, computing the gradient of the objective over all variables can be very expensive such as the tensor decomposition [44]

$$\min_{\mathbf{A}_1,\cdots,\mathbf{A}_N} \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N\|_F^2, \tag{1.3}$$

or even impossible due to the existence of non-smooth terms such as in (1.1). Therefore, direct gradient descent is not a suitable choice to solve these problems. Traditional second-order methods such as interior point method or Newton's method are not good choices either due to non-smoothness. One popular and often efficient method for solving these problems is the alternating minimization method, which alternatively updates each block of variables by minimizing the objective with respect to one block at a time while all the others are fixed. For example, it is applied in [18] to tensor decomposition (1.3) by cyclically updating the factor matrices $\mathbf{A}_1, \cdots, \mathbf{A}_N$ via

$$\mathbf{A}_n^k = \operatorname*{argmin}_{\mathbf{A}_n} \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1^k \circ \cdots \circ \mathbf{A}_{n-1}^k \circ \mathbf{A}_n \circ \mathbf{A}_{n+1}^{k-1} \circ \cdots \circ \mathbf{A}_N^{k-1}\|_F^2, \forall n. \tag{1.4}$$

Each subproblem in (1.4) can be written into a convex quadratic programming by the property of tensor-matrix multiplication (see Section 3.1) and has a closed form solution. Alternating minimization has also been applied in [72] to group Lasso regularized problem (see (1.8) below), [88] to low-rank matrix recovery (see (1.12) below), and [33] to nonnegative matrix factorization (see (1.11) below).

Alternating minimization is favorable for these problems because it is usually difficult to update all variables simultaneously but relatively easy to update one block at a time. This method can be generalized to block coordinate descent (BCD) method, which also updates the variables block by block. Unlike its name would indicate, BCD does not have to decrease the objective during the update of each block. It is really a method by block-coordinate update. There are flexible ways to carry out the block update, by minimizing either the original objective or a relaxed version of the objective with respect to one block at a time with all others fixed. For some applications, minimizing a relaxed problem at each iteration can make overall better performance than minimizing the original one. For example, it was observed in [62] that alternating minimization (1.4) for tensor decomposition may cause a so-called swamp effect, which means the convergence rate dramatically slows down within exceedingly high number of iterations. However, the swamp effect can be reduced if the objective plus a proximal term is minimized at each iteration, namely,

$$\mathbf{A}_n^k = \underset{\mathbf{A}_n}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1^k \circ \cdots \circ \mathbf{A}_n \circ \cdots \circ \mathbf{A}_N^{k-1}\|_F^2 + \frac{\mu_k}{2}\|\mathbf{A}_n - \mathbf{A}_n^{k-1}\|_F^2, \forall n. \quad (1.5)$$

Each subproblem in (1.5) can also be written as a convex quadratic programming and has a closed form solution. In addition, for some applications, it may be difficult to solve block subproblems. For example, if we consider nonnegative tensor decomposition, namely, nonnegativity is enforced in (1.3), then both (1.4) and (1.5) with additional constraints $\mathbf{A}_n \geq 0$ are not easy to solve. However, it will become easy if

the first term in (1.5) is locally linearized, namely,

$$\mathbf{A}_n^k = \underset{\mathbf{A}_n \geq 0}{\operatorname{argmin}} \langle \mathbf{G}_n^k, \mathbf{A}_n - \mathbf{A}_n^k \rangle + \frac{\mu_k}{2} \|\mathbf{A}_n - \mathbf{A}_n^{k-1}\|_F^2, \forall n, \tag{1.6}$$

where $\mathbf{G}_n^k$ is the partial gradient about $\mathbf{A}_n$ of the first term in (1.5) at $\mathbf{A}_n^k$. Each subproblem in (1.6) has a closed form solution. This kind of block-update scheme is new and can be more efficient than alternating minimization as shown later in this thesis for nonnegative tensor decomposition.

BCD has been applied to both convex and non-convex problems. It is relatively easy to establish global convergence for BCD applied to convex smooth optimization. For non-convex smooth problems, only subsequence convergence results have been established for special cases such as [58] considering quadratic functions and [30] assuming strict quasiconvexity of each block subproblem. The global convergence of BCD for non-convex optimization is still an open problem. Non-smoothness also makes it difficult to establish the convergence of BCD even for convex problems (see the review in Section 1.4). This thesis will give a global convergence result of BCD for a special class of non-convex optimization problems which may have non-smooth terms in the objective.

The rest of this chapter first gives a mathematical description of the considered problem and then overviews some interesting examples which arise in applications. These examples are the motivation of my work in this thesis. After that, BCD is formally described, and the existing results of BCD are overviewed. Global convergence analysis and practical performance of BCD will be given in subsequent chapters.

1.1 Problem description

Before describing the problem, let me give a key related definition.

Definition 1.1 (Block multi-convexity)

A set $\mathcal{X} \subset \mathbb{R}^n$ *is called* block multi-convex *under the partition:* $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_s) \in \mathcal{X}$ *if the projection of* $\mathcal{X}$ *to each block of components is convex, namely, for each* i *and fixed* $(s-1)$ *blocks* $\mathbf{x}_1, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_s$*, the set*

$$\mathcal{X}_i(\mathbf{x}_1, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_s) \triangleq \{\mathbf{x}_i \in \mathbb{R}^{n_i} : (\mathbf{x}_1, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_s) \in \mathcal{X}\}$$

is convex. A function $f(\mathbf{x})$ *is called* block multi-convex *if for each* i*,* $f(\mathbf{x})$ *is a convex function of* $\mathbf{x}_i$ *while all the other blocks are fixed.*

In this thesis, I consider the regularized multiconvex optimization problem

$$\min_{\mathbf{x} \in \mathcal{X}} F(\mathbf{x}_1, \cdots, \mathbf{x}_s) \equiv f(\mathbf{x}_1, \cdots, \mathbf{x}_s) + \sum_{i=1}^{s} r_i(\mathbf{x}_i), \tag{1.7}$$

where variable $\mathbf{x}$ is decomposed into s disjoint blocks $\mathbf{x}_1, \cdots, \mathbf{x}_s$, the set $\mathcal{X}$ of feasible points is assumed to be a closed and *block multi-convex* subset of $\mathbb{R}^n$, f is assumed to be a differentiable and *block multi-convex* function, and regularization terms r_i, $i = 1, \cdots, s$, are extended-valued convex functions. The set $\mathcal{X}$ and function f can be non-convex over $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_s)$. However, when all but one blocks are fixed, (1.7) over the free block is a convex problem.

Extended valued means $r_i(\mathbf{x}_i)$ is valued on $\mathbb{R} \cup \{\infty\}$. In particular, r_i (or a part of it) can be an indicator function of convex sets, so r_i can include individual constraints on $\mathbf{x}_i$. I use $\mathbf{x} \in \mathcal{X}$ to model joint constraints and $r_1, \ldots, r_s$ to include

individual constraints of $\mathbf{x}_1, \cdots, \mathbf{x}_s$, respectively, when they are present. In addition, r_i can include non-smooth functions such as ℓ_1-norm $\|\mathbf{x}_i\|_1$ and ℓ_2-norm $\|\mathbf{x}_i\|$, which often give some structures on the solution.

1.2 Motivation by applications

A large number of practical problems can be formulated in the form of (1.7) including both convex and non-convex problems. One convex example arising in signal processing is the basis pursuit (denoising) [19] or more generally, sparse group Lasso [80, 91]

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 + \lambda_1 \sum_{i=1}^{s} \|\mathbf{x}_i\| + \lambda_2 \|\mathbf{x}\|_1, \tag{1.8}$$

where $\mathbf{x}$ has been partitioned into s disjoint blocks: $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_s)$. Another example arising in machine learning is the multi-class logistic regression [27, 12]

$$\min_{\mathbf{W}} -\frac{1}{n} \sum_{i=1}^{n} \left[\sum_{j=1}^{m} y_{ij} (\mathbf{w}_j^\top \mathbf{x}_i) - \log \left(\sum_{j=1}^{m} \exp(\mathbf{w}_j^\top \mathbf{x}_i) \right) \right] + \lambda \|\mathbf{W}\|_F^2,$$

where $y_{ij} = 1$ if data point $\mathbf{x}_i$ belongs to class j and $y_{ij} = 0$ otherwise, and $\mathbf{w}_j$ is the jth column of $\mathbf{W}$.

There are also many non-convex examples such as sparse dictionary learning (1.1) and the ones overviewed below. The work in this thesis is timely and mainly motivated by these non-convex problems.

1.2.1 Blind source separation

Let $\mathbf{s}_1, \ldots, \mathbf{s}_n \in \mathbb{R}^{1\times p}$ be a set of source signals. Given m sensor signals $\mathbf{x}_i = \sum_{j=1}^{n} a_{ij}\mathbf{s}_j + \boldsymbol{\eta}_i, i = 1, \cdots, m$, where $\mathbf{A} = [a_{ij}]_{m\times n} \in \mathbb{R}^{m\times n}$ is an *unknown* mixing matrix and $\boldsymbol{\eta}_i$ is noise, blind source separation (BSS) [36] aims to estimate both $\mathbf{A}$ and $\mathbf{S} = [\mathbf{s}_1^\top, \ldots, \mathbf{s}_n^\top]^\top$. It has found applications in many areas such as artifact removal [35] and image processing [37]. Two classical approaches for BSS are principle component analysis (PCA) [78] and independent component analysis (ICA) [22]. If $m < n$ and no prior information on $\mathbf{A}$ and $\mathbf{S}$ is given, these methods will fail. Assuming $\mathbf{s}_1, \cdots, \mathbf{s}_n$ are sparse under some dictionary $\mathbf{B} \in \mathbb{R}^{T\times p}$, namely, $\mathbf{s}_i = \mathbf{y}_i\mathbf{B}$ and $\mathbf{y}_i \in \mathbb{R}^{1\times T}$ is sparse for $i = 1, \ldots, n$, [96, 15] use the sparse BSS model

$$\min_{\mathbf{A},\mathbf{Y}} \frac{\lambda}{2}\|\mathbf{AYB} - \mathbf{X}\|_F^2 + r(\mathbf{Y}), \text{ subject to } \mathbf{A} \in \mathcal{D} \tag{1.9}$$

where $\mathbf{Y} = [\mathbf{y}_1^\top, \cdots, \mathbf{y}_n^\top]^\top \in \mathbb{R}^{n\times T}$, $r(\mathbf{Y})$ is a sparsity regularizer such as $r(\mathbf{Y}) = \|\mathbf{Y}\|_1$, $\mathcal{D}$ is a convex set to control the scale of $\mathbf{A}$ such as $\|\mathbf{A}\|_F \leq 1$, and λ is a balancing parameter. Note that model (1.9) is block multi-convex in $\mathbf{A}$ and $\mathbf{Y}$ each but jointly non-convex. The block constraint $\mathbf{A} \in \mathcal{D}$ can be included into the objective by adding the indicator function $\delta_{\mathcal{D}}(\mathbf{A})$. A similar model appears in cosmic microwave background analysis [13] which solves

$$\min_{\mathbf{A},\mathbf{Y}} \frac{\lambda}{2}\text{trace}\left((\mathbf{AYB} - \mathbf{X})^\top\mathbf{C}^{-1}(\mathbf{AYB} - \mathbf{X})\right) + r(\mathbf{Y}), \text{ subject to } \mathbf{A} \in \mathcal{D} \tag{1.10}$$

for a certain covariance matrix $\mathbf{C}$. Algorithms for (sparse) BSS include online learning algorithm [1], feature extraction method [52], feature sign algorithm [49], and so on.

1.2.2 Nonnegative matrix factorization

Nonnegative matrix factorization (NMF) was first proposed by Paatero and his coworkers in the area of environmental science [67]. The later popularity of NMF can be partially attributed to the publication of [47] in Nature. It has been widely applied in data mining such as text mining [69] and image mining [50], dimension reduction and clustering [20, 89], hyperspectral endmember extraction, as well as spectral data analysis [68]. A widely used model for (regularized) NMF is

$$\min_{\mathbf{X},\mathbf{Y}} \frac{1}{2}\|\mathbf{XY}-\mathbf{M}\|_F^2 + r_1(\mathbf{X}) + r_2(\mathbf{Y}), \text{ subject to } \mathbf{X}\in\mathbb{R}_+^{m\times r}, \mathbf{Y}\in\mathbb{R}_+^{r\times n} \tag{1.11}$$

where $\mathbf{M}\in\mathbb{R}_+^{m\times n}$ is the input nonnegative matrix, r_1, r_2 are some regularizers promoting solution structures, and

$$\mathbb{R}_+^{m\times n} \triangleq \{\mathbf{A}\in\mathbb{R}^{m\times n} : a_{ij}\geq 0, \forall i,j\}.$$

The block constraints $\mathbf{X}\in\mathbb{R}_+^{m\times r}$ and $\mathbf{Y}\in\mathbb{R}_+^{r\times n}$ can be respectively incorporated into the regularization terms r_1 and r_2 using $\hat{r}_1 = r_1 + \delta_{\mathbb{R}_+^{m\times r}}$ and $\hat{r}_2 = r_2 + \delta_{\mathbb{R}_+^{r\times n}}$. The new regularizers $\hat{r}_1$ and $\hat{r}_2$ both include block constraint and another (non-smooth) term.

Two early popular algorithms for NMF are the projected alternating least squares method [67] and multiplicative updating method [48]. Due to the bi-convexity (multi-convexity under two-block partition) of the objective in (1.11), a series of alternating nonnegative least square (ANLS) methods (alternating minimization) have been proposed such as in [51, 39, 41] to solve (1.11). Recently, the classic alternating direction method (ADM) [29, 28] has been applied in [95] to (1.11).

1.2.3 Low-rank matrix recovery

Similar models of (1.11) also arise in low-rank matrix recovery, such as the one considered in [73]

$$\min_{\mathbf{X},\mathbf{Y}} \frac{1}{2}\|\mathcal{A}(\mathbf{XY}) - \mathbf{b}\|^2 + \alpha\|\mathbf{X}\|_F^2 + \beta\|\mathbf{Y}\|_F^2, \tag{1.12}$$

where $\mathcal{A}$ is a linear operator. A particularly interesting case is the matrix completion problem, where $\mathcal{A}$ is the sampling operator $\mathcal{P}_\Omega$ defined by

$$(\mathcal{P}_\Omega(\mathbf{Z}))_{ij} = \begin{cases} z_{ij}, & \text{if } (i,j) \in \Omega, \\ 0, & \text{otherwise.} \end{cases}$$

A special application of this problem is the famous Netflix problem, which aims to complete users' ratings to all movies from their highly incomplete ratings. Methods for solving (1.12) include augmented Lagrangian method [73], stochastic gradient method [74], and nonlinear succesive over-relaxation (SOR) method [88].

1.2.4 Nonnegative tensor factorization

Nonnegative tensor factorization (NTF) is a generalization of NMF to multi-way arrays. It has been applied in a variety of areas including computer vision [79], hyperspectral analysis [94] and feature selection [9]. One commonly used model for NTF is based on CANDECOMP/PARAFAC tensor decomposition [87]

$$\min_{\mathbf{A}_1,\cdots,\mathbf{A}_N \geq 0} \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N\|_F^2 + \sum_{n=1}^{N} r_n(\mathbf{A}_n); \tag{1.13}$$

and another one is based on Tucker decomposition [43]

$$\min_{\mathcal{G},\mathbf{A}_1,\cdots,\mathbf{A}_N\geq 0} \frac{1}{2}\|\mathcal{M}-\mathcal{G}\times_1\mathbf{A}_1\times_2\mathbf{A}_2\cdots\times_N\mathbf{A}_N\|_F^2+r(\mathcal{G})+\sum_{n=1}^{N} r_n(\mathbf{A}_n), \qquad (1.14)$$

where $\mathcal{M}$ is a given nonnegative tensor, $r, r_1, \cdots, r_N$ are regularizers, and "$\circ$" and "$\times_n$" represent outer product and tensor-matrix multiplication, respectively. The necessary background of tensor will be reviewed in Chapter 3. Using a similar approach as in Section 1.2.2, one can incorporate each block constraint $\mathbf{A}_i \geq 0$ into its corresponding regularization term r_i.

Most algorithms for solving NMF have been directly extended to NTF. For example, the multiplicative update in [67] is extended to solving (1.13) in [79]. The ANLS methods in [39, 41] are extended to solving (1.13) in [40, 42]. Algorithms for solving (1.14) include the column-wise coordinate descent method [53] and the alternating least square method [26]. More about NTF algorithms can be found in [93].

1.3 Method description

For convex optimization, we have a rich set of tools, which own both nice numerical performance and theoretical results. However, very few works have established strong theoretical guarantees for non-convex optimization. Though the methods mentioned in last section are practically efficient for solving the overviewed non-convex examples, to the best of my knowledge, none of them have been shown to globally converge. This motivates me to make an efficient algorithm with global convergence for solving problems in the form of (1.7).

1.3.1 Block coordinate descent with different block updates

Generally, it is difficult to update all the blocks of variables in (1.7) simultaneously but usually easy to do so block by block. My main interest is the *block coordinate descent* (BCD) method of the Gauss-Seidel type, which minimizes F or its relaxation cyclically over each of $\mathbf{x}_1, \cdots, \mathbf{x}_s$ while fixing the remaining blocks at their last updated values. Let $\mathbf{x}_i^k$ denote the value of $\mathbf{x}_i$ after its kth update, and

$$f_i^k(\mathbf{x}_i) \triangleq f(\mathbf{x}_1^k, \cdots, \mathbf{x}_{i-1}^k, \mathbf{x}_i, \mathbf{x}_{i+1}^{k-1}, \cdots, \mathbf{x}_s^{k-1}), \text{ for all } i \text{ and } k. \tag{1.15}$$

Within each cycle, for $i = 1, \ldots, s$, $\mathbf{x}_i$ is updated by one of the next three schemes

$$\text{Block minimization:} \quad \mathbf{x}_i^k = \underset{\mathbf{x}_i \in \mathcal{X}_i^k}{\operatorname{argmin}} f_i^k(\mathbf{x}_i) + r_i(\mathbf{x}_i), \tag{1.16a}$$

$$\text{Block proximal:} \quad \mathbf{x}_i^k = \underset{\mathbf{x}_i \in \mathcal{X}_i^k}{\operatorname{argmin}} f_i^k(\mathbf{x}_i) + \frac{L_i^{k-1}}{2}\|\mathbf{x}_i - \mathbf{x}_i^{k-1}\|^2 + r_i(\mathbf{x}_i), \tag{1.16b}$$

$$\text{Block prox-linear:} \quad \mathbf{x}_i^k = \underset{\mathbf{x}_i \in \mathcal{X}_i^k}{\operatorname{argmin}} \langle \hat{\mathbf{g}}_i^k, \mathbf{x}_i - \hat{\mathbf{x}}_i^{k-1} \rangle + \frac{L_i^{k-1}}{2}\|\mathbf{x}_i - \hat{\mathbf{x}}_i^{k-1}\|^2 + r_i(\mathbf{x}_i), \tag{1.16c}$$

where $L_i^{k-1} > 0$ is some parameter, $\mathcal{X}_i^k \triangleq \mathcal{X}_i(\mathbf{x}_1^k, \cdots, \mathbf{x}_{i-1}^k, \mathbf{x}_{i+1}^{k-1}, \cdots, \mathbf{x}_s^{k-1})$ and in the last type of update (1.16c),

$$\hat{\mathbf{x}}_i^{k-1} = \mathbf{x}_i^{k-1} + \omega_i^{k-1}(\mathbf{x}_i^{k-1} - \mathbf{x}_i^{k-2}) \tag{1.17}$$

denotes an extrapolated point, $\omega_i^{k-1} \geq 0$ is the extrapolation weight, $\hat{\mathbf{g}}_i^k = \nabla f_i^k(\hat{\mathbf{x}}_i^{k-1})$ is the block-partial gradient of f at $\hat{\mathbf{x}}_i^{k-1}$. I consider extrapolation (1.17) for update (1.16c) since it can significantly accelerate the convergence of BCD in the applications; see numerical results in Chapter 4. However, the extrapolation can make the

whole objective increasing and cause numerical instability. Hence, the kth iteration is repeated one time with $\omega_i^{k-1} = 0$ for all blocks updated by (1.16c) whenever $F(\mathbf{x}^k) \geq F(\mathbf{x}^{k-1})$. As long as $\mathbf{x}^k \neq \mathbf{x}^{k-1}$, the algorithm makes $F(\mathbf{x}^k) < F(\mathbf{x}^{k-1})$. The framework of BCD is given in Algorithm 1.

Algorithm 1 Block coordinate descent method for solving (1.7)

Initialization: choose initial points $(\mathbf{x}_1^{-1}, \cdots, \mathbf{x}_s^{-1}) = (\mathbf{x}_1^0, \cdots, \mathbf{x}_s^0)$
for $k = 1, 2, \cdots$ **do**
  for $i = 1, 2, \cdots, s$ **do**
    $\mathbf{x}_i^k \leftarrow$ (1.16a), (1.16b), or (1.16c).
  end for
  if stopping criterion is satisfied **then**
    return $(\mathbf{x}_1^k, \cdots, \mathbf{x}_s^k)$.
  end if
  if $F(\mathbf{x}^k) \geq F(\mathbf{x}^{k-1})$ **then**
    Re-update $\mathbf{x}^k$ with $\omega_i^{k-1} = 0$ for every block i updated by (1.16c).
  end if
end for

Since $\mathcal{X}$ and f are block multi-convex, all three subproblems in (1.16) are convex. In general, the three updates generate different sequences and can thus cause BCD to converge to different solutions. I found in many tests, applying (1.16c) on all or some blocks gives solutions of lower objective values, for a possible reason that its local prox-linear approximation helps avoid the small regions around certain local minima. In addition, it is generally more time consuming to compute (1.16a) and (1.16b) than (1.16c) though each time the former two tend to make larger objective decreases than applying (1.16c) without extrapolation.

There are examples of r_i that make (1.16c) easier to compute than (1.16a) and

(1.16b). For instance, if $r_i = \delta_{\mathcal{D}_i}$ the indicator function of convex set $\mathcal{D}_i$ (equivalent to $\mathbf{x}_i \in \mathcal{D}_i$), (1.16c) reduces to

$$\mathbf{x}_i^k = \mathcal{P}_{\mathcal{X}_i^k \cap \mathcal{D}_i} \left(\hat{\mathbf{x}}_i^{k-1} - \hat{\mathbf{g}}_i^{k-1} / L_i^{k-1} \right), \tag{1.18}$$

where $\mathcal{P}_{\mathcal{X}_i^k \cap \mathcal{D}_i}$ is the project to set $\mathcal{X}_i^k \cap \mathcal{D}_i$. If $r_i(\mathbf{x}_i) = \lambda_i \|\mathbf{x}_i\|_1$ and $\mathcal{X}_i^k = \mathbb{R}^{n_i}$, (1.16c) reduces to

$$\mathbf{x}_i^k = \mathcal{S}_{L_i^{k-1}/\lambda_i} \left(\hat{\mathbf{x}}_i^{k-1} - \hat{\mathbf{g}}_i^{k-1} / L_i^{k-1} \right), \tag{1.19}$$

where $\mathcal{S}_\nu(\cdot)$ is soft-thresholding defined component-wise as $\mathcal{S}_\nu(t) = \text{sign}(t) \max(|t| - \nu, 0)$. More examples arise in joint/group ℓ_1 and nuclear norm minimization, total variation, etc.

1.3.2 Why use different block updates

I consider all of the three updates since they fit different applications, and also different blocks in the same application, yet their convergence can be analyzed in a unified framework. For example, it may be better to apply BCD with (1.16a) to low-rank matrix recovery (1.12), whereas BCD with (1.16c) can be more suitable for nonnegative matrix factorization (1.11). For sparse dictionary learning (1.1), (1.16a) or (1.16b) may be better for $\mathbf{D}$-subproblem, while (1.16c) seems better for $\mathbf{\Theta}$-subproblem.

To ensure the convergence of Algorithm 1, for every block i to which (1.16a) is applied, I will require $f_i^k(\mathbf{x}_i)$ to be strongly convex, and for every block i to which (1.16c) is applied, I will require $\nabla f_i^k(\mathbf{x}_i)$ to be Lipschitz continuous. The parameter L_i^k in both (1.16b) and (1.16c) can be fixed for all k. For generality and faster

convergence, it is allowed to change during the iterations. Use of (1.16b) only requires L_i^k to be uniformly lower bounded from *zero* and uniformly upper bounded. In fact, f_i^k in (1.16b) can be *nonconvex*, and my proofs still go through. (1.16b) is a good replacement of (1.16a) if f_i^k is not strongly convex. Use of (1.16c) requires more conditions on L_i^k; see Lemmas 2.2 and 2.3. (1.16c) is relatively easy to solve and often allows closed form solutions such as in the cases of (1.18) and (1.19). For block i, (1.16c) is prefered over (1.16a) and (1.16b) when the latter two are expensive to solve and f_i^k has Lipschitz continuous gradient. Overall, the three choices cover a large number of cases.

1.4 Overview of existing results

In the literature of BCD, the first two block-update schemes in (1.16) have been widely used and extensively studied, and the third one (1.16c) is quite new and has been used in some very recent works for solving convex problems.

1.4.1 Block minimization scheme

Block minimization (1.16a) is the most-used form in the literature of BCD and has been extensively studied. BCD with block minimization is exactly the alternating minimization method. It is closely related to the Gauss-Seidel or SOR methods in [66] for solving equation systems. It has a long history dating back to 1950s [32], which considers strongly concave quadratic programming. Convergence of alternat-

ing minimization typically requires F to be convex (or pseudoconvex or hemivariate) differentiable and have bounded level sets [86, 92, 4, 70]. With these nice properties, alternating minimization was shown to make the objective converge to optimal value. If F is further assumed to be strictly convex, the iterates themselves will converge to the unique solution [57]. When F is non-convex, alternating minimization may cycle and stagnate [71]. However, subsequence convergence can be obtained for special cases such as quadratic function [58], strict quasi-convexity in each of the first $(s-2)$ blocks [30], unique minimizer per block [56]. If F is non-differentiable, alternating minimization can get stuck at a non-stationary point even if F is convex; see p.94 of [4]. Hence, alternating minimization was generally regarded unsuitable for non-smooth optimization. Nevertheless, if the non-differentiable part is separable such as in the form of (1.7), subsequence convergence can still be obtained for some special cases. The first work considering such a separable nonsmooth structure is by Auslender in [4] which assumes strong convexity of F. In [31], a decomposition method was proposed for minimizing a strictly convex quadratic function over the intersection of convex sets, and it turns out to be alternating minimization for a dual functional of a convex problem in the form of (1.7) as shown in [81]. The work [82] established subsequence convergence result of BCD for non-convex non-smooth problem by assuming separable structure of the non-smooth part and pseudoconvexity of the smooth part. However, global convergence is still unknown for non-convex non-smooth optimization even if the non-smooth part is separable.

1.4.2 Block proximal scheme

Block proximal update (1.16b) can be regarded as alternatively applying one step of the proximal point method to block subproblems. The proximal point method has been extensively studied, and it is usually related to the maximal monotone operator; see [76, 25] and references therein. It has also been used with BCD, and the first work appears to be [5] which applied BCD with block proximal update to convex problems in the form of (1.7). For two-block convex programs, [11] proposed a partial proximal method which iteratively updates both blocks simultaneously by minimizing the objective plus a proximal term involving only one block. The work [30] has extended the method to smooth non-convex optimization with block separable constraints, and it shows that every limit point of the iterates is a critical point. Using the proximal term $\frac{L_i^{k-1}}{2}\|\mathbf{x}_i-\mathbf{x}_i^{k-1}\|^2$ in (1.16b) can often stabilize the iterates as shown in [62] which applied BCD with block proximal update to tensor decomposition and demonstrated that it could reduce a so-called swamp effect. Recently, this method was revisited in [3] for non-convex problems with two blocks and separable non-smooth part, and it was shown that the iterates globally converge to a limit point via the Kurdyka-Łojasiewicz (KL) inequality, whose definition will be given in Chapter 2. However, the global convergence for over-two block non-convex optimization is still unknown.

1.4.3 Block prox-linear scheme

Block prox-linear update (1.16c) with extrapolation is new in the literature of BCD but very similar to the update in the block-coordinate gradient descent (BGD) method of [84], which identifies a block descent direction by gradient projection and then performs an Armijo-type line search. [84] does not use extrapolation (1.17). It considers more general f that is smooth but not necessarily multi-convex, but it does not consider joint constraints. For convex smooth optimization, Nesterov [65] recently proposed a randomized BCD, which randomly selects a block of variables and uses prox-linear method to update the selected block at each step. Meanwhile, he obtained a sublinear convergence of the randomized BCD for convex problems with Lipschitz continuous gradient and linear convergence for strongly convex problems. His work has been extended to convex non-smooth optimization in [75], which obtained similar convergence rate results. However, the convergence rate of cyclic BCD is still unknown, even for strongly convex optimization. In addition, none has applied BCD with block prox-linear update to non-convex optimization. This thesis will also give global convergence of this method and show its high efficiency on nonnegative tensor factorization and completion.

1.5 Contributions

Motivated by many practical problems, I characterize and formulate them into regularized multiconvex optimization. To solve this kind of problem, I choose block

coordinate descent (BCD) method due to its simplicity and efficiency. Also, I generalize BCD by incorporating three different block-update schemes. In addition, I establish global convergence and asymptotic convergence rate in a unified way for BCD with all the three block-update schemes. Then, the algorithm with prox-linear update (1.16c) is applied to two classes of problems (i) nonnegative matrix/tensor *factorization* and (ii) nonnegative matrix/tensor *completion* from incomplete observations, and is demonstrated superior than the state-of-the-art algorithms on both synthetic and real data in both speed and solution quality.

1.6 Organization

The rest of the thesis is organized as follows. Chapter 2 first gives a subsequence convergence result of Algorithm 1. Then it briefly overviews the Kurdyka-Łojasiewicz inequality through which a global convergence result is given. In Chapter 3, tensor is overviewed, and then Algorithm 1 is applied to both the nonnegative matrix/tensor *factorization* problem and the *completion* problem. Numerical results are presented in Chapter 4. Finally, Chapter 5 concludes the thesis.

Chapter 2

Convergence analysis

This chapter will first give some key definitions and then analyze the convergence of Algorithm 1. The convergence analysis takes two steps. Under certain assumptions, the first step establishes the square summable result $\sum_k \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2 < \infty$ and obtains subsequence convergence to Nash points (see Definition 2.6), as well as global convergence to a single Nash point under a fairly strong condition: the sequence of iterates is bounded and the Nash points are isolated. The second step presents a different approach for global convergence; specifically, it assumes the KL inequality [16, 17] and improves the result to $\sum_k \|\mathbf{x}^k - \mathbf{x}^{k+1}\| < \infty$, which gives the algorithm global convergence, as well as asymptotic rates of convergence. The classes of functions that obey the KL inequality are reviewed.

2.1 Elements of analysis

Before starting the analysis, let me give some key definitions which can be found in [77, 83]. Throughout the analysis, I use the vector product $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^\top \mathbf{b} = \sum_{i=1}^n a_i b_i$ for $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ and Euclidean norm $\|\mathbf{a}\| = \sqrt{\langle \mathbf{a}, \mathbf{a} \rangle}$.

Definition 2.1 (Feasible direction)

Let $\mathcal{X}$ be a feasible region and $\mathbf{x} \in \mathcal{X}$. The vector $\mathbf{d}$ is a feasible direction at $\mathbf{x}$ if

there is $\tau > 0$ *such that* $\mathbf{x} + t\mathbf{d} \in \mathcal{X}$ *for any* $t \in [0, \tau)$.

Definition 2.2 (Effective domain)

The effective domain of a function f *on* $\mathbb{R}^n$ *is* $dom(f) = \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) < +\infty\}$.

Definition 2.3 (Limiting Fréchet subdifferential)

A vector $\mathbf{g} \in \mathbb{R}^n$ *is a Fréchet subgradient of continuous function* f *at* $\mathbf{x} \in dom(f)$ *if*

$$\liminf_{\mathbf{y} \to \mathbf{x}, \mathbf{y} \neq \mathbf{x}} \frac{f(\mathbf{y}) - f(\mathbf{x}) - \langle \mathbf{g}, \mathbf{y} - \mathbf{x} \rangle}{\|\mathbf{y} - \mathbf{x}\|} \geq 0.$$

The set of Fréchet subgradient of f *at* $\mathbf{x}$ *is called Fréchet subdifferential and denoted as* $\hat{\partial} f(\mathbf{x})$. *If* $\mathbf{x} \notin dom(f)$, *then* $\partial f(\mathbf{x}) = \emptyset$.

The limiting Fréchet subdifferential is denoted by $\partial f(\mathbf{x})$ *and defined as*

$$\partial f(\mathbf{x}) = \{\mathbf{g} \in \mathbb{R}^n : \text{ there is } \mathbf{x}_n \to \mathbf{x} \text{ and } \mathbf{g}_n \in \hat{\partial} f(\mathbf{x}_n) \text{ such that } \mathbf{g}_n \to \mathbf{g}\}.$$

When f is differentiable at $\mathbf{x}$, $\partial f(\mathbf{x}) = \hat{\partial} f(\mathbf{x}) = \{\nabla f(\mathbf{x})\}$. For convex function f, $\partial f(\mathbf{x}) = \hat{\partial} f(\mathbf{x})$ and is the set of all vectors $\mathbf{g}_\mathbf{x}$ satisfying

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \mathbf{g}_\mathbf{x}, \mathbf{y} - \mathbf{x} \rangle, \ \forall \mathbf{y} \in \text{dom}(f).$$

For problem (1.7), it is shown in [3] that

$$\partial F(\mathbf{x}) = \{\nabla_{\mathbf{x}_1} f(\mathbf{x}) + \partial r_1(\mathbf{x}_1)\} \times \cdots \times \{\nabla_{\mathbf{x}_s} f(\mathbf{x}) + \partial r_s(\mathbf{x}_s)\},$$

where $\mathcal{D}_1 \times \mathcal{D}_2$ denotes the Cartesian product of $\mathcal{D}_1$ and $\mathcal{D}_2$ defined by

$$\mathcal{D}_1 \times \mathcal{D}_2 = \{(\mathbf{d}_1, \mathbf{d}_2) : \mathbf{d}_1 \in \mathcal{D}_1, \mathbf{d}_2 \in \mathcal{D}_2\}.$$

Definition 2.4 (Strong convexity)

A function f is said to be strongly convex with modulus $\mu > 0$ if for any $\mathbf{x}, \mathbf{y} \in dom(f)$,

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \mathbf{g}_{\mathbf{x}}, \mathbf{y} - \mathbf{x} \rangle + \frac{\mu}{2}\|\mathbf{y} - \mathbf{x}\|^2, \text{ for all } \mathbf{g}_{\mathbf{x}} \in \partial f(\mathbf{x}). \tag{2.1}$$

If $\mu = 0$ in (2.1), then f is weakly convex or, commonly called, convex. In addition, if $\mu = 0$ and the inequality holds strictly, f is called strictly convex.

Definition 2.5 (Directional derivative)

For $\mathbf{x} \in dom(f)$, the (lower) directional derivative $f'(\mathbf{x}; \mathbf{d})$ along the direction $\mathbf{d}$ is

$$f'(\mathbf{x}; \mathbf{d}) = \liminf_{t \downarrow 0} \frac{f(\mathbf{x} + t\mathbf{d}) - f(\mathbf{x})}{t}.$$

If f is differentiable at $\mathbf{x}$, then $f'(\mathbf{x}; \mathbf{d}) = \nabla f(\mathbf{x})^\top \mathbf{d}$.

Definition 2.6 (Nash point)

Given a feasible set $\mathcal{X}$, a point $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_s) \in dom(f) \cap \mathcal{X}$ is called a Nash point or block coordinate-wise minimizer if the Nash equilibrium conditions

$$f(\mathbf{x}) \leq f(\mathbf{x} + (\mathbf{0}, \ldots, \mathbf{0}, \mathbf{d}_i, \mathbf{0}, \ldots, \mathbf{0})), \ i = 1, \ldots, s \tag{2.2}$$

hold for any feasible direction $(\mathbf{0}, \ldots, \mathbf{0}, \mathbf{d}_i, \mathbf{0}, \ldots, \mathbf{0})$ at $\mathbf{x}$.

When f is block multi-convex, (2.2) is equivalent to

$$f'(\mathbf{x}; (\mathbf{0}, \ldots, \mathbf{0}, \mathbf{d}_i, \mathbf{0}, \ldots, \mathbf{0})) \geq 0, \ i = 1, \ldots, s. \tag{2.3}$$

According to the definition, a point $\bar{\mathbf{x}}$ is a Nash point of (1.7) if for $i = 1, \cdots, s$,

$$F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i-1}, \bar{\mathbf{x}}_i, \bar{\mathbf{x}}_{i+1}, \cdots, \bar{\mathbf{x}}_s) \leq F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i-1}, \mathbf{x}_i, \bar{\mathbf{x}}_{i+1}, \cdots, \bar{\mathbf{x}}_s), \ \forall \mathbf{x}_i \in \bar{\mathcal{X}}_i, \tag{2.4}$$

or equivalently

$$\langle \nabla_{\mathbf{x}_i} f(\bar{\mathbf{x}}) + \bar{\mathbf{p}}_i, \mathbf{x}_i - \bar{\mathbf{x}}_i \rangle \geq 0, \text{ for all } \mathbf{x}_i \in \bar{\mathcal{X}}_i \text{ and for some } \bar{\mathbf{p}}_i \in \partial r_i(\bar{\mathbf{x}}_i), \tag{2.5}$$

where $\bar{\mathcal{X}}_i = \mathcal{X}_i(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i-1}, \bar{\mathbf{x}}_{i+1}, \cdots, \bar{\mathbf{x}}_s)$.

Definition 2.7 (Critical point)

Given a feasible set $\mathcal{X}$, a point $\mathbf{x} \in dom(f) \cap \mathcal{X}$ is called a critical point or stationary point if $f'(\mathbf{x}; \mathbf{d}) \geq 0$ for any feasible direction $\mathbf{d} \in \mathbb{R}^n$ at $\mathbf{x}$.

If $\mathbf{x}$ is an interior point of $\mathcal{X}$, then $\mathbf{x}$ satisfies (2.3) if and only if $\mathbf{x}$ is a critical point, and in this case, it holds $\mathbf{0} \in \partial f(\mathbf{x})$. A special case is $\mathcal{X} = \mathbb{R}^n$.

Definition 2.8 (Difference measure of two sets)

The difference of two sets $\mathcal{X}, \mathcal{Y}$ is measured by

$$\text{diff}(\mathcal{X}, \mathcal{Y}) = \max \left(\sup_{\mathbf{x} \in \mathcal{X}} \inf_{\mathbf{y} \in \mathcal{Y}} \|\mathbf{x} - \mathbf{y}\|, \sup_{\mathbf{y} \in \mathcal{Y}} \inf_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x} - \mathbf{y}\| \right).$$

If $\lim_{n \to \infty} \text{diff}(\mathcal{X}_n, \mathcal{X}) = 0$, then there is $\mathbf{x}^n \in \mathcal{X}_n$ such that $\lim_{n \to \infty} \|\mathbf{x}^n - \mathbf{x}\| = 0$.

Throughout the analysis, I make the following assumptions.

Assumption 2.1

In (1.7), F is continuous in $dom(F)$ and $\inf_{\mathbf{x} \in dom(F)} F(\mathbf{x}) > -\infty$. Problem (1.7) has a Nash point.

Assumption 2.2

Each block i is updated by the same scheme among (1.16a)–(1.16c) for all k. Let $\mathcal{I}_1, \mathcal{I}_2$ and $\mathcal{I}_3$ denote the set of blocks updated by (1.16a), (1.16b) and (1.16c), respectively. In addition, there exist constants $0 < \ell_i \leq L_i < \infty, i = 1, \cdots, s$ such that

1. *for* $i \in \mathcal{I}_1$, f_i^k *defined in* (1.15) *is* strongly convex *of* $\mathbf{x}_i$ *with modulus* $\ell_i \leq L_i^{k-1} \leq L_i$;

2. *for* $i \in \mathcal{I}_2$, *parameters* L_i^{k-1} *obey* $\ell_i \leq L_i^{k-1} \leq L_i$;

3. *for* $i \in \mathcal{I}_3$, ∇f_i^k *is Lipschitz continuous and parameters* L_i^{k-1} *obey* $\ell_i \leq L_i^{k-1} \leq L_i$ *and*

$$f_i^k(\mathbf{x}_i^k) \leq f_i^k(\hat{\mathbf{x}}_i^{k-1}) + \langle \hat{\mathbf{g}}_i^k, \mathbf{x}_i^k - \hat{\mathbf{x}}_i^{k-1} \rangle + \frac{L_i^{k-1}}{2} \|\mathbf{x}_i^k - \hat{\mathbf{x}}_i^{k-1}\|^2. \tag{2.6}$$

Remark 2.1

The same notation L_i^{k-1} *is used in all three schemes for the simplicity of unified convergence analysis, but I want to emphasize that it has different meanings in the three different schemes. For* $i \in \mathcal{I}_1$, L_i^{k-1} *is not a parameter subject to user selection but a property constant that is determined by the objective and the current values of all other blocks, while for* $i \in \mathcal{I}_2 \cup \mathcal{I}_3$, L_i^{k-1} *is subject to user selection and must meet certain conditions to guarantee convergence. For* $i \in \mathcal{I}_2$, L_i^{k-1} *can be simply fixed to a positive constant or selected by a pre-determined rule to be uniformly lower bounded from* zero *and upper bounded. For* $i \in \mathcal{I}_3$, L_i^{k-1} *is selected to satisfy* (2.6). *Taking* L_i^{k-1} *as the Lipschitz constant of* ∇f_i^k *can satisfy* (2.6). *However, it allows smaller* L_i^{k-1}, *which can speed up the algorithm.*

Remark 2.2

In addition, I want to emphasize that I make different assumptions on the three different schemes. The use of (1.16a) *requires block strong convexity with modulus uniformly away from* zero *and upper bounded, and the use of* (1.16c) *requires block*

Lipschitz continuous gradient. The use of (1.16b) *requires neither strong convexity nor Lipschitz continuity. Even the block convexity is unnecessary for* (1.16b)*, and the proof still goes through. Each assumption on the corresponding scheme guarantees sufficient decrease of the objective and makes square summable; see Lemma 2.2, which plays the key role in the convergence analysis.*

2.2 Subsequence convergence

The analysis in this subsection follows the following steps. First, I show sufficient descent at each step (inequality (2.12) below), from which I establish the square summable result (Lemma 2.2 below). Next, the square summable result is exploited to show that any limit point is a Nash point in Theorem 2.1 below. Finally, with the additional assumptions of isolated Nash points and bounded $\{\mathbf{x}^k\}$, global convergence is obtained in Theorem 2.2 below. The first step is essential while the last two steps use rather standard arguments. I begin with the following lemma similar to Lemma 2.3 of [8]. Since the proof in [8] does not consider constraints, I include a slightly changed proof for completeness.

Lemma 2.1

Let $\xi_1(\mathbf{u})$ and $\xi_2(\mathbf{u})$ be two convex functions defined on the convex set $\mathcal{U}$ and $\xi_1(\mathbf{u})$ be differentiable. Let $\xi(\mathbf{u}) = \xi_1(\mathbf{u}) + \xi_2(\mathbf{u})$ and

$$\mathbf{u}^* = \underset{\mathbf{u}\in\mathcal{U}}{\operatorname{argmin}} \langle \nabla\xi_1(\mathbf{v}), \mathbf{u} - \mathbf{v}\rangle + \frac{L}{2}\|\mathbf{u} - \mathbf{v}\|^2 + \xi_2(\mathbf{u}). \tag{2.7}$$

If

$$\xi_1(\mathbf{u}^*) \leq \xi_1(\mathbf{v}) + \langle \nabla \xi_1(\mathbf{v}), \mathbf{u}^* - \mathbf{v} \rangle + \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2, \tag{2.8}$$

then

$$\xi(\mathbf{u}) - \xi(\mathbf{u}^*) \geq \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2 + L\langle \mathbf{v} - \mathbf{u}, \mathbf{u}^* - \mathbf{v} \rangle \;\; \textit{for any } \mathbf{u} \in \mathcal{U}. \tag{2.9}$$

Proof. The first-order optimality condition of (2.7) holds

$$\langle \nabla \xi_1(\mathbf{v}) + L(\mathbf{u}^* - \mathbf{v}) + \mathbf{g}, \mathbf{u} - \mathbf{u}^* \rangle \geq 0, \text{ for any } \mathbf{u} \in \mathcal{U}, \tag{2.10}$$

for some $\mathbf{g} \in \partial \xi_2(\mathbf{u}^*)$. For any $\mathbf{u} \in \mathcal{U}$, we have

$$\begin{aligned}
\xi(\mathbf{u}) - \xi(\mathbf{u}^*) \geq & \xi(\mathbf{u}) - \big(\xi_1(\mathbf{v}) + \langle \nabla \xi_1(\mathbf{v}), \mathbf{u}^* - \mathbf{v} \rangle + \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2\big) - \xi_2(\mathbf{u}^*) \\
= & \xi_1(\mathbf{u}) - \xi_1(\mathbf{v}) - \langle \nabla \xi_1(\mathbf{v}), \mathbf{u} - \mathbf{v} \rangle + \langle \nabla \xi_1(\mathbf{v}), \mathbf{u} - \mathbf{u}^* \rangle + \xi_2(\mathbf{u}) \\
& - \xi_2(\mathbf{u}^*) - \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2 \\
\geq & \xi_2(\mathbf{u}) - \xi_2(\mathbf{u}^*) - \langle \mathbf{g}, \mathbf{u} - \mathbf{u}^* \rangle - L\langle \mathbf{u}^* - \mathbf{v}, \mathbf{u} - \mathbf{u}^* \rangle - \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2 \\
\geq & - L\langle \mathbf{u}^* - \mathbf{v}, \mathbf{u} - \mathbf{u}^* \rangle - \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2 \\
= & \frac{L}{2}\|\mathbf{u}^* - \mathbf{v}\|^2 + L\langle \mathbf{v} - \mathbf{u}, \mathbf{u}^* - \mathbf{v} \rangle,
\end{aligned}$$

where the first inequality uses (2.8), the second inequality is obtained from the convexity of ξ_1 and (2.10), and the last inequality uses the convexity of ξ_2 and the fact $\mathbf{g} \in \partial \xi_2(\mathbf{u}^*)$. This completes the proof. □

Based on this lemma, I can show the key lemma below.

Lemma 2.2 (Square summable $\|\mathbf{x}^k - \mathbf{x}^{k+1}\|$)

Under Assumptions 2.1 and 2.2, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1

with $0 \le \omega_i^{k-1} \le \delta_\omega \sqrt{\frac{L_i^{k-2}}{L_i^{k-1}}}$ *for* $0 < \delta_\omega < 1$ *uniformly over all* $i \in \mathcal{I}_3$ *and* k. *Then*

$$\sum_{k=0}^{\infty} \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2 < \infty. \tag{2.11}$$

Proof. For $i \in \mathcal{I}_3$, we have the inequality (2.6) by Item 3 of Assumption 2.2. Let $F_i^k \triangleq f_i^k + r_i$ and take $\xi_1 = f_i^k, \xi_2 = r_i, \mathbf{v} = \hat{\mathbf{x}}_i^{k-1}$ and $\mathbf{u} = \mathbf{x}_i^{k-1}$ in (2.9). All the conditions required by Lemma 2.1 are satisfied. Hence, we have

$$\begin{aligned} F_i^k(\mathbf{x}_i^{k-1}) - F_i^k(\mathbf{x}_i^k) &\ge \frac{L_i^{k-1}}{2}\|\hat{\mathbf{x}}_i^{k-1} - \mathbf{x}_i^k\|^2 + L_i^{k-1}\langle \hat{\mathbf{x}}_i^{k-1} - \mathbf{x}_i^{k-1}, \mathbf{x}_i^k - \hat{\mathbf{x}}_i^{k-1}\rangle \\ &= \frac{L_i^{k-1}}{2}\|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2 - \frac{L_i^{k-1}}{2}(\omega_i^{k-1})^2\|\mathbf{x}_i^{k-2} - \mathbf{x}_i^{k-1}\|^2 \qquad (2.12) \\ &\ge \frac{L_i^{k-1}}{2}\|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2 - \frac{L_i^{k-2}}{2}\delta_\omega^2\|\mathbf{x}_i^{k-2} - \mathbf{x}_i^{k-1}\|^2. \qquad (2.13) \end{aligned}$$

For $i \in \mathcal{I}_1 \cup \mathcal{I}_2$, we have $F_i^k(\mathbf{x}_i^{k-1}) - F_i^k(\mathbf{x}_i^k) \ge \frac{L_i^{k-1}}{2}\|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2$, which can be obtained by the strong convexity of $F_i^k(\mathbf{x}_i)$ for $i \in \mathcal{I}_1$ or the update (1.16b) for $i \in \mathcal{I}_2$, and thus the inequality (2.13) still holds. Therefore,

$$\begin{aligned} F(\mathbf{x}^{k-1}) - F(\mathbf{x}^k) &= \sum_{i=1}^{s}\left(F_i^k(\mathbf{x}_i^{k-1}) - F_i^k(\mathbf{x}_i^k)\right) \\ &\ge \sum_{i=1}^{s}\left(\frac{L_i^{k-1}}{2}\|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2 - \frac{L_i^{k-2}\delta_\omega^2}{2}\|\mathbf{x}_i^{k-2} - \mathbf{x}_i^{k-1}\|^2\right). \end{aligned}$$

Summing the above inequality over k from 1 to K, we have

$$\begin{aligned} F(\mathbf{x}^0) - F(\mathbf{x}^K) &\ge \sum_{k=1}^{K}\sum_{i=1}^{s}\left(\frac{L_i^{k-1}}{2}\|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2 - \frac{L_i^{k-2}}{2}\delta_\omega^2\|\mathbf{x}_i^{k-2} - \mathbf{x}_i^{k-1}\|^2\right) \\ &\ge \sum_{k=1}^{K}\sum_{i=1}^{s}\frac{(1-\delta_\omega^2)L_i^{k-1}}{2}\|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2 \ge \sum_{k=1}^{K}\frac{(1-\delta_\omega^2)\ell}{2}\|\mathbf{x}^{k-1} - \mathbf{x}^k\|^2, \end{aligned}$$

where $\ell = \min_i \ell_i > 0$. Since $0 < \delta_\omega < 1$ and F is lower bounded, taking $K \to \infty$ completes the proof. □

Remark 2.3

From the proof above, we can see that Algorithm 1 makes $F(\mathbf{x}^k) > F(\mathbf{x}^{k+1})$ *as long as* $\mathbf{x}^k \neq \mathbf{x}^{k+1}$.

Now, I can establish the following subsequence convergence result.

Theorem 2.1 (Limit point is Nash point)

Under the assumptions of Lemma 2.2 and further assuming the set map $\mathcal{X}_i(\cdot)$ *continuously changes in* $\mathcal{X}$, *namely,* $\mathbf{x}^{k'}, \mathbf{x} \in \mathcal{X}$ *and* $\mathbf{x}^{k'} \to \mathbf{x}$ *imply*

$$\lim_{k\to\infty} \operatorname{diff}\left(\mathcal{X}_i(\mathbf{x}_1^{k'}, \ldots, \mathbf{x}_{i-1}^{k'}, \mathbf{x}_{i+1}^{k'}, \ldots, \mathbf{x}_s^{k'}), \mathcal{X}_i(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_s)\right) = 0, \ \forall i,$$

then any limit point $\bar{\mathbf{x}}$ *of* $\{\mathbf{x}^k\}$ *is a Nash point, namely, satisfying the Nash equilibrium conditions* (2.4) *or* (2.5).

Proof. Let $\bar{\mathbf{x}}$ be a limit point of $\{\mathbf{x}^k\}$ and $\{\mathbf{x}^{k_j}\}$ be the subsequence converging to $\bar{\mathbf{x}}$. The closedness of $\mathcal{X}$ implies $\bar{\mathbf{x}} \in \mathcal{X}$. Since $\{L_i^k\}$ is bounded, passing another subsequence if necessary, we have $L_i^{k_j}$ converges to some $\bar{L}_i$ for $i = 1, \cdots, s$ as $j \to \infty$. Inequality (2.11) implies that $\|\mathbf{x}^{k+1} - \mathbf{x}^k\| \to 0$, so $\{\mathbf{x}^{k_j+1}\}$ also converges to $\bar{\mathbf{x}}$.

For $i \in \mathcal{I}_1$, we have

$$F_i^{k_j+1}(\mathbf{x}_i^{k_j+1}) \leq F_i^{k_j+1}(\mathbf{x}_i), \ \forall \mathbf{x}_i \in \mathcal{X}_i^{k_j+1}. \tag{2.14}$$

Letting $j \to \infty$, we can show (2.4) by the continuity of F and the set map $\mathcal{X}_i(\cdot)$. Suppose otherwise for some block $i_0 \in \mathcal{I}_1$, there is $\mathbf{y}_{i_0} \in \bar{\mathcal{X}}_{i_0}$ such that

$$F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i_0-1}, \bar{\mathbf{x}}_{i_0}, \bar{\mathbf{x}}_{i_0+1}, \cdots, \bar{\mathbf{x}}_s) > F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i_0-1}, \mathbf{y}_{i_0}, \bar{\mathbf{x}}_{i_0+1}, \cdots, \bar{\mathbf{x}}_s). \tag{2.15}$$

Since $\lim_j \operatorname{diff}\left(\mathcal{X}_{i_0}^{k_j+1}, \bar{\mathcal{X}}_{i_0}\right) = 0$, there is $\mathbf{y}_{i_0}^j \in \mathcal{X}_{i_0}^{k_j+1}$ such that $\lim_j \mathbf{y}_{i_0}^j = \mathbf{y}_{i_0}$. From the continuity of F, we have

$$\lim_{j\to\infty} F(\mathbf{x}_1^{k_j+1}, \ldots, \mathbf{x}_{i_0-1}^{k_j+1}, \mathbf{y}_{i_0}^j, \mathbf{x}_{i_0+1}^{k_j}, \ldots, \mathbf{x}_s^{k_j}) = F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i_0-1}, \mathbf{y}_{i_0}, \bar{\mathbf{x}}_{i_0+1}, \cdots, \bar{\mathbf{x}}_s). \tag{2.16}$$

Note that (2.14) implies

$$F(\mathbf{x}_1^{k_j+1}, \ldots, \mathbf{x}_{i_0-1}^{k_j+1}, \mathbf{x}_{i_0}^{k_j+1}, \mathbf{x}_{i_0+1}^{k_j}, \ldots, \mathbf{x}_s^{k_j}) \le F(\mathbf{x}_1^{k_j+1}, \ldots, \mathbf{x}_{i_0-1}^{k_j+1}, \mathbf{y}_{i_0}^j, \mathbf{x}_{i_0+1}^{k_j}, \ldots, \mathbf{x}_s^{k_j}).$$

Letting $j \to \infty$ and using (2.16), we get

$$F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i_0-1}, \bar{\mathbf{x}}_{i_0}, \bar{\mathbf{x}}_{i_0+1}, \cdots, \bar{\mathbf{x}}_s) \le F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i_0-1}, \mathbf{y}_{i_0}, \bar{\mathbf{x}}_{i_0+1}, \cdots, \bar{\mathbf{x}}_s),$$

which contradicts to (2.15). Hence, (2.4) holds.

Similarly, for $i \in \mathcal{I}_2$, we have for any $\mathbf{x}_i \in \bar{\mathcal{X}}_i$

$$F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i-1}, \bar{\mathbf{x}}_i, \bar{\mathbf{x}}_{i+1}, \cdots, \bar{\mathbf{x}}_s) \le F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i-1}, \mathbf{x}_i, \bar{\mathbf{x}}_{i+1}, \cdots, \bar{\mathbf{x}}_s) + \frac{\bar{L}_i}{2}\|\mathbf{x}_i - \bar{\mathbf{x}}_i\|^2,$$

namely,

$$\bar{\mathbf{x}}_i = \operatorname*{argmin}_{\mathbf{x}_i \in \bar{\mathcal{X}}_i} F(\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_{i-1}, \mathbf{x}_i, \bar{\mathbf{x}}_{i+1}, \cdots, \bar{\mathbf{x}}_s) + \frac{\bar{L}_i}{2}\|\mathbf{x}_i - \bar{\mathbf{x}}_i\|^2. \tag{2.17}$$

Thus, $\bar{\mathbf{x}}_i$ satisfies the first-order optimality condition of (2.17), which is precisely (2.5).

For $i \in \mathcal{I}_3$, we have

$$\mathbf{x}_i^{k_j+1} = \operatorname*{argmin}_{\mathbf{x}_i \in \mathcal{X}_i^{k_j+1}} \langle \nabla f_i^{k_j+1}(\hat{\mathbf{x}}_i^{k_j}), \mathbf{x}_i - \hat{\mathbf{x}}_i^{k_j} \rangle + \frac{L_i^{k_j}}{2}\|\mathbf{x}_i - \hat{\mathbf{x}}_i^{k_j}\|^2 + r_i(\mathbf{x}_i).$$

The convex proximal minimization is continuous in the sense that the output $\mathbf{x}_i^{k_j+1}$ depends continuously on the input $\hat{\mathbf{x}}_i^{k_j}$ [76]. Letting $j \to \infty$, from $\mathbf{x}_i^{k_j+1} \to \bar{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_i^{k_j} \to \bar{\mathbf{x}}_i$, we get

$$\bar{\mathbf{x}}_i = \underset{\mathbf{x}_i \in \bar{\mathcal{X}}_i}{\operatorname{argmin}} \ \langle \nabla_{\mathbf{x}_i} f(\bar{\mathbf{x}}), \mathbf{x}_i - \bar{\mathbf{x}}_i \rangle + \frac{\bar{L}_i}{2} \|\mathbf{x}_i - \bar{\mathbf{x}}_i\|^2 + r_i(\mathbf{x}_i). \tag{2.18}$$

Hence, $\bar{\mathbf{x}}_i$ satisfies the first-order optimality condition of (2.18), which is precisely (2.5). This completes the proof. □

Remark 2.4

The continuity of $\mathcal{X}_i(\cdot)$ holds if $\mathcal{X}$ is convex; see Theorem 4.32 in [77]. A special case is $\mathcal{X} = \mathbb{R}^n$, in which case we can immediately have the following corollary.

Corollary 2.1

Let $\mathcal{X} = \mathbb{R}^n$ in (1.7). Under the assumptions of Lemma 2.2, any limit point $\bar{\mathbf{x}}$ of $\{\mathbf{x}^k\}$ is a critical point of (1.7), namely, $\mathbf{0} \in \partial F(\bar{\mathbf{x}})$.

Theorem 2.2 (Global convergence given isolated Nash points)

Let $\mathcal{N}$ be the set of Nash points of (1.7). Under the assumptions of Lemma 2.2, we have $\operatorname{dist}(\mathbf{x}^k, \mathcal{N}) \to 0$, if $\{\mathbf{x}^k\}$ is bounded. Further, if $\mathcal{N}$ contains uniformly isolated points, namely, there is $\eta > 0$ such that $\|\mathbf{x} - \mathbf{y}\| \geq \eta$ for any distinct points $\mathbf{x}, \mathbf{y} \in \mathcal{N}$, then $\mathbf{x}^k$ converges to a point in $\mathcal{N}$.

Moreover, if F is strictly convex and $\mathcal{X} = \mathbb{R}^n$, then $\mathbf{x}^k$ converges to the unique solution of (1.7).

Proof. Suppose $\operatorname{dist}(\mathbf{x}^k, \mathcal{N})$ does not converge to 0. Then there exists $\varepsilon > 0$ and a subsequence $\{\mathbf{x}^{k_j}\}$ such that $\operatorname{dist}(\mathbf{x}^{k_j}, \mathcal{N}) \geq \varepsilon$ for all j. However, the boundedness of

$\{\mathbf{x}^{k_j}\}$ implies that it must have a limit point $\bar{\mathbf{x}} \in \mathcal{N}$ according to Theorem 2.1, which is a contradiction.

From $\text{dist}(\mathbf{x}^k, \mathcal{N}) \to 0$, it follows that there is an integer $K_1 > 0$ such that $\mathbf{x}^k \in \cup_{\mathbf{y}\in\mathcal{N}} B(\mathbf{y}, \frac{\eta}{3})$ for all $k \geq K_1$, where $B(\mathbf{y}, \frac{\eta}{3}) \triangleq \{\mathbf{x} \in \mathcal{X} : \|\mathbf{x} - \mathbf{y}\| < \frac{\eta}{3}\}$. In addition, Lemma 2.2 implies that there exists another integer $K_2 > 0$ such that $\|\mathbf{x}^k - \mathbf{x}^{k+1}\| < \frac{\eta}{3}$ for all $k \geq K_2$. Take $K = \max(K_1, K_2)$ and assume $\mathbf{x}^K \in B(\bar{\mathbf{x}}, \frac{\eta}{3})$ for some $\bar{\mathbf{x}} \in \mathcal{N}$. We claim that for any $\bar{\mathbf{x}} \neq \mathbf{y} \in \mathcal{N}$, $\|\mathbf{x}^k - \mathbf{y}\| > \frac{\eta}{3}$ holds for all $k \geq K$. This claim can be shown by induction on $k \geq K$. If some $\mathbf{x}^k \in B(\bar{\mathbf{x}}, \frac{\eta}{3})$, then

$$\|\mathbf{x}^{k+1} - \bar{\mathbf{x}}\| \leq \|\mathbf{x}^{k+1} - \mathbf{x}^k\| + \|\mathbf{x}^k - \bar{\mathbf{x}}\| < \frac{2\eta}{3},$$

and

$$\|\mathbf{x}^{k+1} - \mathbf{y}\| \geq \|\bar{\mathbf{x}} - \mathbf{y}\| - \|\mathbf{x}^{k+1} - \bar{\mathbf{x}}\| > \frac{\eta}{3}, \text{ for any } \bar{\mathbf{x}} \neq \mathbf{y} \in \mathcal{N}.$$

Therefore, $\mathbf{x}^k \in B(\bar{\mathbf{x}}, \frac{\eta}{3})$ for all $k \geq K$ since $\mathbf{x}^k \in \cup_{\mathbf{y}\in\mathcal{N}} B(\mathbf{y}, \frac{\eta}{3})$, and thus $\{\mathbf{x}^k\}$ has the unique limit point $\bar{\mathbf{x}}$, which means $\mathbf{x}^k \to \bar{\mathbf{x}}$.

When F is strictly convex and $\mathcal{X} = \mathbb{R}^n$, there is only one critical point, which is the unique solution. Hence, $\mathbf{x}^k$ converges to the unique solution. □

Remark 2.5

The boundedness of $\{\mathbf{x}^k\}$ is guaranteed if the level set $\{\mathbf{x} \in \mathcal{X} : F(\mathbf{x}) \leq F(\mathbf{x}^0)\}$ is bounded. However, the isolation assumption does not hold, or holds but is difficult to verify, for many functions. This motivates another approach below for global convergence.

2.3 Kurdyka-Łojasiewicz inequality

Before proceeding with the analysis, let me briefly review the Kurdyka-Łojasiewicz inequality, which is central to the global convergence analysis in the next subsection.

Definition 2.9 (Kurdyka-Łojasiewicz property)

A function $\psi(\mathbf{x})$ satisfies the Kurdyka-Łojasiewicz (KL) property at point $\bar{\mathbf{x}} \in \mathrm{dom}(\partial\psi)$ if there exists $\theta \in [0, 1)$ such that

$$\frac{|\psi(\mathbf{x}) - \psi(\bar{\mathbf{x}})|^\theta}{\mathrm{dist}(\mathbf{0}, \partial\psi(\mathbf{x}))} \tag{2.19}$$

is bounded around $\bar{\mathbf{x}}$ under the notational conventions: $0^0 = 1, \infty/\infty = 0/0 = 0$. In other words, in a certain neighborhood $\mathcal{U}$ of $\bar{\mathbf{x}}$, there exists $\phi(s) = cs^{1-\theta}$ for some $c > 0$ and $\theta \in [0, 1)$ such that the KL inequality holds

$$\phi'(|\psi(\mathbf{x}) - \psi(\bar{\mathbf{x}})|)\mathrm{dist}(\mathbf{0}, \partial\psi(\mathbf{x})) \geq 1, \forall \mathbf{x} \in \mathcal{U} \cap \mathrm{dom}(\partial\psi) \text{ and } \psi(\mathbf{x}) \neq \psi(\bar{\mathbf{x}}), \tag{2.20}$$

where $\mathrm{dom}(\partial\psi) \triangleq \{\mathbf{x} : \partial\psi(\mathbf{x}) \neq \emptyset\}$ and $\mathrm{dist}(\mathbf{0}, \partial\psi(\mathbf{x})) \triangleq \min\{\|\mathbf{y}\| : \mathbf{y} \in \partial\psi(\mathbf{x})\}$.

This property was introduced by Łojasiewicz [55] on real analytic functions, for which the term with $\theta \in [\frac{1}{2}, 1)$ in (2.19) is bounded around any critical point $\bar{\mathbf{x}}$. Kurdyka extended this property to functions on the o-minimal structure in [45]. Recently, the KL inequality was extended to nonsmooth sub-analytic functions [16]. Since it is not trivial to check the conditions in the definition, I give some examples below that satisfy the KL inequality.

Real analytic functions

A smooth function $\varphi(t)$ on $\mathbb{R}$ is analytic if $\left(\frac{\varphi^{(k)}(t)}{k!}\right)^{\frac{1}{k}}$ is bounded for all k and on any compact set $\mathcal{D} \subset \mathbb{R}$. One can verify whether a real function $\psi(\mathbf{x})$ on $\mathbb{R}^n$ is analytic by checking the analyticity of $\varphi(t) \triangleq \psi(\mathbf{x} + t\mathbf{y})$ for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. For example, any polynomial function is real analytic such as $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ and the first terms in the objectives of (1.13) and (1.14). In addition, it is not difficult to verify that the non-convex function $\sum_{i=1}^n (x_i^2 + \varepsilon^2)^{q/2} + \frac{1}{2\lambda}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ with $0 < q < 1$ and $\varepsilon > 0$ considered in [46] for sparse vector recovery is a real analytic function (the first term is the ε-smoothed ℓ_q semi-norm). The logistic loss function $\log(1 + e^{-t})$ is also analytic. Therefore, all the above functions satisfy the KL property with $\theta \in [\frac{1}{2}, 1)$ in (2.19).

Locally strongly convex functions

A function $\psi(\mathbf{x})$ is strongly convex in a neighborhood $\mathcal{D}$ with modulus $\mu > 0$ if

$$\psi(\mathbf{y}) \geq \psi(\mathbf{x}) + \langle \gamma(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\mu}{2}\|\mathbf{x} - \mathbf{y}\|^2, \text{ for all } \gamma(\mathbf{x}) \in \partial\psi(\mathbf{x}) \text{ and for any } \mathbf{x}, \mathbf{y} \in \mathcal{D}.$$

According to the definition and using the Cauchy-Schwarz inequality, we have

$$\psi(\mathbf{y}) - \psi(\mathbf{x}) \geq \langle \gamma(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\mu}{2}\|\mathbf{x} - \mathbf{y}\|^2 \geq -\frac{1}{\mu}\|\gamma(\mathbf{x})\|^2, \text{ for all } \gamma(\mathbf{x}) \in \partial\psi(\mathbf{x}).$$

Hence, $\mu(\psi(\mathbf{x}) - \psi(\mathbf{y})) \leq \operatorname{dist}(\mathbf{0}, \partial\psi(\mathbf{x}))^2$, and ψ satisfies the KL inequality (2.20) at any point $\mathbf{y} \in \mathcal{D}$ with $\phi(s) = \frac{2}{\mu}\sqrt{s}$ and $\mathcal{U} = \mathcal{D} \cap \{\mathbf{x} : \psi(\mathbf{x}) \geq \psi(\mathbf{y})\}$. For example, the logistic loss function $\log(1 + e^{-t})$ is strongly convex in any bounded set $\mathcal{D}$, so it

has the KL property with $\theta = \frac{1}{2}$.

Semi-algebraic functions

A set $\mathcal{D} \subset \mathbb{R}^n$ is called *semi-algebraic* [14] if it can be represented as

$$\mathcal{D} = \bigcup_{i=1}^{s} \bigcap_{j=1}^{t} \{\mathbf{x} \in \mathbb{R}^n : p_{ij}(\mathbf{x}) = 0, q_{ij}(\mathbf{x}) > 0\},$$

where p_{ij}, q_{ij} are real polynomial functions for $1 \leq i \leq s$, $1 \leq j \leq t$. A function ψ is called *semi-algebraic* if its graph $\mathrm{Gr}(\psi) \triangleq \{(\mathbf{x}, \psi(\mathbf{x})) : \mathbf{x} \in \mathrm{dom}(\psi)\}$ is a *semi-algebraic* set.

Semi-algebraic functions are sub-analytic, so they satisfy the KL inequality according to [16, 17]. I list some elementary properties of semi-algebraic sets and functions below as they help identify semi-algebraic functions.

1. If a set $\mathcal{D}$ is semi-algebraic, so is its closure $\mathrm{cl}(\mathcal{D})$.

2. If $\mathcal{D}_1$ and $\mathcal{D}_2$ are both semi-algebraic, so are $\mathcal{D}_1 \cup \mathcal{D}_2$, $\mathcal{D}_1 \cap \mathcal{D}_2$ and $\mathbb{R}^n \backslash \mathcal{D}_1$.

3. Indicator functions of semi-algebraic sets are semi-algebraic.

4. Finite sums and products of semi-algebraic functions are semi-algebraic.

5. The composition of semi-algebraic functions is semi-algebraic.

From items 1 and 2, any polyhedral set is semi-algebraic such as the nonnegative orthant $\mathbb{R}^n_+ = \{\mathbf{x} \in \mathbb{R}^n : x_i \geq 0, \forall i\}$. Hence, the indicator function $\delta_{\mathbb{R}^n_+}$ is a semi-algebraic function. The absolute value function $\varphi(t) = |t|$ is also semi-algebraic since

its graph is $\mathrm{cl}(\mathcal{D})$, where

$$\mathcal{D} = \{(t,s) : t+s=0, -t>0\} \cup \{(t,s) : t-s=0, t>0\}.$$

Hence, the ℓ_1-norm $\|\mathbf{x}\|_1$ is semi-algebraic since it is the finite sum of absolute functions. In addition, the sup-norm $\|\mathbf{x}\|_\infty$ is semi-algebraic, which can be shown by observing

$$\mathrm{Graph}(\|\mathbf{x}\|_\infty) = \{(\mathbf{x},t) : t = \max_j |x_j|\} = \bigcup_i \{(\mathbf{x},t) : |x_i| = t, |x_j| \leq t, \forall j \neq i\}.$$

Further, the Euclidean norm $\|\mathbf{x}\|$ is shown to be semi-algebraic in [14]. According to item 5, $\|\mathbf{Ax}-\mathbf{b}\|_1$, $\|\mathbf{Ax}-\mathbf{b}\|_\infty$ and $\|\mathbf{Ax}-\mathbf{b}\|$ are all semi-algebraic functions.

Sum of real analytic and semi-algebraic functions

Both real analytic and semi-algebraic functions are sub-analytic. According to [14], if ψ_1 and ψ_2 are both sub-analytic and ψ_1 maps bounded sets to bounded sets, then $\psi_1+\psi_2$ is also sub-analytic. Since real analytic functions map bounded set to bounded set, the sum of a real analytic function and a semi-algebraic function is sub-analytic, so the sum satisfies the KL property. For example, the sparse logistic regression function

$$\psi(\mathbf{x},b) = \frac{1}{n}\sum_{i=1}^{n} \log\left(1+\exp\left(-c_i(\mathbf{a}_i^\top \mathbf{x}+b)\right)\right) + \lambda\|\mathbf{x}\|_1$$

is sub-analytic and satisfies the KL inequality.

2.4 Global convergence and rate

If $\{\mathbf{x}^k\}$ is bounded, then Theorem 2.1 guarantees that there exists one subsequence converging to a Nash point of (1.7). In this subsection, I assume $\mathcal{X} = \mathbb{R}^n$ and strengthen this result for problems with F obeying the KL inequality. Recall that any Nash point is a critical point when $\mathcal{X} = \mathbb{R}^n$. The analysis here was motivated by [3], which applies the inequality to establish the global convergence of the alternating proximal point method — the special case of BCD with two blocks and using update (1.16b).

In the sequel, I use the notation $F_k = F(\mathbf{x}^k)$ and $\bar{F} = F(\bar{\mathbf{x}})$. First, let me establish the following pre-convergence result, the proof of which is given in the Appendix.

Lemma 2.3 (Pre-convergence)

Under Assumptions 2.1 and 2.2, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1. Assume

1. *$L_i^k \geq \ell^{k-1} = \min_{i\in\mathcal{I}_3} L_i^{k-1}$ and $\omega_i^k \leq \delta_\omega \sqrt{\frac{\ell^{k-1}}{L_i^k}}$, $\delta_\omega < 1$, for all $i \in \mathcal{I}_3$ and k;*

2. *∇f is Lipschitz continuous on any bounded set;*

3. *F satisfies the KL inequality at $\bar{\mathbf{x}}$, namely, there exists $\phi(s) = cs^{1-\theta}$ for some $c > 0$ and $\theta \in [0, 1)$ such that within some neighborhood $\mathcal{U}$ of $\bar{\mathbf{x}}$, it holds*

$$\phi'(|F(\mathbf{x})-F(\bar{\mathbf{x}})|)\mathrm{dist}(\mathbf{0}, \partial F(\mathbf{x})) \geq 1, \forall \mathbf{x} \in \mathcal{U}\cap\mathrm{dom}(\partial F) \text{ and } F(\mathbf{x}) \neq F(\bar{\mathbf{x}}); \quad (2.21)$$

4. *$\mathbf{x}_0$ is* sufficiently close *to $\bar{\mathbf{x}}$ and $F_k > \bar{F}$ for $k \geq 0$.*

Then there exists some $\mathcal{B} \subset \mathcal{U} \cap \mathrm{dom}(\partial F)$ *such that* $\{\mathbf{x}^k\} \subset \mathcal{B}$ *and* $\mathbf{x}^k$ *converges to a point in* $\mathcal{B}$.

Remark 2.6

In the lemma, the required closeness of $\mathbf{x}^0$ *to* $\bar{\mathbf{x}}$ *depends on* $\mathcal{U}, \phi$ *and* F*; see the inequality in* (A.1)*. The extrapolation weight* ω_i^k *must be smaller than it is in Lemma 2.2 in order to guarantee sufficient decrease at each iteration.*

The following corollary is a straightforward application of Lemma 2.3.

Corollary 2.2 (Local convergence to global minimizer)

Under the assumptions of Lemma 2.3, $\{\mathbf{x}^k\}$ *converges to a global minimizer of* (1.7) *if the initial point* $\mathbf{x}^0$ *is sufficiently close to any global minimizer* $\bar{\mathbf{x}}$.

Proof. Suppose $F(\mathbf{x}^{k_0}) = F(\bar{\mathbf{x}})$ at some k_0. Then $\mathbf{x}^k = \mathbf{x}^{k_0}$ for all $k \geq k_0$, according to Remark 2.3. Now consider $F(\mathbf{x}^k) > F(\bar{\mathbf{x}})$ for all $k \geq 0$, and thus Lemma 2.3 implies that $\mathbf{x}^k$ converges to some critical point $\mathbf{x}^*$ if $\mathbf{x}^0$ is sufficiently close to $\bar{\mathbf{x}}$, where $\mathbf{x}^0, \mathbf{x}^*, \bar{\mathbf{x}} \in \mathcal{B}$. If $F(\mathbf{x}^*) > F(\bar{\mathbf{x}})$, then the KL inequality (2.21) indicates $\phi'\left(F(\mathbf{x}^*) - F(\bar{\mathbf{x}})\right) \mathrm{dist}\left(\mathbf{0}, \partial F(\mathbf{x}^*)\right) \geq 1$, which is impossible since $\mathbf{0} \in \partial F(\mathbf{x}^*)$. □

Next, I give the global convergence result of Algorithm 1.

Theorem 2.3 (Global convergence)

Under the assumptions of Lemma 2.3 and that $\{\mathbf{x}^k\}$ *has a finite limit point* $\bar{\mathbf{x}}$ *where* F *satisfies the KL inequality* (2.21)*, the sequence* $\{\mathbf{x}^k\}$ *converges to* $\bar{\mathbf{x}}$*, which is a critical point of* (1.7).

Proof. Note that $F(\mathbf{x}^k)$ is monotonically nonincreasing and converges to $F(\bar{\mathbf{x}})$. If

$F(\mathbf{x}^{k_0}) = F(\bar{\mathbf{x}})$ at some k_0, then $\mathbf{x}^k = \mathbf{x}^{k_0} = \bar{\mathbf{x}}$ for all $k \geq k_0$ from Remark 2.3. It remains to consider $F(\mathbf{x}^k) > F(\bar{\mathbf{x}})$ for all $k \geq 0$. Since $\bar{\mathbf{x}}$ is a limit point and $F(\mathbf{x}^k) \to F(\bar{\mathbf{x}})$, there must exist an integer k_0 such that $\mathbf{x}^{k_0}$ is sufficiently close to $\bar{\mathbf{x}}$ as required in Lemma 2.3 (see the inequality in (A.1)). The conclusion now directly follows from Lemma 2.3. □

I can also estimate the asymptotic rate of convergence, and the proof is given in the Appendix.

Theorem 2.4 (Convergence rate)

Assume the assumptions of Lemma 2.3, and suppose that $\mathbf{x}^k$ *converges to a critical point* $\bar{\mathbf{x}}$*, at which* F *satisfies the KL inequality* (2.21) *with* $\phi(s) = cs^{1-\theta}$ *for* $c > 0$ *and* $\theta \in [0, 1)$*. Then*

1. *If* $\theta = 0$, $\mathbf{x}^k$ *converges to* $\bar{\mathbf{x}}$ *in finite iterations;*

2. *If* $\theta \in (0, \frac{1}{2}]$, $\|\mathbf{x}^k - \bar{\mathbf{x}}\| \leq C\tau^k,\ \forall k \geq k_0$, *for certain* $k_0 > 0,\ C > 0,\ \tau \in [0, 1)$*;*

3. *If* $\theta \in (\frac{1}{2}, 1)$, $\|\mathbf{x}^k - \bar{\mathbf{x}}\| \leq Ck^{-(1-\theta)/(2\theta-1)},\ \forall k \geq k_0$, *for certain* $k_0 > 0,\ C > 0$.

When F is strongly convex, global linear convergence can be obtained. The result is summarized in the next theorem.

Theorem 2.5 (Global linear convergence for strongly convex optimization)

Under Assumptions 2.1 and 2.2, let $\{\mathbf{x}^k\}$ *be the sequence generated by Algorithm 1. Let* $\ell^k = \min_{i \in \mathcal{I}_3} L_i^k$*, and choose* $L_i^k \geq \ell^{k-1}$ *and* $\omega_i^k \leq \delta_\omega \sqrt{\frac{\ell^{k-1}}{L_i^k}}$, $\delta_\omega < 1$, *for all* $i \in \mathcal{I}_3$ *and* k*. If* F *is strongly convex, then* $\mathbf{x}^k$ *globally linearly converges to the unique solution* $\mathbf{x}^*$ *of* (1.7).

Proof. When F is strongly convex, it has the KL property (2.19) with $\theta = \frac{1}{2}$ at $\mathbf{x}^*$. Hence, item 2 in Theorem 2.4 holds. In addition, note that the neighborhood $\mathcal{U}$ in (2.20) is the whole space as F is strongly convex, so $k_0 = 0$ in item 2. This completes the proof. □

Chapter 3

Nonnegative tensor factorization and completion

In this section, I apply Algorithm 1 to the factorization and completion of nonnegative matrices and tensors. Since a matrix is a two-way tensor, I present the algorithm for tensors. I will first overview tensor and its two popular factorizations and then describe in details how to apply Algorithm 1 to nonnegative tensor factorization and its completion. Global convergence results are obtained directly from the analysis in Chapter 2.

3.1 Overview of tensor

A *tensor* is a multi-dimensional array. For example, a *vector* is a first-order *tensor*, and a *matrix* is a second-order *tensor*. The *order* of a tensor is the number of dimensions, also called *way* or *mode*. For an N-way *tensor* $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, denote its $(i_1, i_2, \cdots, i_N)$th element by $x_{i_1 i_2 \cdots i_N}$. Below I list some concepts related to tensor. For more details about tensor, the reader is referred to the review paper [44].

1. **fiber:** a *fiber* of a tensor $\boldsymbol{\mathcal{X}}$ is a vector obtained by fixing all indices of $\boldsymbol{\mathcal{X}}$ except one. For example, a row of a matrix is a mode-2 fiber (the 1st index is fixed), and a column is a mode-1 fiber (the 2nd index is fixed). I use $\mathbf{x}_{i_1 \cdots i_{n-1} : i_{n+1} \cdots i_N}$

to denote a mode-n fiber of an Nth-order tensor $\boldsymbol{\mathcal{X}}$.

2. **slice:** a *slice* of a tensor $\boldsymbol{\mathcal{X}}$ is a matrix obtained by fixing all indices of $\boldsymbol{\mathcal{X}}$ except two. Take a third-order tensor $\boldsymbol{\mathcal{X}}$ for example. $\mathbf{X}_{i::}, \mathbf{X}_{:j:}$, and $\mathbf{X}_{::k}$ denote horizontal, lateral, and frontal slices of $\boldsymbol{\mathcal{X}}$, respectively.

3. **matricization:** the mode-n *matricization* of a tensor $\boldsymbol{\mathcal{X}}$ is a matrix whose columns are the mode-n fibers of $\boldsymbol{\mathcal{X}}$ in the lexicographical order. Let $\mathbf{X}_{(n)}$ denote the mode-n matricization of $\boldsymbol{\mathcal{X}}$.

4. **tensor-matrix product:** the mode-n product of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{A} \in \mathbb{R}^{J \times I_n}$ is a tensor of size $I_1 \times \cdots I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$ defined as

$$(\boldsymbol{\mathcal{X}} \times_n \mathbf{A})_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_N} a_{j i_n}. \tag{3.1}$$

In addition, let me briefly review the matrix Kronecker, Khatri-Rao and Hadamard products below, which are used to derive tensor-related computations.

The *Kronecker product* of matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{p \times q}$ is an $mp \times nq$ matrix defined by

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2n}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & a_{m2}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix}.$$

The *Khatri-Rao product* of matrices $\mathbf{A} \in \mathbb{R}^{m\times q}$ and $\mathbf{B} \in \mathbb{R}^{p\times q}$ is an $mp \times q$ matrix:

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1,\ \mathbf{a}_2 \otimes \mathbf{b}_2,\ \cdots,\ \mathbf{a}_q \otimes \mathbf{b}_q],$$

where $\mathbf{a}_i, \mathbf{b}_i$ are the ith columns of $\mathbf{A}$ and $\mathbf{B}$, respectively. The *Hadamard product* of matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$ is the componentwise product defined by

$$\mathbf{A} * \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \cdots & a_{1n}b_{1n} \\ a_{21}b_{21} & a_{22}b_{22} & \cdots & a_{2n}b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{m1} & a_{m2}b_{m2} & \cdots & a_{mn}b_{mn} \end{bmatrix}.$$

Two important tensor decompositions are the CANDECOMP/PARAFAC (CP) [38] and Tucker [85] decompositions. The former one decomposes a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ in the form of $\boldsymbol{\mathcal{X}} = \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N$, where $\mathbf{A}_n \in \mathbb{R}^{I_n\times r}, n = 1, \cdots, N$ are factor matrices, r is the tensor rank of $\boldsymbol{\mathcal{X}}$, and the outer product "$\circ$" is defined as

$$x_{i_1 i_2 \cdots i_N} = \sum_{j=1}^{r} a^{(1)}_{i_1 j} a^{(2)}_{i_2 j} \cdots a^{(N)}_{i_N j},\ \text{for } i_n \in [I_n], n = 1, \cdots, N,$$

where $a^{(n)}_{ij}$ is the (i,j)th element of $\mathbf{A}_n$ and $[I] \triangleq \{1, 2, \cdots, I\}$. The latter Tucker decomposition decomposes a tensor $\boldsymbol{\mathcal{X}}$ in the form of $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{G}} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 \cdots \times_N \mathbf{A}_N$, where $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$ is called the core tensor and $\mathbf{A}_n \in \mathbb{R}^{I_n\times J_n}, n = 1, \cdots, N$ are factor matrices.

3.2 An algorithm for nonnegative tensor factorization

One can obtain a nonnegative CP decomposition of a nonnegative tensor $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ by solving

$$\min \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N\|_F^2, \text{ subject to } \mathbf{A}_n \in \mathbb{R}_+^{I_n \times r}, n = 1, \cdots, N \tag{3.2}$$

where r is a specified order and the Frobenius norm of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is defined as

$$\|\boldsymbol{\mathcal{X}}\|_F = \sqrt{\sum_{i_1, i_2, \cdots, i_N} x_{i_1 i_2 \cdots i_N}^2}.$$

Similar models based on the CP decomposition can be found in [40, 23, 42]. One can obtain a nonnegative Tucker decomposition of $\boldsymbol{\mathcal{M}}$ by solving

$$\min \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \boldsymbol{\mathcal{G}} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 \cdots \times_N \mathbf{A}_N\|_F^2, \text{ subject to } \boldsymbol{\mathcal{G}} \in \mathbb{R}_+^{J_1 \times \cdots \times J_N}, \mathbf{A}_n \in \mathbb{R}_+^{I_n \times J_n}, \forall n, \tag{3.3}$$

as in [43, 61, 53]. Usually, it is computationally expensive to update $\boldsymbol{\mathcal{G}}$. Since applying Algorithm 1 to problem (3.3) involves lots of computing details, I focus on applying it with block-update (1.16c) to problem (3.2).

Let $\mathbf{A} = (\mathbf{A}_1, \cdots, \mathbf{A}_N)$ and

$$F(\mathbf{A}) = F(\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_N) = \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N\|_F^2$$

be the objective of (3.2). Consider updating $\mathbf{A}_n$ at iteration k. Using the fact that if $\boldsymbol{\mathcal{X}} = \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N$, then $\mathbf{X}_{(n)} = \mathbf{A}_n \left(\mathbf{A}_N \odot \cdots \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \cdots \mathbf{A}_1\right)^\top$, we have

$$F(\mathbf{A}) = \frac{1}{2}\left\|\mathbf{M}_{(n)} - \mathbf{A}_n \left(\mathbf{A}_N \odot \cdots \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \cdots \mathbf{A}_1\right)^\top\right\|_F^2,$$

and

$$\nabla_{\mathbf{A}_n} F = \left(\mathbf{A}_n \left(\mathbf{A}_N \odot \cdots \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \cdots \mathbf{A}_1 \right)^\top - \mathbf{M}_{(n)} \right) \left(\mathbf{A}_N \odot \cdots \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \cdots \mathbf{A}_1 \right).$$

Let

$$\mathbf{B}_n^{k-1} = \mathbf{A}_N^{k-1} \odot \cdots \mathbf{A}_{n+1}^{k-1} \odot \mathbf{A}_{n-1}^{k} \cdots \mathbf{A}_1^{k}. \tag{3.4}$$

Take $L_n^{k-1} = \max(\ell^{k-2}, \|(\mathbf{B}_n^{k-1})^\top \mathbf{B}_n^{k-1}\|)$, where $\ell^{k-2} = \min_n L_n^{k-2}$ and $\|\mathbf{A}\|$ is the spectral norm of $\mathbf{A}$. Let

$$\omega_n^{k-1} = \min \left(\hat{\omega}_{k-1}, \delta_\omega \sqrt{\frac{\ell^{k-2}}{L_n^{k-1}}} \right) \tag{3.5}$$

where $\delta_\omega < 1$ is pre-selected and $\hat{\omega}_{k-1} = \frac{t_{k-1}-1}{t_k}$ with $t_0 = 1$ and $t_k = \frac{1}{2}\left(1 + \sqrt{1 + 4t_{k-1}^2}\right)$. In addition, let

$$\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1} + \omega_n^{k-1}(\mathbf{A}_n^{k-1} - \mathbf{A}_n^{k-2})$$

and

$$\hat{\mathbf{G}}_n^{k-1} = \left(\hat{\mathbf{A}}_n^{k-1} (\mathbf{B}_n^{k-1})^\top - \mathbf{M}_{(n)} \right) \mathbf{B}_n^{k-1} \tag{3.6}$$

be the gradient. Then we derive the update (1.16c):

$$\mathbf{A}_n^k = \underset{\mathbf{A}_n \geq 0}{\operatorname{argmin}} \left\langle \hat{\mathbf{G}}_n^{k-1}, \mathbf{A}_n - \hat{\mathbf{A}}_n^{k-1} \right\rangle + \frac{L_n^{k-1}}{2} \left\| \mathbf{A}_n - \hat{\mathbf{A}}_n^{k-1} \right\|_F^2,$$

which can be written in the closed form

$$\mathbf{A}_n^k = \max \left(0, \hat{\mathbf{A}}_n^{k-1} - \hat{\mathbf{G}}_n^{k-1} / L_n^{k-1} \right). \tag{3.7}$$

At the end of iteration k, check whether $F\left(\mathbf{A}^k\right) \geq F\left(\mathbf{A}^{k-1}\right)$. If so, re-update $\mathbf{A}_n^k$ by (3.7) with $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1}$, for $n = 1, \cdots, N$.

Remark 3.1

In (3.7), $\hat{\mathbf{G}}_n^{k-1}$ *is most expensive to compute. To efficiently compute it, write*

$$\hat{\mathbf{G}}_n^{k-1} = \hat{\mathbf{A}}_n^{k-1}(\mathbf{B}_n^{k-1})^\top \mathbf{B}_n^{k-1} - \mathbf{M}_{(n)}\mathbf{B}_n^{k-1}$$

Using the fact

$$(\mathbf{A} \odot \mathbf{B})^\top (\mathbf{A} \odot \mathbf{B}) = (\mathbf{A}^\top \mathbf{A}) * (\mathbf{B}^\top \mathbf{B})$$

we can compute $(\mathbf{B}_n^{k-1})^\top \mathbf{B}_n^{k-1}$ *by*

$$(\mathbf{B}_n^{k-1})^\top \mathbf{B}_n^{k-1} = \big((\mathbf{A}_1^k)^\top \mathbf{A}_1^k\big) * \cdots * \big((\mathbf{A}_{n-1}^k)^\top \mathbf{A}_{n-1}^k\big) * \big((\mathbf{A}_{n+1}^{k-1})^\top \mathbf{A}_{n+1}^{k-1}\big) * \cdots * \big((\mathbf{A}_N^{k-1})^\top \mathbf{A}_N^{k-1}\big).$$

Then, $\mathbf{M}_{(n)}\mathbf{B}_n^{k-1}$ *can be obtained by the so-called* matricized-tensor-times-Khatri-Rao-product [7].

Algorithm 2 summarizes how to apply Algorithm 1 with block-update (1.16c) to problem (3.2).

3.3 Convergence results

Since problem (3.2) is a special case of problem (1.7), the convergence results in Chapter 2 apply to Algorithm 2. Let $\mathcal{D}_n = \mathbb{R}_+^{I_n \times r}$ and $\delta_{\mathcal{D}_n}(\cdot)$ be the indicator function on $\mathcal{D}_n$ for $n = 1, \cdots, N$. Then (3.2) is equivalent to

$$\min_{\mathbf{A}_1, \cdots, \mathbf{A}_N} Q(\mathbf{A}) \equiv F(\mathbf{A}) + \sum_{n=1}^{N} \delta_{\mathcal{D}_n}(\mathbf{A}_n). \tag{3.8}$$

According to the discussion in Section 2.3, Q is a semi-algebraic function and satisfies the KL property (2.19) at any feasible point. Further, we get $\theta \neq 0$ in (2.19) for Q

Algorithm 2 Alternating proximal gradient method for solving (3.2)

Input: nonnegative N-way tensor $\boldsymbol{\mathcal{M}}$ and rank r.
Output: nonnegative factors $\mathbf{A}_1, \cdots, \mathbf{A}_N$.
Initialization: choose positive number $\delta_\omega < 1$ and $\mathbf{A}_n^{-1} = \mathbf{A}_n^0, n = 1, \cdots, N$, as nonnegative matrices of appropriate sizes.
for $k = 1, 2, \cdots$ **do**
  for $n = 1, 2, \cdots, N$ **do**
    Compute L_n^{k-1} and set ω_n^{k-1} according to (3.5);
    Let $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1} + \omega_n^{k-1}(\mathbf{A}_n^{k-1} - \mathbf{A}_n^{k-2})$;
    Update $\mathbf{A}_n^k$ according to (3.7).
  end for
  if $F\left(\mathbf{A}^k\right) \geq F\left(\mathbf{A}^{k-1}\right)$ **then**
    Re-update $\mathbf{A}_n^k$ according to (3.7) with $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1}, n = 1, \cdots, N$
  end if
  if stopping criterion is satisfied **then**
    Return $\mathbf{A}_1^k, \cdots, \mathbf{A}_N^k$.
  end if
end for

at any critical point. This claim can be shown by the argument: writing the first-order optimality conditions of (3.8), one can find that if $\bar{\mathbf{A}} = \left(\bar{\mathbf{A}}_1, \cdots, \bar{\mathbf{A}}_N\right)$ is a critical point, then so is $\bar{\mathbf{A}}_t \left(t\bar{\mathbf{A}}_1, \frac{1}{t}\bar{\mathbf{A}}_2, \bar{\mathbf{A}}_3, \cdots, \bar{\mathbf{A}}_N\right)$ for any $t > 0$. Hence within any neighborhood of $\bar{\mathbf{A}}$, there is other critical point and (2.19) cannot be bounded with $\theta = 0$. Therefore, from Theorems 2.3 and 2.4 and the above discussions, we have

Theorem 3.1

Let $\{\mathbf{A}^k\}$ be the sequence generated by Algorithm 2. Assume $\{\mathbf{A}^k\}$ is bounded and there is a positive constant ℓ such that $\ell \leq \ell^k$ for all k. Then $\{\mathbf{A}^k\}$ converges to a critical point $\bar{\mathbf{A}}$, and the asymptotic convergence rates in parts 2 and 3 of Theorem 2.4 apply.

Remark 3.2

The boundedness of $\{\mathbf{A}^k\}$ *guarantees that* L_n^k *is upper bounded. A simple way to make* $\{\mathbf{A}^k\}$ *bounded is to scale* $(\mathbf{A}_1, \cdots, \mathbf{A}_N)$ *so that* $\|\mathbf{A}_1\|_F = \cdots = \|\mathbf{A}_N\|_F$ *after each iteration. The existence of a positive* ℓ *can be satisfied if one changes* L_n^k *to* $\max(L_n^k, L_{\min})$ *for a positive constant* $L_{\min}$.

3.4 An algorithm for nonnegative tensor completion

Algorithm 2 can be easily modified for solving the nonnegative tensor completion problem

$$\min_{\mathbf{A}_1,\cdots,\mathbf{A}_N \geq 0} \frac{1}{2}\|\mathcal{P}_\Omega(\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N)\|_F^2, \tag{3.9}$$

where $\Omega \subset [I_1] \times [I_2] \times \cdots [I_N]$ is the index set of the observed entries of $\boldsymbol{\mathcal{M}}$ and $\mathcal{P}_\Omega(\boldsymbol{\mathcal{X}})$ keeps the entries of $\boldsymbol{\mathcal{X}}$ in Ω and sets the remaining ones to zero. Nonnegative matrix completion (corresponding to $N = 2$) has been proposed in [90], where it is demonstrated that a low-rank and nonnegative matrix can be recovered from a small set of its entries by taking advantages of both low-rankness and nonnegative factors. To solve (3.9), I transform it into the equivalent problem

$$\min_{\boldsymbol{\mathcal{X}},\mathbf{A}_n \geq 0, n=1,\cdots,N} G(\mathbf{A}, \boldsymbol{\mathcal{X}}) \equiv \frac{1}{2}\|\boldsymbol{\mathcal{X}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N\|_F^2, \text{ subject to } \mathcal{P}_\Omega(\boldsymbol{\mathcal{X}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}}). \tag{3.10}$$

My algorithm shall cycle through the decision variables $\mathbf{A}_1, \cdots, \mathbf{A}_N$ and $\boldsymbol{\mathcal{X}}$. It is summarized in Algorithm 3, which is simply modified from Algorithm 2. At each iteration of Algorithm 2, set its $\boldsymbol{\mathcal{M}}$ to $\boldsymbol{\mathcal{X}}^{k-1}$ and, after its updates (1.16c) on $\mathbf{A}_1, \cdots, \mathbf{A}_N$, per-

form update (1.16a) on $\boldsymbol{\mathcal{X}}$ as

$$\boldsymbol{\mathcal{X}}^k = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}}) + \mathcal{P}_{\Omega^c}(\mathbf{A}_1^k \circ \cdots \circ \mathbf{A}_N^k), \tag{3.11}$$

where Ω^c is the complement of Ω.

Note that for a fixed $\mathbf{A}$, $G(\mathbf{A}, \boldsymbol{\mathcal{X}})$ is a strongly convex function of $\boldsymbol{\mathcal{X}}$ with modulus 1 and

$$G(\mathbf{A}^k, \boldsymbol{\mathcal{X}}^{k-1}) - G(\mathbf{A}^k, \boldsymbol{\mathcal{X}}^k) = \frac{1}{2}\|\boldsymbol{\mathcal{X}}^{k-1} - \boldsymbol{\mathcal{X}}^k\|_F^2.$$

Hence, the convergence result for Algorithm 2 still holds for this algorithm with extra update (3.11).

Algorithm 3 Alternating proximal gradient method for solving (3.9)

Input: partially observed nonnegative N-way tensor $\mathcal{P}_\Omega(\boldsymbol{\mathcal{M}})$, set Ω and rank r.
Output: recovered nonnegative tensor $\boldsymbol{\mathcal{M}}^r$.
Initialization: choose positive number $\delta_\omega < 1$ and $\mathbf{A}_n^{-1} = \mathbf{A}_n^0, n = 1, \cdots, N$, as nonnegative matrices of appropriate sizes.
for $k = 1, 2, \cdots$ **do**
  for $n = 1, 2, \cdots, N$ **do**
    Compute L_n^{k-1} and set ω_n^{k-1} according to (3.5);
    Let $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1} + \omega_n^{k-1}(\mathbf{A}_n^{k-1} - \mathbf{A}_n^{k-2})$;
    Update $\mathbf{A}_n^k$ according to (3.7) with $\hat{\mathbf{G}}_n^{k-1}$ computed from (3.6) where $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{X}}^{k-1}$ is used.
  end for
  Update $\boldsymbol{\mathcal{X}}^k$ according to (3.11);
  if $G\left(\mathbf{A}^k, \boldsymbol{\mathcal{X}}^k\right) \geq G\left(\mathbf{A}^{k-1}, \boldsymbol{\mathcal{X}}^{k-1}\right)$ **then**
    Repeat this iteration with $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1}$ while updating $\mathbf{A}_n^k$ for $n = 1, \cdots, N$.
  end if
  if stopping criterion is satisfied **then**
    Return $\mathbf{A}_1^k, \cdots, \mathbf{A}_N^k$.
  end if
end for

Chapter 4

Numerical results

In this section, I test Algorithm 2 for nonnegative matrix factorization taking $N = 2$ in (3.2) and three-way tensor factorization taking $N = 3$ in (3.2), as well as Algorithm 3 for their completion. In the implementations, I simply choose $\delta_\omega = 1$. The algorithm is terminated whenever

$$\frac{F_k}{\|\boldsymbol{\mathcal{M}}\|_F} \leq tol, \quad \text{or} \quad \frac{F_k - F_{k+1}}{1 + F_k} \leq tol \text{ holds for three consecutive iterations}$$

where F_k is the objective value after iteration k and tol is specified below. I test

- APG-MF: nonnegative matrix factorization (NMF) by Algorithm 2;
- APG-TF: nonnegative tensor factorization (NTF) by Algorithm 2;
- APG-MC: nonnegative matrix completion (NMC) by Algorithm 3;
- APG-TC: nonnegative tensor completion (NTC) by Algorithm 3.

All the tests were performed on a laptop with an i7-620m CPU and 3GB RAM and running 32-bit Windows 7 and Matlab 2010b with Tensor Toolbox of version 2.5 [6].

4.1 Nonnegative matrix factorization

This section tests Algorithm 2 on nonnegative matrix factorization

$$\min_{\mathbf{A}_1,\mathbf{A}_2} \frac{1}{2}\|\mathbf{A}_1\mathbf{A}_2 - \mathbf{M}\|_F^2, \text{ subject to } \mathbf{A}_1 \in \mathbb{R}_+^{I_1\times r}, \mathbf{A}_2 \in \mathbb{R}_+^{r\times I_2}. \tag{4.1}$$

Before comparing with other methods, let us see how the extrapolation technique in (1.16c) affects the algorithm. In Figure 4.1, I test Algorithm 2 on nonnegative matrix factorization (NMF). The accelerated version corresponds to extrapolation weight ω_i^k specified by (3.5), and no-acceleration corresponds to $\omega_i^k \equiv 0$. The tested matrix has the form of

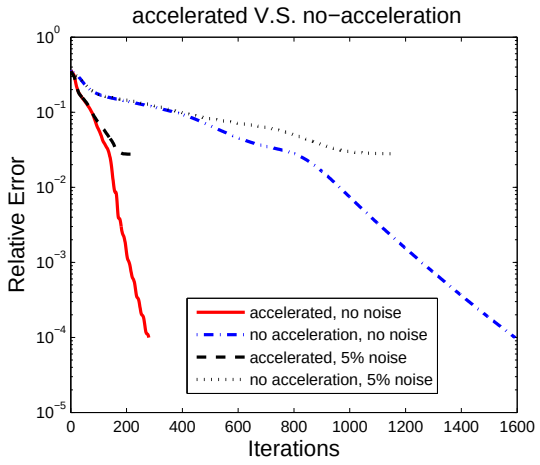
$$\mathbf{M} = \mathbf{LR} + \sigma\frac{\|\mathbf{LR}\|_F}{\|\mathbf{N}\|_F}\mathbf{N}$$

where σ is noise level and $\mathbf{L} \in \mathbb{R}^{500\times 30}, \mathbf{R} \in \mathbb{R}^{30\times 500}, \mathbf{N} \in \mathbb{R}^{500\times 500}$ are Gaussian randomly generated matrices. We can see that the extrapolation technique significantly speeds up the algorithm in both noisy and noiseless cases.

4.1.1 Overview of some algorithms

Next, I choose to compare the most popular and recent algorithms. The first two compared ones are the alternating least square method (Als-MF) [67, 10] and multiplicative updating method (Mult-MF) [48], which are available as MATLAB's function `nnmf` with specifiers `als` and `mult`, respectively. Als-MF alternatively updates

Figure 4.1 : How extrapolation improves the algorithm



$\mathbf{A}_1$ and $\mathbf{A}_2$ by

$$\mathbf{A}_1^k = \max\left(0, \mathbf{M}(\mathbf{A}_2^{k-1})^\top \left(\mathbf{A}_2^{k-1}(\mathbf{A}_2^{k-1})^\top\right)^\dagger\right),$$
$$\mathbf{A}_2^k = \max\left(0, \left((\mathbf{A}_1^k)^\top \mathbf{A}_1^k\right)^\dagger (\mathbf{A}_1^k)^\top \mathbf{M}\right),$$

where † denotes the Moore-Penrose pseudo-inverse. Mult-MF has cheaper multiplicative updates

$$(\mathbf{A}_1^k)_{ij} = \frac{(\mathbf{A}_1^{k-1})_{ij}(\mathbf{M}(\mathbf{A}_2^{k-1})^\top)_{ij}}{(\mathbf{A}_1^{k-1}\mathbf{A}_2^{k-1}(\mathbf{A}_2^{k-1})^\top + \varepsilon)_{ij}},$$
$$(\mathbf{A}_2^k)_{ij} = \frac{(\mathbf{A}_2^{k-1})_{ij}((\mathbf{A}_1^k)^\top \mathbf{M})_{ij}}{((\mathbf{A}_1^k)^\top(\mathbf{A}_1^k)\mathbf{A}_2^{k-1} + \varepsilon)_{ij}}, \ \forall i, j$$

where $\varepsilon > 0$ is used to avoid division by zero.

One recent ANLS method Blockpivot-MF is compared since it outperforms all other compared ANLS methods in both speed and solution quality [41]. ANLS meth-

ods belong to the framework of Algorithm 1 and use block-update (1.16a):

$$\mathbf{A}_1^k = \underset{\mathbf{A}_1 \geq 0}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}_1\mathbf{A}_2^{k-1} - \mathbf{M}\|_F^2, \tag{4.2a}$$

$$\mathbf{A}_2^k = \underset{\mathbf{A}_2 \geq 0}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}_1^k\mathbf{A}_2 - \mathbf{M}\|_F^2. \tag{4.2b}$$

Blockpivot uses an active-set like method to solve subproblems in (4.2).

Another compared algorithm is the recent ADM-based method ADM-MF [95]. With auxiliary variables $\mathbf{U}$ and $\mathbf{V}$, ADM-MF solves the equivalent problem

$$\min_{\mathbf{A}_1,\mathbf{A}_2,\mathbf{U},\mathbf{V}} \frac{1}{2}\|\mathbf{A}_1\mathbf{A}_2 - \mathbf{M}\|_F^2, \text{ subject to } \mathbf{A}_1 = \mathbf{U}, \mathbf{A}_2 = \mathbf{V}, \mathbf{U} \geq 0, \mathbf{V} \geq 0. \tag{4.3}$$

It is derived by alternatively minimizing the augmented Lagrangian function

$$\begin{aligned}\mathcal{L}(\mathbf{A}_1, \mathbf{A}_2, \mathbf{U}, \mathbf{V}, \mathbf{\Lambda}, \mathbf{\Pi}) =& \frac{1}{2}\|\mathbf{A}_1\mathbf{A}_2 - \mathbf{M}\|_F^2 + \langle \mathbf{\Lambda}, \mathbf{A}_1 - \mathbf{U}\rangle + \frac{\alpha}{2}\|\mathbf{A}_1 - \mathbf{U}\|_F^2 \\ &+ \langle \mathbf{\Pi}, \mathbf{A}_2 - \mathbf{V}\rangle + \frac{\beta}{2}\|\mathbf{A}_2 - \mathbf{V}\|_F^2\end{aligned}$$

with respect to $\mathbf{A}_1, \mathbf{A}_2, \mathbf{U}, \mathbf{V}$, one at a time by fixing others, and updating the multipliers $\mathbf{\Lambda}, \mathbf{\Pi}$

$$\mathbf{\Lambda}^k = \mathbf{\Lambda}^{k-1} + \gamma\alpha(\mathbf{A}_1^k - \mathbf{U}^k), \quad \mathbf{\Pi}^k = \mathbf{\Pi}^{k-1} + \gamma\beta(\mathbf{A}_2^k - \mathbf{V}^k),$$

where step length $\gamma \in (0, 1.618)$ and α, β are penalty parameters.

Although both Blockpivot-MF and ADM-MF have superior performance than Als-MF and Mult-MF, I include them in the first two tests below due to their popularity.

4.1.2 Parameter setting

I set $tol = 10^{-4}$ for all the compared algorithms except ADM-MF, for which I set $tol = 10^{-5}$ since it is a dual algorithm and 10^{-4} is too loose. The maximum number of

iterations is set to 2000 for all the compared algorithms. The same random starting points are used for all the algorithms except for Mult-MF. Since Mult-MF is very sensitive to initial points, I set the initial point by running Mult-MF 10 iterations for 5 independent times and choose the best one. In the implementation of ADM-MF, it scales $\mathbf{M}$ to have $\|\mathbf{M}\|_F = 5 \times 10^6$, and step length $\gamma = 1.618$ and penalty parameters $\alpha = \beta = 1.25\frac{\max(I_1, I_2)}{r} \times 10^4$ are used. All the other parameters for Als-MF, Mult-MF, Blockpivot-MF and ADM-MF are set to their default values.

4.1.3 Synthetic data

Each matrix in this test is exactly low-rank and can be written in the form of $\mathbf{M} = \mathbf{LR}$, where $\mathbf{L}$ and $\mathbf{R}$ are generated by MATLAB commands `max(0,randn(m,q))` and `rand(q,n)`, respectively. It is worth mentioning that generating $\mathbf{R}$ by `rand(q,n)` makes the problems more difficult than `max(0,randn(q,n))` or `abs(randn(q,n))`. The algorithms are compared with fixed $n = 1000$ and m chosen from $\{200, 500, 1000\}$, q from $\{10, 20, 30\}$. The parameter r is set to q in (4.1). I use relative error relerr $= \|\mathbf{A}_1\mathbf{A}_2 - \mathbf{M}\|_F / \|\mathbf{M}\|_F$ and CPU time (in seconds) to measure the performance of each algorithm. Table 4.1 lists the average results of 20 independent trials. From the table, we can see that APG-MF outperforms all the other algorithms in both CPU time and solution quality. Figure 4.2 plots one trial for which $m = 500, q = 30$ and each algorithm runs to sufficiently long time. It shows that APG-MF, ADM-MF and Blockpivot-MF can all reach a high accuracy while Als-MF and Mult-MF are stuck

Table 4.1 : Average relative errors and running time (sec) of each algorithm on nonnegative random $m \times n$ matrices for $n = 1000$; **bold** are **large error** or **slow time**.

		APG-MF† (prop'd)		ADM-MF		Blockpivot-MF		Als-MF		Mult-MF	
m	r	relerr	time	relerr	time	relerr	time	relerr	time	relerr	time
200	10	9.98e-5	7.16e-1	**2.24e-3**	1.04e+0	5.36e-4	1.30e+0	**7.39e-3**	1.04e+0	**3.61e-2**	**2.67e+0**
200	20	9.97e-5	2.09e+0	**3.02e-3**	2.80e+0	**1.02e-3**	4.71e+0	**1.01e-2**	2.33e+0	**4.64e-2**	3.61e+0
200	30	9.97e-5	4.72e+0	**4.55e-3**	5.70e+0	**1.75e-3**	1.06e+1	**1.04e-2**	4.54e+0	**4.09e-2**	5.53e+0
500	10	9.98e-5	1.61e+0	**2.26e-3**	2.39e+0	5.11e-4	2.38e+0	**1.15e-2**	2.99e+0	**3.58e-2**	**7.76e+0**
500	20	9.98e-5	3.66e+0	**2.82e-3**	4.38e+0	5.53e-4	6.86e+0	**1.08e-2**	6.31e+0	**4.96e-2**	**7.99e+0**
500	30	9.98e-5	7.75e+0	**3.51e-3**	8.34e+0	5.75e-4	1.37e+1	**1.29e-2**	9.95e+0	**4.42e-2**	1.20e+1
1000	10	9.98e-5	2.86e+0	**2.11e-3**	3.44e+0	4.99e-4	3.18e+0	**1.54e-3**	8.04e+0	**3.25e-2**	**1.55e+1**
1000	20	9.98e-5	7.44e+0	**2.82e-3**	7.19e+0	5.46e-4	1.05e+1	**1.74e-2**	**1.75e+1**	**4.96e-2**	**1.61e+1**
1000	30	9.98e-5	1.27e+1	**3.01e-3**	1.28e+1	5.76e-4	2.00e+1	**1.99e-2**	**2.61e+1**	**4.57e-2**	2.21e+1

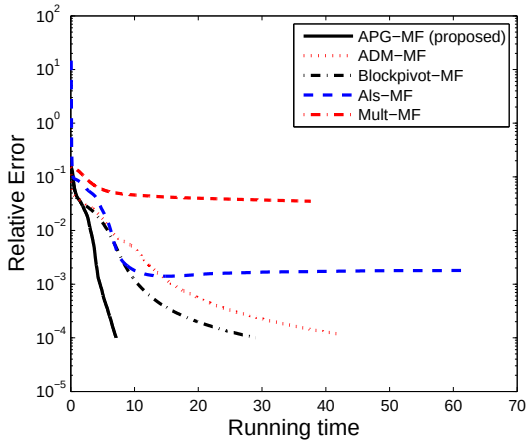
†: the relerr values of APG-MF are nearly the same due to the use of the same stopping tolerence.

at local minima.

4.1.4 Image data

In this subsection, I compare APG-MF (proposed), ADM-MF, Blockpivot-MF, Als-MF and Mult-MF on the CBCL and ORL image databases used in [34, 51]. There are 6977 face images in the training set of CBCL, each having 19×19 pixels. Multiple images of each face are taken with varying illuminations and facial expressions. The first 2000 images are used for test. I vectorize every image and obtain a matrix $\mathbf{M}$ of size 361×2000. Figure 4.3 (a) shows the first 36 faces corresponding to the first 36 columns of matrix $\mathbf{M}$. Dimension parameter r in (4.1) is chosen from $\{30, 60, 90\}$. The average relative errors and running time (sec) of 10 independent trials are given in Table 4.2. We can see that APG-MF outperforms ADM-MF in both speed and solution quality. APG-MF is as accurate as Blockpivot-MF but runs much faster.

Figure 4.2 : One trial on nonnegative matrix factorization with $m = 500, q = 30$



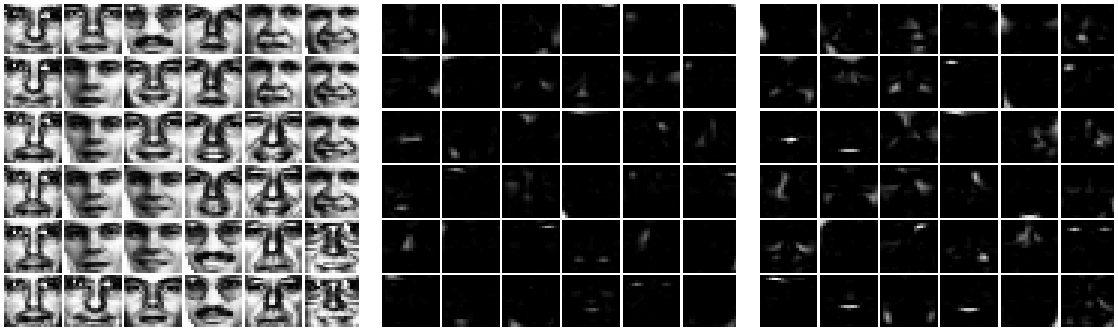
Als-MF and Mult-MF produce very bad results in this test, and Als-MF stagnates at solutions of low quality at the very beginning. Figure 4.3 (b)-(f) plot 36 base images corresponding to the first 36 columns of $\mathbf{A}_1$ obtained by each method for $r = 90$. Each $\mathbf{A}_1$ is scaled to have the unit maximum element. APG-MF, ADM-MF and Blockpivot-MF all get relatively sparse $\mathbf{A}_1$'s, as demonstrated in [47] that NMF can be used to learn local features of images. Due to the poor performance of Als-MF and Mult-MF, only APG-MF, ADM-MF and Blockpivot-MF are compared in the remaining tests.

The ORL database has 400 images divided into 40 groups. Each image has 112×92 pixels, and each group has 10 images of one face taken from 10 different directions and with different expressions. All the images are used for test. I vectorize each image and obtain a matrix $\mathbf{M}$ of size 10304×400. Figure 4.4 (a) depicts 50 images corresponding

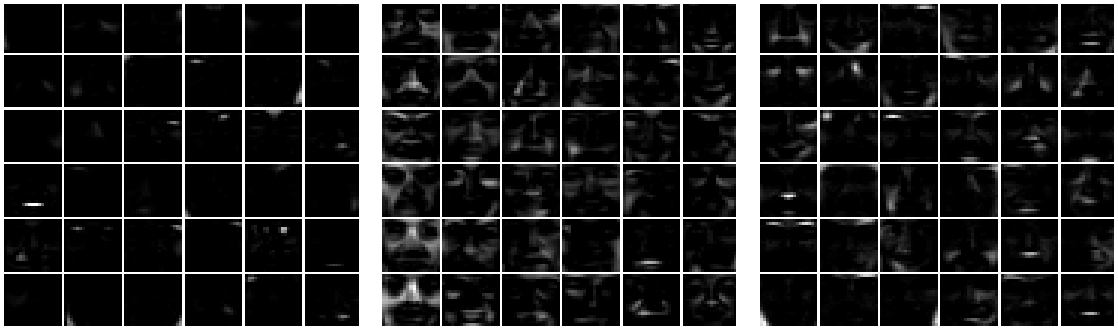
Table 4.2 : Average relative errors and running time (sec) of each algorithm on 2000 selected images from the CBCL face database; **bold** are **large error** or **slow time**.

	APG-MF (proposed)		ADM-MF		Blockpivot-MF		Als-MF		Mult-MF	
r	relerr	time	relerr	time	relerr	time	relerr	time	relerr	time
30	1.91e-1	3.68	1.92e-1	7.33	1.90e-1	**21.5**	**3.53e-1**	3.15	**2.13e-1**	6.51
60	1.42e-1	12.5	1.43e-1	19.5	1.40e-1	**63.2**	**4.59e-1**	1.80	**1.74e-1**	12.1
90	1.13e-1	26.7	1.15e-1	34.2	1.12e-1	**111**	**6.00e-1**	2.15	**1.52e-1**	18.4

Figure 4.3 : CBCL database and base images: (a) 36 images selected from the 2000 tested images; (b)-(f) 36 base images corresponding to the first 36 columns of $\mathbf{A}_1$ obtained by APG-MF (prop'd), ADM-MF, Blockpivot-MF, Als-MF and Mult-MF at $r = 90$.



(a) CBCL data (b) APG-MF (prop'd) (c) ADM-MF



(d) Blockpivot-MF (e) Als-MF (f) Mult-MF

Table 4.3 : Comparison on the images from the ORL face database; **bold** are **slow time**.

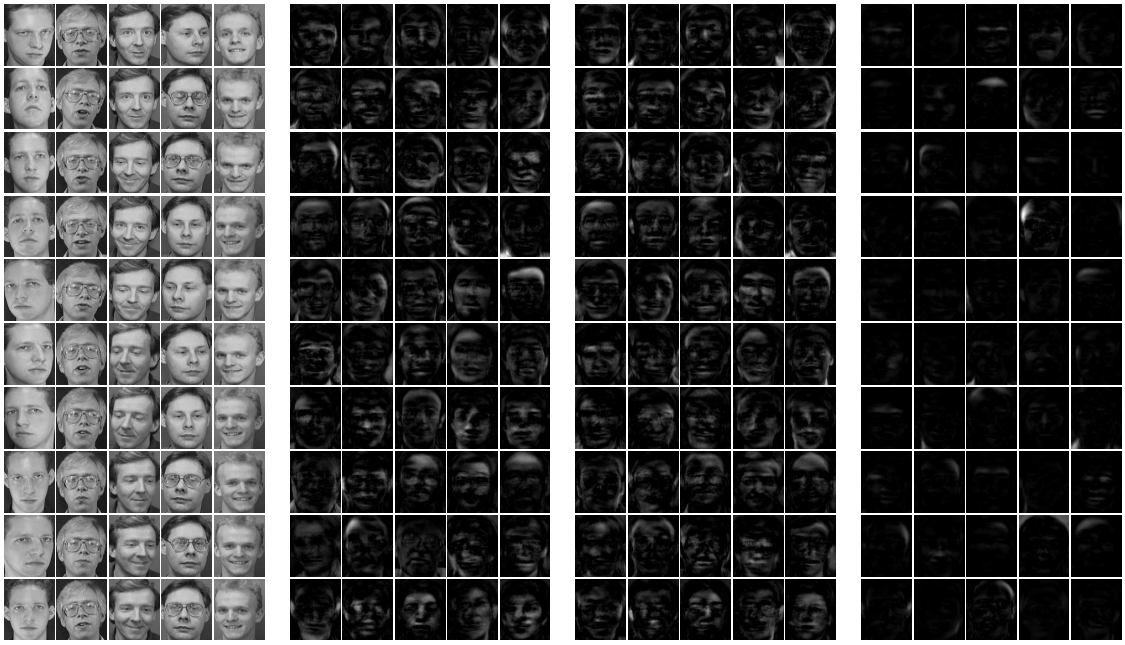
	APG-MF (proposed)		ADM-MF		Blockpivot-MF	
r	relerr	time	relerr	time	relerr	time
30	1.67e-1	15.8	1.71e-1	46.5	1.66e-1	**74.3**
60	1.41e-1	42.7	1.45e-1	88.0	1.40e-1	**178**
90	1.26e-1	76.4	1.30e-1	127	1.25e-1	**253**

to the first 50 columns of $\mathbf{M}$. As in last test, I choose r from $\{30, 60, 90\}$. The average results of 10 independent trials are listed in Table 4.3. From the results, we can see again that APG-MF is better than ADM-MF in both speed and solution quality, and in far less time APG-MF achieves comparable accuracy as Blockpivot-MF. Figure 4.4 (b)-(d) depict 50 base images corresponding to the first 50 columns of $\mathbf{A}_1$ obtained by each algorithm for $r = 90$. All the three algorithms get the frames of faces instead of local features. None of them get a sparse $\mathbf{A}_1$. The sparsest one obtained by Blockpivot-MF has about 60.6% non-zeros.

4.1.5 Hyperspectral data

It has been shown in [68] that NMF can be applied to spectral data analysis. In [68], a regularized NMF model is also considered with penalty terms $\alpha\|\mathbf{A}_1\|_F^2$ and $\beta\|\mathbf{A}_2\|_F^2$ added in the objective of (3.2). The parameters α and β can be tuned for specific purposes in practice. Here, I focus on the original NMF model to show the effectiveness of the algorithm. However, my method can also solve the regularized NMF model. In this test, I use a $150 \times 150 \times 163$ hyperspectral cube to test the

Figure 4.4 : ORL database and base images: (a) 50 images selected from 400 tested images; (b)-(d) 50 base images corresponding to the first 50 columns of $\mathbf{A}_1$ obtained from APG-MF, ADM-MF and Blockpivot-MF at $r = 90$.



(a) ORL data (b) APG-MF (c) ADM-MF (d) Blockpivot-MF

Figure 4.5 : Hyperspectral data of $150 \times 150 \times 163$: four selected slices are shown

Table 4.4 : Average relative errors and running time (sec) of each algorithm on hyperspectral data of size $150 \times 150 \times 163$; **bold** are **large error** or **slow time**.

	APG-MF (proposed)		ADM-MF		Blockpivot-MF	
r	relerr	time	relerr	time	relerr	time
20	1.18e-2	34.2	**2.34e-2**	**87.5**	1.38e-2	62.5
30	9.07e-3	63.2	**2.02e-2**	116	1.10e-2	**143**
40	7.56e-3	86.2	**1.78e-2**	140	9.59e-3	**194**
50	6.45e-3	120	**1.58e-2**	182	8.00e-3	**277**

compared algorithms. Each slice of the cube is reshaped as a column vector, and a 22500×163 matrix $\mathbf{M}$ is obtained. In addition, the cube is scaled to have a unit maximum element. Four selected slices before scaling are shown in Figure 4.5 corresponding to the 1st, 50th, 100th and 150th columns of $\mathbf{M}$. The dimension r is chosen from $\{20, 30, 40, 50\}$, and Table 4.4 lists the average results of 10 independent trials. We can see from the table that APG-MF is superior to ADM-MF and Blockpivot-MF in both speed and solution quality.

4.2 Nonnegative matrix completion

In this section, I test Algorithm 3 on nonnegative matrix completion

$$\min_{\mathbf{A}_1, \mathbf{A}_2} \frac{1}{2} \|\mathcal{P}_\Omega(\mathbf{A}_1\mathbf{A}_2) - \mathcal{P}_\Omega(\mathbf{M})\|_F^2, \text{ subject to } \mathbf{A}_1 \in \mathbb{R}_+^{I_1 \times r}, \mathbf{A}_2 \in \mathbb{R}_+^{r \times I_2}, \tag{4.4}$$

and compare APG-MC to the ADM-based algorithm (ADM-MC) proposed in [90] on the hyperspectral data used in last test. ADM-MC is derived in essentially the same way as ADM-MF. Specifically, it applies the classic alternating direction method to

$$\min_{\mathbf{A}_1,\mathbf{A}_2,\mathbf{U},\mathbf{V},\mathbf{Z}} \frac{1}{2}\|\mathbf{A}_1\mathbf{A}_2 - \mathbf{Z}\|_F^2,$$
$$\text{subject to } \mathbf{A}_1 = \mathbf{U}, \mathbf{A}_2 = \mathbf{V}, \mathcal{P}_\Omega(\mathbf{Z}) = \mathcal{P}_\Omega(\mathbf{M}), \mathbf{U} \geq 0, \mathbf{V} \geq 0.$$

It is demonstrated in [90] that ADM-MC outperforms other matrix completion solvers such as FPCA [59] and LMaFit [88] on recovering nonnegative matrices because ADM-MC takes advantages of data nonnegativity while the latter two do not. I fix the dimension $r = 40$ in (4.4) and choose sample ratio $\text{SR} \triangleq \frac{|\Omega|}{mn}$ from $\{0.20, 0.30, 0.40\}$, where the samples in Ω are chosen at random. The parameter δ_ω for APG-MC is set to 1, and all the parameters for ADM-MC are set to their default values. To make the comparison consistent, I let both of the algorithms run to a maximum time (in second) $T = 50, 100$, and I employ relative error: $\text{relerr} = \|\mathbf{A}_1\mathbf{A}_2 - \mathbf{M}\|_F / \|\mathbf{M}\|_F$ to measure the performance of the two algorithms. Table 4.5 lists the average results of 10 independent trials. From the table, we can see that APG-MC is significantly better than ADM-MC in all cases.

4.3 Nonnegative three-way tensor factorization

This section tests Algorithm 2 on nonnegative three-way tensor factorization

$$\min_{\mathbf{A}_1,\mathbf{A}_2,\mathbf{A}_3} \frac{1}{2}\|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \mathbf{A}_3\|_F^2, \text{ subject to } \mathbf{A}_n \in \mathbb{R}_+^{I_n \times r}, n = 1, 2, 3. \tag{4.5}$$

Table 4.5 : Average relative errors and running time (sec) of each algorithm on a hyperspectral data at stopping time $T = 50, 100$ (sec); **bold** are **large error**.

$T = 50$	APG-MC (proposed)	ADM-MC	$T = 100$	APG-MC (proposed)	ADM-MC
Smpl. Rate	relerr	relerr	Smpl. Rate	relerr	relerr
0.20	1.08e-1	**1.64e-1**	0.20	1.05e-1	**1.62e-1**
0.30	4.11e-2	**9.42e-2**	0.30	3.84e-2	**9.30e-2**
0.40	2.31e-2	**5.22e-2**	0.40	2.25e-2	**5.12e-2**

To the best of my knowledge, all the existing algorithms for nonnegative tensor factorizations are extensions of those for nonnegative matrix factorization including multiplicative updating method [87], hierachical alternating least square algorithm [23], alternaing Poisson regression algorithm [21] and alternating nonnegative least square (ANLS) methods [40, 42]. I compare APG-TF with two ANLS methods AS-TF [40] and Blockpivot-TF [42], which are also proposed based on the CP decomposition and superior over many other algorithms. Both of the two ANLS methods apply block-update (1.16a), but they solve the subproblem in different ways. AS-TF solves every subproblem by using active-set method to each column of the factor matrices while Blockpivot-TF uses active-set method to a block of columns. I set $tol = 10^{-4}$ and $maxit = 2000$ for all the compared algorithms, and the same initial points are used for all three algorithms. All the other parameters for Blockpivot-TF and AS-TF are set to their default values.

Table 4.6 : Average relative errors and running time (sec) of each algorithm on synthetic three-way tensors; **bold** are **large error** or **slow time**.

Problem Setting				APG-TF (proposed)		AS-TF		Blockpivot-TF	
N_1	N_2	N_3	q	relerr	time	relerr	time	relerr	time
80	80	80	10	8.76e-005	4.39e-001	7.89e-005	**8.64e-001**	8.62e-005	8.19e-001
80	80	80	20	9.47e-005	1.26e+000	**1.97e-004**	1.45e+000	**1.77e-004**	1.21e+000
80	80	80	30	9.65e-005	**2.83e+000**	**2.05e-004**	2.13e+000	**2.07e-004**	1.95e+000
50	50	500	10	9.15e-005	1.27e+000	1.07e-004	**1.91e+000**	9.54e-005	**1.87e+000**
50	50	500	20	9.44e-005	3.42e+000	**1.86e-004**	3.17e+000	**1.77e-004**	3.47e+000
50	50	500	30	9.74e-005	**7.11e+000**	**1.89e-004**	5.04e+000	**1.88e-004**	4.54e+000

4.3.1 Synthetic data

I compare APG-TF, Blockpivot-TF and AS-TF on randomly generated three-way tensors. Each tensor has the form of $\boldsymbol{\mathcal{M}} = \mathbf{L} \circ \mathbf{C} \circ \mathbf{R}$, where $\mathbf{L}, \mathbf{C}$ are generated by MATLAB commands `max(0,randn(N1,q))` and `max(0,randn(N2,q))`, respectively, and $\mathbf{R}$ by `rand(N3,q)`. The algorithms are compared with two sets of (N_1, N_2, N_3) and $q = 10, 20, 30$. The dimension parameter in (4.5) is set to $r = q$. The relative error relerr $= \|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \mathbf{A}_3\|_F / \|\boldsymbol{\mathcal{M}}\|_F$ and CPU time (sec) measure the performance of the algorithms. The average results of 10 independent runs are shown in Table 4.6, from which we can see that all the algorithms give similar results.

4.3.2 Image test

NMF does not utilize the spatial redundancy of high-dimensional data. Its factors tend to form the invariant parts of all images as ghosts while NTF factors can correctly resolve all the parts demonstrated in [79]. Figure 4.7 and 4.8 show the factors obtained

Figure 4.6 : 16 selected images in Swimmer dataset

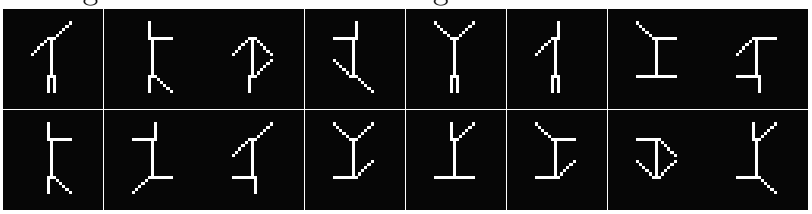
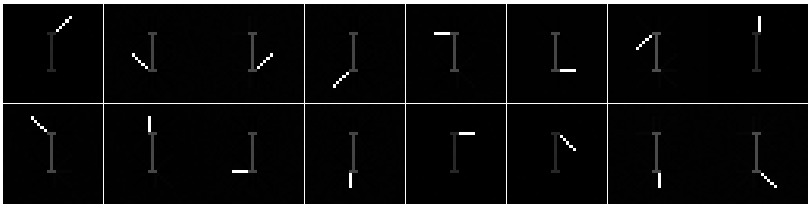


Figure 4.7 : Factor images obtained by doing NMF on Swimmer database; $r = 16$ is set in (4.1)



by doing NMF and NTF on the Swimmer database [24]. Some selected images in the Swimmer database are shown in Figure 4.6. We can see that each factor of NMF has the "torso" part as a ghost while NTF clearly factors all the parts.

This subsection compares APG-TF, Blockpivot-TF and AS-TF on two nonnegative three-way tensors used in [79]. Each slice of the tensors corresponds to an image. The first tensor is $19 \times 19 \times 2000$ and is formed from 2000 images in the CBCL

Figure 4.8 : Factor images obtained by doing NTF on Swimmer database; $r = 60$ is set in (4.5). The "limb" parts are obtained by superimposing the corresponding rank-1 factors of NTF.

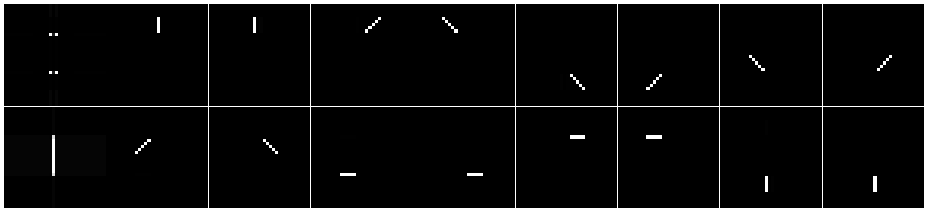


Table 4.7 : Average relative errors and running time (sec) of each algorithm on CBCL database; **bold** are **slow time**.

	APG-TF (proposed)		AS-TF		Blockpivot-TF	
r	relerr	time	relerr	time	relerr	time
40	1.85e-001	9.95e+000	1.86e-001	**2.99e+001**	1.85e-001	**2.04e+001**
50	1.68e-001	1.65e+001	1.68e-001	**4.55e+001**	1.69e-001	2.47e+001
60	1.53e-001	2.13e+001	1.56e-001	**4.16e+001**	1.56e-001	2.85e+001

database, used in Section 4.1.4. The average performance of 10 independent runs with $r = 40, 50, 60$ are shown in Table 4.7. Another one has the size of $32 \times 32 \times 256$ and is formed with the 256 images in the Swimmer dataset mentioned at the beginning of this subsection. The results of 10 independent runs with $r = 40, 50, 60$ are listed in Table 4.8. Both tests show that APG-TF is consistently faster than Blockpivot-TF and AS-TF. In particular, APG-TF is much faster than Blockpivot-TF and AS-TF with better solution quality in the second test.

In the second test, one interesting phenomenon is that APG-TF can always achieve a high accuracy if it runs sufficiently long when $r = 60$. However, Blockpivot-TF and AS-TF sometimes only achieve a low accuracy. Figure 4.9 plots the results of 8 independent runs. It implies that APG-TF is more often to avoid local minima.

4.3.3 Hyperspectral data

NTF is employed in [94] for hyperspectral unmixing. It is demonstrated that the cubic data can be highly compressed and NTF is efficient to identify the material

Figure 4.9 : Relative error versus running time (in seconds) for 8 independent trials on Swimmer dataset

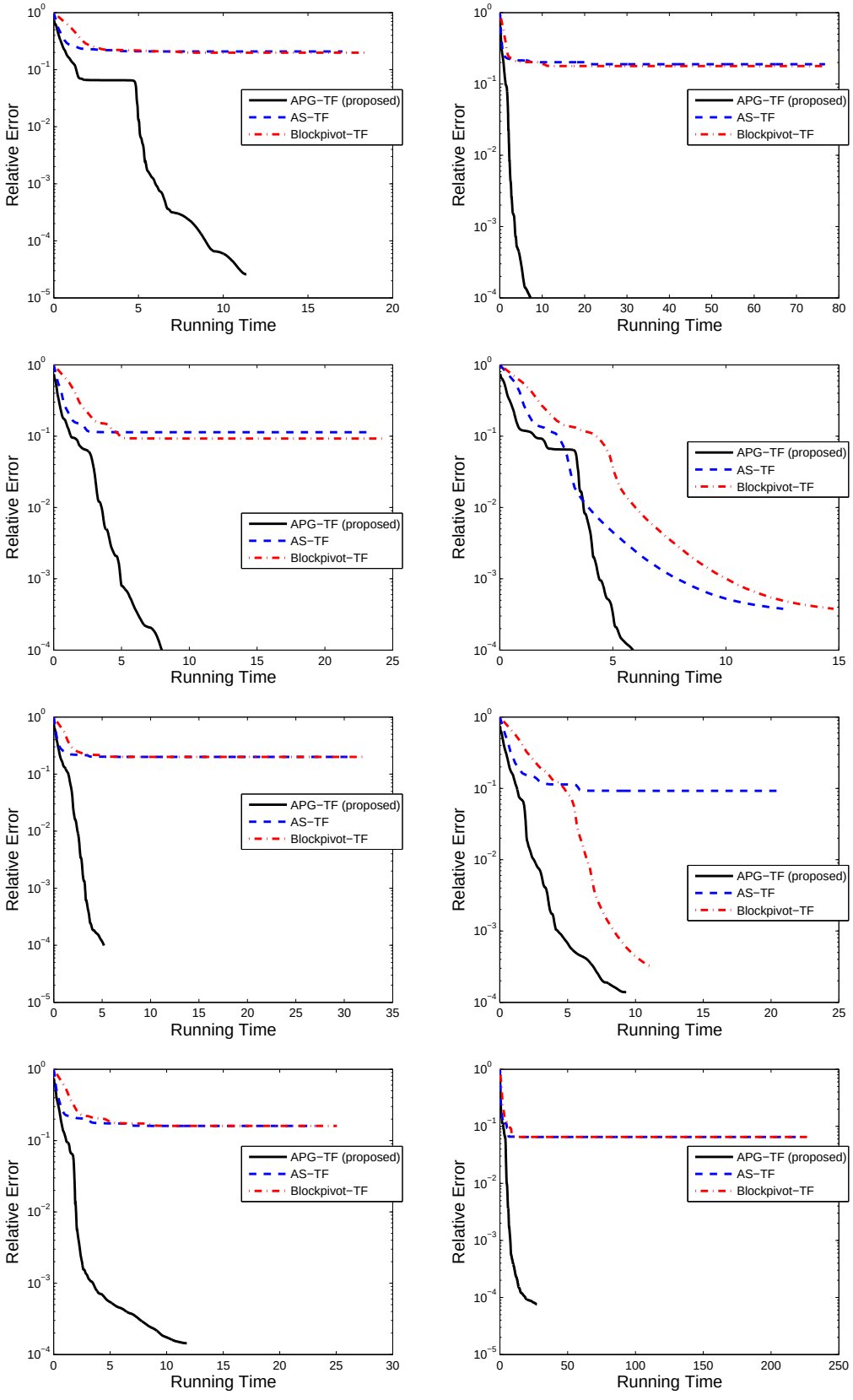


Table 4.8 : Average relative errors and running time (sec) of each algorithm on Swimmer database; **bold** are **large error** or **slow time**.

	APG-TF (proposed)		AS-TF		Blockpivot-TF	
r	relerr	time	relerr	time	relerr	time
40	2.43e-001	2.01e+000	2.71e-001	**2.09e+001**	2.53e-001	**2.50e+001**
50	1.45e-001	3.21e+000	**2.00e-001**	**5.54e+001**	1.87e-001	**3.23e+001**
60	3.16e-002	6.91e+000	**1.10e-001**	**3.55e+001**	7.63e-002	**3.74e+001**

Table 4.9 : Relative errors on hyperspectral data.

	APG-TF (proposed)				AS-TF				Blockpivot-TF			
$r \setminus T$	10	25	50	100	10	25	50	100	10	25	50	100
30	2.56e-1	2.53e-1	2.53e-1	2.53e-1	2.60e-1	2.56e-1	2.54e-1	2.53e-1	2.60e-1	2.56e-1	2.54e-1	2.53e-1
40	2.32e-1	2.27e-1	2.26e-1	2.26e-1	2.37e-1	2.30e-1	2.28e-1	2.26e-1	2.36e-1	2.29e-1	2.28e-1	2.27e-1
50	2.14e-1	2.07e-1	2.04e-1	2.04e-1	2.20e-1	2.11e-1	2.07e-1	2.06e-1	2.17e-1	2.10e-1	2.07e-1	2.05e-1
60	2.00e-1	1.91e-1	1.87e-1	1.86e-1	2.04e-1	1.95e-1	1.91e-1	1.88e-1	2.01e-1	1.94e-1	1.90e-1	1.88e-1

signatures. I compare APG-TF with Blockpivot-TF and AS-TF on the $150 \times 150 \times 163$ hyperspectral cube, which is used in Section 4.1.5. For consistency, I let them run to a maximum time T (in seconds) and compare the relative errors. The dimension parameter r is chosen from $\{30, 40, 50, 60\}$. The relative errors corresponding to $T = 10, 25, 50, 100$ are shown in Table 4.9, as the average of 10 independent trials. We can see from the table that APG-TF achieves the same accuracy much earlier than Blockpivot-TF and AS-TF.

4.4 Nonnegative tensor completion

This section tests Algorithm 3 on nonnegative three-way tensor completion

$$\min_{\mathbf{A}_1,\mathbf{A}_2,\mathbf{A}_3} \frac{1}{2}\| \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}}) - \mathcal{P}_\Omega(\mathbf{A}_1 \circ \mathbf{A}_2 \circ \mathbf{A}_3)\|_F^2, \text{ subject to } \mathbf{A}_n \in \mathbb{R}_+^{I_n \times r}, n = 1, 2, 3. \quad (4.6)$$

Recently, [54] proposed tensor completion based on minimizing tensor n-rank, the matrix rank of mode-n matricization of a tensor. Using the matrix nuclear norm instead of matrix rank, they solve the convex program

$$\min_{\boldsymbol{\mathcal{X}}} \sum_{n=1}^{N} \alpha_n \|\mathbf{X}_{(n)}\|_*, \text{ subject to } \mathcal{P}_\Omega(\boldsymbol{\mathcal{X}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}}), \quad (4.7)$$

where α_n's are pre-specified weights satisfying $\sum_n \alpha_n = 1$ and $\|\mathbf{A}\|_*$ is the nuclear norm of $\mathbf{A}$ defined as the sum of its singular values. Meanwhile, to solve (4.7) and its relaxed versions, they proposed simple low-rank tensor completion (SiLRTC), fast low-rank tensor completion (FaLRTC) and high accuracy low-rank tensor completion (HaLRTC).

SiLRTC solves the penalized problem

$$\min_{\boldsymbol{\mathcal{X}},\mathbf{Y}_1,\ldots,\mathbf{Y}_N} \sum_{n=1}^{N} \alpha_n \|\mathbf{Y}_n\|_* + \beta_n \|\mathbf{X}_{(n)} - \mathbf{Y}_n\|_F^2, \text{ subject to } \mathcal{P}_\Omega(\boldsymbol{\mathcal{X}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}})$$

by BCD using block minimization scheme (1.16a). FaLRTC first uses the technique in [63] to smooth each nuclear norm $\|\mathbf{X}_{(n)}\|_*$ by

$$g_{\mu_n}(\mathbf{X}_{(n)}) = \max_{\|\mathbf{Y}_n\| \le 1} \langle \mathbf{X}_{(n)}, \mathbf{Y}_n \rangle - \frac{\mu_n}{2}\|\mathbf{Y}_n\|_F^2.$$

It is shown in [63] that $g_{\mu_n}(\mathbf{X}_{(n)})$ is differentiable. Then FaLRTC is derived by

applying an accelerated proximal gradient method [64, 8] to

$$\min_{\boldsymbol{\mathcal{X}}} \sum_{n=1}^{N} \max_{\|\mathbf{Y}_n\| \leq 1} \alpha_n \langle \mathbf{X}_{(n)}, \mathbf{Y}_n \rangle - \frac{\mu_n}{2} \|\mathbf{Y}_n\|_F^2, \text{ subject to } \mathcal{P}_\Omega(\boldsymbol{\mathcal{X}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}}), \tag{4.8}$$

where $\|\mathbf{Y}\|$ denotes spectral norm of $\mathbf{Y}$ and equals its maximum singular value. HaLRTC applies the alternating direction method to an equivalent problem of (4.6)

$$\min_{\boldsymbol{\mathcal{X}}, \mathbf{Y}_1, \ldots, \mathbf{Y}_N} \sum_{n=1}^{N} \alpha_n \|\mathbf{Y}_n\|_*, \text{ subject to } \mathcal{P}_\Omega(\boldsymbol{\mathcal{X}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}}), \mathbf{X}_{(n)} = \mathbf{Y}_n, \forall n.$$

It is derived by alternatively minimizing the augmented Lagrangian function

$$\mathcal{L}_\rho = \sum_{n=1}^{N} \alpha_n \|\mathbf{Y}_n\|_* + \langle \mathbf{X}_{(n)} - \mathbf{Y}_n, \mathbf{\Lambda}_n \rangle + \frac{\rho}{2} \|\mathbf{X}_{(n)} - \mathbf{Y}_n\|_F^2 \tag{4.9}$$

with respect to $\boldsymbol{\mathcal{X}}$ over the constraints $\mathcal{P}_\Omega(\boldsymbol{\mathcal{X}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{M}})$ and $(\mathbf{Y}_1, \ldots, \mathbf{Y}_N)$, followed by updates of multipliers $\mathbf{\Lambda}_n$'s.

I compare APG-TC with FaLRTC and HaLRTC on synthetic three-way tensors since FaLRTC is more stable and HaLRTC gives more accurate solutions. Each tensor is generated similarly as in Section 4.3.1. Rank q is chosen from $\{5, 10, 20, 30\}$ and sampling ratio SR $= |\Omega|/(N_1 N_2 N_3)$ from $\{0.10, 0.30, 0.50\}$. Two sets of (N_1, N_2, N_3) are tested: $(N_1, N_2, N_3) = (80, 80, 80)$ and $(N_1, N_2, N_3) = (50, 50, 500)$. For APG-TC, I use exact rank estimate $r = q$ and over-estimate $r = \lfloor 1.25q \rfloor$ in (4.6). I set $tol = 10^{-4}$ and $maxit = 2000$ for all the three algorithms. The weights α_n's in (4.7) are set to $\alpha_n = \frac{1}{3}, n = 1, 2, 3$. The smoothing parameters in (4.8) for FaLRTC are set to $\mu_n = \frac{5\alpha_n}{I_n}, n = 1, 2, 3$. HaLRTC gradually increases its penalty parameter ρ in (4.9) by $\rho_{k+1} = 1.05\rho_k$, and the initial penalty parameter for HaLRTC is set to

$\rho_0 = 10^{-2}$. Its default initial value is $\rho = 10^{-6}$. However, this default setting gives very bad results for this test. All other parameters of FaLRTC and HaLRTC are set to their default values. The average relative error and running time (in seconds) of 10 independent trials are shown in Table 4.10. For APG-TC, the relative error is computed by relerr $= \|\boldsymbol{\mathcal{M}} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \mathbf{A}_3\|_F / \|\boldsymbol{\mathcal{M}}\|_F$, and for FaLRTC and HaLRTC it is computed by relerr $= \|\boldsymbol{\mathcal{M}} - \boldsymbol{\mathcal{X}}\|_F / \|\boldsymbol{\mathcal{M}}\|_F$.

From the results, we can see that APG-TC with exact rank estimate almost always produces most accurate solutions within least time. The only exception happens when $q = 30, \text{SR} = 0.1$ and $(N_1, N_2, N_3) = (80, 80, 80)$, for which I observe it gives solutions of high accuracy in all trials except one. APG-TC with over-estimated rank also gives very accurate solution while it needs a little more time. In most cases, FaLRTC and HaLRTC are slower and sometimes much slower than APG-TC. FaLRTC never gives highly accurate solutions, which may be because it uses a smoothing parameter and changes the original objective. HaLRTC can produce highly accurate solutions when there are sufficiently many samples. However, it performs very bad with low sample ratio such as when SR = 0.1. In addition, the size of the second set of problems is about 3 times larger than the first one, but FaLRTC and HaLRTC become at least 20 times slower while APG-TC is more scalable.

Table 4.10 : Comparison results on synthetic nonnegative tensor completion; **bold** are **bad** or **slow**.

Prob. Set		APG-TC $r = q$		APG-TC $r = \lfloor 1.25q \rfloor$		FaLRTC		HaLRTC	
q	SR	relerr	time	relerr	time	relerr	time	relerr	time
$N_1 = 80$, $N_2 = 80$, $N_3 = 80$									
5	0.10	2.94e-4	2.42e+0	6.79e-4	3.18e+0	**5.90e-2**	**2.10e+1**	**1.73e-1**	**1.93e+1**
5	0.30	9.38e-5	1.36e+0	5.54e-4	2.42e+0	**1.48e-2**	**1.41e+1**	1.61e-4	5.64e+0
5	0.50	8.75e-5	1.06e+0	3.03e-4	2.86e+0	8.96e-3	**1.13e+1**	1.15e-4	3.61e+0
10	0.10	1.94e-4	4.10e+0	5.80e-4	6.99e+0	**3.87e-1**	**2.74e+1**	**4.20e-1**	**1.43e+1**
10	0.30	9.73e-5	2.25e+0	3.31e-4	5.75e+0	**1.66e-2**	**1.75e+1**	2.11e-4	6.71e+0
10	0.50	9.42e-5	2.10e+0	2.57e-4	6.02e+0	**1.04e-2**	**1.43e+1**	1.13e-4	3.99e+0
20	0.10	1.38e-4	9.05e+0	4.56e-4	1.56e+1	**4.17e-1**	1.90e+1	**4.27e-1**	1.26e+1
20	0.30	1.04e-4	5.82e+0	2.86e-4	1.33e+1	**4.34e-2**	1.74e+1	**9.48e-2**	1.61e+1
20	0.50	9.72e-5	5.15e+0	2.15e-4	**1.27e+1**	**1.29e-2**	**1.74e+1**	1.91e-4	5.27e+0
30	0.10	8.71e-3	1.51e+1	3.98e-4	**2.62e+1**	**3.81e-1**	1.77e+1	**3.88e-1**	1.28e+1
30	0.30	1.03e-4	1.21e+1	1.85e-4	**2.31e+1**	**1.75e-1**	8.42e+0	**1.91e-1**	1.26e+1
30	0.50	9.98e-5	9.21e+0	1.37e-4	**1.97e+1**	**1.72e-2**	**2.08e+1**	**1.70e-2**	1.38e+1
$N_1 = 50$, $N_2 = 50$, $N_3 = 500$									
5	0.10	4.54e-4	7.48e+0	5.67e-4	1.01e+1	**3.77e-2**	**1.60e+2**	**1.03e-1**	**3.37e+2**
5	0.30	9.99e-5	3.26e+0	2.98e-4	8.77e+0	**1.21e-2**	**8.16e+1**	2.21e-4	**1.43e+2**
5	0.50	9.29e-5	2.92e+0	1.96e-4	6.44e+0	7.59e-3	**6.55e+1**	1.47e-4	**8.94e+1**
10	0.10	2.93e-4	1.14e+1	5.06e-4	1.92e+1	**1.08e-1**	**1.63e+2**	**2.80e-1**	**3.00e+2**
10	0.30	9.93e-5	6.43e+0	2.72e-4	1.68e+1	**1.37e-2**	**1.11e+2**	2.49e-4	**1.63e+2**
10	0.50	9.32e-5	5.50e+0	1.66e-4	1.76e+1	9.07e-3	**8.41e+1**	1.50e-4	9.66e+1
20	0.10	1.65e-4	2.25e+1	3.87e-4	4.62e+1	**3.13e-1**	**1.40e+2**	**3.56e-1**	**2.55e+2**
20	0.30	1.06e-4	1.38e+1	1.69e-4	3.65e+1	**1.73e-2**	**1.53e+2**	1.42e-3	**2.24e+2**
20	0.50	1.01e-4	1.33e+1	1.14e-4	3.46e+1	**1.14e-2**	**1.07e+2**	1.95e-4	**1.17e+2**
30	0.10	1.44e-4	4.33e+1	3.23e-4	7.13e+1	**3.18e-1**	**1.33e+2**	**3.45e-1**	**2.51e+2**
30	0.30	1.11e-4	2.84e+1	1.31e-4	6.34e+1	**3.75e-2**	**1.81e+2**	**8.63e-2**	**2.72e+2**
30	0.50	1.00e-4	2.32e+1	1.06e-4	5.51e+1	**1.40e-2**	**1.31e+2**	2.49e-4	**1.42e+2**

4.5 Summary

Although the test results are obtained with a given set of parameters, it is clear from the results that, compared to the existing algorithms, the proposed ones can return solutions of similar or better quality in less time. Tuning the parameters of the compared algorithms can hardly obtain much improvement in both solution quality and time. I believe that the superior performance of the proposed algorithms is due to the use of prox-linear steps, which are not only easy to compute but also, as a local approximate, help avoid the small regions around certain local minima.

Chapter 5

Conclusions

I have proposed a block coordinate descent method with three choices of block-update schemes for regularized multi-convex optimization. Under the assumptions of block strong convexity and/or Lipschitz continuity, I obtain a square summable result about the difference of iterates, from which a subsequence convergence is established. Further assuming isolation of Nash points, I establish a global convergence result. Due to the difficulty of verifying the isolation condition, another property called Kurdyka-Łojasiewicz property is reviewed and employed to show the global convergence and asymptotic convergence rate. The algorithm has been applied to nonnegative matrix/tensor factorization and completion problems, which have the Kurdyka-Łojasiewicz property and thus have global convergence. Numerical results on both synthetic and real image data illustrate the high efficiency of the proposed algorithm. Compared to some state-of-the-art algorithms, the proposed ones are not only faster but also can achieve higher accuracy.

Appendix A

Some proofs

A.1 Proof of Lemma 2.3

Without loss of generality, we assume $\bar{F} = 0$. Otherwise, we can consider $F - \bar{F}$. Let $B(\bar{\mathbf{x}}, \rho) \triangleq \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \rho\} \subset \mathcal{U}$ for some $\rho > 0$ where $\mathcal{U}$ is the neighborhood of $\bar{\mathbf{x}}$ in (2.21), and let L_G be the global Lipschitz constant for $\nabla_{\mathbf{x}_i} f(\mathbf{x}), i = 1, \cdots, s$ within $B(\bar{\mathbf{x}}, \sqrt{10}\rho)$, namely,

$$\|\nabla_{\mathbf{x}_i} f(\mathbf{x}) - \nabla_{\mathbf{x}_i} f(\mathbf{y})\| \leq L_G \|\mathbf{x} - \mathbf{y}\|, \ i = 1, \cdots, s$$

for any $\mathbf{x}, \mathbf{y} \in B(\bar{\mathbf{x}}, \sqrt{10}\rho)$.

The proof will follow two steps. The first step will show

Claim A.1
Let $\ell = \min_i \ell_i, \ L = \max_i L_i$ *and*

$$C_1 = \frac{9(L + sL_G)}{2\ell(1 - \delta_\omega)^2}, \ C_2 = 2\sqrt{\frac{2}{\ell}} + \frac{3}{1 - \delta_\omega}\sqrt{\frac{2 + 2\delta_\omega^2}{\ell}},$$

where ℓ_i, L_i*'s are the constants in Assumption 2.2. If* $F_k > \bar{F}$ *and*

$$C_1 \phi(F_0 - \bar{F}) + C_2 \sqrt{F_0 - \bar{F}} + \|\mathbf{x}^0 - \bar{\mathbf{x}}\| < \rho, \tag{A.1}$$

then

$$\mathbf{x}^k \in B(\bar{\mathbf{x}}, \rho), \ \forall k. \tag{A.2}$$

Note that (A.1) quantifies how close to $\bar{\mathbf{x}}$ the initial point $\mathbf{x}^0$ is required. The second step will establish

Claim A.2

$$\sum_{k=N}^{\infty} \|\mathbf{x}^{k+1} - \mathbf{x}^k\| \leq C_1 \phi(F_N - \bar{F}) + \|\mathbf{x}^{N-1} - \mathbf{x}^{N-2}\| + \frac{2 + \delta_\omega}{1 - \delta_\omega} \|\mathbf{x}^N - \mathbf{x}^{N-1}\|, \forall N \geq 2, \tag{A.3}$$

where C_1 *is specified in Claim A.1.*

Note (A.3) implies $\{\mathbf{x}^k\}$ is a Cauchy sequence, and thus $\mathbf{x}^k$ converges. Hence, if (A.2) and (A.3) both hold, then letting $\mathcal{B} = B(\bar{\mathbf{x}}, \rho)$ will prove the results of Lemma 2.3.

Proof of Claim A.1

We will prove $\mathbf{x}^k \in B(\bar{\mathbf{x}}, \rho)$ by induction on k.

Obviously, $\mathbf{x}^0 \in B(\bar{\mathbf{x}}, \rho)$ from (A.1). Hence, (A.2) holds for $k = 0$.

For $k = 1$, we have from (2.12) that

$$F_0 \geq F_0 - F_1 \geq \sum_{i=1}^{s} \frac{L_i^0}{2} \|\mathbf{x}_i^0 - \mathbf{x}_i^1\|^2 \geq \frac{\ell}{2} \|\mathbf{x}^0 - \mathbf{x}^1\|^2.$$

Hence, $\|\mathbf{x}^0 - \mathbf{x}^1\| \leq \sqrt{\frac{2}{\ell} F_0}$, and

$$\|\mathbf{x}^1 - \bar{\mathbf{x}}\| \leq \|\mathbf{x}^0 - \mathbf{x}^1\| + \|\mathbf{x}^0 - \bar{\mathbf{x}}\| \leq \sqrt{\frac{2}{\ell} F_0} + \|\mathbf{x}^0 - \bar{\mathbf{x}}\|,$$

which indicates $\mathbf{x}^1 \in B(\bar{\mathbf{x}}, \rho)$.

For $k = 2$, we have from (2.12) that (regard $\omega_i^k \equiv 0$ for $i \in \mathcal{I}_1 \cup \mathcal{I}_2$)

$$F_0 \geq F_1 - F_2 \geq \sum_{i=1}^{s} \frac{L_i^1}{2} \|\mathbf{x}_i^1 - \mathbf{x}_i^2\|^2 - \sum_{i=1}^{s} \frac{L_i^1}{2} (\omega_i^1)^2 \|\mathbf{x}_i^0 - \mathbf{x}_i^1\|^2.$$

Note $L_i^1 (\omega_i^1)^2 \leq \delta_\omega^2 \ell^0$ for $i = 1, \cdots, s$. Thus, it follows from the above inequality that

$$\frac{\ell^1}{2} \|\mathbf{x}^1 - \mathbf{x}^2\|^2 \leq \sum_{i=1}^{s} \frac{L_i^1}{2} \|\mathbf{x}_i^1 - \mathbf{x}_i^2\|^2 \leq F_0 + \frac{\ell^0}{2} \delta_\omega^2 \|\mathbf{x}^0 - \mathbf{x}^1\|^2 \leq (1 + \frac{\ell^0}{\ell} \delta_\omega^2) F_0,$$

which implies $\|\mathbf{x}^1 - \mathbf{x}^2\| \leq \sqrt{\frac{2+2\delta_\omega^2}{\ell} F_0}$. Therefore,

$$\|\mathbf{x}^2 - \bar{\mathbf{x}}\| \leq \|\mathbf{x}^1 - \mathbf{x}^2\| + \|\mathbf{x}^1 - \bar{\mathbf{x}}\| \leq \left(\sqrt{\frac{2}{\ell}} + \sqrt{\frac{2 + 2\delta_\omega^2}{\ell}} \right) \sqrt{F_0} + \|\mathbf{x}^0 - \bar{\mathbf{x}}\|,$$

and thus $\mathbf{x}^2 \in B(\bar{\mathbf{x}}, \rho)$.

Suppose $\mathbf{x}^k \in B(\bar{\mathbf{x}}, \rho)$ for $0 \leq k \leq K$. We go to show $\mathbf{x}^{K+1} \in B(\bar{\mathbf{x}}, \rho)$. For $k \leq K$, note

$$\begin{aligned}
-\nabla f_i^k(\mathbf{x}_i^k) + \nabla_{\mathbf{x}_i} f(\mathbf{x}^k) &\in \partial r_i(\mathbf{x}_i^k) + \nabla_{\mathbf{x}_i} f(\mathbf{x}^k), \ i \in \mathcal{I}_1, \\
-L_i^{k-1}(\mathbf{x}_i^k - \mathbf{x}_i^{k-1}) - \nabla f_i^k(\mathbf{x}_i^k) + \nabla_{\mathbf{x}_i} f(\mathbf{x}^k) &\in \partial r_i(\mathbf{x}_i^k) + \nabla_{\mathbf{x}_i} f(\mathbf{x}^k), \ i \in \mathcal{I}_2, \\
-L_i^{k-1}(\mathbf{x}_i^k - \hat{\mathbf{x}}_i^{k-1}) - \nabla f_i^k(\hat{\mathbf{x}}_i^{k-1}) + \nabla_{\mathbf{x}_i} f(\mathbf{x}^k) &\in \partial r_i(\mathbf{x}_i^k) + \nabla_{\mathbf{x}_i} f(\mathbf{x}^k), \ i \in \mathcal{I}_3,
\end{aligned}$$

and

$$\partial F(\mathbf{x}^k) = \left\{\partial r_1(\mathbf{x}_1^k) + \nabla_{\mathbf{x}_1} f(\mathbf{x}^k)\right\} \times \cdots \times \left\{\partial r_s(\mathbf{x}_s^k) + \nabla_{\mathbf{x}_s} f(\mathbf{x}^k)\right\},$$

so (for $i \in \mathcal{I}_1 \cup \mathcal{I}_2$, regard $\hat{\mathbf{x}}_i^{k-1} = \mathbf{x}_i^{k-1}$ in $\mathbf{x}_i^k - \hat{\mathbf{x}}_i^{k-1}$ and $\hat{\mathbf{x}}_i^{k-1} = \mathbf{x}_i^k$ in $\nabla f_i^k(\hat{\mathbf{x}}_i^{k-1}) - \nabla_{\mathbf{x}_i} f(\mathbf{x}^k)$, respectively)

$$\begin{aligned} & \operatorname{dist}\left(0, \partial F(\mathbf{x}^k)\right) \\ \leq\ & \left\|\left(L_1^{k-1}(\mathbf{x}_1^k - \hat{\mathbf{x}}_1^{k-1}), \cdots, L_s^{k-1}(\mathbf{x}_s^k - \hat{\mathbf{x}}_s^{k-1})\right)\right\| \\ & + \textstyle\sum_{i=1}^s \left\|\nabla f_i^k(\hat{\mathbf{x}}_i^{k-1}) - \nabla_{\mathbf{x}_i} f(\mathbf{x}^k)\right\|. \end{aligned} \tag{A.4}$$

For the first term in (A.4), plugging in $\hat{\mathbf{x}}_i^{k-1}$ and recalling $L_i^{k-1} \leq L, \omega_i^{k-1} \leq 1$ for $i = 1, \cdots, s$, we can easily get

$$\begin{aligned} & \left\|\left(L_1^{k-1}(\mathbf{x}_1^k - \hat{\mathbf{x}}_1^{k-1}), \cdots, L_s^{k-1}(\mathbf{x}_s^k - \hat{\mathbf{x}}_s^{k-1})\right)\right\| \\ \leq\ & L\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right). \end{aligned} \tag{A.5}$$

For the second term in (A.4), it is not difficult to verify

$$\left(\mathbf{x}_1^k, \cdots, \mathbf{x}_{i-1}^k, \hat{\mathbf{x}}_i^{k-1}, \cdots, \mathbf{x}_s^{k-1}\right) \in B(\bar{\mathbf{x}}, \sqrt{10}\rho).$$

In addition, note

$$\nabla f_i^k(\hat{\mathbf{x}}_i^{k-1}) = \nabla_{\mathbf{x}_i} f\left(\mathbf{x}_1^k, \cdots, \mathbf{x}_{i-1}^k, \hat{\mathbf{x}}_i^{k-1}, \cdots, \mathbf{x}_s^{k-1}\right).$$

Hence,

$$\begin{aligned} & \sum_{i=1}^s \left\|\nabla_{\mathbf{x}_i} f_i^k(\hat{\mathbf{x}}_i^{k-1}) - \nabla_{\mathbf{x}_i} f(\mathbf{x}^k)\right\| \\ \leq\ & \sum_{i=1}^s L_G \left\|\left(\mathbf{x}_1^k, \cdots, \mathbf{x}_{i-1}^k, \hat{\mathbf{x}}_i^{k-1}, \cdots, \mathbf{x}_s^{k-1}\right) - \mathbf{x}^k\right\| \\ \leq\ & sL_G\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right). \end{aligned} \tag{A.6}$$

Combining (A.4), (A.5) and (A.6) gives

$$\operatorname{dist}(\mathbf{0}, \partial F(\mathbf{x}^k)) \leq (L + sL_G)\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right),$$

which together with the KL inequality (2.21) implies

$$\phi'(F_k) \geq (L + sL_G)^{-1}\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right)^{-1}. \tag{A.7}$$

Note that ϕ is concave and $\phi'(F_k) > 0$. Thus it follows from (2.12) and (A.7) that

$$\begin{aligned}\phi(F_k) - \phi(F_{k+1}) \geq & \phi'(F_k)(F_k - F_{k+1}) \\ \geq & \frac{\sum_{i=1}^{s} \left(L_i^k \|\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\|^2 - \ell^{k-1}\delta_\omega^2 \|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2\right)}{2(L + sL_G)\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right)},\end{aligned}$$

or equivalently

$$\begin{aligned}\sum_{i=1}^{s} L_i^k \|\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\|^2 \leq & 2(L + sL_G)\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right)(\phi(F_k) - \phi(F_{k+1})) \\ & + \sum_{i=1}^{s} \ell^{k-1}\delta_\omega^2 \|\mathbf{x}_i^{k-1} - \mathbf{x}_i^k\|^2.\end{aligned}$$

Recalling $\ell \leq \ell^{k-1} \leq \ell^k \leq L_i^k \leq L$ for all i, k, we have from the above inequality that

$$\begin{aligned}& \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2 \\ \leq & \tfrac{2(L+sL_G)}{\ell}\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right)(\phi(F_k) - \phi(F_{k+1})) \\ & + \delta_\omega^2 \|\mathbf{x}^{k-1} - \mathbf{x}^k\|^2.\end{aligned} \tag{A.8}$$

Using inequalities $a^2 + b^2 \leq (a+b)^2$ and $ab \leq ta^2 + \frac{b^2}{4t}$ for $t > 0$, we get from (A.8) that

$$\begin{aligned}& \|\mathbf{x}^k - \mathbf{x}^{k+1}\| \\ \leq & \left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right)^{\frac{1}{2}} \left(\frac{2(L + sL_G)}{\ell}(\phi(F_k) - \phi(F_{k+1}))\right)^{\frac{1}{2}} \\ & + \delta_\omega \|\mathbf{x}^{k-1} - \mathbf{x}^k\| \\ \leq & \frac{1 - \delta_\omega}{3}\left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right) + \frac{3(L + sL_G)}{2\ell(1 - \delta_\omega)}(\phi(F_k) - \phi(F_{k+1})) \\ & + \delta_\omega \|\mathbf{x}^{k-1} - \mathbf{x}^k\|,\end{aligned}$$

or equivalently

$$\begin{aligned}& 3\|\mathbf{x}^k - \mathbf{x}^{k+1}\| \\ \leq & (1 + 2\delta_\omega)\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + (1 - \delta_\omega)\|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\| \\ & + \tfrac{9(L+sL_G)}{2\ell(1-\delta_\omega)}(\phi(F_k) - \phi(F_{k+1})).\end{aligned} \tag{A.9}$$

Summing up (A.9) over k from 2 to K and doing some eliminations give

$$\begin{aligned}&\sum_{k=2}^{K}(1-\delta_\omega)\|\mathbf{x}^k-\mathbf{x}^{k+1}\|+(2+\delta_\omega)\|\mathbf{x}^K-\mathbf{x}^{K+1}\|+(1-\delta_\omega)\|\mathbf{x}^{K-1}-\mathbf{x}^K\|\\ \leq&(1-\delta_\omega)\|\mathbf{x}^0-\mathbf{x}^1\|+(2+\delta_\omega)\|\mathbf{x}^1-\mathbf{x}^2\|+\frac{9(L+sL_G)}{2\ell(1-\delta_\omega)}(\phi(F_2)-\phi(F_{K+1})).\end{aligned}$$

Recalling $\|\mathbf{x}^0-\mathbf{x}^1\|\leq\sqrt{\frac{2}{\ell}F_0}$ and $\|\mathbf{x}^1-\mathbf{x}^2\|\leq\sqrt{\frac{2+2\delta_\omega^2}{\ell}F_0}$, we have from the above inequality that

$$\begin{aligned}\|\mathbf{x}^{K+1}-\bar{\mathbf{x}}\|\leq&\sum_{k=2}^{K}\|\mathbf{x}^k-\mathbf{x}^{k+1}\|+\|\mathbf{x}^2-\bar{\mathbf{x}}\|\\ \leq&\sqrt{\frac{2}{\ell}F_0}+\frac{2+\delta_\omega}{1-\delta_\omega}\sqrt{\frac{2+2\delta_\omega^2}{\ell}F_0}+\frac{9(L+sL_G)}{2\ell(1-\delta_\omega)^2}(\phi(F_2)-\phi(F_{K+1}))\\ &+\|\mathbf{x}^2-\bar{\mathbf{x}}\|\\ \leq&\frac{9(L+sL_G)}{2\ell(1-\delta_\omega)^2}\phi(F_0)+\left(2\sqrt{\frac{2}{\ell}}+\frac{3}{1-\delta_\omega}\sqrt{\frac{2+2\delta_\omega^2}{\ell}}\right)\sqrt{F_0}+\|\mathbf{x}^0-\bar{\mathbf{x}}\|.\end{aligned}$$

Hence, $\mathbf{x}^{K+1}\in B(\bar{\mathbf{x}},\rho)$, and this completes the proof of Claim A.1.

Proof of Claim A.2

We will prove (A.3) from (A.9). Indeed, (A.9) holds for all $k\geq 0$. Summing it over k from N to T and doing some eliminations yield

$$\begin{aligned}&\sum_{k=N}^{T}(1-\delta_\omega)\|\mathbf{x}^k-\mathbf{x}^{k+1}\|+(2+\delta_\omega)\|\mathbf{x}^T-\mathbf{x}^{T+1}\|+(1-\delta_\omega)\|\mathbf{x}^{T-1}-\mathbf{x}^T\|\\ \leq&(1-\delta_\omega)\|\mathbf{x}^{N-2}-\mathbf{x}^{N-1}\|+(2+\delta_\omega)\|\mathbf{x}^{N-1}-\mathbf{x}^N\|+\frac{9(L+sL_G)}{2\ell(1-\delta_\omega)}(\phi(F_N)-\phi(F_{T+1})),\end{aligned}$$

which implies

$$\sum_{k=N}^{\infty}\|\mathbf{x}^k-\mathbf{x}^{k+1}\|\leq\|\mathbf{x}^{N-2}-\mathbf{x}^{N-1}\|+\frac{2+\delta_\omega}{1-\delta_\omega}\|\mathbf{x}^{N-1}-\mathbf{x}^N\|+\frac{9(L+sL_G)}{2\ell(1-\delta_\omega)^2}\phi(F_N)$$

by letting $T\to\infty$. This completes the proof of Claim A.2.

A.2 Proof of Theorem 2.4

If $\theta = 0$, we must have $F(\mathbf{x}^{k_0}) = F(\bar{\mathbf{x}})$ for some k_0. Otherwise, $F(\mathbf{x}^k) > F(\bar{\mathbf{x}})$ for all sufficiently large k. The Kurdyka-Łojasiewicz inequality gives $c \cdot \text{dist}(\mathbf{0}, \partial F(\mathbf{x}^k)) \geq 1$ for all $k \geq 0$, which is impossible since $\mathbf{x}^k \to \bar{\mathbf{x}}$ and $\mathbf{0} \in \partial F(\bar{\mathbf{x}})$. The finite convergence now follows from the fact that $F(\mathbf{x}^{k_0}) = F(\bar{\mathbf{x}})$ implies $\mathbf{x}^k = \mathbf{x}^{k_0} = \bar{\mathbf{x}}$ for all $k \geq k_0$.

For $\theta \in (0, 1)$, we assume $F(\mathbf{x}^k) > F(\bar{\mathbf{x}}) = 0$ and use the same notation as in the proof of Lemma 3. Define

$$S_k = \sum_{i=k}^{\infty} \|\mathbf{x}^i - \mathbf{x}^{i+1}\|.$$

Then (A.3) can be written as

$$S_k \leq C_1 \phi(F_k) + \frac{2 + \delta_\omega}{1 - \delta_\omega}(S_{k-1} - S_k) + S_{k-2} - S_{k-1}, \text{ for } k \geq 2,$$

which implies

$$S_k \leq C_1 \phi(F_k) + \frac{2 + \delta_\omega}{1 - \delta_\omega}(S_{k-2} - S_k), \text{ for } k \geq 2, \tag{A.10}$$

since $S_{k-2} - S_{k-1} \geq 0$. Using $\phi(s) = cs^{1-\theta}$, we have from (A.7) for sufficiently large k that

$$c(1-\theta)(F_k)^{-\theta} \geq (L + sL_G)^{-1} \left(\|\mathbf{x}^k - \mathbf{x}^{k-1}\| + \|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\|\right)^{-1},$$

or equivalently $(F_k)^\theta \leq c(1-\theta)(L + sL_G)(S_{k-2} - S_k)$. Then,

$$\phi(F_k) = c(F_k)^{1-\theta} \leq c\big(c(1-\theta)(L + sL_G)(S_{k-2} - S_k)\big)^{\frac{1-\theta}{\theta}}. \tag{A.11}$$

Letting $C_3 = C_1 c\big(c(1-\theta)(L + sL_G)\big)^{\frac{1-\theta}{\theta}}$ and $C_4 = \frac{2+\delta_\omega}{1-\delta_\omega}$, we have from (A.10) and (A.11) that

$$S_k \leq C_3 \left(S_{k-2} - S_k\right)^{\frac{1-\theta}{\theta}} + C_4 \left(S_{k-2} - S_k\right). \tag{A.12}$$

When $\theta \in (0, \frac{1}{2}]$, i.e., $\frac{1-\theta}{\theta} \geq 1$, (A.12) implies that $S_k \leq (C_3 + C_4)(S_{k-2} - S_k)$ for sufficiently large k since $S_{k-2} - S_k \to 0$, and thus

$$S_k \leq \frac{C_3 + C_4}{1 + C_3 + C_4} S_{k-2}.$$

Note that $\|\mathbf{x}^k - \bar{\mathbf{x}}\| \leq S_k$. Therefore, item 2 holds with $\tau = \sqrt{\frac{C_3+C_4}{1+C_3+C_4}} < 1$ and sufficiently large C.

When $\theta \in (\frac{1}{2}, 1)$, i.e., $\frac{1-\theta}{\theta} < 1$, we can show

$$S_N^\nu + S_{N-1}^\nu - S_{K+1}^\nu - S_K^\nu \geq \mu(N - K), \tag{A.13}$$

for $\nu = \frac{1-2\theta}{1-\theta} < 0$, some constant $\mu > 0$ and any $N > K$ with sufficiently large K by the same argument as in the proof of Theorem 2 of [2]. Note $S_N \leq S_{N-1}$ and $\nu < 0$. Hence, (A.13) implies

$$S_N \leq \left(\frac{1}{2} \left(S_{K+1}^{\nu} + S_K^{\nu} + \mu(N-K) \right) \right)^{\frac{1}{\nu}} \leq C N^{-\frac{1-\theta}{2\theta-1}},$$

for sufficiently large C and N. This completes the proof.

For completeness, I give the proof of (A.13) below by repeating the arguments in Theorem 2 of [2].

Proof of (A.13)

Note that $S_k \to 0$ and $\frac{1-\theta}{\theta} < 1$. We have from (A.12) that

$$S_k^{\frac{\theta}{1-\theta}} \leq C_5 (S_{k-2} - S_k), \tag{A.14}$$

for $C_5 = \max(C_3, C_4) + 1$ and all $k \geq K$ with sufficiently large K such that $S_{K-2} < 1$. Define

$$h(s) = s^{-\frac{\theta}{1-\theta}}$$

and let $c \in (1, +\infty)$. Take $k \geq K$ and go to show

$$S_k^{\nu} - S_{k-2}^{\nu} \geq \mu > 0, \tag{A.15}$$

for some μ and all $k \geq K$.

Case 1: $h(S_k) \leq c h(S_{k-2})$. Writing (A.14) to

$$1 \leq C_5 (S_{k-2} - S_k) S_k^{-\frac{\theta}{1-\theta}},$$

we have

$$\begin{aligned} 1 &\leq C_5 (S_{k-2} - S_k) h(S_k) \\ &\leq c C_5 (S_{k-2} - S_k) h(S_{k-2}) \\ &\leq c C_5 \int_{S_k}^{S_{k-2}} h(s) ds \\ &= c C_5 \frac{1-\theta}{1-2\theta} \left[S_{k-2}^{\nu} - S_k^{\nu} \right]. \end{aligned}$$

Letting $\hat{\mu} = \frac{\nu}{c C_5}$, we get

$$0 < \hat{\mu} \leq S_{k-2}^{\nu} - S_k^{\nu}.$$

Case 2: $h(S_k) > ch(S_{k-2})$. Set $q = c^{\frac{\theta}{1-\theta}} \in (0,1)$. Then $S_k \leq qS_{k-2}$ and $S_k^\nu \geq q^\nu S_{k-2}^\nu$ or equivalently

$$S_k^\nu - S_{k-2}^\nu \geq (q^\nu - 1)S_{k-2}^\nu$$

by noting $\nu < 0$. Since $S_{k-2} < 1, \forall k > K$ and $q^\nu > 1$, there exists $\bar{\mu} > 0$ such that $(q^\nu - 1)S_{k-2}^\nu > \bar{\mu}$ for all $k > K$. Hence,

$$S_{k-2}^\nu - S_k^\nu > \bar{\mu}.$$

Taking $\mu = \min(\hat{\mu}, \bar{\mu})$, we get (A.15). Summing (A.15) from K to some integer $N > K$ gives (A.13).

Bibliography

[1] S. Amari, A. Cichocki, H.H. Yang, et al. A new learning algorithm for blind signal separation. *Advances in neural information processing systems*, pages 757–763, 1996.

[2] H. Attouch and J. Bolte. On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Mathematical Programming*, 116(1):5–16, 2009.

[3] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Lojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010.

[4] A. Auslender. *Optimisation: méthodes numériques*. Masson, 1976.

[5] A. Auslender. Asymptotic properties of the fenchel dual functional and applications to decomposition problems. *Journal of optimization theory and applications*, 73(3):427–449, 1992.

[6] B. W. Bader, T. G. Kolda, et al. Matlab tensor toolbox version 2.5, January 2012.

[7] B.W. Bader and T.G. Kolda. Efficient matlab computations with sparse and factored tensors. *SIAM Journal on Scientific Computing*, 30(1):205–231, 2009.

[8] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[9] E. Benetos and C. Kotropoulos. Non-negative tensor factorization applied to music genre classification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(8):1955–1967, 2010.

[10] M.W. Berry, M. Browne, A.N. Langville, V.P. Pauca, and R.J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis*, 52(1):155–173, 2007.

[11] D.P. Bertsekas and P. Tseng. Partial proximal minimization algorithms for convex pprogramming. *SIAM Journal on Optimization*, 4(3):551–572, 1994.

[12] C.M. Bishop et al. *Pattern recognition and machine learning*, volume 4. springer New York, 2006.

[13] J. Bobin, Y. Moudden, J.L. Starck, J. Fadili, and N. Aghanim. SZ and CMB reconstruction using generalized morphological component analysis. *Statistical Methodology*, 5(4):307–317, 2008.

[14] J. Bochnak, M. Coste, and M.F. Roy. *Real algebraic geometry*, volume 36. Springer Verlag, 1998.

[15] P. Bofill and M. Zibulevsky. Underdetermined blind source separation using sparse representations. *Signal processing*, 81(11):2353–2362, 2001.

[16] J. Bolte, A. Daniilidis, and A. Lewis. The Lojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. *SIAM Journal on Optimization*, 17(4):1205–1223, 2007.

[17] J. Bolte, A. Daniilidis, A. Lewis, and M. Shiota. Clarke subgradients of stratifiable functions. *SIAM Journal on Optimization*, 18(2):556–572, 2007.

[18] J.D. Carroll and J.J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika*, 35(3):283–319, 1970.

[19] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, pages 129–159, 2001.

[20] Y. Chen, M. Rege, M. Dong, and J. Hua. Non-negative matrix factorization for semi-supervised data clustering. *Knowledge and Information Systems*, 17(3):355–379, 2008.

[21] E.C. Chi and T.G. Kolda. On tensors, sparsity, and nonnegative factorizations. *Arxiv preprint arXiv:1112.2414*, 2011.

[22] S. Choi, A. Cichocki, H.M. Park, and S.Y. Lee. Blind source separation and independent component analysis: A review. *Neural Information Processing-Letters and Reviews*, 6(1):1–57, 2005.

[23] A. Cichocki and A.H. Phan. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE transactions on fundamentals of electronics, communications and computer science*, 92(3):708–721, 2009.

[24] D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts. *Advances in neural information processing systems*, 16, 2003.

[25] Jonathan Eckstein and Dimitri P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Programming*, 55(3, Ser. A):293–318, 1992.

[26] M.P. Friedlander and K. Hatz. Computing non-negative tensor factorizations. *Optimisation Methods and Software*, 23(4):631–647, 2008.

[27] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The annals of statistics*, 28(2):337–407, 2000.

[28] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Computers and Mathematics with Applications*, 2:17–40, 1976.

[29] R. Glowinski and A. Marrocco. Sur lapproximation par elements finis dordre un, et la resolution par penalisation-dualite dune classe de problemes de Dirichlet nonlineaires. *Rev. Francaise dAut. Inf. Rech. Oper.*, 2:41–76, 1975.

[30] L. Grippo and M. Sciandrone. On the convergence of the block nonlinear Gauss-Seidel method under convex constraints. *Oper. Res. Lett.*, 26(3):127–136, 2000.

[31] S.P. Han. A successive projection method. *Mathematical Programming*, 40(1):1–14, 1988.

[32] C. Hildreth. A quadratic programming procedure. *Naval Research Logistics Quarterly*, 4(1):79–85, 1957.

[33] N.D. Ho, P. Van Dooren, and V.D. Blondel. Descent methods for nonnegative matrix factorization. *Numerical Linear Algebra in Signals, Systems and Control*, pages 251–293, 2011.

[34] P.O. Hoyer. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004.

[35] T.P. Jung, S. Makeig, C. Humphries, T.W. Lee, M.J. Mckeown, V. Iragui, and T.J. Sejnowski. Removing electroencephalographic artifacts by blind source separation. *Psychophysiology*, 37(02):163–178, 2000.

[36] C. Jutten and J. Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal processing*, 24(1):1–10, 1991.

[37] J. Karhunen, A. Hyvarinen, R. Vigário, J. Hurri, and E. Oja. Applications of neural blind separation to signal and image processing. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, pages 131–134. IEEE, 1997.

[38] H.A.L. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, 14(3):105–122, 2000.

[39] H. Kim and H. Park. Non-negative matrix factorization based on alternating non-negativity constrained least squares and active set method. *SIAM J. Matrix Anal. Appl*, 30(2):713–730, 2008.

[40] H. Kim, H. Park, and L. Eldén. Non-negative tensor factorization based on alternating large-scale non-negativity-constrained least squares. In *Bioinformatics and Bioengineering, 2007. BIBE 2007. Proceedings of the 7th IEEE International Conference on*, pages 1147–1151. IEEE, 2007.

[41] J. Kim and H. Park. Toward faster nonnegative matrix factorization: A new algorithm and comparisons. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 353–362. IEEE, 2008.

[42] J. Kim and H. Park. Fast nonnegative tensor factorization with an active-set-like method. *High-Performance Scientific Computing*, pages 311–326, 2012.

[43] Y.D. Kim and S. Choi. Nonnegative Tucker decomposition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[44] T.G. Kolda and B.W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455, 2009.

[45] K. Kurdyka. On gradients of functions definable in o-minimal structures. In *Annales de l'institut Fourier*, volume 48, pages 769–784. Chartres: L'Institut, 1950-, 1998.

[46] M. Lai and Y. Wang. An unconstrained ℓ_q minimization with $0 < q < 1$ for sparse solution of under-determined linear systems. *SIAM J. Optimization*, 21:82–101, 2011.

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

[48] D.D. Lee and H.S. Seung. Algorithms for Non-Negative Matrix Factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[49] H. Lee, A. Battle, R. Raina, and A.Y. Ng. Efficient sparse coding algorithms. *Advances in neural information processing systems*, 19:801–808, 2007.

[50] S.Z. Li, X.W. Hou, H.J. Zhang, and Q.S. Cheng. Learning spatially localized, parts-based representation. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages 207–212. IEEE, 2001.

[51] C.J. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Computation*, 19(10):2756–2779, 2007.

[52] J.K. Lin, D.G. Grier, and J.D. Cowan. Feature extraction approach to blind source separation. In *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, pages 398–405. IEEE, 1997.

[53] J. Liu, J. Liu, P. Wonka, and J. Ye. Sparse non-negative tensor factorization using columnwise coordinate descent. *Pattern Recognition*, 2011.

[54] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[55] S. Łojasiewicz. Sur la géométrie semi-et sous-analytique. *Ann. Inst. Fourier (Grenoble)*, 43(5):1575–1595, 1993.

[56] D.G. Luenberger. Introduction to linear and nonlinear programming. 1973.

[57] Z. Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *J. Optim. Theory Appl.*, 72(1):7–35, 1992.

[58] Z.Q. Luo and P. Tseng. Error bounds and convergence analysis of feasible descent methods: A general approach. *Annals of Operations Research*, 46(1):157–178, 1993.

[59] S. Ma, D. Goldfarb, and L. Chen. Fixed point and Bregman iterative methods for matrix rank minimization. *Mathematical Programming*, pages 1–33, 2009.

[60] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696. ACM, 2009.

[61] M. Mørup, L.K. Hansen, and S.M. Arnfred. Algorithms for sparse nonnegative Tucker decompositions. *Neural computation*, 20(8):2112–2131, 2008.

[62] C. Navasca, L. De Lathauwer, and S. Kindermann. Swamp reducing technique for tensor decomposition. In *Proc. of the 16th European Signal Processing Conference (EUSIPCO 2008)*, 2008.

[63] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[64] Y. Nesterov. Gradient methods for minimizing composite objective function. *CORE Discussion Papers*, 2007.

[65] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[66] J.M. Ortega and W.C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. Academic Press, 1970.

[67] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[68] V.P. Pauca, J. Piper, and R.J. Plemmons. Nonnegative matrix factorization for spectral data analysis. *Linear Algebra and its Applications*, 416(1):29–47, 2006.

[69] V.P. Pauca, F. Shahnaz, M.W. Berry, and R.J. Plemmons. Text mining using nonnegative matrix factorizations. In *Proc. SIAM Inter. Conf. on Data Mining, Orlando, FL*, 2004.

[70] E. Polak, R.W.H. Sargent, and D.J. Sebastian. On the convergence of sequential minimization algorithms. *Journal of Optimization Theory and Applications*, 12(6):567–575, 1973.

[71] M.J.D. Powell. On search directions for minimization algorithms. *Mathematical Programming*, 4(1):193–201, 1973.

[72] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso. *Preprint*, 2010.

[73] B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[74] B. Recht and C. Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. *Optimization Online*, 2011.

[75] P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *arXiv preprint arXiv:1107.2848*, 2011.

[76] R.T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14:877–898, 1976.

[77] R.T. Rockafellar and R.J.B. Wets. *Variational analysis*, volume 317. Springer Verlag, 1998.

[78] C. Serviere and P. Fabry. Principal component analysis and blind source separation of modulated sources for electro-mechanical systems diagnostic. *Mechanical systems and signal processing*, 19(6):1293–1311, 2005.

[79] A. Shashua and T. Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799. ACM, 2005.

[80] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[81] P. Tseng. Dual coordinate ascent methods for non-strictly convex minimization. *Mathematical Programming*, 59(1):231–247, 1993.

[82] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

[83] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.*, 109(3):475–494, 2001.

[84] Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Math. Program.*, 117(1-2, Ser. B):387–423, 2009.

[85] L.R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[86] J. Warga. Minimizing certain convex functions. *Journal of the Society for Industrial and Applied Mathematics*, 11(3):588–593, 1963.

[87] M. Welling and M. Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001.

[88] Z. Wen, W. Yin, and Y. Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, pages 1–29, 2012.

[89] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273. ACM, 2003.

[90] Y. Xu, W. Yin, Z. Wen, and Y. Zhang. An alternating direction algorithm for matrix completion with nonnegative factors. *Journal of Frontiers of Mathematics in China, Special Issue on Computational Mathematics*, 7(2):365–384, 2011.

[91] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[92] N. Zadeh. A note on the cyclic coordinate ascent method. *Management Science*, 16(9):642–644, 1970.

[93] S. Zafeiriou. Algorithms for nonnegative tensor factorization. *Tensors in Image Processing and Computer Vision*, pages 105–124, 2009.

[94] Q. Zhang, H. Wang, R.J. Plemmons, and V. Pauca. Tensor methods for hyperspectral data analysis: a space object material identification study. *JOSA A*, 25(12):3001–3012, 2008.

[95] Y. Zhang. An alternating direction algorithm for nonnegative matrix factorization. *Rice Technical Report*, 2010.

[96] M. Zibulevsky and B.A. Pearlmutter. Blind source separation by sparse decomposition in a signal dictionary. *Neural computation*, 13(4):863–882, 2001.