Linear-quadratic Blind Source Separation Using NMF to Unmix Urban Hyperspectral Images

Inès Meganem[†], Yannick Deville[†], *IEEE member*, Shahram Hosseini[†], Philippe Déliot[*], and Xavier Briottet[*]

***Abstract*—In this work, we propose algorithms to perform Blind Source Separation (BSS) for the linear-quadratic mixing model. The linear-quadratic model is less studied in the literature than the linear one. In this paper, we propose original methods that are based on Non-negative Matrix Factorization (NMF). This class of methods is well suited to many applications where the data are non-negative. We are here particularly interested in spectral unmixing (extracting reflectance spectra of materials present in pixels and associated abundance fractions) for urban hyperspectral images. The originality of our work is that we developed extensions of NMF, which is initially suited to the linear model, for the linear-quadratic model. The proposed algorithms are tested with simulated hyperspectral images using real reflectance spectra and the obtained results are very satisfactory.**

***Index Terms*—blind source separation, linear-quadratic mixing model, non-negative matrix factorization, hyperspectral images, spectral unmixing.**

I. Introduction

Many applications involve observed signals that are mixtures of original signals. Those mixtures generally occur during the signal propagation to the sensors. Signals received by the sensors are thus composed of the original signals, called "source" signals. Blind Source Separation (BSS) [1], [2] aims at retrieving the "source" signals, with limited prior information about the sources and the mixing model. Generally the class of the mixing model (linear, post-linear, linear-quadratic...) is known but not its parameters.

Most existing BSS methods rely on the linear instantaneous mixing model, which assumes that the observed signals (mixtures) are linear combinations of the source signals. There are four principal classes of methods for the linear model, depending on the assumptions made on the sources and/or mixing parameters. The first class, and the oldest one, is Independent Component Analysis (ICA) [1]–[6]. It corresponds to methods which are based on the statistical independence of the sources, i.e. which can be used only if this condition is met by the sources. There are then the methods based on the "joint" sparsity properties of the sources, i.e. often on the possibility of isolating each source in a certain zone of the signal. This class is often called Sparse Component Analysis (SCA) [1], [7]–[9]. The third class consists of methods based on Non-negative Matrix Factorization (NMF) which needs the non-negativity of the observations, mixing coefficients and sources [10]–[12]. The last class corresponds to Bayesian methods, that can use different prior information about the observations, sources or mixing coefficients and usually need to associate probability laws with the involved components, so they need to have enough prior information to do it [13], [14].

In this article, we are especially interested in the problem of unmixing hyperspectral images, i.e. extracting reflectance spectra[1] of materials present in an image. Indeed, the resolution of hyperspectral images often results in pixels that are not homogeneous but correspond to a ground zone with different materials. The pixel spectra thus result from the contributions of those different material reflectance spectra. In the reflective domain (wavelengths from 0.4 to 2.5 μm), a linear mixing model of reflectances is generally used. In this model, if the reflectance spectra are considered as the "sources", the mixing coefficients thus correspond to the proportions of the materials in each pixel (see e.g [15]–[17]). Such a model is essentially valid when the scene is flat.

In this paper, we aim at unmixing reflectances in urban images and the involved model is then linear-quadratic. This is due to the 3D structures that induce multiple scattering of light between surfaces in urban environments [18], [19].

There exist much less BSS methods for the linear-quadratic model than for the linear one. Most approaches are based on ICA [20]–[23]. But there are also some methods using sparsity as [24], [25]. Some Bayesian approaches have also been proposed, as [26], but using the statistical independence of sources as ICA methods. Existing BSS approaches for the linear-quadratic model, to our knowledge, are thus based at least on sparsity or statistical independence. We stress this point since, in our studied problem of unmixing urban images, we know that the spectra are often very correlated (which excludes methods using the statistical independence assumption) and that the sparsity can hardly be used. Concerning sparsity, on the one hand, this is due to the fact that the large variability in urban environment makes the use of methods exploiting the spatial "joint" sparsity difficult. On the other hand, we also know that our spectra are not sparse.

Considering all this, we chose to take advantage of the non-negativity of observations, sources and mixing coefficients. In this paper, we propose non-negativity-based BSS methods dedicated to the linear-quadratic model, also applicable when sources are correlated. Our approach consists in extending the principles of several NMF methods, initially intended for linear models, to our non-linear model.

[†] IRAP, CNRS, Université de Toulouse, UPS-OMP, 14, av. Edouard Belin, 31400 Toulouse, France.
[*] ONERA, The French Aerospace Lab, 31055 Toulouse, France.

[1] Each material is characterized by a reflectance spectrum which corresponds to the proportion of light power reflected by this material, for each wavelength (values between 0 and 1).

In Section II, we first present the considered mixing model and the associated constraints. We then show how this model is adapted to the use of NMF and present the criterion that we aim to minimise. In Section III, the first approach using NMF for our problem is presented. Section IV describes our second proposed NMF method suited to the linear-quadratic model, based on a projected gradient descent algorithm. In Section V, this gradient method is extended with a Newton update for the mixing matrix. Section VI then presents our fourth proposed method, which is a multiplicative NMF algorithm. All these methods are then compared and evaluated in Section VII, where results obtained with simulated images of real reflectance spectra are presented.

II. Problem statement and how to extend NMF

A. Linear-Quadratic Model

In this work, we aim at extracting the material spectra composing an urban hyperspectral image. In this case, each pixel spectrum is a linear-quadratic mixture of reflectance spectra (sources) of different materials. The justification and validation of this mixing model as well as the constraints related to the mixing coefficients have been presented in [18], [19]. The spectrum of a given pixel i ($i = 1...P$, where P is the number of pixels) thus reads:

$$\boldsymbol{x_i} = \sum_{j=1}^{M} a_j(i)\boldsymbol{s_j} + \sum_{j=1}^{M}\sum_{\ell=j}^{M} a_{j,\ell}(i)\boldsymbol{s_j} \odot \boldsymbol{s_\ell} \ , \tag{1}$$

$$\text{with} \begin{cases} \boldsymbol{s_j} \geq 0, & j = 1...M \\ \sum_{j=1}^{M} a_j(i) = 1 & \\ a_j(i) \geq 0, & j = 1...M \\ 0 \leq a_{j,\ell}(i) \leq 0.5, & 1 \leq j \leq \ell \leq M \end{cases}$$

where

- $\boldsymbol{x_i}$: observed signal i, which is the reflectance spectrum of pixel i.
- $\boldsymbol{s_j}$: source j, which is the reflectance spectrum of material j ($\odot$ corresponds to an element-wise multiplication).
- $\boldsymbol{x_i}$ and $\boldsymbol{s_j}$: column vectors of size L, with L the number of spectral bands in the image.
- M: number of materials present in the image.
- $a_j(i)$ and $a_{j,\ell}(i)$: mixing coefficients respectively corresponding to the linear and quadratic parts of the model.

We aim at performing blind source separation, i.e. at estimating both sources (spectra) and mixing coefficients in an unsupervised way. Note that our final goal is to accurately estimate the linear part of the model since it corresponds to the real ground pixel composition, whereas quadratic terms correspond to reflections [18], [19]. Taking into account the quadratic part only aims at improving the estimation of the linear part.

As in many BSS methods, we assume the number of materials M is known.

B. Non-negative Matrix Factorization (NMF)

Principle of NMF: Given a non-negative matrix $\boldsymbol{V}$, Non-negative Matrix Factorization consists in finding non-negative matrix factors $\boldsymbol{W}$ and $\boldsymbol{H}$ that verify $\boldsymbol{V} \approx \boldsymbol{WH}$.

Different NMF methods have been proposed in the literature for linear mixtures [10], based on different criteria, without or with constraints, and several updating rules for the matrix factors (NMF algorithms are iterative) have been used [10], [12], [27]–[29].

C. Rewriting the linear-quadratic model for NMF

To use NMF with the above linear-quadratic model, we first write the observed data as the product of two matrices. We thus write the model (1) in matrix form as follows (for P pixels, $P \geq 2$):

$$\boldsymbol{X} = \boldsymbol{AS} = \boldsymbol{A_a S_a} + \boldsymbol{A_b S_b} \tag{2}$$

with:

- $\boldsymbol{X} = [\boldsymbol{x_1} \cdots \boldsymbol{x_P}]^T$.
- $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A_a} & \boldsymbol{A_b} \end{bmatrix}$: mixing matrix,

  with $\boldsymbol{A_a} = \begin{bmatrix} a_1(1) & \cdots & a_M(1) \\ \vdots & & \vdots \\ a_1(P) & \cdots & a_M(P) \end{bmatrix}$

  and $\boldsymbol{A_b} = \begin{bmatrix} a_{1,1}(1) & a_{1,2}(1) & \cdots & a_{M,M}(1) \\ \vdots & & & \vdots \\ a_{1,1}(P) & a_{1,2}(P) & \cdots & a_{M,M}(P) \end{bmatrix}$.
- $\boldsymbol{S} = \begin{bmatrix} \boldsymbol{S_a} \\ \boldsymbol{S_b} \end{bmatrix}$: source matrix,

  with $\boldsymbol{S_a} = \begin{bmatrix} \boldsymbol{s_1} & \cdots & \boldsymbol{s_M} \end{bmatrix}^T$

  and $\boldsymbol{S_b} = \begin{bmatrix} \boldsymbol{s_1} \odot \boldsymbol{s_1} & \boldsymbol{s_1} \odot \boldsymbol{s_2} & \cdots & \boldsymbol{s_M} \odot \boldsymbol{s_M} \end{bmatrix}^T$.

The way we wrote our model in Eq. (2), first introduced in [30] in the case of only two sources, permits us to extend the principle of NMF methods, which are originally dedicated to linear models, to the case of linear-quadratic models.

The criterion we aim at minimising is:

$$J = \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{AS}\|_F^2 = \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{A_a S_a} - \boldsymbol{A_b S_b}\|_F^2. \tag{3}$$

We here chose the classical criterion of Frobenius norm for simplicity reasons and propose four algorithms to optimise it. The algorithm initialisation is discussed in Section VII-B.

III. First algorithm: extension of multiplicative linear NMF

Our first approach to minimise the criterion (3) is to apply a linear NMF method from the literature without using the structure of matrix $\boldsymbol{S}$, but with an extended number of sources: the quadratic terms (terms of matrix $\boldsymbol{S_b}$) are considered as additional "pseudo-sources", without taking into account the link between these "pseudo-sources" and the real ones (those of matrix $\boldsymbol{S_a}$). We chose to apply Lee and Seung's multiplicative algorithm [11] to our problem.

Our proposed extended algorithm will here be denoted *Linear ext* ("ext" for "extended"). It is iterative, and the matrix updates read as follows for a given iteration:

***Linear ext* algorithm**

1) Update of matrix $\boldsymbol{S}$:

$$\boldsymbol{S} \leftarrow \boldsymbol{S} \odot ((\boldsymbol{A^T X}) \oslash (\boldsymbol{A^T AS} + \varepsilon)).$$

2) Update of matrix $\boldsymbol{A}$:

$$\boldsymbol{A} \leftarrow \boldsymbol{A} \odot ((\boldsymbol{X}\boldsymbol{S}^{\boldsymbol{T}}) \oslash (\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^{\boldsymbol{T}} + \varepsilon)).$$

3) Constraints on matrix $\boldsymbol{A}$:

$$\begin{aligned} [a_1(i) \cdots a_M(i)] &\leftarrow [a_1(i) \cdots a_M(i)] \,/ \sum_{j=1}^{M} a_j(i) \\ a_{j,\ell}(i) &\leftarrow \min\{a_{j,\ell}(i), 0.5\}\,. \end{aligned}$$

The notation $\oslash$ corresponds to element-wise division. The term ε in the denominator is chosen to be very small and positive and aims at avoiding possible divisions by zero.

Step 3 in the algorithm aims at satisfying the constraints related to the mixing coefficients in (1). This step is not present in the original Lee and Seung method [11] and was not used in our previous work [18] since we had not yet performed the physical analysis [19] which proved the validity of these constraints (especially the sum-to-one condition).

IV. Second algorithm: Gradient LQ NMF

We now propose an NMF algorithm really adapted to the linear-quadratic (LQ) model: this algorithm takes into account that the quadratic terms are products of the real sources present in the mixing model. For the sake of simplicity, we here start with a projected gradient descent algorithm with a fixed learning rate. The principle is to take into account the quadratic terms in the gradient calculation corresponding to the criterion (3). The algorithm proposed in this section has already been presented in our conference paper [30], but only in the case of 2 sources. We here extend it to a general case of M sources and add the new constraints related to the mixing coefficients (essentially the sum-to-one condition).

A. Gradient calculations

Our mixing model does not induce any modifications in gradient calculation with respect to the mixing matrix, compared to the linear case [10]. Indeed, our criterion has the same dependence with respect to the mixing matrix $\boldsymbol{A}$ as in the linear case. Writing the criterion (3) as follows

$$\begin{aligned} J &= \frac{1}{2} Tr((\boldsymbol{X} - \boldsymbol{A}\boldsymbol{S})(\boldsymbol{X} - \boldsymbol{A}\boldsymbol{S})^T) \\ &= \frac{1}{2} Tr(\boldsymbol{X}\boldsymbol{X}^T - \boldsymbol{X}\boldsymbol{S}^T\boldsymbol{A}^T - \boldsymbol{A}\boldsymbol{S}\boldsymbol{X}^T + \boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^T\boldsymbol{A}^T) \end{aligned}$$

permits us to easily derive the gradient expression (using the properties in [31]):

$$\begin{aligned} \frac{\partial \boldsymbol{J}}{\partial \boldsymbol{A}} &= \frac{1}{2}(-\boldsymbol{X}\boldsymbol{S}^T - \boldsymbol{X}\boldsymbol{S}^T + 2\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^T) \\ &= -(\boldsymbol{X} - \boldsymbol{A}\boldsymbol{S})\boldsymbol{S}^T. \end{aligned} \quad (4)$$

However, the gradient calculation with respect to the sources changes as compared with the linear case, since each source here also yields quadratic terms in the mixing model. To perform this calculation, we write our criterion J in (3) in scalar form:

$$J = \frac{1}{2} \sum_{i,n} [X_{in} - \sum_{j=1}^{M} a_j(i) S_{jn} - \sum_{j=1}^{M} \sum_{\ell=j}^{M} a_{j,\ell}(i) S_{jn} S_{\ell n}]^2 \quad (5)$$

where n is the index for the spectra components, i.e. the wavelengths. S_{jn} is element (j, n) of matrix $\boldsymbol{S}$, with $j = 1..M$; more precisely, the terms S_{jn} are the elements of matrix $\boldsymbol{S_a}$, since they correspond to the real sources.

Note that a given pair of sources $\{j, \ell\} = \{\ell, j\}$ only appears once in our model (1) and in the following equations, so the corresponding coefficient is hereafter denoted equally as $a_{j,\ell}$ or $a_{\ell,j}$.
Using (5), the elements of the gradient of J with respect to source $\boldsymbol{s_p}$ can be computed as follows:

$$\begin{aligned} \frac{\partial J}{\partial S_{pn}} = & \sum_{i=1}^{P} \left(X_{in} - \sum_{j=1}^{M} a_j(i) S_{jn} - \sum_{j=1}^{M} \sum_{\ell=j}^{M} a_{j,\ell}(i) S_{jn} S_{\ell n} \right) \\ & \times \left(-a_p(i) - \sum_{j=1, j \neq p}^{M} a_{j,p}(i) S_{jn} - 2a_{p,p}(i) S_{pn} \right) \quad (6 \\ = & -\sum_{i=1}^{P} a_p(i) \left(X_{in} - \sum_{j=1}^{M} a_j(i) S_{jn} - \sum_{j=1}^{M} \sum_{\ell=j}^{M} a_{j,\ell}(i) S_{jn} S_{\ell n} \right) \\ & - \sum_{j=1, j \neq p}^{M} \left[S_{jn} \sum_{i=1}^{P} a_{j,p}(i) \left(X_{in} - \sum_{j=1}^{M} a_j(i) S_{jn} \right.\right. \\ & \left.\left. - \sum_{j=1}^{M} \sum_{\ell=j}^{M} a_{j,\ell}(i) S_{jn} S_{\ell n} \right) \right] \\ & -2S_{pn} \sum_{i=1}^{P} a_{p,p}(i) \left(X_{in} - \sum_{j=1}^{M} a_j(i) S_{jn} \right. \\ & \left. - \sum_{j=1}^{M} \sum_{\ell=j}^{M} a_{j,\ell}(i) S_{jn} S_{\ell n} \right). \quad (7 \end{aligned}$$

Since the S_{jn} correspond to the elements of matrix $\boldsymbol{S_a}$ they can also be denoted $[\boldsymbol{S_a}]_{jn}$. Replacing the scalar expressions in (7) by matrix products, we obtain the following matrix form:

$$\begin{aligned} \left[\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{S_a}}\right]_{pn} = & -[\boldsymbol{A_a}^T(\boldsymbol{X} - \boldsymbol{A}\boldsymbol{S})]_{pn} \\ & - \sum_{j=1, j \neq p}^{M} [\boldsymbol{S_a}]_{jn} \times [\boldsymbol{A_b}^T(\boldsymbol{X} - \boldsymbol{A}\boldsymbol{S})]_{(jp)n} \\ & -2[\boldsymbol{S_a}]_{pn} \times [\boldsymbol{A_b}^T(\boldsymbol{X} - \boldsymbol{A}\boldsymbol{S})]_{(pp)n} \end{aligned} \quad (8)$$

where (jp) is the index of the column of matrix $\boldsymbol{A_b}$ corresponding to coefficients $a_{jp}(i)$. It is, of course, also the index of the row containing $\boldsymbol{s_j} \odot \boldsymbol{s_p}$ in matrix $\boldsymbol{S_b}$ (it is e.g. equal to one for $j = p = 1$).

The algorithm proposed here supposes that we impose the quadratic term positions in matrix $\boldsymbol{S_b}$. Indeed, these terms are computed using the M sources of the linear part, as we will show in Section IV-B.

Note that the calculation here is presented in a general case of a linear-quadratic model. If one faces a bilinear model, i.e. without squared terms, the presented result can easily be adapted by removing the last term (3^{rd} line) in expression (8), which is the one due to the squared terms.

B. Update rules

Now applying the gradient descent algorithm with a fixed learning rate [10], we obtain the following update rule for the mixing matrix (for the $(m+1)^{th}$ iteration):

$$\boldsymbol{A}^{(m+1)} \leftarrow \boldsymbol{A}^{(m)} - \alpha_A \frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}} \tag{9}$$

where α_A is a small positive learning rate. In the same way, we obtain for the linear part of the source matrix, $\boldsymbol{S_a}$, the following update rule (keeping in mind that the gradient expression, given in (8), is different from that in the linear case):

$$[\boldsymbol{S_a}]_{pn}^{(m+1)} \leftarrow [\boldsymbol{S_a}]_{pn}^{(m)} - \alpha_S \left[\frac{\boldsymbol{\partial J}}{\boldsymbol{\partial S_a}}\right]_{pn} \tag{10}$$

where α_S is the learning rate corresponding to matrix $\boldsymbol{S}$. The learning rates can indeed be different for update rules of the two matrices $\boldsymbol{A}$ and $\boldsymbol{S}$ [10].

Concerning updating rules in Equations (9) and (10), applying the gradient descent algorithm in this way is not sufficient because it does not guarantee non-negativity. To enforce the non-negativity constraint, we chose a solution proposed in the literature (see [10], [12]) which consists in comparing, at each iteration, each updated matrix with a very small positive value ϵ, keeping the maximum between them as follows

$$\begin{aligned} \boldsymbol{A}^{(m+1)} &\leftarrow max\{\boldsymbol{A}^{(m+1)}, \epsilon\} && (11) \\ \boldsymbol{S_a}^{(m+1)} &\leftarrow max\{\boldsymbol{S_a}^{(m+1)}, \epsilon\}. && (12) \end{aligned}$$

This must be understood as a comparison of each element of the matrix with ϵ, and we only replace the negative elements by ϵ. We thus eliminate all negative matrix entries and guarantee the non-negativity needed in NMF algorithms. This is why the algorithm is called a "projected" gradient descent [10], [12], since comparing each estimated value to ϵ is like projecting the solution in the interval $[\epsilon, +\infty]$.

For the quadratic terms, i.e. for each row of matrix $\boldsymbol{S_b}$, we just copy the element-wise product of the two rows corresponding to the two real concerned sources. We thus force the elements of matrix $\boldsymbol{S_b}$ to be in keeping with the reality in our mixtures:

$$[\boldsymbol{S_b}]_{(jp)n}^{(m+1)} \leftarrow [\boldsymbol{S_a}]_{jn}^{(m+1)} [\boldsymbol{S_a}]_{pn}^{(m+1)}. \tag{13}$$

C. Final algorithm

To complete the algorithm, we have to add the constraints related to the mixing coefficient, as in the *Linear ext* algorithm in Section III. We thus have, for each iteration:

***Grd LQ* algorithm**

1) Update of matrix $\boldsymbol{S_a}$:

$$\begin{aligned} [\boldsymbol{S_a}]_{pn} \;\leftarrow\; & [\boldsymbol{S_a}]_{pn} + \alpha_S \; [\boldsymbol{A_a}^T(\boldsymbol{X} - \boldsymbol{AS})]_{pn} \\ & + \alpha_S \sum_{j=1, j\neq p}^{M} [\boldsymbol{S_a}]_{jn} \times [\boldsymbol{A_b}^T(\boldsymbol{X} - \boldsymbol{AS})]_{(jp)n} \\ & + 2\;\alpha_S \; [\boldsymbol{S_a}]_{pn} \times [\boldsymbol{A_b}^T(\boldsymbol{X} - \boldsymbol{AS})]_{(pp)n} \\ \boldsymbol{S_a} \;\leftarrow\; & max\{\boldsymbol{S_a}, \epsilon\}. \end{aligned}$$

2) Update of matrix $\boldsymbol{S_b}$ using $\boldsymbol{S_a}$:

$$[\boldsymbol{S_b}]_{(jp)n} \leftarrow [\boldsymbol{S_a}]_{jn} \times [\boldsymbol{S_a}]_{pn}.$$

3) Update of matrix $\boldsymbol{A}$:

$$\begin{aligned} \boldsymbol{A} &\leftarrow \boldsymbol{A} + \alpha_A \; (\boldsymbol{X} - \boldsymbol{AS})\boldsymbol{S}^T \\ \boldsymbol{A} &\leftarrow max\{\boldsymbol{A}, \epsilon\}. \end{aligned}$$

4) Constraints on matrix $\boldsymbol{A}$:

$$\begin{aligned} [a_1(i) \cdots a_M(i)] &\leftarrow [a_1(i) \cdots a_M(i)] \,/ \sum_{j=1}^{M} a_j(i) \\ a_{j,\ell}(i) &\leftarrow \min\{a_{j,\ell}(i), 0.5\}\,. \end{aligned}$$

We thus here proposed an NMF algorithm suited to the linear-quadratic mixing model, when the data, i.e. the observations, sources and mixing coefficients, are non-negative. However, this algorithm has the disadvantage of depending on two learning rates that are manually fixed. This can reduce performance if they are not well chosen, and its convergence can be very long. In Sections V and VI, we then propose two extensions of this algorithm that aim at reducing these effects.

V. Third algorithm: Gradient-Newton LQ NMF

In the general case of optimisation based on gradient descent [32], [33], there exist different algorithms permitting one to derive an adequate learning rate at each iteration. However, these methods do not guarantee an optimal learning rate and can slow down the algorithm. We thus here prefer to apply a Newton update, which is better suited if the Hessian calculation is not too difficult and if it doesn't increase the overall computational cost.

However, we here only apply the Newton update to the mixing matrix $\boldsymbol{A}$, since it yields a simple expression, as shown in Section V-A. The Newton matrix update is much less easily applicable to matrix $\boldsymbol{S}$, as discussed in Section V-B.

A. Hessian for matrix $\boldsymbol{A}$

We here show how the Newton update of matrix $\boldsymbol{A}$ can be derived. In [10], a less detailed demonstration is given.

We first rewrite the gradient expression with respect to $\boldsymbol{A}$, given in (4), in scalar form:

$$\frac{\partial J}{\partial A_{pk}} = \left[-(\boldsymbol{X} - \boldsymbol{AS})\boldsymbol{S}^T\right]_{pk} = -\sum_n \left(X_{pn} - \sum_{j=1}^{K} A_{pj} S_{jn} \right) S_{kn} \tag{14}$$

where K is the total number of columns of $\boldsymbol{A}$, i.e. the overall number of real sources and quadratic terms.

The calculation of the second derivative gives:

$$\frac{\partial^2 J}{\partial A_{pk} \partial A_{m\ell}} = \begin{cases} \sum_n S_{\ell n} S_{kn} = \left[\boldsymbol{SS}^T\right]_{k\ell} & \text{if } p = m,\ \forall p \\ 0 & \text{if } p \neq m. \end{cases} \tag{15}$$

Note that $\boldsymbol{SS}^T \in \mathbb{R}_+^{K \times K}$, and that the gradient matrix, $\frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}} \in \mathbb{R}^{P \times K}$, has the same dimension as $\boldsymbol{A}$.

A Hessian is generally defined with respect to a vector. Let's then vectorise $\boldsymbol{A}$ row by row. We obtain a vector whose size is

$1 \times KP$ that can be denoted by $\boldsymbol{vec(A)}$. We then can consider the total Hessian matrix associated with $\boldsymbol{vec(A)}$ (denoted by $\frac{\boldsymbol{\partial^2 J}}{\boldsymbol{\partial A^2}}$) and whose size is $KP \times KP$.

We can notice from (15) that:
- The second derivative of J is equal to zero if $p \neq m$, so the total Hessian matrix associated with $\boldsymbol{vec(A)}$ is block-diagonal.
- It doesn't depend on the value of p, i.e. it doesn't depend on the row index in the matrix. In $\frac{\boldsymbol{\partial^2 J}}{\boldsymbol{\partial A^2}}$, the same block is thus repeated.

Therefore, the result in (15) permits us to write the total Hessian matrix $\frac{\boldsymbol{\partial^2 J}}{\boldsymbol{\partial A^2}}$ as follows:

$$\frac{\boldsymbol{\partial^2 J}}{\boldsymbol{\partial A^2}} = \begin{bmatrix} \boldsymbol{SS}^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{SS}^T & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{SS}^T \end{bmatrix}. \quad (16)$$

This shows that there are no interferences between rows of $\boldsymbol{A}$ and we can reason as follows.
Let $\boldsymbol{A_p}$ be the p-th row of $\boldsymbol{A}$, $\left[\frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}}\right]_p$ be the p-th row of the gradient matrix and $\boldsymbol{H}$ be the matrix defined as $H_{k\ell} = \frac{\partial^2 J}{\partial A_{pk} \partial A_{p\ell}}$ (i.e. $\boldsymbol{H} = \boldsymbol{SS}^T$). The matrix $\boldsymbol{H}$ is thus the Hessian corresponding to the vector $\boldsymbol{A_p}$. We then obtain the following update rule for this row vector, applying the usual Newton expression [32], [33]:

$$\boldsymbol{A_p}^T \leftarrow \boldsymbol{A_p}^T - \boldsymbol{H}^{-1} \left[\frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}}\right]_p^T. \quad (17)$$

Since $\boldsymbol{H}$ is symmetric, expression (17) can also be written as follows (we switch from column vectors to row vectors):

$$\boldsymbol{A_p} \leftarrow \boldsymbol{A_p} - \left[\frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}}\right]_p \boldsymbol{H}^{-1}. \quad (18)$$

The matrix $\boldsymbol{H}$ is the same for each row of $\boldsymbol{A}$. The Newton update of $\boldsymbol{A}$ thus reads:

$$\begin{aligned} \boldsymbol{A} &\leftarrow \boldsymbol{A} - \frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}} \boldsymbol{H}^{-1} \\ &= \boldsymbol{A} - \frac{\boldsymbol{\partial J}}{\boldsymbol{\partial A}} (\boldsymbol{SS}^T)^{-1} \\ &= \boldsymbol{A} + (\boldsymbol{X} - \boldsymbol{AS})\boldsymbol{S}^T(\boldsymbol{SS}^T)^{-1} \\ &= \boldsymbol{A} + \boldsymbol{XS}^T(\boldsymbol{SS}^T)^{-1} - \boldsymbol{ASS}^T(\boldsymbol{SS}^T)^{-1} \\ &= \boldsymbol{XS}^T(\boldsymbol{SS}^T)^{-1}. \end{aligned} \quad (19)$$

We obtain a very simple expression for the update of $\boldsymbol{A}$, thanks to the particular structure of the Hessian.

B. Hessian for matrix $\boldsymbol{S}$

The Hessian expression for matrix $\boldsymbol{S}$ is far less simple. Because of the complex expression of the gradient with respect to $\boldsymbol{S}$, we can not obtain such a convenient structure for the Hessian (see Appendix A). As we are here dealing with matrix and not vector updates, applying Newton algorithm is quite complicated if the Hessian does not have a special structure as it is the case for the mixing matrix.

Indeed, we obtain a Hessian that is block-diagonal, but (see results and conclusions in Appendix A):
- the expression for each block is complicated, it is not a simple matrix product, since there are several terms involved,
- the blocks are different from one column of matrix $\boldsymbol{S_a}$ to another, which implies an inversion of the Hessian for each column separately. The number of matrix inversions to be performed at each iteration is thus equal to the number of wavelengths. As a comparison, the Newton update rule of matrix $\boldsymbol{A}$ only requires one matrix inversion at each iteration.

This means that applying a Newton update rule for matrix $\boldsymbol{S}$ can considerably slow down the algorithm and the gain in performance could not be worth it because of the heavy computational cost.

The algorithm considered below thus uses Newton update only for matrix $\boldsymbol{A}$. We will see in Section VII that this already improves results.

C. Final algorithm

As compared with our *Grd LQ* algorithm presented in Section IV-C, the extended *Grd-Newt LQ* algorithm obtained here only has one modified step, namely the 3^{rd} step. It reads:

Grd-Newt LQ Algorithm

1) Update of matrix $\boldsymbol{S_a}$:
   same as *Grd LQ algorithm*
2) Update of matrix $\boldsymbol{S_b}$ using $\boldsymbol{S_a}$:
   same as *Grd LQ algorithm*
3) Update of matrix $\boldsymbol{A}$:

$$\begin{aligned} \boldsymbol{A} &\leftarrow \boldsymbol{XS}^T(\boldsymbol{SS}^T)^{-1} \\ \boldsymbol{A} &\leftarrow max\{\boldsymbol{A}, \epsilon\}. \end{aligned}$$

4) Constraints on matrix $\boldsymbol{A}$:
   same as *Grd LQ algorithm*

VI. Fourth algorithm: Multiplicative LQ NMF

Another way to get rid of the learning rates in the *Grd LQ* algorithm (Section IV) is to develop a multiplicative version of this NMF algorithm. This multiplicative algorithm can be derived from the gradient-based algorithm by expressing the learning rates as functions of the matrices $\boldsymbol{A}$ and $\boldsymbol{S}$ (see e.g. [27] for the linear model). This is detailed hereafter for the case of the linear-quadratic model considered in this paper.

The update of matrix $\boldsymbol{A}$ is the same as in the linear case (same as in [27]), since $\boldsymbol{A}$ is not concerned by the quadratic terms. Its update is thus the same as in step 2 of our multiplicative *Linear ext* algorithm (Section III).

We now detail the update calculation for matrix $\boldsymbol{S_a}$. We use the gradient update expression corresponding to the first step of the *Grd LQ* algorithm (Section IV-C), and replace the fixed learning rate α_S by ϕ_{pn} which depends on p and n. This gives, for source p:

$$[\boldsymbol{S_a}]_{pn} \leftarrow [\boldsymbol{S_a}]_{pn} - \phi_{pn} \left[\frac{\boldsymbol{\partial J}}{\boldsymbol{\partial S_a}}\right]_{pn} \quad (20)$$

with $\left[\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{S_a}}\right]_{pn} = -[\boldsymbol{A_a}^T(\boldsymbol{X}-\boldsymbol{AS})]_{pn} - \sum_{j=1,j\neq p}^{M}[\boldsymbol{S_a}]_{jn} \times [\boldsymbol{A_b}^T(\boldsymbol{X}-\boldsymbol{AS})]_{(jp)n} - 2[\boldsymbol{S_a}]_{pn} \times [\boldsymbol{A_b}^T(\boldsymbol{X}-\boldsymbol{AS})]_{(pp)n}$.

The expression of the learning rate ϕ_{pn} is chosen so as to yield a multiplicative update and to preserve the non-negativity. The first point is verified if the learning rate contains the term $[\boldsymbol{S_a}]_{pn}$ in the numerator, to permit a factorisation of $[\boldsymbol{S_a}]_{pn}$ in the expression (20). To fulfil the second point, we have to get rid, in the right-hand term of (20), of all terms preceded by a minus sign, to obtain a sum of positive terms. This is achieved by placing those terms in the denominator of ϕ_{pn} (without the minus sign), which will permit us to eliminate them when reducing the whole expression to the same denominator.

Based on this principle, we chose the learning rate ϕ_{pn} as follows:

$$\phi_{pn} = \frac{[\boldsymbol{S_a}]_{pn}}{D_{pn}} \tag{21}$$

with

$$\begin{aligned} D_{pn} &= [\boldsymbol{A_a^T AS}]_{pn} + \sum_{j=1,j\neq p}^{M}[\boldsymbol{S_a}]_{jn} \times [\boldsymbol{A_b^T AS}]_{(jp)n} \\ &\quad + 2[\boldsymbol{S_a}]_{pn} \times [\boldsymbol{A_b^T AS}]_{(pp)n}. \end{aligned} \tag{22}$$

The update of source p is then computed by replacing, in Equation (20), the learning rate ϕ_{pn} by its expression (21):

$$\begin{aligned} [\boldsymbol{S_a}]_{pn} \leftarrow\ & \frac{[\boldsymbol{S_a}]_{pn}[\boldsymbol{A_a^T AS}]_{pn} + 2[\boldsymbol{S_a}]_{pn}^2[\boldsymbol{A_b^T AS}]_{(pp)n}}{D_{pn}} \\ & + \frac{[\boldsymbol{S_a}]_{pn} \times \left(\sum_{j=1,j\neq p}^{M}[\boldsymbol{S_a}]_{jn} \times [\boldsymbol{A_b^T AS}]_{(jp)n}\right)}{D_{pn}} \\ & + \frac{[\boldsymbol{S_a}]_{pn}[\boldsymbol{A_a}^T(\boldsymbol{X}-\boldsymbol{AS})]_{pn}}{D_{pn}} \\ & + \frac{2[\boldsymbol{S_a}]_{pn}^2 \times [\boldsymbol{A_b}^T(\boldsymbol{X}-\boldsymbol{AS})]_{(pp)n}}{D_{pn}} \\ & + \frac{[\boldsymbol{S_a}]_{pn}\left(\sum_{j=1,j\neq p}^{M}[\boldsymbol{S_a}]_{jn}[\boldsymbol{A_b}^T(\boldsymbol{X}-\boldsymbol{AS})]_{(jp)n}\right)}{D_{pn}}. \end{aligned}$$

We thus finally obtain the following update rule:

$$\begin{aligned} [\boldsymbol{S_a}]_{pn} \leftarrow\ & [\boldsymbol{S_a}]_{pn}\left(\frac{[\boldsymbol{A_a^T X}]_{pn} + 2[\boldsymbol{S_a}]_{pn}[\boldsymbol{A_b^T X}]_{(pp)n}}{D_{pn}+\varepsilon}\right. \\ & \left. + \frac{\sum_{j=1,j\neq p}^{M}[\boldsymbol{S_a}]_{jn}[\boldsymbol{A_b^T X}]_{(jp)n}}{D_{pn}+\varepsilon}\right). \end{aligned} \tag{23}$$

As in the linear case, ε (very small and positive) is added to the denominator to avoid possible divisions by zero.

It is easy to verify that, in the update rule (23), if the convergence is met (i.e. $\boldsymbol{X} = \boldsymbol{AS}$), the entire term in parentheses is equal to one (disregarding the ε we added) and the value in the right-hand side is equal to $[\boldsymbol{S_a}]_{pn}$.

This algorithm has the advantage of eliminating the learning rate from the update rule. Besides, the multiplicative construction composed of only sums of non-negative terms, guarantees the non-negativity at each iteration, if the initialisation is non-negative. We thus do not need here the steps, in algorithms *Grd LQ* and *Grd-Newt LQ*, which consist in forcing the non-negativity at each iteration (Eq. (11-12)).

The final algorithm reads, for one iteration:

***Mult LQ* algorithm**

1) Update of matrix $\boldsymbol{S_a}$:
   Apply update rule given by (23)
2) Update of matrix $\boldsymbol{S_b}$ using $\boldsymbol{S_a}$:

$$[\boldsymbol{S_b}]_{(jp)n} \leftarrow [\boldsymbol{S_a}]_{jn} \times [\boldsymbol{S_a}]_{pn}.$$

3) Update of matrix $\boldsymbol{A}$:

$$\boldsymbol{A} \leftarrow \boldsymbol{A} \odot ((\boldsymbol{XS}^T) \oslash (\boldsymbol{ASS}^T + \varepsilon)).$$

4) Constraints on matrix $\boldsymbol{A}$:

$$\begin{aligned} [a_1(i)\cdots a_M(i)] &\leftarrow [a_1(i)\cdots a_M(i)] \Big/ \sum_{j=1}^{M} a_j(i) \\ a_{j,\ell}(i) &\leftarrow \min\{a_{j,\ell}(i), 0.5\}. \end{aligned}$$

VII. Test results

We here compare the performance of the different proposed methods. We also compare them with the results obtained with the classical linear method of Lee and Seung [11]. It will be denoted *Linear*.

A. Performance criteria

Convergence criterion: To check the algorithm convergence, the normalised RMSE (Root Mean Square Error) is used:

$$Err_{tot} = \frac{\|\boldsymbol{X} - \widehat{\boldsymbol{A}}\widehat{\boldsymbol{S}}\|_F}{\|\boldsymbol{X}\|_F} \tag{24}$$

where the notation "ˆ" corresponds to the estimated matrix or vector (hereafter).

Spectra estimation: To compare the real spectra to their estimates, we use the SAM (Spectral Angle Mapper), in radians, after taking into account the possible permutations between the $\widehat{\boldsymbol{s_j}}$[2]:

$$\begin{aligned} SAM_{OUT} &= mean_j(SAM_{OUTj}) \\ &= mean_j\left(arccos\left(\frac{<\boldsymbol{s_j}, \widehat{\boldsymbol{s_j}}>}{\|\boldsymbol{s_j}\|\|\widehat{\boldsymbol{s_j}}\|}\right)\right) \end{aligned} \tag{25}$$

with $j = 1...M$.

[2] To this end, we select the $\boldsymbol{s_j}$ and $\widehat{\boldsymbol{s_k}}$ giving the smallest SAM_{OUTj}, we discard them and search for the next best couple, and so on for all sources.

Coefficient estimation: The mixing coefficient estimation is evaluated using the RMSE:

$$RMSE = \sqrt{\frac{1}{MP}\sum_{i,j}(a_j(i) - \hat{a}_j(i))^2} \tag{26}$$

where $j = 1...M$ and $i = 1...P$.

The SAM and RMSE are chosen because they are commonly used in spectral unmixing. Concerning the spectra and coefficients, we only evaluate the estimation performance of the linear part of the model.

B. Algorithm initialisation

It is well known that the convergence of NMF algorithms often depends on the initialisation and that the NMF solution is not unique [10], [34]. We here assume we have no prior information about the spectra or coefficients (except the constraints in (1)). We thus chose to initialise our algorithms as follows:

- matrix $\widehat{\boldsymbol{A}}$: randomly initialised with uniformly distributed values between 0 and 1 for the linear part (0 and 0.5 for the quadratic part), then the linear part is normalised to fulfil the sum-to-one condition.
- matrix $\widehat{\boldsymbol{S}}$, 2 cases:
  - *init 1*: initialised with constant spectra equal to 0.5.
  - *init 2*: initialised with the spectra resulting from the N-FINDR extraction algorithm [35], used to extract K spectra (K being the total number of spectra in matrix $\boldsymbol{S}$, including the quadratic terms).

C. Data presentation

The results presented here correspond to images created by numerically mixing real reflectance spectra[3]. We simulate little images involving two or three materials (a real complete scene can be unmixed zone by zone), with contributions of reflections. Squared terms are not considered here, but this does not change the principle (we just assume there are no reflections between surfaces of the same material).

1) Data construction: Each image $\boldsymbol{X}$ is computed as $\boldsymbol{X} = \boldsymbol{AS}$, where:

- $\boldsymbol{S}$ contains K spectra: the M spectra of the involved materials and their products (quadratic terms).
- The mixing matrix $\boldsymbol{A}$, of size $P \times K$ (with P number of pixels), is generated randomly as follows. For each pixel:
  - The coefficients of the linear part are generated following uniformly distributed values between 0 and 1, then normalised to fulfil the sum-to-one condition.
  - The coefficients of the quadratic part follow a Gaussian distribution centred on 0.1, with a standard deviation 0.15, $\mathcal{N}(0.1, 0.15)$ (to approximately follow the distribution of values encountered in [19]), then are projected on the interval [0, 0.5] (i.e. if a value is outside this interval, it is reset to the nearest limit).

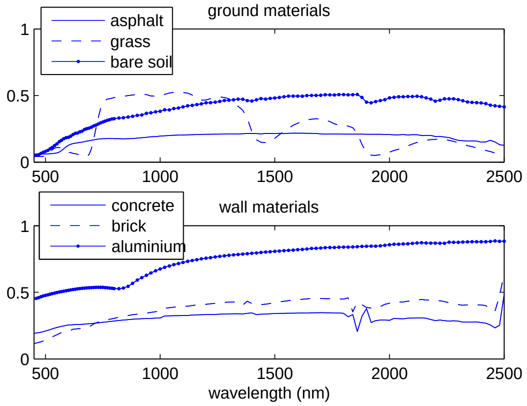


Fig. 1. Reflectance spectra of used materials

2) Used materials: Figure 1 shows the reflectance spectra of the materials used to generate our images, with 126 bands in the spectral domain [0.4, 2.5 μm]. The chosen materials are representative of what can be encountered in urban environments.

It can be seen that, except for the vegetation spectrum, the spectra are very similar. The correlation coefficients measured between them (except those involving the vegetation) are higher than 0.65.

3) Studied cases: The studied images contain two or three materials.

- Case 1: 2 materials
  The pixels correspond to a part of a wall and a part of the ground. The resulting three contributions correspond to these two materials and reflections between them. For each image, we thus here use a couple of spectra involving a ground spectrum and a wall spectrum, which yields 9 possible couples.
- Case 2: 3 materials
  Each pixel corresponds to two ground materials and one wall material, which also yields 9 possible combinations.

For each combination of spectra (involving 2 or 3 materials), we generate 20 different mixing matrices. This set of matrices is the same for all spectra combinations. We thus obtain 9x20 different images in case 1, and also 9x20 images in case 2. All generated images contain 16 pixels (4x4).

D. Performance of the algorithms

As stated above, NMF results can depend on algorithm initialisation. However, in a blind context, it is hard to tell which initialisation gives the best results, since a low reconstruction error Err_{tot} is not a sufficient condition to have the right solution, even if it is a necessary condition (the NMF solution is not unique). We thus propose, for each processed image, to run each algorithm with 30 different initialisations of matrix $\widehat{\boldsymbol{A}}$ (see Section VII-B) and we present results obtained following two protocols:

- Protocol 1: in this case, each run (one initialisation) leads to one final result $\widehat{\boldsymbol{S}}$ and $\widehat{\boldsymbol{A}}$. The 30 obtained results

[3] The used spectra are from the MEMOIRES data base (http://www.onera.fr/dota/memoires).

for each image are used to compute statistics of the performance criteria.

- Protocol 2: for each image, we use the 30 obtained results to derive only one final estimate. Since it is well known that estimation in BSS methods is up to a permutation indeterminacy, we propose to handle that by performing a clustering over the sources estimated in the 30 runs. We here choose to use the K-means method [36] performed over 5 extracted parameters representing each estimated source: its mean, its variance, its kurtosis, the maximum and variance of its derivative[4]. The medians of the estimated sources associated with these clusters then represent the final estimated sources.

Here are the detailed protocols for the presented tests:

Protocol 1

- For each combination of spectra (2 or 3 spectra):
  - ○ For each of the 20 mixing matrices:
    - → A hyperspectral image is created (1 combination of spectra mixed with the mixing matrix)
    - → The algorithm is applied using 30 different initialisations of $\widehat{\boldsymbol{A}}$
    - → The algorithm performance is evaluated for each of these 30 cases
- Mean and standard deviation of the performance criteria are computed over the 9x20x30 obtained results

Protocol 2

- For each combination of spectra (2 or 3 spectra):
  - ○ For each of the 20 mixing matrices:
    - → A hyperspectral image is created (1 combination of spectra mixed with the mixing matrix)
    - → The algorithm is applied using 30 different initialisations of $\widehat{\boldsymbol{A}}$
    - → One result ($\widehat{\boldsymbol{S}}_{\boldsymbol{m}}$,$\widehat{\boldsymbol{A}}_{\boldsymbol{m}}$) is estimated, using the 30 obtained results as follows:
      - · A clustering is performed over the 30 obtained matrices $\widehat{\boldsymbol{S}}$ (applying the clustering algorithm to the associated extracted parameters)
      - · The median of each source cluster is computed to obtain one new matrix $\widehat{\boldsymbol{S}}_{\boldsymbol{m}}$
      - · An estimation of $\widehat{\boldsymbol{A}}_{\boldsymbol{m}}$ is obtained using least squares with non-negativity and sum-to-one (for the linear mixing coefficients) constraints[5].
    - → The algorithm performance is evaluated using the new matrices $\widehat{\boldsymbol{A}}_{\boldsymbol{m}}$ and $\widehat{\boldsymbol{S}}_{\boldsymbol{m}}$
- Mean and standard deviation of the performance criteria are computed over the 9x20 obtained results

1) Case of 2 materials: Tables I and II show, for each configuration (one algorithm with one kind of initialisation for $\widehat{\boldsymbol{S}}$), the mean and standard deviation of the results obtained respectively with Protocols 1 and 2. Table III permits to evaluate the convergence precision and speed (mean CPU time)[6], following Protocol 1.

The *Grd LQ* algorithm gives almost the lowest mean performance here, which is coherent with its high mean total error (Table III), as compared with all other methods. This value represents the error computed at the "convergence point" (when the stop condition is reached). The used stop condition here corresponds to the iteration when the matrices $\widehat{\boldsymbol{A}}$ and $\widehat{\boldsymbol{S}}$ and the criterion (3) become almost constant (with a precision of 10^{-6} for the criterion and of 10^{-5} for the matrices). This means the algorithm had not completely converged towards a minimum when it stopped evolving, and this is a disadvantage of using a fixed learning rate. This algorithm has thus just been a starting point for our methods *Grd-Newt LQ* and *Mult LQ* which are preferable. Indeed, it can be seen that these two algorithms give the best results and their mean performance is about twice better than that of the *Linear* and *Linear ext* algorithms. This means that we can not here neglect the presence of the quadratic terms and that the linear method is not suited. Then, this shows that taking into account the presence of the quadratic terms in gradient calculation presents a non negligible gain.

What can seem weird here, is that the *Linear* algorithm gives better performance than the *Linear ext* one, concerning coefficient estimation (mean RMSE). This can be due to the fact that only one quadratic term is involved here (model with only two sources), so it seems not to disturb much the two-component *Linear* method, which converges with a reasonable precision. This interpretation is verified below in the 3 material case. The *Linear ext* algorithm, which estimates the quadratic term as a third "pseudo-source", can be disturbed by the possible correlation between this term and the real sources, and this can degrade the coefficient estimation.

These first results show the advantage of *Grd-Newt LQ* and *Mult LQ* compared with *Linear ext* or *Linear*.

Comparing results of Tables I and II, it can be seen that Protocol 2 permits us to obtain lower RMSE with our linear-quadratic methods. But the differences are not very important in this case.

Finally, the methods are here almost insensitive to the kind of initialisation chosen for the spectra (constant or N-FINDR).

2) Case of 3 materials: Tables IV and V present, as above, performance obtained using respectively Protocols 1 and 2. The best mean performance is obtained with *Grd-Newt LQ* and *Mult LQ* algorithms, and especially the latter. The total error obtained with *Mult LQ* is among the lowest ones (see Table VI).

Figure 2 shows an example of estimated spectra (for one image among the 9x20 ones) obtained with *Linear ext* and *Mult LQ*, compared to the real spectra. It can be seen that *Mult LQ* gives globally the best estimation, essentially for sources 2 and 3. Note that this example corresponds to values of SAM close to the mean performance in Table V.

The less satisfying method is *Linear* and, in this case, its performance is lower than that of *Linear ext*. This confirms

[4] What we here call derivative calculation of the spectra is made using the *diff* function of Matlab.

[5] The '*lsqnonneg*' Matlab function is used here.

[6] Computation has been performed with Matlab, on a computer with a double core processor, with a frequency of 2.8 GHz and a RAM of 4 GB.

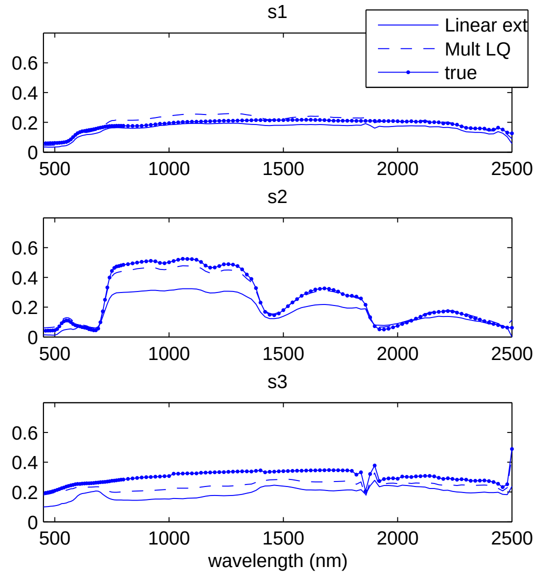


Fig. 2. Example of estimated spectra with *Linear ext* (SAM = 0.12) and *Mult LQ* (SAM = 0.09)

SNR (dB)	SAM		RMSE		Err_{tot} (in %)
	mean	std	mean	std	mean
40	0.03	0.03	0.07	0.06	1.2
30	0.04	0.03	0.07	0.05	3.0
25	0.05	0.03	0.08	0.05	5.4
20	0.08	0.03	0.12	0.09	9.8

TABLE VII
CASE M = 2 MATERIALS - PERFORMANCE FUNCTION OF THE SNR - *Mult LQ* METHOD, WITH INITIALIZATION OF TYPE INIT 1 - PROTOCOL 2

the interpretation made about the results in the case of two materials: We here have three quadratic terms, the linear model is not sufficient at all to model the mixtures and to give an acceptable estimation. Indeed, it can be seen that the total error is high for the *Linear* algorithm (see Table VI).

SAM results obtained using Protocol 2 (Table V) are here much better than those obtained using Protocol 1 (Table IV). Mean SAM values obtained with our *Grd LQ* and *Mult LQ* methods are almost 30 % lower with Protocol 2, with an increased reliability (lower std). The RMSE are almost the same for both protocols. Using Protocol 2 can thus be a good solution to alleviate the problem of NMF dependence on the initialisation. Besides, this solution is unsupervised.

As in the previous case, the methods have low sensitivity to the choice of initialisation for $\widehat{S}$.

The convergence of our algorithms, especially the final one, i.e. *Mult LQ*, has not been proved theoretically in this paper. However, in all our tests, the proposed algorithms made the considered criterion J monotonically decrease. Besides the criterion seems to converge towards a local minimum since the obtained final total errors are very low. A theoretical convergence analysis could be the subject of future work.

The last tests presented in this paper concern the robustness of our *Mult LQ* method to the presence of noise in data. Tables VII and VIII show results obtained following Protocol 2, for different SNR (Signal to Noise Ratio) values. With each of our 9x20 previous images, we generated 10 noisy images[7] for each SNR value (10 runs for each SNR). Protocol 2 has thus been applied to 9x20x10 images this time (instead of 9x20). In both cases, the results are almost unchanged for a 40 dB SNR, as compared to the noiseless case, and performance remains very good down to a 30 dB SNR. In the case of two materials, the results are good even for 20 dB, which is a quite low SNR, since it corresponds to noise representing 10% of the signal magnitude. In [37], for example, where different linear unmixing methods are compared, the lowest considered SNR is of 30 dB. Further, in all cases here, it can be noticed that the mean value of the total error[8] almost corresponds to the ratio of noise vs. signal magnitudes (1, 3, 5 and 10%), which means that the residual error essentially corresponds to the noise. It thus can be concluded that the proposed method is quite robust to noise.

SNR (dB)	SAM		RMSE		Err_{tot} (in %)
	mean	std	mean	std	mean
40	0.09	0.04	0.16	0.08	1.0
30	0.10	0.04	0.17	0.09	2.8
25	0.16	0.07	0.22	0.09	5.0
20	0.33	0.11	0.33	0.10	8.7

TABLE VIII
CASE M = 3 MATERIALS - PERFORMANCE FUNCTION OF THE SNR - *Mult LQ* METHOD, WITH INITIALIZATION OF TYPE INIT 1 - PROTOCOL 2

VIII. CONCLUSION

In this work, we proposed NMF methods for the linear-quadratic model. The proposed algorithms, especially *Grd-Newt LQ* and *Mult LQ* (*Grd LQ* algorithm has essentially been a starting point for the other two proposed ones), gave very satisfactory results. Mean SAM are mostly not above 0.1 radian with our NMF methods using the multiple initialisation procedure (Protocol 2). Note that we here address a difficult configuration since we face correlated sources (spectra), and even NMF is sensitive to this problem.

The proposed algorithms have here been specially developed for our problem related to hyperspectral unmixing for urban images, using the corresponding constraints on the mixing coefficients. However, one can easily adapt them to another application (with a linear-quadratic model and non-negative data), by removing these constraints from the method. We can also imagine adding other constraints to the criterion, depending on the application.

As future work, it would be interesting to test other kinds of initialisations and other constraints (as smoothness). Trying the *Grd LQ* algorithm with a varying learning rate could also give a useful comparison with *Grd-Newt LQ*. Finally, beyond the development and validation of the methods reported in this paper, their practical application to the remote sensing field will consist in applying them to real data, but their evaluation then necessitates information about the real scene composition.

[7] We added a Normal centred noise, whose magnitude corresponds to a percentage of the signal magnitude depending on the desired SNR value.

[8] The total error is computed following Protocol 1, as in Tables III and VI, to evaluate the algorithm convergence for each run.

Appendix A
Hessian calculation for matrix $\boldsymbol{S}$

We use the scalar expression of the gradient of J with respect to S_{pn}, given in Eq. (6).

This first entails that, for $m \neq n$:

$$\frac{\partial^2 J}{\partial S_{km}\partial S_{pn}} = 0. \tag{27}$$

We then obtain, for $k \neq p$:

$$\begin{aligned}\frac{\partial^2 J}{\partial S_{kn}\partial S_{pn}} = &\sum_{i=1}^{P}\left(X_{in} - \sum_{j=1}^{M} a_j(i)S_{jn}\right.\\ &\left.-\sum_{j=1}^{M}\sum_{\ell=j}^{M} a_{j,\ell}(i)S_{jn}S_{\ell n}\right) \times \left(-a_{k,p}(i)\right)\\ &+\sum_{i=1}^{P}\left(-a_k(i) - \sum_{j=1,j\neq k}^{M} a_{j,k}(i)S_{jn} - 2a_{k,k}(i)S_{kn}\right)\\ &\times\left(-a_p(i) - \sum_{j=1,j\neq p}^{M} a_{j,p}(i)S_{jn} - 2a_{p,p}(i)S_{pn}\right).\end{aligned} \tag{28}$$

And finally, for $k = p$:

$$\begin{aligned}\frac{\partial^2 J}{\partial S_{pn}\partial S_{pn}} = &\sum_{i=1}^{P}\left(X_{in} - \sum_{j=1}^{M} a_j(i)S_{jn}\right.\\ &\left.-\sum_{j=1}^{M}\sum_{\ell=j}^{M} a_{j,\ell}(i)S_{jn}S_{\ell n}\right) \times \left(-2a_{p,p}(i)\right)\\ &+\sum_{i=1}^{P}\left(-a_p(i) - \sum_{j=1,j\neq p}^{M} a_{j,p}(i)S_{jn} - 2a_{p,p}(i)S_{pn}\right)^2.\end{aligned} \tag{29}$$

We can thus here reason on the columns of matrix $\boldsymbol{S_a}$ and vectorise this matrix column by column. This permits us to conclude that the total Hessian associated with $vec(\boldsymbol{S_a})$ [9] has the following properties:

- It is block-diagonal, because of result given by Eq. (27).
- The elements of the n^{th} block are given by Equations (28) and (29).
- The blocks are all different, since results in Equations (28) and (29) depend on n.

References

[1] P. Comon and C. Jutten, *Handbook of Blind Source Separation, Independent Component Analysis and Applications*. Oxford UK, Burlington USA: Academic Press, 2010.

[2] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent component analysis*. John Wiley & Sons Inc, a volume in Series on Adaptive and Learning Systems for Signal Processing, Communications and Control, 2001.

[3] C. Jutten and J. Herault, "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, 1991.

[4] A. Hyvärinen, "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis," *IEEE Transactions on Neural Networks*, vol. 10, no. 3, pp. 626–634, 1999.

[5] D.-T. Pham and J.-F. Cardoso, "Blind separation of instantaneous mixtures of non stationary sources," *IEEE Transactions on Signal Processing*, vol. 49, pp. 1837–1848, 2000.

[6] A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second order statistics," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 434–444, 1997.

[7] R. Gribonval and S. Lesage, "A survey of sparse component analysis for blind source separation: principles, perspectives, and new challenges," in *ESANN*, 2006, pp. 323–330.

[8] O. Yilmaz and S. Rickard, "Blind separation of speech mixtures via time-frequency masking," *IEEE Transactions on Signal Processing*, vol. 52, pp. 1830–1847, 2004.

[9] Y. Deville and M. Puigt, "Temporal and time-frequency correlation-based blind source separation methods. part I: Determined and underdetermined linear instantaneous mixtures," *Signal Processing*, vol. 87, no. 3, pp. 374–407, 2007.

[10] A. Cichocki, R. Zdunek, A. H. Phan, and S.-I. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. John Wiley and Sons, UK, 2009.

[11] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," in *Nature*, vol. 401, 1999, pp. 788–791.

[12] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, pp. 2756–2779, 2007.

[13] S. Moussaoui, D. Brie, A. Mohammad-Djafari, and C. Carteret, "Separation of non-negative mixture of non-negative sources using a Bayesian approach and MCMC sampling," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4133–4145, 2006.

[14] K. H. Knuth, "A Bayesian approach to source separation," in *Proceedings of the First International Workshop on Independent Component Analysis and Signal Separation: ICA*, 1999, pp. 283–288.

[15] J. Bayliss, J. A. Gualtieri, and R. F. Cromp, "Analyzing hyperspectral data with independent component analysis," in *Proc. SPIE AIPR Workshop*, vol. 9, 1997, pp. 133–143.

[16] V. P. Pauca, J. Piper, and R. J. Plemmons, "Nonnegative matrix factorization for spectral data analysis," in *Linear algebra and its applications*, vol. 416, no. 1, 2006, pp. 29–47.

[17] J. M. P. Nascimento and J. M. Bioucas-Dias, "Learning dependent sources using mixtures of Dirichlet: Applications on hyperspectral unmixing," in *IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2009.

[18] I. Meganem, P. Déliot, X. Briottet, Y. Deville, and S. Hosseini, "Physical modelling and non-linear unmixing method for urban hyperspectral images," in *Proceedings of 3rd IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), Lisbon, Portugal*, 2011.

[19] ——, "Linear-quadratic mixing model for reflectances in urban environments," *IEEE Transactions on Geoscience And Remote Sensing*, vol. 51, no. 10, 2013.

[20] M. Krob and M. Benidir, "Blind identification of a linear-quadratic mixture: application to quadratic phase coupling estimation," in *IEEE Signal Processing Workshop on Higher-Order Statistics*, 1993, pp. 351–355.

[21] K. Abed-Meraim, A. Belouchrani, and Y. Hua, "Blind identification of a linear-quadratic mixture of independent components based on joint diagonalization procedure," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 5, 1996, pp. 2718 –2721.

[22] Y. Deville and S. Hosseini, "Recurrent networks for separating extractable-target nonlinear mixtures. Part I: non-blind configurations," *Signal Processing*, vol. 89, no. 4, pp. 378–393, 2009.

[23] S. Hosseini and Y. Deville, "Recurrent networks for separating extractable-target nonlinear mixtures. part ii. blind configurations," *Signal Processing*, vol. 93, no. 4, pp. 671 – 683, 2013.

[24] Y. Deville and S. Hosseini, "Blind identification and separation methods for linear-quadratic mixtures and/or linearly independent non-stationary signals," in *9th International Symposium on Signal Processing and Its Applications, ISSPA*, 2007, pp. 1–4.

[25] L. Duarte, R. A. Ando, R. Attux, Y. Deville, and C. Jutten, "Separation of sparse signals in overdetermined linear-quadratic mixtures," in *Proceedings of the 10th international conference on Latent Variable Analysis and Signal Separation*, ser. LVA/ICA, 2012, pp. 239–246.

[9] We here use the same notation vec as for matrix $\boldsymbol{A}$, but note that this time the vectorisation is made column by column.

[26] L. T. Duarte, C. Jutten, and S. Moussaoui, “Bayesian source separation of linear and linear-quadratic mixtures using truncated priors,” *Journal of Signal Processing Systems*, vol. 65, pp. 311–323, 2011.

[27] D. D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in *Proc. NIPS*, vol. 13. MIT Press, 2001.

[28] C. Févotte and J. Idier, “Algorithms for nonnegative matrix factorization with the beta-divergence,” *Neural Computation*, vol. 23, no. 9, pp. 2421–2456, 2011.

[29] I. Dhillon and S. Sra, “Generalized nonnegative matrix approximations with Bregman divergences,” *Sciences New York*, vol. 18, no. 2, pp. 283–290, 2006.

[30] I. Meganem, Y. Deville, S. Hosseini, P. Déliot, and X. Briottet, “Linear-quadratic and polynomial non-negative matrix factorization; application to spectral unmixing,” in *Proceedings of the 19th European Signal Processing Conference (EUSIPCO), Barcelona, Spain*, 2011.

[31] K. B. Petersen and M. S. Pedersen, “The matrix cookbook,” *Technical University of Denmark. URL:http://matrixcookbook.com/.*, 2008.

[32] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, Belmont, Mass., U.S.A., 1999.

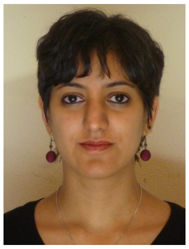
[33] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, U.S.A., 2000.

[34] S. Moussaoui, D. Brie, and J. Idier, “Non-negative source separation: range of admissible solutions and conditions for the uniqueness of the solution,” in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 5, 2005, pp. 289–292.

[35] M. E. Winter, “N-FINDR: an algorithm for fast autonomous spectral end-member determination in hyperspectral data,” in *Proc. SPIE, Imaging spectrometry 5*, vol. 3753, 1999, pp. 266–275.

[36] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, 2008.

[37] J. Plaza, E. M. T. Hendrix, I. García, G. Martín, and A. Plaza, “On endmember identification in hyperspectral images without pure pixels: A comparison of algorithms,” *Journal of Mathematical Imaging and Vision, Springer US*, vol. 42, no. 2-3, pp. 163–175, 2012.



Inès Meganem received the Engineering degree in electronics and signal processing from ENSEEIHT, Toulouse, France, and the M.Sc. degree in signal processing from the National Polytechnical Institute of Toulouse, both in 2009 and, in December 2012, the Ph.D. degree in signal and image processing from the University of Toulouse. She was with IRAP and ONERA lab during its Ph.D., she then worked as a teaching and research assistant in signal and image processing at the University of Toulouse, and with the IRAP lab. Her main research interests concern the study of Blind Source Separation methods for the analysis of hyperspectral images, with applications to astrophysics and spectral unmixing in remote sensing.

Yannick Deville was born in Lyon, France, in 1964. He graduated from the Ecole Nationale Supérieure des Télécommunications de Bretagne (Brest, France) in 1986. He received the D.E.A and Ph.D. degrees, both in Microelectronics, from the University of Grenoble (France), in 1986 and 1989 respectively. From 1986 to 1997, he was a Research Scientist at Philips Research Labs (Limeil, France). His investigations during this period concerned various fields, including GaAs integrated microwave RC active filters, VLSI cache memory architectures and replacement algorithms, neural network algorithms and applications, and nonlinear systems. Since 1997, he has been a Professor at the University of Toulouse (France). From 1997 to 2004, he was with the Acoustics lab of that University. Since 2004, he has been with the Astrophysics lab in Toulouse, which is part of the University and of the French National Center for Scientific Research (CNRS). His current major research interests include signal and image processing, higher-order statistics, time-frequency analysis, neural networks, and especially Blind Source Separation methods (including Independent or Sparse Component Analysis) and their applications to remote sensing, astrophysics, quantum information processing, acoustics and communication/electromagnetic signals.

Shahram Hosseini was born in Shiraz, Iran, in 1968. He received the B.Sc. and M.Sc. degrees in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 1991 and 1993, respectively, and the Ph.D. degree in signal processing from the Institut National Polytechnique, Grenoble, France, in 2000. He is currently an Associate Professor at the Université Paul Sabatier Toulouse 3, Toulouse, France. His research interests include BSS, artificial neural networks, and adaptive signal processing.

Philippe Déliot obtained a physical engineer grade in 1987. He has been working at ONERA (French Aerospace Lab) since 1989 in the applied and theoretical optics department. He was firstly in charge of infrared sensor conception and realization. Since 2000, he has been working on subjects in relation with image quality in remote sensing and particularly with calibration of airborne system (ONERA-IGN-CNES PELICAN) and development of defocusing correction method for sensor after launch (SPOT, PLEIADES). He is currently involved in hyperspectral acquisition and calibration in lab and airborne system implementation (SETHI). Its main interests are in relation with hyperspectral image processing and its application to security and environment.

Xavier Briottet Xavier Briottet received his PhD degree in Electonics from the ”Ecole Nationale Supérieure de l’Aéronautique et de l’Espace” (Toulouse, France). In 1987, he became an assistant professor in signal and image processing at the ”Ecole Nationale Supérieure d’Electricité”. Since 1988, he is working in remote sensing at ONERA in the optic department (Toulouse, France). He received his ”Habilitation à Diriger les Recherches” from Paul Sabatier University (Toulouse, France) in 1999. He is now Director of Research and the advisor of the scientific politic at DOTA. His current research interests are on 3D radiative transfer in urban area, shadow detection, atmosphere, unmixing in the reflective and thermal domain, using high spatial hyperspectral imagery and lidar. He is one of the 4 PI of HYPXIM hyperspectral space mission.

Methods		Linear		Linear ext		Grd LQ		Grd-Newt LQ		Mult LQ	
		init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2
SAM_{OUT} (rad)	mean	0.07	0.07	0.09	0.08	0.10	0.10	0.04	0.04	**0.03**	**0.03**
	std	0.09	0.09	0.06	0.06	0.07	0.07	0.04	0.05	0.03	0.03
RMSE	mean	0.14	0.14	0.19	0.19	0.28	0.28	0.09	0.09	**0.09**	**0.09**
	std	0.13	0.12	0.11	0.11	0.09	0.09	0.05	0.07	0.06	0.06

TABLE I
Case M = 2 materials - Performance - Protocol 1

Methods		Linear		Linear ext		Grd LQ		Grd-Newt LQ		Mult LQ	
		init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2
SAM_{OUT} (rad)	mean	0.07	0.07	0.08	0.07	0.10	0.10	0.04	0.04	**0.03**	**0.03**
	std	0.10	0.09	0.06	0.05	0.06	0.05	0.04	0.05	0.03	0.03
RMSE	mean	0.13	0.13	0.37	0.34	0.25	0.25	0.08	0.09	**0.07**	**0.07**
	std	0.13	0.13	0.22	0.20	0.08	0.08	0.05	0.07	0.06	0.06

TABLE II
Case M = 2 materials - Performance - Protocol 2

Methods		Linear		Linear ext		Grd LQ		Grd-Newt LQ		Mult LQ	
Err_{tot} (in %)	mean	4.9	4.9	0.2	0.2	8.7	8.6	0.7	0.9	0.5	0.5
CPU time (s)	mean	2.6	2.6	1.7	1.7	4.3	4.4	1.7	1.2	1.3	1.3

TABLE III
Case M = 2 materials - Convergence precision and speed

Methods		Linear		Linear ext		Grd LQ		Grd-Newt LQ		Mult LQ	
		init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2
SAM_{OUT} (rad)	mean	0.19	0.19	0.15	0.15	0.14	0.14	0.16	0.13	**0.12**	**0.12**
	std	0.11	0.11	0.06	0.06	0.06	0.06	0.12	0.09	0.06	0.06
RMSE	mean	0.20	0.20	0.20	0.20	0.23	0.23	0.17	0.16	**0.15**	**0.15**
	std	0.07	0.06	0.05	0.05	0.05	0.05	0.11	0.08	0.07	0.07

TABLE IV
Case M = 3 materials - Performance - Protocol 1

Methods		Linear		Linear ext		Grd LQ		Grd-Newt LQ		Mult LQ	
		init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2	init 1	init 2
SAM_{OUT} (rad)	mean	0.18	0.18	0.14	0.14	0.12	0.12	0.11	0.10	**0.09**	**0.09**
	std	0.11	0.10	0.05	0.05	0.04	0.04	0.07	0.06	0.04	0.04
RMSE	mean	0.23	0.23	0.43	0.43	0.26	0.25	0.18	0.17	**0.16**	**0.16**
	std	0.09	0.09	0.11	0.10	0.07	0.07	0.10	0.08	0.09	0.08

TABLE V
Case M = 3 materials - Performance - Protocol 2

Methods		Linear		Linear ext		Grd LQ		Grd-Newt LQ		Mult LQ	
Err_{tot} (in %)	mean	6.0	6.0	0.3	0.3	4.7	4.6	3.1	2.0	0.4	0.4
CPU time (s)	mean	6.4	6.2	1.1	1.2	5.6	5.5	6.6	4.3	2.6	2.6

TABLE VI
Case M = 3 materials - Convergence precision and speed