Nonnegativity constraints in numerical analysis*

Donghui Chen[1] and Robert J. Plemmons[2]

[1] Department of Mathematics, Wake Forest University, Winston-Salem, NC 27109.
Presently at Dept. Mathematics Tufts University.
[2] Departments of Computer Science and Mathematics, Wake Forest University,
Winston-Salem, NC 27109.
Medford, MA 02155
plemmons@wfu.edu

Abstract. A survey of the development of algorithms for enforcing nonnegativity constraints in scientific computation is given. Special emphasis is placed on such constraints in least squares computations in numerical linear algebra and in nonlinear optimization. Techniques involving nonnegative low-rank matrix and tensor factorizations are also emphasized. Details are provided for some important classical and modern applications in science and engineering. For completeness, this report also includes an effort toward a literature survey of the various algorithms and applications of nonnegativity constraints in numerical analysis.

Key Words: nonnegativity constraints, nonnegative least squares, matrix and tensor factorizations, image processing, optimization.

1 Historical comments on enforcing nonnegativity

Nonnegativity constraints on solutions, or approximate solutions, to numerical problems are pervasive throughout science, engineering and business. In order to preserve inherent characteristics of solutions corresponding to amounts and measurements, associated with, for instance frequency counts, pixel intensities and chemical concentrations, it makes sense to respect the nonnegativity so as to avoid physically absurd and unpredictable results. This viewpoint has both computational as well as philosophical underpinnings. For example, for the sake of interprentation one might prefer to determine solutions from the same space, or a subspace thereof, as that of the input data.

In numerical linear algebra, nonnegativity constraints very often arise in least squares problems, which we denote as **nonnegative least squares** (NNLS). The design and implementation of NNLS algorithms has been the subject of considerable work the seminal book of Lawson and Hanson [51]. This book seems to contain the first widely used method for solving NNLS. A variation of their algorithm is available as **lsqnonneg** in Matlab. (For a history of NNLS computations in Matlab see [84].)

* Research supported by the Air Force Office of Scientific Research under grant FA9550-08-1-0151.

More recently, beginning in the 1990s, NNLS computations have been generalized to approximate nonnegative matrix or tensor factorizations, in order to obtain low-dimensional representations of nonnegative data. A suitable representation for data is essential to applications in fields such as statistics, signal and image processing, machine learning, and data mining. (See, e.g., the survey by Berry, et al. [9]). Low rank constraints on high dimensional massive data sets are prevalent in dimensionality reduction and data analysis across numerous scientific disciplines. Techniques for dimensionality reduction and feature extraction include Principal Component Analysis (PCA), Independent Component Analysis (ICA), and **(approximate) Nonnegative Matrix Factorization** (NMF).

In this paper we are concerned primarily with NNLS as well as NMF and its extension to **Nonnegative Tensor Factorization** (NTF). A tensor can be thought of as a multi-way array, and our interest is in the natural extension of concepts involving data sets represented by 2-D arrays to 3-D arrays represented by tensors. Tensor analysis became immensely popular after Einstein used tensors as the natural language to describe laws of physics in a way that does not depend on the initial frame of reference. Recently, tensor analysis techniques have become a widely applied tool, especially in the processing of massive data sets. (See the work of Cichocki et al. [22] and Ho [39], as well as the program for the 2008 Stanford Workshop on Modern Massive Data Sets on the web page http://www.stanford.edu/group/mmds/). Together, NNLS, NMF and NTF are used in various applications which will be discussed and referenced in this survey.

2 Preliminaries

We begin this survey with a review of some notation and terminology, some useful theoretical issues associated with nonnegative matrices arising in the mathematical sciences, and the Karush-Kuhn-Tucker conditions used in optimization. All matrices discussed are over the real numbers. For $\mathbf{A} = (\mathbf{a_{ij}})$ we write $\mathbf{A} \geq \mathbf{0}$ if $\mathbf{a_{ij}} \geq \mathbf{0}$ for each i and j. We say that $\mathbf{A}$ is a **nonnegative matrix**. The notation naturally extends to vectors, and to the term positive matrix.

Aspects of the theory of nonnegative matrices, such as the classical Perron-Frobenious theory, have been included in various books. For more details the reader is referred to the books, in chronological order, by Varga [90], by Berman and Plemmons. [8], and by Bapat and Raghavan [6]. This topic leads naturally to the concepts of inverse-positivity, monotonicity and iterative methods, and M-matrix computations. For example, M-Matrices $\mathbf{A}$ have positive diagonal entries and non-positive off-diagonal entries, with the added condition that $\mathbf{A^{-1}}$ is a nonnegative matrix. Associated linear systems of equations $\mathbf{Ax} = \mathbf{b}$ thus have nonnegative solutions whenever $\mathbf{b} \geq \mathbf{0}$. Applications of M-Matrices abound in numerical analysis topics such as numerical PDEs and Markov Chain analysis, as well as in economics, operations research, and statistics, see e.g., [8, 90].

For the sake of completeness we state the classical Perron-Frobenious Theorem for irreducible nonnegative matrices. Here, an $n \times n$ matrix $\mathbf{A}$ is said to be

reducible if $n \geq 2$ and there exists a permutation matrix $\mathbf{P}$ such that

$$\mathbf{PAP^T} = \begin{bmatrix} \mathbf{B} & 0 \\ \mathbf{C} & \mathbf{D} \end{bmatrix}, \tag{1}$$

where $\mathbf{B}$ and $\mathbf{D}$ are square matrices and 0 is a zero matrix. The matrix $\mathbf{A}$ is **irreducible** if it is not reducible.

Perron-Frobenius theorem:

Let A be a $n \times n$ nonnegative irreducible matrix. Then there exists a real number $\lambda_0 > 0$ and a positive vector y such that

- $\mathbf{Ay} = \lambda_\mathbf{0}\mathbf{y}$.
- The eigenvalue λ_0 is geometrically simple. That is, any two eigenvectors corresponding to λ_0 are linearly dependent.
- The eigenvalue λ_0 is maximal in modulus among all the eigenvalues of $\mathbf{A}$. That is, for any eigenvalue μ of $\mathbf{A}$, $|\mu| \leq \lambda_0$.
- The only nonnegative, nonzero eigenvectors of $\mathbf{A}$ are just the positive scalar multiplies of y.
- The eigenvalue λ_0 is algebraically simple. That is, λ_0 is a simple root of the characteristic polynomial of $\mathbf{A}$.
- Let $\lambda_0, \lambda_1, \ldots, \lambda_{k-1}$ be the distinct eigenvalues of $\mathbf{A}$ with $|\lambda_i| = \lambda_0, i = 1, 2, \ldots, k-1$. Then they are precisely the solutions of the equation $\lambda^k - \lambda_0^k = 0$.

As a simple illustration of one application of this theorem, we mention that a finite irreducible Markov process associated with a probability matrix $\mathbf{S}$ must have a positive stationary distribution vector, which is associated with the eigenvalue 1 of $\mathbf{S}$. (See, e.g., [8].)

Another concept that will be useful in this paper is the classical Karush-Kuhn-Tucker conditions (also known as the Kuhn-Tucker or the KKT conditions). The set of conditions is a generalization of the method of Lagrange multipliers.

Karush-Kuhn-Tucker conditions:

The Karush-Kuhn-Tucker(KKT) conditions are necessary for a solution in nonlinear programming to be optimal. Consider the following nonlinear optimization problem:

Let $\mathbf{x}^*$ be a local minimum of

$$\min_{\mathbf{x}} f(\mathbf{x}) \; \textit{subject to} \begin{cases} h(\mathbf{x}) = 0 \\ g(\mathbf{x}) \leq 0 \end{cases}$$

and suppose $\mathbf{x}^*$ is a regular point for the constraints, i.e. the Jacobian of the binding constraints at that point is of full rank. Then $\exists$ λ and μ such that

$$\begin{gathered}\nabla f(\mathbf{x}^*) + \lambda^T \nabla h(\mathbf{x}^*) + \mu^T \nabla g(\mathbf{x}^*) = 0 \\ \mu^T g(\mathbf{x}^*) = 0 \\ h(\mathbf{x}^*) = 0 \\ \mu \geq 0.\end{gathered} \tag{2}$$

Next we move to the topic of least squares computations with nonnegativity constraints, NNLS. Both old and new algorithms are outlined. We will see that NNLS leads in a natural way to the topics of approximate low-rank nonnegative matrix and tensor factorizations, NMF and NTF.

3 Nonnegative least squares

3.1 Introduction

A fundamental problem in data modeling is the estimation of a parameterized model for describing the data. For example, imagine that several experimental observations that are linear functions of the underlying parameters have been made. Given a sufficiently large number of such observations, one can reliably estimate the true underlying parameters. Let the unknown model parameters be denoted by the vector $\mathbf{x} = (x_1, \cdots, x_n)^T$, the different experiments relating $\mathbf{x}$ be encoded by the measurement matrix $\mathbf{A} \in R^{m \times n}$, and the set of observed values be given by $\mathbf{b}$. The aim is to reconstruct a vector $\mathbf{x}$ that explains the observed values as well as possible. This requirement may be fulfilled by considering the linear system

$$\mathbf{Ax} = \mathbf{b},$$

where the system may be either under-determined ($m < n$) or over-determined ($m \geq n$). In the latter case, the technique of least-squares proposes to compute $\mathbf{x}$ so that the reconstruction error

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|^2 \tag{3}$$

is minimized, where $\|\cdot\|$ denotes the $\mathbf{L}_2$ norm. However, the estimation is not always that straightforward because in many real-world problems the underlying parameters represent quantities that can take on only nonnegative values, e.g., amounts of materials, chemical concentrations, pixel intensities, to name a few. In such a case, problem (3) must be modified to include nonnegativity constraints on the model parameters $\mathbf{x}$. The resulting problem is called Nonnegative Least Squares (NNLS), and is formulated as follows:

NNLS problem:

Given a matrix $\mathbf{A} \in R^{m\times n}$ *and the set of observed values given by* $\mathbf{b} \in R^m$*, find a nonnegative a vector* $\mathbf{x} \in R^n$ *to minimize the functional* $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|^2$*, i.e.*

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x}) = \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|^2, \\ &\textit{subject to } \ \mathbf{x} \geq 0. \end{aligned} \tag{4}$$

The gradient of $f(\mathbf{x})$ is $\nabla f(\mathbf{x}) = \mathbf{A}^T(\mathbf{Ax} - \mathbf{b})$ and the KKT optimality conditions for NNLS problem (4) are

$$\begin{aligned} \mathbf{x} &\geq 0 \\ \nabla f(\mathbf{x}) &\geq 0 \\ \nabla f(\mathbf{x})^T\mathbf{x} &= 0. \end{aligned} \tag{5}$$

Some of the iterative methods for solving (4) are based on the solution of the corresponding linear complementarity problem (LCP).

Linear Complementarity Problem:

Given a matrix $\mathbf{A} \in R^{m\times n}$ *and the set of observed values be given by* $\mathbf{b} \in R^m$*, find a vector* $\mathbf{x} \in R^n$ *to minimize the functional*

$$\begin{aligned} \lambda = \nabla f(\mathbf{x}) = \mathbf{A}^T\mathbf{Ax} - \mathbf{A}^T\mathbf{b} \geq 0 \\ \mathbf{x} \geq 0 \\ \lambda^T\mathbf{x} = 0. \end{aligned} \tag{6}$$

Problem (6) is essentially the set of KKT optimality conditions (5) for quadratic programming. The problem reduces to finding a nonnegative $\mathbf{x}$ which satisfies $(\mathbf{Ax} - \mathbf{b})^T\mathbf{Ax} = 0$. Handling nonnegative constraints is computationally nontrivial because we are dealing with expansive nonlinear equations. An equivalent but sometimes more tractable formulation of NNLS using the residual vector variable $\mathbf{p} = \mathbf{b} - \mathbf{Ax}$ is as follows:

$$\begin{aligned} &\min_{\mathbf{x},\mathbf{p}} \ \frac{1}{2}\mathbf{p}^T\mathbf{p} \\ &\text{s. } t. \ \mathbf{Ax} + \mathbf{p} = \mathbf{b}, \ \ \mathbf{x} \geq 0. \end{aligned} \tag{7}$$

The advantage of this formulation is that we have a simple and separable objective function with linear and nonnegtivity constraints.

The NNLS problem is fairly old. The algorithm of Lawson and Hanson [51] seems to be the first method to solve it. (This algorithm is available as the **lsqnonneg** in Matlab, see [84].) An interesting thing about NNLS is that it is solved iteratively, but as Lawson and Hanson show, the iteration always converges and terminates. There is no cutoff in iteration required. Sometimes it

might run too long, and have to be terminated, but the solution will still be "fairly good", since the solution improves smoothly with iteration. Noise, as expected, increases the number of iterations required to reach the solution.

3.2 Numerical approaches and algorithms

Over the years a variety of methods have been applied to tackle the NNLS problem. Although those algorithms can straddle more than one class, in general they can be roughly divided into active-set methods and iterative approaches. (See Table 1 for a listing of some approaches to solving the NNLS problem.)

Table 1 Some Numerical Approaches and Algorithms for NNLS

Active Set Methods	*Iterative Approaches*	*Other Methods*
lsqnonneg in Matlab	*Projected Quasi-Newton NNLS*	*Interior Point Method*
Bro and Jong's Fast NNLS	*Projected Landweber method*	*Principal Block Pivoting method*
Fast Combinatorial NNLS	*Sequential Coordinate-wise Alg.*	

3.2.1 Active set methods

Active-set methods [31] are based on the observation that only a small subset of constraints are usually active (i.e. satisfied exactly) at the solution. There are n inequality constraints in NNLS problem. The *ith* constraint is said to be *active*, if the *ith* regression coefficient will be is negative (or zero) if unconstrained, otherwise the constraint is passive. An *active set algorithm* uses the fact that if the true active set is known, the solution to the least squares problem will simply be the unconstrained least squares solution to the problem using only the variables corresponding to the passive set, setting the regression coefficients of the active set to zero. This can also be stated as: if the active set is known, the solution to the NNLS problem is obtained by treating the active constraints as equality constraints rather than inequality constraints. To find this solution, an alternating least squares algorithm is applied. An initial feasible set of regression coefficients is found. A feasible vector is a vector with no elements violating the constraints. In this case the vector containing only zeros is a feasible starting vector as it contains no negative values. In each step of the algorithm, variables are identified and removed from the active set in such a way that the least least squares fit strictly decreases. After a finite number of iterations the true active set is found and the solution is found by simple linear regression on the unconstrained subset of the variables.

The NNLS algorithm of Lawson and Hanson [51] is an *active set method*, and was the *de facto* method for solving (4) for many years. Recently, Bro and Jong [15] modified it and developed a method called Fast NNLS (FNNLS), which often speeds up the basic algorithm, especially in the presence of multiple

right-hand sides, by avoiding unnecessary re-computations. A recent variant of FNNLS, called fast combinatorial NNLS [4], appropriately rearranges calculations to achieve further speedups in the presence of multiple right hand sides. However, all of these approaches still depend on $\mathbf{A}^T\mathbf{A}$, or the normal equations in factored form, which is infeasible for ill-conditioned problems.

Lawson and Hanson's algorithm:

In their landmark text [51], Lawson and Hanson give the Standard algorithm for NNLS which is an *active set method* [31]. Mathworks [84] modified the algorithm NNLS, which ultimately was renamed to "*lsqnonneg*".

Notation: The matrix $\mathbf{A}^P$ is a matrix associated with only the variables currently in the passive set P.

Algorithm *lsqnonneg* **:**

Input: $\mathbf{A} \in \mathbf{R}^{m\times n}$, $\mathbf{b} \in \mathbf{R}^m$
Output: $\mathbf{x}^* \geq 0$ such that $\mathbf{x}^* = \arg\min \|\mathbf{Ax} - \mathbf{b}\|^2$.
Initialization: $P = \varnothing$, $R = \{1, 2, \cdots, n\}$, $\mathbf{x} = \mathbf{0}$, $\mathbf{w} = \mathbf{A}^T(\mathbf{b} - \mathbf{Ax})$
repeat

1. Proceed if $R \neq \varnothing \wedge [\max_{i\in R}(w_i) > tolerance]$
2. $j = \arg\max_{i\in R}(w_i)$
3. Include the index j in P and remove it from R
4. $\mathbf{s}^P = [(\mathbf{A}^P)^T\mathbf{A}^P]^{-1}(\mathbf{A}^P)^T\mathbf{b}$
   4.1. Proceed if $\min(\mathbf{s}^P) \leq 0$
   4.2. $\alpha = -\min_{i\in P}[x_i/(x_i - s_i)]$
   4.3. $\mathbf{x} := \mathbf{x} + \alpha(\mathbf{s} - \mathbf{x})$
   4.4. Update R and P
   4.5. $\mathbf{s}^P = [(\mathbf{A}^P)^T\mathbf{A}^P]^{-1}(\mathbf{A}^P)^T\mathbf{b}$
   4.6. $\mathbf{s}^R = \mathbf{0}$
5. $\mathbf{x} = \mathbf{s}$
6. $\mathbf{w} = \mathbf{A}^T(\mathbf{b} - \mathbf{Ax})$

It is proved by Lawson and Hanson that the iteration of the NNLS algorithm is finite. Given sufficient time, the algorithm will reach a point where the Kuhn-Tucker conditions are satisfied, and it will terminate. There is no arbitrary cutoff in iteration required; in that sense it is a direct algorithm. It is not direct in the sense that the upper limit on the possible number of iterations that the algorithm might need to reach the point of optimum solution is impossibly large. There is no good way of telling exactly how many iterations it will require in a practical sense. The solution does improve smoothly as the iteration continues. If it is terminated early, one will obtain a sub-optimal but likely still fairly good image.

However, when applied in a straightforward manner to large scale NNLS problems, this algorithm's performance is found to be unacceptably slow owing to the need to perform the equivalent of a full pseudo-inverse calculation for each observation vector. More recently, Bro and de Jong [15] have made a substantial speed improvement to Lawson and Hanson's algorithm for the case of a large number of observation vectors, by developing a modified NNLS algorithm.

Fast NNLS *fnnls* :

In the paper [15], Bro and de Jong give a modification of the standard algorithm for NNLS by Lawson and Hanson. Their algorithm, called Fast Nonnegative Least Squares, *fnnls*, is specifically designed for use in multiway decomposition methods for tensor arrays such as PARAFAC and N-mode PCA. (See the material on tensors given later in this paper.) They realized that large parts of the pseudoinverse could be computed once but used repeatedly. Specifically, their algorithm precomputes the cross-product matrices that appear in the normal equation formulation of the least squares solution. They also observed that, during alternating least squares (ALS) procedures (to be discussed later), solutions tend to change only slightly from iteration to iteration. In an extension to their NNLS algorithm that they characterized as being for "advanced users", they retained information about the previous iteration's solution and were able to extract further performance improvements in ALS applications that employ NNLS. These innovations led to a substantial performance improvement when analyzing large multivariate, multiway data sets.

Algorithm *fnnls* :

Input: $\mathbf{A} \in \mathbf{R}^{m \times n}$, $\mathbf{b} \in \mathbf{R}^m$
Output: $\mathbf{x}^* \geq 0$ such that $\mathbf{x}^* = \arg\min \|\mathbf{Ax} - \mathbf{b}\|^2$.
Initialization: $P = \varnothing, R = \{1, 2, \cdots, n\}, \mathbf{x} = \mathbf{0}, \mathbf{w} = \mathbf{A}^T\mathbf{b} - (\mathbf{A}^T\mathbf{A})\mathbf{x}$
repeat

1. Proceed if $R \neq \varnothing \wedge [\max_{i \in R}(w_i) > tolerance]$
2. $j = \arg\max_{i \in R}(w_i)$
3. Include the index j in P and remove it from R
4. $\mathbf{s}^P = [(\mathbf{A}^T\mathbf{A})^P]^{-1}(\mathbf{A}^T\mathbf{b})^P$
   4.1. Proceed if $\min(\mathbf{s}^P) \leq 0$
   4.2. $\alpha = -\min_{i \in P}[x_i/(x_i - s_i)]$
   4.3. $\mathbf{x} := \mathbf{x} + \alpha(\mathbf{s} - \mathbf{x})$
   4.4. Update R and P
   4.5. $\mathbf{s}^P = [(\mathbf{A}^T\mathbf{A})^P]^{-1}(\mathbf{A}^T\mathbf{b})^P$
   4.6. $\mathbf{s}^R = \mathbf{0}$
5. $\mathbf{x} = \mathbf{s}$
6. $\mathbf{w} = \mathbf{A}^T(\mathbf{b} - \mathbf{Ax})$

While Bro and de Jong's algorithm precomputes parts of the pseudoinverse, the algorithm still requires work to complete the pseudoinverse calculation once for each vector observation. A recent variant of *fnnls*, called fast combinatorial NNLS [4], appropriately rearranges calculations to achieve further speedups in the presence of multiple observation vectors $\mathbf{b}_i, i = 1, 2 \cdots l$. This new method rigorously solves the constrained least squares problem while exacting essentially no performance penalty as compared with Bro and Jong's algorithm. The new algorithm employs combinatorial reasoning to identify and group together all observations $\mathbf{b}_i$ that share a common pseudoinverse at each stage in the NNLS iteration. The complete pseudoinverse is then computed just once per group and, subsequently, is applied individually to each observation in the group. As a result, the computational burden is significantly reduced and the time required to perform ALS operations is likewise reduced. Essentially, if there is only one observation, this new algorithm is no different from Bro and Jong's algorithm.

In the paper [25], Dax concentrates on two problems that arise in the implementation of an active set method. One problem is the choice of a good starting point. The second problem is how to move away from a "*dead point*". The results of his experiments indicate that the use of *Gauss-Seidel* iterations to obtain a starting point is likely to provide large gains in efficiency. And also, dropping one constraint at a time is advantageous to dropping several constraints at a time.

However, all these *active set methods* still depend on the normal equations, rendering them infeasible for ill-conditioned. In contrast to an active set method, iterative methods, for instance gradient projection, enables one to incorporate multiple active constraints at each iteration.

3.2.2 Algorithms based on iterative methods

The main advantage of this class of algorithms is that by using information from a projected gradient step along with a good guess of the active set, one can handle multiple active constraints per iteration. In contrast, the active-set method typically deals with only one active constraint at each iteration. Some of the iterative methods are based on the solution of the corresponding LCP (6). In contrast to an active set approach, iterative methods like gradient projection enables the incorporation of multiple active constraints at each iteration.

Projective Quasi-Newton NNLS (PQN-NNLS)

In the paper [48], Kim, et al. proposed a projection method with non-diagonal gradient scaling to solve the NNLS problem (4). In contrast to an active set approach, gradient projection avoids the pre-computation of $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{b}$, which is required for the use of the active set method *fnnls*. It also enables their method to incorporate multiple active constraints at each iteration. By employing non-diagonal gradient scaling, **PQN-NNLS** overcomes some of the deficiencies of a projected gradient method such as slow convergence and zigzagging. An important characteristic of **PQN-NNLS** algorithm is that despite the efficiencies,

it still remains relatively simple in comparison with other optimization-oriented algorithms. Also in this paper, Kim et al. gave experiments to show that their method outperforms other standard approaches to solving the NNLS problem, especially for large-scale problems.

Algorithm PQN-NNLS:

Input: $\mathbf{A} \in \mathbf{R}^{m \times n}$, $\mathbf{b} \in \mathbf{R}^m$
Output: $\mathbf{x}^* \geq 0$ such that $\mathbf{x}^* = \arg\min \|\mathbf{Ax} - \mathbf{b}\|^2$.
Initialization: $\mathbf{x}^0 \in \mathbf{R}^n_+, \mathbf{S}^0 \leftarrow \mathbf{I}$ and $k \leftarrow 0$
repeat

1. Compute fixed variable set $\mathbf{I}^k = \{i : x_i^k = 0, [\nabla f(\mathbf{x}^k)]_i > 0\}$
2. Partition $\mathbf{x}^k = [\mathbf{y}^k; \mathbf{z}^k]$, where $y_i^k \notin \mathbf{I}^k$ and $z_i^k \in \mathbf{I}^k$
3. Solve equality-constrained subproblem:
   3.1. Find appropriate values for α^k and β^k
   3.2. $\gamma^k(\beta^k; \mathbf{y}^k) \leftarrow \mathcal{P}(\mathbf{y}^k - \beta^k \bar{\mathbf{S}}^k \nabla f(\mathbf{y}^k))$
   3.3. $\tilde{\mathbf{y}} \leftarrow \mathbf{y}^k + \alpha(\gamma^k(\beta^k; \mathbf{y}^k) - \mathbf{y}^k)$
4. Update gradient scaling matrix $\mathbf{S}^k$ to obtain $\mathbf{S}^{k+1}$
5. Update $\mathbf{x}^{k+1} \leftarrow [\tilde{\mathbf{y}}; \mathbf{z}^k]$
6. $k \leftarrow k + 1$

until Stopping criteria are met.

Sequential coordinate-wise algorithm for NNLS

In [30], the authors propose a novel sequential coordinate-wise (SCA) algorithm which is easy to implement and it is able to cope with large scale problems. They also derive stopping conditions which allow control of the distance of the solution found to the optimal one in terms of the optimized objective function. The algorithm produces a sequence of vectors $\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^t$ which converges to the optimal x^*. The idea is to optimize in each iteration with respect to a single coordinate while the remaining coordinates are fixed. The optimization with respect to a single coordinate has an analytical solution, thus it can be computed efficiently.

Notation: $\mathcal{I} = \{1, 2, \cdots, n\}$, $\mathcal{I}_k = \mathcal{I}/k$, $\mathbf{H} = \mathbf{A}^T\mathbf{A}$ which is semi-positive definite, and $\mathbf{h}_k$ denotes the kth column of $\mathbf{H}$.

Algorithm SCA-NNLS:

Input: $\mathbf{A} \in \mathbf{R}^{m \times n}$, $\mathbf{b} \in \mathbf{R}^m$
Output: $\mathbf{x}^* \geq 0$ such that $\mathbf{x}^* = \arg\min \|\mathbf{Ax} - \mathbf{b}\|^2$.
Initialization: $\mathbf{x}^0 = \mathbf{0}$ and $\mu^0 = f = -\mathbf{A}^T\mathbf{b}$
repeat For $k = 1$ to n

1. $\mathbf{x}_k^{t+1} = \max\left(0, \mathbf{x}_k^t - \frac{\mu_k^t}{\mathbf{H}_{k,k}}\right)$, and $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t$, $\forall i \in \mathcal{I}_k$
2. $\mu^{t+1} = \mu^t + (\mathbf{x}_k^{t+1} - \mathbf{x}_k^t)\mathbf{h}_k$

until Stopping criteria are met.

3.2.3 Other methods:

Principal block pivoting method

In the paper [17], the authors gave a block principal pivoting algorithm for large and sparse NNLS. They considered the linear complementarity problem (6). The n indices of the variables in $\mathbf{x}$ are divided into complementary sets F and G, and let $\mathbf{x}_F$ and $\mathbf{y}_G$ denote pairs of vectors with the indices of their nonzero entries in these sets. Then the pair $(\mathbf{x}_F, \mathbf{y}_G)$ is a *complementary basic solution* of Equation (6) if $\mathbf{x}_F$ is a solution of the unconstrained least squares problem

$$\min_{\mathbf{x}_F \in \mathbb{R}^{|F|}} \|\mathbf{A}_F\mathbf{x}_F - \mathbf{b}\|_2^2 \tag{8}$$

where $\mathbf{A}_F$ is formed from $\mathbf{A}$ by selecting the columns indexed by F, and $\mathbf{y}_G$ is obtained by

$$\mathbf{y}_G = \mathbf{A}_G^T(\mathbf{A}_F\mathbf{x}_F - \mathbf{b}). \tag{9}$$

If $\mathbf{x}_F \geq 0$ and $\mathbf{y}_G \geq 0$, then the solution is *feasible*. Otherwise it is *infeasible*, and we refer to the negative entries of $\mathbf{x}_F$ and $\mathbf{y}_G$ as *infeasible variables*. The idea of the algorithm is to proceed through infeasible complementary basic solutions of (6) to the unique feasible solution by exchanging infeasible variables between F and G and updating $\mathbf{x}_F$ and $\mathbf{y}_G$ by (8) and (9). To minimize the number of solutions of the least-squares problem in (8), it is desirable to exchange variables in large groups if possible. The performance of the algorithm is several times faster than Matstoms' Matlab implementation [59] of the same algorithm. Further, it matches the accuracy of Matlab's built-in *lsqnonneg* function. (The program is available online at http://plato.asu.edu/sub/nonlsq.html).

Block principal pivoting algorithm:

Input: $\mathbf{A} \in \mathbf{R}^{m\times n}$, $\mathbf{b} \in \mathbf{R}^m$
Output: $\mathbf{x}^* \geq 0$ such that $\mathbf{x}^* = \arg\min \|\mathbf{Ax} - \mathbf{b}\|^2$.
Initialization: $F = \varnothing$ and $G = 1, \ldots, n, x = 0, y = -\mathbf{A}^T b$, and $p = 3, N = \infty$.
repeat:

1. Proceed if (x_F, y_G) is an infeasible solution.
2. Set n to the number of negative entries in x_F and y_G.
   - 2.1 Proceed if $n < N$,
     - 2.1.1 Set $N = n, p = 3$,
     - 2.1.2 Exchange all infeasible variables between F and G.
   - 2.2 Proceed if $n \geq N$
     - 2.2.1 Proceed if $p > 0$,
       - 2.2.1.1 set $p = p - 1$
       - 2.2.1.2 Exchange all infeasible variables between F and G.
     - 2.2.2 Proceed if $p \leq 0$,
       - 2.2.2.1 Exchange only the infeasible variable with largest index.
3. Update x_F and y_G by Equation (8) and (9).
4. Set Variables in $x_F < 10^{-12}$ and $y_G < 10^{-12}$ to zero.

Interior point Newton-like method:

In addition to the methods above, Interior Point methods can be used to solve NNLS problems. They generate an infinite sequence of strictly feasible points converging to the solution and are known to be competitive with active set methods for medium and large problems. In the paper [1], the authors present an interior-point approach suited for NNLS problems. Global and locally fast convergence is guaranteed even if a degenerate solution is approached and the structure of the given problem is exploited both in the linear algebra phase and in the globalization strategy. Viable approaches for implementation are discussed and numerical results are provided. Here we give an interior algorithm for NNLS, more detailed discussion could be found in the paper [1].

Notation: $g(x)$ is the gradient of the objective function (4), i.e. $g(x) = \nabla f(x) = \mathbf{A}^T(\mathbf{Ax} - \mathbf{b})$. Therefore, by the KKT conditions, x^* can be found by searching for the positive solution of the system of nonlinear equations

$$D(x)g(x) = 0, \tag{10}$$

where $D(x) = diag(d_1(x), \ldots, d_n(x))$, has entries

$$d_i(x) = \begin{cases} x_i & if\ g_i(x) \geq 0, \\ 1 & otherwise. \end{cases} \tag{11}$$

The matrix $W(x)$ is defined by $W(x) = diag(w_1(x), \ldots, w_n(x))$, where $w_i(x) = \frac{1}{d_i(x)+e_i(x)}$ and for $1 < s \leq 2$

$$e_i(x) = \begin{cases} g_i(x) & if\ 0 \leq g_i(x) < x_i^s\ or\ g_i(x)^s > x_i, \\ 1 & otherwise. \end{cases} \tag{12}$$

Newton Like method for NNLS:

Input: $\mathbf{A} \in \mathbf{R}^{m \times n}$, $\mathbf{b} \in \mathbf{R}^m$
Output: $\mathbf{x}^* \geq 0$ such that $\mathbf{x}^* = \arg\min \|\mathbf{Ax} - \mathbf{b}\|^2$.
Initialization: $\mathbf{x}_0 > \mathbf{0}$ and $\sigma < 1$
repeat

1. Choose $\eta_k \in [0, 1)$
2. Solve $Z_k\tilde{p} = -W_k^{\frac{1}{2}} D_k^{\frac{1}{2}} g_k + \tilde{r}_k$, $\|\tilde{r}_k\|_2 \leq \eta_k \|W_k D_k g_k\|_2$
3. Set $p = W_k^{\frac{1}{2}} D_k^{\frac{1}{2}} \tilde{p}$
4. Set $p_k = max\{\sigma, 1 - \|p(x_k + p) - x_k\|_2\}(p(x_k + p) - x_k)$
5. Set $x_{k+1} = x_k + p_k$

until Stopping criteria are met.

We next move to the extension of Problem NNLS to approximate low-rank nonnegative matrix factorization and later extend that concept to approximate low-rank nonnegative tensor (multiway array) factorization.

4 Nonnegative matrix and tensor factorizations

As indicated earlier, NNLS leads in a natural way to the topics of approximate nonnegative matrix and tensor factorizations, NMF and NTF. We begin by discussing algorithms for approximating an $m \times n$ nonnegative matrix $\mathbf{X}$ by a low-rank matrix, say $\mathbf{Y}$, that is factored into $\mathbf{Y} = \mathbf{WH}$, where $\mathbf{W}$ has $k \leq \min\{m, n\}$ columns, and $\mathbf{H}$ has k rows.

4.1 Nonnegative matrix factorization

In Nonnegative Matrix Factorization (NMF), an $m \times n$ (nonnegative) mixed data matrix $\mathbf{X}$ is approximately factored into a product of two nonnegative rank-k matrices, with k small compared to m and n, $\mathbf{X} \approx \mathbf{WH}$. This factorization has the advantage that $\mathbf{W}$ and $\mathbf{H}$ can provide a physically realizable representation of the mixed data. NMF is widely used in a variety of applications, including air emission control, image and spectral data processing, text mining, chemometric

analysis, neural learning processes, sound recognition, remote sensing, and object characterization, see, e.g. [9].

NMF problem: *Given a nonnegative matrix* $\mathbf{X} \in \mathbf{R}^{m\times n}$ *and a positive integer* $k \leq \min\{m, n\}$*, find nonnegative matrices* $W \in \mathbf{R}^{m\times k}$ *and* $H \in \mathbf{R}^{k\times n}$ *to minimize the function* $f(\mathbf{W}, \mathbf{H}) = \frac{1}{2}\|\mathbf{X} - \mathbf{WH}\|_F^2$*, i.e.*

$$\min_{\mathbf{H}} f(\mathbf{H}) = \|\mathbf{X} - \sum_{i=1}^{k} \mathbf{W}^{(i)} \circ \mathbf{H}^{(i)}\| \quad \textit{subject to} \quad \mathbf{W}, \mathbf{H} \geq 0 \tag{13}$$

where $'\circ'$ *denotes outer product,* $\mathbf{W}^{(i)}$ *is ith column of* $\mathbf{W}$*,* $\mathbf{H}^{(i)}$ *is ith column of* $\mathbf{H^T}$

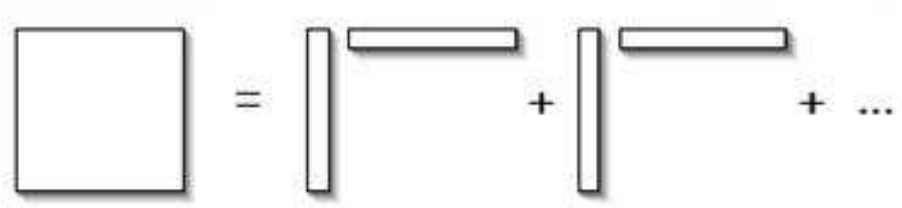

Fig. 1 An illustration of nonnegative matrix factorization.

See Figure 1 which provides an illustration of matrix approximation by a sum of rank one matrices determined by $\mathbf{W}$ and $\mathbf{H}$. The sum is truncated after k terms.

Quite a few numerical algorithms have been developed for solving the NMF. The methodologies adapted are following more or less the principles of alternating direction iterations, the projected Newton, the reduced quadratic approximation, and the descent search. Specific implementations generally can be categorized into alternating least squares algorithms [65], multiplicative update algorithms [42, 53, 54], gradient descent algorithms, and hybrid algorithms [68, 70]. Some general assessments of these methods can be found in [20, 57]. It appears that there is much room for improvement of numerical methods. Although schemes and approaches are different, any numerical method is essentially centered around satisfying the first order optimality conditions derived from the Kuhn-Tucker theory. Note that the computed factors W and H may only be local minimizers of (13).

Theorem 1. *Necessary conditions for* $(W, H) \in \mathbf{R}_+^{m\times p} \times \mathbf{R}_+^{p\times n}$ *to solve the nonnegative matrix factorization problem (13) are*

$$\begin{aligned} W. * ((X - WH)H^T) &= \mathbf{0} \in \mathbf{R}^{m\times p}, \\ H. * (W^T(X - WH)) &= \mathbf{0} \in \mathbf{R}^{p\times n}, \\ (X - WH)H^T &\leq \mathbf{0}, \\ W^T(X - WH) &\leq \mathbf{0}, \end{aligned} \tag{14}$$

where $'.*'$ *denotes the Hadamard product.*

Alternating Least Squares (ALS) algorithms for NMF

Since the Frobenius norm of a matrix is just the sum of Euclidean norms over columns (or rows), minimization or descent over either $\mathbf{W}$ or $\mathbf{H}$ boils down to solving a sequence of nonnegative least squares (NNLS) problems. In the class of ALS algorithms for NMF, a least squares step is followed by another least squares step in an alternating fashion, thus giving rise to the ALS name. ALS algorithms were first used by Paatero [65], exploiting the fact that, while the optimization problem of (13) is not convex in both $\mathbf{W}$ and $\mathbf{H}$, it is convex in either $\mathbf{W}$ or $\mathbf{H}$. Thus, given one matrix, the other matrix can be found with NNLS computations. An elementary ALS algorithm in matrix notation follows.

ALS algorithm for NMF:

Initialization: Let $\mathbf{W}$ be a random matrix $\mathbf{W} = rand(m, k)$ or use another initialization from [52]
repeat: for $i = 1 : maxiter$

1. (NNLS) Solve for $\mathbf{H}$ in the matrix equation $\mathbf{W}^T\mathbf{W}\mathbf{H} = \mathbf{W}^T\mathbf{X}$ by solving

$$\min_{\mathbf{H}} f(\mathbf{H}) = \frac{1}{2}\|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2 \quad subject\ \ to\ \ \mathbf{H} \geq 0,$$

   with $\mathbf{W}$ fixed,
2. (NNLS) Solve for $\mathbf{W}$ in the matrix equation $\mathbf{H}\mathbf{H}^T\mathbf{W}^T = \mathbf{H}\mathbf{X}^T$ by solving

$$\min_{\mathbf{W}} f(\mathbf{W}) = \frac{1}{2}\|\mathbf{X}^T - \mathbf{H}^T\mathbf{W}^T\|_F^2 \quad subject\ \ to\ \ \mathbf{W} \geq 0$$

   with $\mathbf{H}$ fixed.

end

Compared to other methods for NMF, the ALS algorithms are more flexible, allowing the iterative process to escape from a poor path. Depending on the

implementation, ALS algorithms can be very fast. The implementation shown above requires significantly less work than other NMF algorithms and slightly less work than an SVD implementation. Improvements to the basic ALS algorithm appear in [52, 66].

We conclude this section with a discussion of the convergence of ALS algorithms. Algorithms following an alternating process, approximating $\mathbf{W}$, then $\mathbf{H}$, and so on, are actually variants of a simple optimization technique that has been used for decades, and are known under various names such as alternating variables, coordinate search, or the method of local variation [63]. While statements about global convergence in the most general cases have not been proven for the method of alternating variables, a bit has been said about certain special cases. For instance, [74] proved that every limit point of a sequence of alternating variable iterates is a stationary point. Others [72, 73, 91] proved convergence for special classes of objective functions, such as convex quadratic functions. Furthermore, it is known that an ALS algorithm that properly enforces nonnegativity, for example, through the nonnegative least squares (NNLS) algorithm of Lawson and Hanson [51], will converge to a local minimum [11, 33, **?**].

4.2 Nonnegative tensor decomposition

Nonnegative Tensor Factorization (NTF) is a natural extension of NMF to higher dimensional data. In NTF, high-dimensional data, such as hyperspectral or other image cubes, is factored directly, it is approximated by a sum of rank 1 nonnegative tensors. The ubiquitous tensor approach, originally suggested by Einstein to explain laws of physics without depending on inertial frames of reference, is now becoming the focus of extensive research. Here, we develop and apply NTF algorithms for the analysis of spectral and hyperspectral image data. The algorithm given here combines features from both NMF and NTF methods.

Notation: The symbol $*$ denotes the **Hadamard** (i.e., elementwise) matrix product,

$$\mathbf{A} * \mathbf{B} = \begin{pmatrix} \mathbf{A}_{11}\mathbf{B}_{11} & \cdots & \mathbf{A}_{1n}\mathbf{B}_{1n} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{m1}\mathbf{B}_{m1} & \cdots & \mathbf{A}_{mn}\mathbf{B}_{mn} \end{pmatrix} \tag{15}$$

The symbol $\otimes$ denotes the **Kronecker** product, i.e.

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} \mathbf{A}_{11}\mathbf{B} & \cdots & \mathbf{A}_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{m1}\mathbf{B} & \cdots & \mathbf{A}_{mn}\mathbf{B} \end{pmatrix} \tag{16}$$

And the symbol $\odot$ denotes the **Khatri-Rao** product (columnwise Kronecker)[44],

$$\mathbf{A} \odot \mathbf{B} = (\mathbf{A}_1 \otimes \mathbf{B}_1 \quad \cdots \quad \mathbf{A}_n \otimes \mathbf{B}_n). \tag{17}$$

where $\mathbf{A}_i, \mathbf{B}_i$ are the columns of $\mathbf{A}, \mathbf{B}$ respectively.

The concept of matricizing or unfolding is simply a rearrangement of the entries of $\boldsymbol{\mathcal{T}}$ into a matrix. For a three-dimensional array $\boldsymbol{\mathcal{T}}$ of size $m \times n \times p$, the notation $\boldsymbol{\mathcal{T}}^{(m \times np)}$ represents a matrix of size $m \times np$ in which the n-index runs the fastest over columns and p the slowest. The norm of a tensor, $||\boldsymbol{\mathcal{T}}||$, is the same as the Frobenius norm of the matricized array, i.e., the square root of the sum of squares of all its elements.

Nonnegative Rank-k Tensor Decomposition Problem:

$$\min_{x^{(i)}, y^{(i)}, z^{(i)}} ||\boldsymbol{\mathcal{T}} - \sum_{i=1}^{r} \boldsymbol{x}^{(i)} \circ \boldsymbol{y}^{(i)} \circ \boldsymbol{z}^{(i)}||, \tag{18}$$

subject to:

$$\boldsymbol{x}^{(i)} \geq \mathbf{0}, \boldsymbol{y}^{(i)} \geq \mathbf{0}, \boldsymbol{z}^{(i)} \geq \mathbf{0}$$

where $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{m \times n \times p}, \boldsymbol{x}^{(i)} \in \mathbb{R}^m, \boldsymbol{y}^{(i)} \in \mathbb{R}^n, \boldsymbol{z}^{(i)} \in \mathbb{R}^p$.

Note that Equation (18) defines matrices $\mathbf{X}$ which is $m \times k$, $\mathbf{Y}$ which is $n \times k$, and $\mathbf{X}$ which is $p \times k$. Also, see Figure 2 which provides an illustration of 3D tensor approximation by a sum of rank one tensors. When the sum is truncated after, say, k terms, it then provides a rank k approximation to the tensor $\boldsymbol{\mathcal{T}}$.

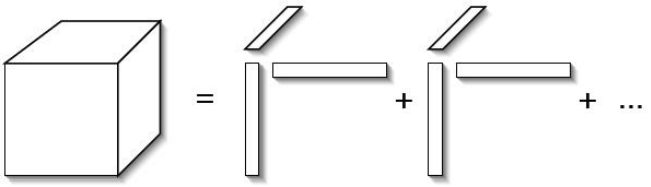

Fig. 2 An illustration of 3-D tensor factorization.

Alternating least squares for NTF

A common approach to solving Equation (18) is an alternating least squares(ALS) algorithm [29, 37, 85], due to its simplicity and ability to handle constraints. At each inner iteration, we compute an entire factor matrix while holding all the others fixed.

Starting with random initializations for $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$, we update these quantities in an alternating fashion using the method of normal equations. The minimization problem involving $\boldsymbol{X}$ in Equation (18) can be rewritten in matrix form

as a least squares problem:

$$\min_{\boldsymbol{X}} ||\boldsymbol{T}^{(m\times np)} - \boldsymbol{X}\boldsymbol{C}||^2. \tag{19}$$

where $\boldsymbol{T}^{(m\times np)} = \boldsymbol{X}(\boldsymbol{Z} \odot \boldsymbol{Y})^T, \boldsymbol{C} = (\boldsymbol{Z} \odot \boldsymbol{Y})^T$.

The least squares solution for Equation (18) involves the pseudo-inverse of $\boldsymbol{C}$, which may be computed in a special way that avoids computing $\boldsymbol{C}^T\boldsymbol{C}$ with an explicit $\boldsymbol{C}$, so the solution to Equation (18) is given by

$$\boldsymbol{X} = \boldsymbol{T}^{(m\times np)}(\boldsymbol{Z} \odot \boldsymbol{Y})(\boldsymbol{Y}^T\boldsymbol{Y} * \boldsymbol{Z}^T\boldsymbol{Z})^{-1}. \tag{20}$$

Furthermore, the product $\boldsymbol{T}^{(m\times np)}(\boldsymbol{Z} \odot \boldsymbol{Y})$ may be computed efficiently if $\boldsymbol{\mathcal{T}}$ is sparse by not forming the Khatri-Rao product $(\boldsymbol{Z} \odot \boldsymbol{Y})$. Thus, computing $\boldsymbol{X}$ essentially reduces to several matrix inner products, tensor-matrix multiplication of $\boldsymbol{Y}$ and $\boldsymbol{Z}$ into $\boldsymbol{\mathcal{T}}$, and inverting an $\mathbf{R} \times \mathbf{R}$ matrix.

Analogous least squares steps may be used to update $\boldsymbol{Y}$ and $\boldsymbol{Z}$. Following is a summary of the complete NTF algorithm.

ALS algorithm for NTF:

---

1. Group $\mathbf{x}_i$'s, $\mathbf{y}_i$'s and $\mathbf{z}_i$'s as columns in $\boldsymbol{X} \in \mathbb{R}_+^{m\times r}$, $\boldsymbol{Y} \in \mathbb{R}_+^{n\times r}$ and $\boldsymbol{Z} \in \mathbb{R}_+^{p\times r}$ respectively.
2. Initialize $\boldsymbol{X}, \boldsymbol{Y}$.
   (a) Nonnegative Matrix Factorization of the mean slice,

   $$\min ||\boldsymbol{A} - \boldsymbol{X}\boldsymbol{Y}||_F^2. \tag{21}$$

   where $\boldsymbol{A}$ is the mean of $\boldsymbol{\mathcal{T}}$ across the 3^{rd} dimension.
3. Iterative Tri-Alternating Minimization
   (a) Fix $\boldsymbol{\mathcal{T}}, \boldsymbol{X}, \boldsymbol{Y}$ and fit $\boldsymbol{Z}$ by solving a NMF problem in an alternating fashion.

   $$\boldsymbol{X}_{i\rho} \leftarrow \boldsymbol{X}_{i\rho}\frac{(\boldsymbol{T}^{(m\times np)}\boldsymbol{C})_{i\rho}}{(\boldsymbol{X}\boldsymbol{C}^T\boldsymbol{C})_{i\rho} + \epsilon}, \quad \boldsymbol{C} = (\boldsymbol{Z} \odot \boldsymbol{Y}) \tag{22}$$

   (b) Fix $\boldsymbol{\mathcal{T}}, \boldsymbol{X}, \boldsymbol{Z}$, fit for $\boldsymbol{Y}$,

   $$\boldsymbol{Y}_{j\rho} \leftarrow \boldsymbol{Y}_{j\rho}\frac{(\boldsymbol{T}^{(m\times np)}\boldsymbol{C})_{j\rho}}{(\boldsymbol{Y}\boldsymbol{C}^T\boldsymbol{C})_{j\rho} + \epsilon}, \quad \boldsymbol{C} = (\boldsymbol{Z} \odot \boldsymbol{X}) \tag{23}$$

   (c) Fix $\boldsymbol{\mathcal{T}}, \boldsymbol{Y}, \boldsymbol{Z}$, fit for $\boldsymbol{X}$.

   $$\boldsymbol{Z}_{k\rho} \leftarrow \boldsymbol{Z}_{k\rho}\frac{(\boldsymbol{T}^{(m\times np)}\boldsymbol{C})_{k\rho}}{(\boldsymbol{Z}\boldsymbol{C}^T\boldsymbol{C})_{k\rho} + \epsilon}, \quad \boldsymbol{C} = (\boldsymbol{Y} \odot \boldsymbol{X}) \tag{24}$$

Here ϵ is a small number like 10^{-9} that adds stability to the calculation and guards against introducing a negative number from numerical underflow.

If $\boldsymbol{\mathcal{T}}$ is sparse a simpler computation in the procedure above can be obtained. Each matricized version of $\boldsymbol{\mathcal{T}}$ is a sparse matrix. The matrix $\boldsymbol{C}$ from each step should not be formed explicitly because it would be a large, dense matrix. Instead, the product of a matricized $\boldsymbol{\mathcal{T}}$ with $\boldsymbol{C}$ should be computed specially, exploiting the inherent Kronecker product structure in $\boldsymbol{C}$ so that only the required elements in $\boldsymbol{C}$ need to be computed and multiplied with the nonzero elements of $\boldsymbol{\mathcal{T}}$.

5 Some applications of nonnegativity constraints

5.1 Support vector machines

Support Vector machines were introduced by Vapnik and co-workers [13, 24] theoretically motivated by Vapnik-Chervonenkis theory (also known as VC theory [88, 89]). Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. They belong to a family of generalized linear classifiers. They are based on the following idea: input points are mapped to a high dimensional feature space, where a separating hyperplane can be found. The algorithm is chosen in such a way to maximize the distance from the closest patterns, a quantity that is called the margin. This is achieved by reducing the problem to a quadratic programming problem,

$$F(v) = \frac{1}{2}\mathbf{v}^T\mathbf{A}\mathbf{v} + \mathbf{b}^T\mathbf{v}, \quad \mathbf{v} \geq 0. \tag{25}$$

Here we assume that the matrix $\mathbf{A}$ is symmetric and semipositive definite. The problem (25) is then usually solved with optimization routines from numerical libraries. SVMs have a proven impressive performance on a number of real world problems such as optical character recognition and face detection.

We briefly review the problem of computing the maximum margin hyperplane in SVMs [88]. Let $\{(x_i, y_i)\}_i^N = 1\}$ denote labeled examples with binary class labels $y_i = \pm 1$, and let $K(x_i, x_j)$ denote the kernel dot product between inputs. For brevity, we consider only the simple case where in the high dimensional feature space, the classes are linearly separable and the hyperplane is required to pass through the origin. In this case, the maximum margin hyperplane is obtained by minimizing the loss function:

$$L(\alpha) = -\sum_i \alpha_i + \frac{1}{2}\sum_{ij} \alpha_i\alpha_j y_i y_j K(x_i, x_j), \tag{26}$$

subject to the nonnegativity constraints $\alpha_i \geq 0$. Let α^* denote the minimum of equation (26). The maximal margin hyperplane has normal vector $w = \sum_i \alpha_i^* y_i x_i$ and satisfies the margin constraints $y_i K(w, x_i) \geq 1$ for all examples in the training set.

The loss function in equation (26) is a special case of the non-negative quadratic programming (25) with $A_{ij} = y_i y_j K(x_i, x_j)$ and $\mathbf{b}_i = -\mathbf{1}$. Thus, the multiplicative updates in the paper [80] are easily adapted to SVMs. This

algorithm for training SVMs is known as Multiplicative Margin Maximization (M^3). The algorithm can be generalized to data that is not linearly separable and to separating hyper-planes that do not pass through the origin.

Many iterative algorithms have been developed for nonnegative quadratic programming in general and for SVMs as a special case. Benchmarking experiments have shown that M^3 is a feasible algorithm for small to moderately sized data sets. On the other hand, it does not converge as fast as leading subset methods for large data sets. Nevertheless, the extreme simplicity and convergence guarantees of M^3 make it a useful starting point for experimenting with SVMs.

5.2 Image processing and computer vision

Digital images are represented nonnegative matrix arrays, since pixel intensity values are nonnegative. It is sometimes desirable to process data sets of images represented by column vectors as composite objects in many articulations and poses, and sometimes as separated parts for in, for example, biometric identification applications such as face or iris recognition. It is suggested that the factorization in the linear model would enable the identification and classification of intrinsic "parts" that make up the object being imaged by multiple observations [19, 43, 53, 55]. More specifically, each column x_j of a nonnegative matrix $\mathbf{X}$ now represents m pixel values of one image. The columns w_i of $\mathbf{W}$ are basis elements in R^m. The columns of $\mathbf{H}$, belonging to R^k, can be thought of as coefficient sequences representing the n images in the basis elements. In other words, the relationship

$$\mathbf{x}_j = \sum_{i=1}^{k} w_i h_{ij}, \tag{27}$$

can be thought of as that there are standard parts $\mathbf{w}_i$ in a variety of positions and that each image represented as a vector $\mathbf{x}_j$ making up the factor $\mathbf{W}$ of basis elements is made by superposing these parts together in specific ways by a mixing matrix represented by $\mathbf{H}$. Those parts, being images themselves, are necessarily nonnegative. The superposition coefficients, each part being present or absent, are also necessarily nonnegative. A related application to the identification of object materials from spectral reflectance data at different optical wavelengths has been investigated in [69–71].

As one of the most successful applications of image analysis and understanding, face recognition has recently received significant attention, especially during the past few years. Recently, many papers, like [9, 43, 53, 56, 65] have proved that Nonnegative Matrix Factorization (NMF) is a good method to obtain a representation of data using non-negativity constraints. These constraints lead to a part-based representation because they allow only additive, not subtractive, combinations of the original data. Given an initial database expressed by a $n \times m$ matrix $\mathbf{X}$, where each column is an n-dimensional nonnegative vector of the original database (m vectors), it is possible to find two new matrices $\mathbf{W}$ and $\mathbf{H}$ in

order to approximate the original matrix

$$X_{i\mu} \approx (WH)_{i\mu} = \sum_{a=1}^{k} W_{ia} H_{a\mu}. \tag{28}$$

The dimensions of the factorized matrices $\mathbf{W}$ and $\mathbf{H}$ are $n \times k$ and $k \times m$, respectively. Usually, k is chosen so that $(n+m)k < nm$. Each column of matrix $\mathbf{W}$ contains a basis vector while each column of $\mathbf{H}$ contains the weights needed to approximate the corresponding column in $\mathbf{X}$ using the bases from $\mathbf{W}$.

Other image processing work that uses non-negativity constraint includes the work image restorations. Image restoration is the process of approximating an original image from an observed blurred and noisy image. In image restoration, image formation is modeled as a first kind integral equation which, after discretization, results in a large scale linear system of the form

$$\mathbf{Ax} + \eta = \mathbf{b}. \tag{29}$$

The vector $\mathbf{x}$ represents the true image, $\mathbf{b}$ is the blurred noisy copy of $\mathbf{x}$, and η models additive noise, matrix $\mathbf{A}$ is a large ill-conditioned matrix representing the blurring phenomena.

In the absence of information of noise, we can model the image restoration problem as NNLS problem,

$$\begin{aligned} &\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|^2, \\ &subject \;\; to \;\; \mathbf{x} \geq 0. \end{aligned} \tag{30}$$

Thus, we can use NNLS to solve this problem. Experiments show that enforcing a nonnegativity constraint can produce a much more accurate approximate solution, see e.g., [36, 45, 61, 78].

5.3 Text mining

Assume that the textual documents are collected in an *matrix* $\mathbf{Y} = [y_{ij}] \in R^{m \times n}$. Each document is represented by one column in $\mathbf{Y}$. The entry y_{ij} represents the *weight* of one particular *term* i in document j whereas each term could be defined by just one single word or a string of phrases. To enhance discrimination between various documents and to improve retrieval effectiveness, a term-weighting scheme of the form,

$$y_{ij} = t_{ij} g_i d_j, \tag{31}$$

is usually used to define $\mathbf{Y}$ [10], where t_{ij} captures the relative importance of term i in document j, g_i weights the overall importance of term i in the entire set of documents, and $d_j = (\sum_{i=1}^{m} t_{ij} g_i)^{-1/2}$ is the scaling factor for normalization. The normalization by d_j per document is necessary, otherwise one could artificially inflate the prominence of document j by padding it with repeated pages or volumes. After the normalization, the columns of $\mathbf{Y}$ are of unit length and usually nonnegative.

The indexing matrix contains lot of information for retrieval. In the context of latent semantic indexing (LSI) application [10, 38], for example, suppose a query represented by a row vector $\mathbf{q}^T = [q_1, ..., q_m] \in R^m$, where q_i denotes the weight of term i in the query $\mathbf{q}$, is submitted. One way to measure how the query $\mathbf{q}$ matches the documents is to calculate the row vector $\mathbf{s}^T = \mathbf{q}^T\mathbf{Y}$ and rank the relevance of documents to $\mathbf{q}$ according to the *scores* in $\mathbf{s}$.

The computation in the LSI application seems to be merely the vector-matrix multiplication. This is so only if $\mathbf{Y}$ is a "reasonable" representation of the relationship between documents and terms. In practice, however, the matrix $\mathbf{Y}$ is never exact. A major challenge in the field has been to represent the indexing matrix and the queries in a more compact form so as to facilitate the computation of the scores [26, 67]. The idea of representing $\mathbf{Y}$ by its nonnegative matrix factorization approximation seems plausible. In this context, the standard parts w_i indicated in (27) may be interpreted as subcollections of some "general concepts" contained in these documents. Like images, each document can be thought of as a linear composition of these general concepts. The column-normalized matrix $\mathbf{A}$ itself is a term-concept indexing matrix.

5.4 Environmetrics and chemometrics

In the air pollution research community, one observational technique makes use of the ambient data and source profile data to apportion sources or source categories [41, 46, 76]. The fundamental principle in this model is that mass conservation can be assumed and a mass balance analysis can be used to identify and apportion sources of airborne particulate matter in the atmosphere. For example, it might be desirable to determine a large number of chemical constituents such as elemental concentrations in a number of samples. The relationships between p sources which contribute m chemical species to n samples leads to a mass balance equation

$$y_{ij} = \sum_{k=1}^{p} a_{ik} f_{kj}, \tag{32}$$

where y_{ij} is the elemental concentration of the ith chemical measured in the jth sample, a_{ik} is the gravimetric concentration of the ith chemical in the kth source, and f_{kj} is the airborne mass concentration that the kth source has contributed to the jth sample. In a typical scenario, only values of y_{ij} are observable whereas neither the sources are known nor the compositions of the local particulate emissions are measured. Thus, a critical question is to estimate the number p, the compositions a_{ik}, and the contributions f_{kj} of the sources. Tools that have been employed to analyze the linear model include principal component analysis, factor analysis, cluster analysis, and other multivariate statistical techniques. In this receptor model, however, there is a physical constraint imposed upon the data. That is, the source compositions a_{ik} and the source contributions f_{kj} must all be nonnegative. The identification and apportionment problems thus become a nonnegative matrix factorization problem for the matrix $\mathbf{Y}$.

5.5 Speech recognition

Stochastic language modeling plays a central role in large vocabulary speech recognition, where it is usually implemented using the n-gram paradigm. In a typical application, the purpose of an n-gram language model may be to constrain the acoustic analysis, guide the search through various (partial) text hypotheses, and/or contribute to the determination of the final transcription.

In language modeling one has to model the probability of occurrence of a predicted word given its history $Pr(w_n|\mathbf{H})$. N-gram based Language Models have been used successfully in Large Vocabulary Automatic Speech Recognition Systems. In this model, the word history consists of the $N-1$ immediately preceding words. Particularly, tri-gram language models ($Pr(w_n|w_{n-1}; w_{n-2})$) offer a good compromise between modeling power and complexity. A major weakness of these models is the inability to model word dependencies beyond the span of the n-grams. As such, n-gram models have limited semantic modeling ability. Alternate models have been proposed with the aim of incorporating long term dependencies into the modeling process. Methods such as word trigger models, high-order n-grams, cache models, etc., have been used in combination with standard n-gram models.

One such method, a *Latent Semantic Analysis* based model has been proposed [2]. A word-document occurrence matrix $\mathbf{X}_{m\times n}$ is formed (m = size of the vocabulary, n = number of documents), using a training corpus explicitly segmented into a collection of documents. A Singular Value Decomposition $\mathbf{X} = \mathbf{USV}^T$ is performed to obtain a low dimensional linear space $\mathcal{S}$, which is more convenient to perform tasks such as word and document clustering, using an appropriate metric. Bellegarda [2] gave the detailing explanation about this method.

In the paper [64], Novak and Mammone introduce a new method with NMF. In addition to the non-negativity, another property of this factorization is that the columns of $\mathbf{W}$ tend to represent groups of associated words. This property suggests that the columns of $\mathbf{W}$ can be interpreted as conditional word probability distributions, since they satisfy the conditions of a probability distribution by the definition. Thus the matrix $\mathbf{W}$ describes a hidden document space $\mathcal{D} = \{d_j\}$ by providing conditional distributions $\mathbf{W} = \mathbf{P}(w_i|d_j)$. The task is to find a matrix $\mathbf{W}$, given the word document count matrix $\mathbf{X}$. The second term of the factorization, matrix $\mathbf{H}$, reflects the properties of the explicit segmentation of the training corpus into individual documents. This information is not of interest in the context of Language Modeling. They provide an experimental result where the NMF method results in a perplexity reduction of 16% on a database of biology lecture transcriptions.

5.6 Spectral unmixing by NMF and NTF

Here we discuss applications of NMF and NTF to numerical methods for the classification of remotely sensed objects. We consider the identification of space satellites from non-imaging data such as spectra of visible and NIR range, with

different spectral resolutions and in the presence of noise and atmospheric turbulence (See, e.g., [69] or [70, 71].). This is the research area of space object identification (SOI).

A primary goal of using remote sensing image data is to identify materials present in the object or scene being imaged and quantify their abundance estimation, i.e., to determine concentrations of different signature spectra present in pixels. Also, due to the large quantity of data usually encountered in hyperspectral datasets, compressing the data is becoming increasingly important. In this section we discuss the use of MNF and NTF to reach these major goals: material identification, material abundance estimation, and data compression.

For safety and other considerations in space, non-resolved space object characterization is an important component of Space Situational Awareness. The key problem in non-resolved space object characterization is to use spectral reflectance data to gain knowledge regarding the physical properties (e.g., function, size, type, status change) of space objects that cannot be spatially resolved with telescope technology. Such objects may include geosynchronous satellites, rocket bodies, platforms, space debris, or nano-satellites. rendition of a JSAT type satellite in a 36,000 kilometer high synchronous orbit around the Earth. Even with adaptive optics capabilities, this object is generally not resolvable using ground-based telescope technology.

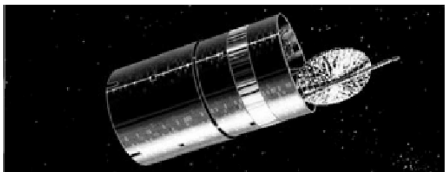

Fig. 3 Artist rendition of a JSAT satellite. Image obtained from the Boeing Satellite Development Center.

Spectral reflectance data of a space object can be gathered using ground-based spectrometers and contains essential information regarding the make up or types of materials comprising the object. Different materials such as aluminum, mylar, paint, etc. possess characteristic wavelength-dependent absorption features, or spectral *signatures*, that mix together in the spectral reflectance measurement of an object. Figure 4 shows spectral signatures of four materials typically used in satellites, namely, aluminum, mylar, white paint, and solar cell.

The objective is then, given a set of spectral measurements or traces of an object, to determine i) the type of constituent materials and ii) the proportional amount in which these materials appear. The first problem involves the detection of material spectral signatures or *endmembers* from the spectral data. The second problem involves the computation of corresponding proportional amounts or

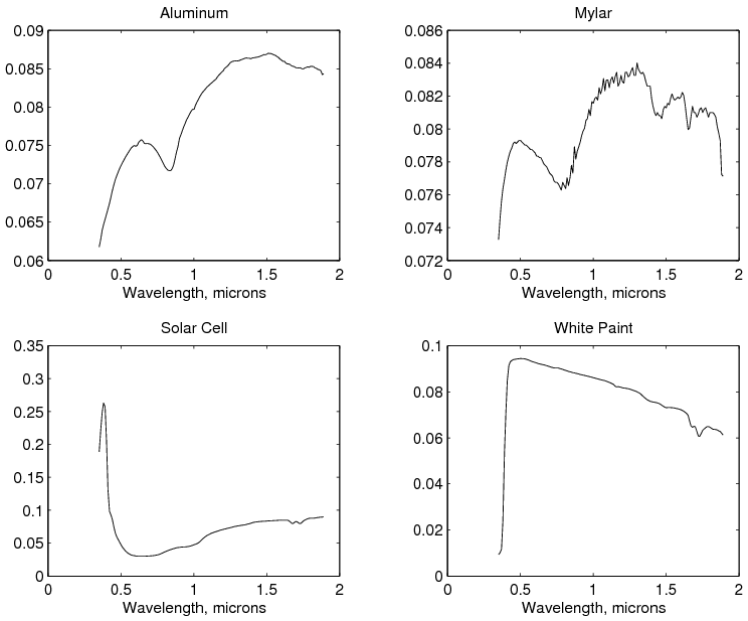

Fig. 4 Laboratory spectral signatures for aluminum, mylar, solar cell, and white paint. For details see [71].

fractional abundances. This is known as the *spectral unmixing* problem in the hyperspectral imaging community.

Recall that in In Nonnegative Matrix Factorization (NMF), an $m \times n$ (nonnegative) mixed data matrix $\mathbf{X}$ is approximately factored into a product of two nonnegative rank-k matrices, with k small compared to m and n, $\mathbf{X} \approx \mathbf{WH}$. This factorization has the advantage that $\mathbf{W}$ and $\mathbf{H}$ can provide a physically realizable representation of the mixed data, see e.g. [69]. Two sets of factors, one as endmembers and the other as fractional abundances, are optimally fitted simultaneously. And due to reduced sizes of factors, data compression, spectral signature identification of constituent materials, and determination of their corresponding fractional abundances, can be fulfilled at the same time.

Spectral reflectance data of a space object can be gathered using ground-based spectrometers, such as the SPICA system located on the 1.6 meter Gemini telescope and the ASIS system located on the 3.67 meter telescope at the Maui Space Surveillance Complex (MSSC), and contains essential information regarding the make up or types of materials comprising the object. Different materials, such as aluminum, mylar, paint, plastics and solar cell, possess characteristic wavelength-dependent absorption features, or spectral *signatures*, that mix together in the spectral reflectance measurement of an object. A new spectral imaging sensor, capable of collecting hyperspectral images of space objects, has

been installed on the 3.67 meter Advanced Electrocal-optical System (AEOS) at the MSSC. The AEOS Spectral Imaging Sensor (ASIS) is used to collect adaptive optics compensated spectral images of astronomical objects and satellites. See Figure 4 for a simulated hyperspectral image of the Hubble Space Telescope similar to that collected by ASIS.

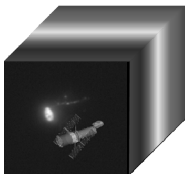

Fig. 5 A blurred and noisy simulated hyperspectral image above the original simulated image of the Hubble Space Telescope representative of the data collected by the Maui ASIS system.

In [92] and [93] Zhang, et al. develop NTF methods for identifying space objects using hyperspectral data. Illustrations of material identification, material abundance estimation, and data compression are demonstrated for data similar to that shown in Figure 5.

6 Summary

We have outlined some of what we consider the more important and interesting problems for enforcing nonnegativity constraints in numerical analysis. Special emphasis has been placed nonnegativity constraints in least squares computations in numerical linear algebra and in nonlinear optimization. Techniques involving nonnegative low-rank matrix and tensor factorizations and their many applications were also given. This report also includes an effort toward a literature survey of the various algorithms and applications of nonnegativity constraints in numerical analysis. As always, such an overview is certainly incomplete, and we apologize for omissions. Hopefully, this work will inform the reader about the importance of nonnegativity constraints in many problems in numerical analysis, while pointing toward the many advantages of enforcing nonnegativity in practical applications.

References

1. S. Bellavia, M. Macconi, and B. Morini, *An interior point newton-like method for nonnegative least squares problems with degenerate solution*, Numerical Linear Algebra with Applications, 13, pp. 825–846, 2006.
2. J. R. Bellegarda, *A multispan language modelling framework for large vocabulary speech recognition*, IEEE Transactions on Speech and Audio Processing, September 1998, Vol. 6, No. 5, pp. 456–467.

3. P. Belhumeur, J. Hespanha, and D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, IEEE PAMI, Vol. 19, No. 7, 1997.
4. M. H. van Benthem, M. R. Keenan, *Fast algorithm for the solution of large-scale non-negativity constrained least squares problems*, Journal of Chemometrics, Vol. 18, pp. 441–450, 2004.
5. M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski, *Face recognition by independent component analysis*, IEEE Trans. Neural Networks, Vol. 13, No. 6, pp. 1450–1464, 2002.
6. R. B. Bapat, T. E. S. Raghavan, *Nonnegative Matrices and Applications*, Cambridge University Press, UK, 1997.
7. A. Berman and R. Plemmons, *Rank factorizations of nonnegative matrices, Problems and Solutions*, 73-14 (Problem), SIAM Rev., Vol. 15:655, 1973.
8. A. Berman, R. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, Academic Press, NY, 1979. Revised version in SIAM Classics in Applied Mathematics, Philadelphia, 1994.
9. M. Berry, M. Browne, A. Langville, P. Pauca, and R. Plemmons, *Algorithms and applications for approximate nonnegative matrix factorization*, Computational Statistics and Data Analysis, Vol. 52, pp. 155-173, 2007. Preprint available at `http://www.wfu.edu/~plemmons`
10. M. W. Berry, *Computational Information Retrieval*, SIAM, Philadelphia, 2000.
11. D. Bertsekas, *Nonlinear Programming*, Athena Scientific, Belmont, MA., 1999.
12. M. Bierlaire, Ph. L. Toint, and D. Tuyttens, *On iterative algorithms for linear least squares problems with bound constraints*, Linear Algebra and its Applications, Vol. 143, pp. 111–143, 1991.
13. B. Boser, I. Guyon, and V. Vapnik, *A training algorithm for optimal margin classifiers*, Fifth Annual Workshop on Computational Learning Theory, ACM Press, 1992.
14. D. S. Briggs, *High fidelity deconvolution of moderately resolved radio sources*, Ph.D. thesis, New Mexico Inst. of Mining & Technology, 1995.
15. R. Bro, S. D. Jong, *A fast non-negativity-constrained least squares algorithm*, Journal of Chemometrics, Vol. 11, No. 5, pp. 393–401, 1997.
16. M. Catral, L. Han, M. Neumann, and R. Plemmons, *On reduced rank nonnegative matrix factorization for symmetric nonnegative matrices*, Lin. Alg. Appl., 393:107–126, 2004.
17. J. Cantarella, M. Piatek, *Tsnnls: A solver for large sparse least squares problems with non-negative variables*, ArXiv Computer Science e-prints, 2004.
18. R. Chellappa, C. Wilson, and S. Sirohey, *Human and Machine Recognition of Faces: A Survey*, Proc, *IEEE*, Vol. 83, No. 5, pp. 705–740, 1995.
19. X. Chen, L. Gu, S. Z. Li, and H. J. Zhang, *Learning representative local features for face detection*, *IEEE* Conference on Computer Vision and Pattern Recognition, Vol.1, pp. 1126–1131, 2001.
20. M. T. Chu, F. Diele, R. Plemmons, S. Ragni, *Optimality, computation, and interpretation of nonnegative matrix factorizations*, preprint. Available at: `http://www.wfu.edu/~plemmons`
21. M. Chu and R.J. Plemmons, *Nonnegative matrix factorization and applications*, Appeared in IMAGE, Bulletin of the International Linear Algebra Society, Vol. 34, pp. 2-7, July 2005. Available at: `http://www.wfu.edu/~plemmons`
22. A. Cichocki, R. Zdunek, and S. Amari, *Hierarchical ALS Algorithms for Nonnegative Matrix and 3D Tensor Factorization*, In: Independent Component Analysis, ICA07, London, UK, September 9-12, 2007, Lecture Notes in Computer Science, Vol. LNCS 4666, Springer, pp. 169-176, 2007.

23. I. B. Ciocoiu, H. N. Costin, *Localized versus locality-preserving subspace projections for face recognition*, EURASIP Journal on Image and Video Processing Volume 2007, Article ID 17173.
24. C. Cortes, V. Vapnik, *Support Vector networks*, Machine Learning, Vol. 20, pp. 273 - 297, 1995.
25. A. Dax, *On computational aspects of bounded linear least squares problems*, ACM Trans. Math. Softw. Vol. 17, pp. 64–73, 1991.
26. I. S. Dhillon, D. M. Modha, *Concept decompositions for large sparse text data using clustering*, Machine Learning J., Vol. 42, pp. 143–175, 2001.
27. C. Ding and X. He, and H. Simon, *On the equivalence of nonnegative matrix factorization and spectral clustering*, Proceedings of the Fifth SIAM International Conference on Data Mining, Newport Beach, CA, 2005.
28. B. A. Draper, K. Baek, M. S. Bartlett, and J. R. Beveridge, *Recognizing faces with PCA and ICA*, Computer Vision and Image Understanding, Vol. 91, No. 1, pp. 115–137, 2003.
29. N. K. M. Faber, R. Bro, and P. K. Hopke, *Recent developments in CANDECOMP/PARAFAC algorithms: a critical review*, Chemometr. Intell. Lab., Vol. 65, No. 1, pp. 119–137, 2003.
30. V. Franc, V. Hlavač, and M. Navara, *Sequential coordinate-wise algorithm for nonnegative least squares problem*, Research report CTU-CMP-2005-06, Center for Machine Perception, Czech Technical University, Prague, Czech Republic, February 2005.
31. P. E. Gill, W. Murray and M. H. Wright, *Practical Optimization*, Academic, London, 1981.
32. A. A. Giordano, F. M. Hsu, *Least Square Estimation With Applications To Digital Signal Processing*, John Wiley & Sons, 1985.
33. L. Grippo, M. Sciandrone, *On the convergence of the block nonlinear Gauss-Seidel method under convex constraints*, Oper. Res. Lett. Vol. 26, N0. 3, pp. 127–136, 2000.
34. D. Guillamet, J. Vitrià, *Classifying faces with non-negative matrix factorization*, Accepted CCIA 2002, Castelló de la Plana, Spain.
35. D. Guillamet, J. Vitrià, *Non-negative matrix factorization for face recognition*, Lecture Notes in Computer Science. Vol. 2504, 2002, pp. 336–344.
36. M. Hanke, J. G. Nagy and C. R. Vogel, *Quasi-newton approach to nonnegative image restorations*, Linear Algebra Appl., Vol. 316, pp. 223–236, 2000.
37. R. A. Harshman, *Foundations of the PARAFAC procedure: models and conditions for an "explantory" multi-modal factor analysis*, UCLA working papers in phonetics, Vol. 16, pp. 1–84, 1970.
38. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer-Verlag, New York, 2001.
39. N.-D. Ho, *Nonnegative Matrix Factorization Algorithms and Applications*, PhD thesis, Univ. Catholique de Louvain, June 2008. (Available from *edoc.bib.ucl.ac.be:81/ETD-db/collection/available/BelnUcetd-06052008-235205/*).
40. P. K. Hopke, *Receptor Modeling in Environmental Chemistry*, Wiley and Sons, New York, 1985.
41. P. K. Hopke, *Receptor Modeling for Air Quality Management*, Elsevier, Amsterdam, Netherlands, 1991.
42. P. O. Hoyer, *Nonnegative sparse coding, neural networks for signal processing XII*, Proc. IEEE Workshop on Neural Networks for Signal Processing, Martigny, 2002.

43. P. Hoyer, *Nonnegative matrix factorization with sparseness constraints*, J. of Mach. Leanring Res., vol.5, pp.1457–1469, 2004.
44. C. G. Khatri, C. R. Rao, *Solutions to some functional equations and their applications to. characterization of probability distributions*, Sankhya, Vol. 30, pp. 167–180, 1968.
45. B. Kim, *Numerical optimization methods for image restoration*, Ph.D. thesis, Stanford University, 2002.
46. E. Kim, P. K. Hopke, E. S. Edgerton, *Source identification of Atlanta aerosol by positive matrix factorization*, J. Air Waste Manage. Assoc., Vol. 53, pp. 731–739, 2003.
47. H. Kim, H. Park, *Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis*, Bioinformatics, Vol. 23, No. 12, pp. 1495–1502, 2007.
48. D. Kim, S. Sra, and I. S. Dhillon, *A new projected quasi-newton approach for the nonnegative least squares problem*, Dept. of Computer Sciences, The Univ. of Texas at Austin, Technical Report # TR-06-54, Dec. 2006.
49. D. Kim, S. Sra, and I. S. Dhillon, *Fast newton-type methods for the least squares nonnegative matrix approximation problem*, Statistical Analysis and Data Mining, Vol. 1, No. 1, pp. 38-51, (2008).
50. S. Kullback, and R. Leibler, *On information and sufficiency*, Annals of Mathematical Statistics Vol. 22 pp. 79–86, 1951.
51. C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, 1987.
52. A. Langville, C. Meyer, R. Albright, J. Cox, D. Duling, *Algorithms, initializations, and convergence for the nonnegative matrix factorization*, NCSU Technical Report Math 81706, 2006.
53. D. Lee and H. Seung, *Learning the parts of objects by non-negative matrix factorization*, Nature Vol. 401, pp. 788–791, 1999.
54. D. Lee and H. Seung, *Algorithms for nonnegative matrix factorization*, Advances in Neural Information Processing Systems, Vol. 13, pp. 556–562, 2001.
55. S. Z. Li, X. W. Hou and H. J. Zhang, *Learning spatially localized, parts-based representation*, *IEEE* Conference on Computer Vision and Pattern Recognition, pp. 1–6, 2001.
56. C. J. Lin, *Projected gradient methods for non-negative matrix factorization*, Neural Computation, Vol. 19, No. 10, pp. 2756-2779, (2007).
57. W. Liu, J. Yi, *Existing and new algorithms for nonnegative matrix factorization*, University of Texas at Austin, 2003, report.
58. A. Mazer, M. Martin, M. Lee and J. Solomon, *Image processing software for imaging spectrometry data analysis*, Remote Sensing of Environment, Vol. 24, pp. 201–220, 1988.
59. P. Matstoms, *snnls: a matlab toolbox for Solve sparse linear least squarest problem with nonnegativity constraints by an active set method*, 2004, available at `http://www.math.liu.se/~milun/sls/`.
60. J. J. Moré, G. Toraldo, *On the solution of large quadratic programming problems with bound constraints*, SIAM Journal on Optimization, Vol. 1, No. 1, pp. 93–113, 1991.
61. J. G. Nagy, Z. Strakoš, *Enforcing nonnegativity in image reconstruction algorithms*, in Mathematical Modeling, Estimation, and Imaging, 4121, David C. Wilson, et al, eds., pp. 182–190, 2000.
62. P. Niyogi, C. Burges, P. Ramesh, *Distinctive feature detection using support vector machines*, In Proceedings of ICASSP-99, pages 425-428, 1999.

63. J. Nocedal, S. Wright, *Numerical Optimization*, Springer, Berlin, 2006.
64. M. Novak, R. Mammone, *Use of non-negative matrix factorization for language model adaptation in a lecture transcription task*, *IEEE* Workshop on ASRU 2001, pp. 190–193, 2001.
65. P. Paatero and U. Tapper, *Positive matrix factorization – a nonnegative factor model with optimal utilization of error-estimates of data value*, Environmetrics, Vol. 5, pp. 111–126, 1994.
66. P. Paatero, *The multilinear engine – a table driven least squares program for solving mutilinear problems, including the n-way parallel factor analysis model*, J. Comput. Graphical Statist. Vol. 8, No. 4, pp.854–888, 1999.
67. H. Park, M. Jeon, J. B. Rosen, *Lower dimensional representation of text data in vector space based information retrieval*, in Computational Information Retrieval, ed. M. Berry, Proc. Comput. Inform. Retrieval Conf., SIAM, pp. 3–23, 2001.
68. V. P. Pauca, F. Shahnaz, M. W. Berry, R. J. Plemmons, *Text mining using nonnegative matrix factorizations*, In Proc. SIAM Inter. Conf. on Data Mining, Orlando, FL, April 2004.
69. P. Pauca, J. Piper, and R. Plemmons, *Nonnegative matrix factorization for spectral data analysis*, Lin. Alg. Applic., Vol. 416, Issue 1, pp. 29–47, 2006.
70. P. Pauca, J. Piper R. Plemmons, M. Giffin, *Object characterization from spectral data using nonnegative factorization and information theory*, Proc. AMOS Technical Conf., Maui HI, September 2004. Available at `http://www.wfu.edu/~plemmons`
71. P. Pauca, R. Plemmons, M. Giffin and K. Hamada, *Unmixing spectral data using nonnegative matrix factorization*, Proc. AMOS Technical Conference, Maui, HI, September 2004. Available at `http://www.wfu.edu/~plemmons`
72. M. Powell, *An efficient method for finding the minimum of a function of several variables without calculating derivatives*, Comput. J. Vol. 7, pp. 155-162, 1964.
73. M. Powell, *On Search Directions For Minimization*, Math. Programming Vol. 4, pp. 193-201, 1973.
74. E. Polak, *Computational Methos in Optimization: A Unified Approach*, Academic Press, New York, 1971.
75. L. F. Portugal, J. J. Judice, and L. N. Vicente, *A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables*, Mathematics of Computation, Vol. 63, No. 208, pp. 625-643, 1994.
76. Z. Ramadan, B. Eickhout, X. Song, L. M. C. Buydens, P. K. Hopke *Comparison of positive matrix factorization and multilinear engine for the source apportionment of particulate pollutants*, Chemometrics and Intelligent Laboratory Systems 66, pp. 15-28, 2003.
77. R. Ramath, W. Snyder, and H. Qi, *Eigenviews for object recognition in multispectral imaging systems*, 32nd Applied Imagery Pattern Recognition Workshop, Washington D.C., pp. 33–38, 2003.
78. M. Rojas, T. Steihaug, *Large-Scale optimization techniques for nonnegative image restorations*, Proceedings of SPIE, 4791: 233-242, 2002.
79. K. Schittkowski, *The numerical solution of constrained linear least-squares problems*, IMA Journal of Numerical Analysis, Vol. 3, pp. 11–36, 1983.
80. F. Sha, L. K. Saul, D. D. Lee, *Multiplicative updates for large margin classifiers*, Technical Report MS-CIS-03-12, Department of Computer and Information Science, University of Pennsylvania, 2003.
81. A. Shashua and T. Hazan, *Non-negative tensor factorization with applications to statistics and computer vision*, Proceedings of the 22^{nd} International Conference on Machine Learning, Bonn, Germany, pp. 792–799, 2005.

82. A. Shashua and A. Levin, *Linear image coding for regression and classification using the tensor-rank principal*, Proceedings of the *IEEE* Conference on Computer Vision and Pattern Recognition, 2001.
83. N. Smith, M. Gales, *Speech recognition using SVMs*, in Advances in Neural and Information Processing Systems, Vol. 14, Cambridge, MA, 2002, MIT Press.
84. L. Shure, *Brief History of Nonnegative Least Squares in MATLAB*, Blog available at: http://blogs.mathworks.com/loren/2006/ .
85. G. Tomasi, R. Bro, *PARAFAC and missing values*, Chemometr. Intell. Lab., Vol. 75, No. 2, pp. 163–180, 2005.
86. M. A. Turk, A. P. Pentland, *Eigenfaces for recognition*, Cognitive Neuroscience, Vol. 3, No. 1, pp.71-86, 1991.
87. M. A. Turk, A. P. Pentland, *Face recognition using eigenfaces*, Proc. *IEEE* Conference on Computer Vision and Pattern Recognition, Maui, Hawaii, 1991.
88. V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.
89. V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, 1999.
90. R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.
91. W. Zangwill, *Minimizing a function without calculating derivatives*. Comput. J. Vol. 10, pp. 293–296, 1967.
92. P. Zhang, H. Wang, R. Plemmons, and P. Pauca, *Spectral unmixing using nonnegative tensor factorization*, Proc. ACM, Conference, Winston-Salem, NC, March 2007.
93. P. Zhang, H. Wang, R. Plemmons, and P. Pauca, *Hyperspectral Data Analysis: A Space Object Material Identification Study*, Journal of the Optical Soc. Amer., Series A, Vol. 25, pp. 3001-3012, Dec. (2008).
94. W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, *Face recognition: a literature survey*, ACM Computing Surveys, Vol. 35, No. 4, pp. 399–458, 2003.