Démélange d'images hyperspectrales à l'aide de la NMF en-ligne avec contrainte de dispersion minimale

Ludivine NUS, Sebastian MIRON, David BRIE *

CRAN, Université de Lorraine, CNRS, Vandœuvre-lès-Nancy, France
prénom.nom@univ-lorraine.fr

Résumé – Nous proposons une nouvelle méthode de démélange *en-ligne* d'images hyperspectrales fondée sur une approche de type ADMM (*Alternating Direction Method of Multipliers*), particulièrement bien adaptée aux systèmes d'imagerie *pushbroom*. L'algorithme proposé présente une convergence plus rapide et une complexité de calcul plus faible par rapport aux algorithmes fondés sur des règles de mise à jour multiplicatives. En raison du caractère généralement mal posé du problème de démélange, nous intégrons dans la méthode une contrainte de dispersion minimale des *endmembers* ; cette contrainte peut être interprétée comme une relaxation convexe de la contrainte de volume minimal. Des tests sur des données réelles permettent d'illustrer les bonnes performances de la méthode proposée comparées à l'état de l'art.

Abstract – We propose a new method for *on-line* hyperspectral images unmixing based on an ADMM (*Alternating Direction Method of Multipliers*) approach, particularly well-adapted to *pushbroom* imaging systems. The proposed algorithm presents a faster convergence and a lower computational complexity compared to the algorithms based on multiplicative update rules. Due to the general ill-posed nature of the unmixing problem, we impose a minimal *endmembers* dispersion constraint; this constraint can be interpreted as a convex relaxation of the minimal volume constraint. Real data tests illustrate the good performance of the proposed method compared to the state of the art.

1 Introduction

Le démélange hyperspectral consiste à décomposer une image (scène) observée à plusieurs longueurs d'onde en une somme de termes de rang 1 correspondant au produit matriciel des *endmembers* par leurs abondances respectives, qui sont les grandeurs d'intérêt. De nombreuses méthodes ont été proposées pour le démélange hyperspectral (voir *e.g.,* [1]). En raison de la nature non-négative des données, la *Factorisation en Matrices Non-négatives (Non-negative Matrix Factorization (NMF))* a été largement utilisée pour ce type de problème. Néanmoins, cette méthode exige que l'intégralité des données réside dans la mémoire de la machine lors du traitement, ce qui, pour des données de grande dimension, peut devenir prohibitif en ce qui concerne la capacité de stockage et le temps de calcul.

Afin de minimiser les besoins en mémoire liés aux données hyperspectrales de grande dimension, une série d'algorithmes NMF *en-ligne* a été proposée. L'objectif est de traiter un échantillon à la fois, et de mettre à jour de façon itérative les *endmembers* et les abondances, sans stocker les échantillons passés. La méthode *Incremental NMF (INMF)* [4] considère que la matrice des *endmembers* n'évolue que très peu entre deux échantillons consécutifs, ce qui permet de diminuer le coût de calcul. C'est à ce jour l'hypothèse la plus largement adoptée dans les algorithmes NMF en-ligne. Cependant, la contrainte de non-négativité seule ne permet pas de garantir l'unicité de la solution. Pour restreindre le domaine des solutions possibles, plusieurs méthodes NMF en-ligne sous contrainte ont vu le jour telles que [10, 7]. Récemment, nous avons proposé l'algorithme *Minimal Volume Simplex-NMF (MVS-NMF)* [6] qui impose une contrainte de volume minimal sur la matrice des *endmembers*. Cependant, fondé sur des règles de mise à jour multiplicatives [5], cet algorithme souffre d'une convergence lente et donc d'un temps de calcul conséquent et présente des risques d'instabilités numériques liées à la concavité de la contrainte de volume minimal.

Ainsi, dans cet article, nous introduisons un nouvel algorithme pour le démélange en-ligne d'images hyperspectrales, spécialement conçu pour les systèmes d'acquisition *pushbroom*. Les originalités de cet algorithme par rapport à l'état de l'art et en particulier aux méthodes proposées dans [6, 9] sont les suivantes : i) l'ajout d'une *Contrainte de Dispersion Minimale (Minimal Dispersion Constraint (MDC))* [8] sur la matrice des *endmembers*, qui peut être interprétée comme une relaxation convexe de la contrainte de volume minimal utilisée dans [6] ; ii) l'utilisation de l'optimisation ADMM dans le contexte du démélange en-ligne de données hyperspectrales. Dans la suite, nous désignons cette nouvelle approche comme étant l'algorithme *MDC-ADMM en-ligne*.

Cet communication est organisée de la façon suivante : la section 2 est réservée à la présentation de l'algorithme proposé, dont la convergence et la complexité de calcul sont étudiées en section 3. La section 4 présente des résultats obtenus sur une image hyperspectrale réelle. Enfin, les conclusions et les perspectives de ce travail sont élaborées dans la section 5.

*Nous bénéficions du soutien de l'ANR-OPTIFIN (Agence Nationale de la Recherche-OPTimisation des FINitions) ainsi que du projet CNRS Imag'in ALOHA.

2 L'approche proposée

2.1 Représentation des données acquises à l'aide d'un système *pushbroom*

L'algorithme de démélange spectral proposé dans cet article est spécialement conçu pour les systèmes d'acquisition de type *pushbroom*. La principale caractéristique de ces systèmes d'imagerie est l'acquisition de l'image hyperspectrale tranche par tranche, séquentiellement dans le temps. Chaque tranche (représentée en pointillé sur la figure 1) est une image de dimensions $n_x \times n_\lambda$, où n_x désigne la dimension spatiale (une ligne de la scène) et n_λ la dimension spectrale (nombre de longueurs d'onde). Le cube de données est acquis par balayage selon la direction spatiale OY. L'algorithme de démélange en-ligne que

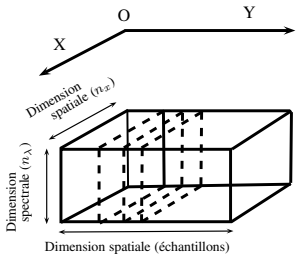


FIGURE 1 – Acquisition des données *pushbroom*

nous proposons est fondé sur le modèle NMF. Pour une matrice non-négative $\mathbf{X} \in \mathbb{R}_+^{n_\lambda \times n_x}$, la NMF consiste à estimer deux matrices non-négatives $\mathbf{S} \in \mathbb{R}_+^{n_\lambda \times r}$ et $\mathbf{A} \in \mathbb{R}_+^{r \times n_x}$ avec $r \leq \min(n_\lambda, n_x)$, tel que [5] : $\mathbf{X} \approx \mathbf{SA}$. En imagerie hyperspectrale, les colonnes n_x de $\mathbf{X}$ représentent les échantillons enregistrés aux longueurs d'onde n_λ. $\mathbf{S}$ est une matrice contenant les r *endmembers* normalisés et $\mathbf{A}$ est une matrice contenant sur ses colonnes les abondances relatives à chaque échantillon. Le principe de la méthode en-ligne proposée est de mettre à jour alternativement les matrices des *endmembers* et des abondances lorsqu'un nouvel échantillon arrive. Afin de représenter les données sous forme matricielle, l'image hyperspectrale est dépliée, comme illustrée sur la figure 2. La matrice $\tilde{\mathbf{X}}^{(1)} = \mathbf{X}^{(1)}$

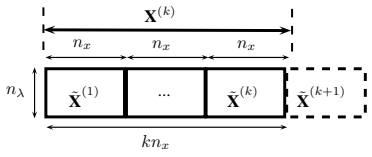


FIGURE 2 – Image hyperspectrale dépliée

fait référence à la première tranche de l'image hyperspectrale et $\tilde{\mathbf{X}}^{(k)}$ à la $k^{ème}$ tranche. L'ensemble des données à l'instant $k+1$, $\mathbf{X}^{(k+1)}$, peut s'écrire comme la concaténation des k premiers échantillons avec le nouvel échantillon à l'instant $k+1$, c'est à dire $\mathbf{X}^{(k+1)} = \left[\mathbf{X}^{(k)}\ \tilde{\mathbf{X}}^{(k+1)}\right]$. De même, nous définissons $\mathbf{S}^{(k+1)} = \left[\mathbf{S}^{(k)}\ \tilde{\mathbf{S}}^{(k+1)}\right]$ et $\mathbf{A}^{(k+1)} = \left[\mathbf{A}^{(k)}\ \tilde{\mathbf{A}}^{(k+1)}\right]$.

2.2 L'algorithme MDC-ADMM en-ligne

Afin d'estimer $\tilde{\mathbf{S}}^{(k+1)}$ et $\tilde{\mathbf{A}}^{(k+1)}$, nous considérons la fonction coût suivante :

$$\begin{aligned}\mathcal{J}^{(k+1)}\left(\tilde{\mathbf{S}}^{(k+1)}, \tilde{\mathbf{A}}^{(k+1)}\right) &= \alpha \sum_{\ell=1}^{k} \left\|\tilde{\mathbf{X}}^{(\ell)} - \tilde{\mathbf{S}}^{(\ell)}\tilde{\mathbf{A}}^{(\ell)}\right\|_F^2 \\ &+(1-\alpha)\left\|\tilde{\mathbf{X}}^{(k+1)} - \tilde{\mathbf{S}}^{(k+1)}\tilde{\mathbf{A}}^{(k+1)}\right\|_F^2 \\ &+\mu\ \mathrm{trace}\left(\tilde{\mathbf{S}}^{(k+1)}\mathbf{P}\tilde{\mathbf{S}}^{(k+1)T}\right), \end{aligned} \quad (1)$$

où le coefficient α ($0 \leq \alpha \leq 1$) ajoute une capacité de suivi à l'algorithme. Ainsi, les estimations à l'instant $k+1$ dépendent de celles à l'instant k. Selon [2], nous supposons que les *endmembers* ne varient que légèrement entre deux échantillons consécutifs, c'est-à-dire $\tilde{\mathbf{S}}^{(k+1)} \approx \tilde{\mathbf{S}}^{(k)}$. Le terme $\mu\ \mathrm{trace}\left(\tilde{\mathbf{S}}^{(k+1)}\mathbf{P}\,\tilde{\mathbf{S}}^{(k+1)T}\right)$ pénalise la somme des distances entre les *endmembers* et leur barycentre, avec la matrice $\mathbf{P}$ donnée par $\mathbf{P} = \mathbf{I} - \frac{1}{r}\mathbf{1}_r\mathbf{1}_r^T$ ($\mathbf{I}$ est la matrice identité et $\mathbf{1}_r$, un vecteur de dimension $r \times 1$ ne contenant que des 1). Cette contrainte de dispersion minimale possède des propriétés intéressantes d'un point de vue optimisation car c'est une fonction convexe ; elle permet une mise à jour explicite de la matrice des *endmembers* (cf. **Algorithme 1**). Afin de minimiser la fonction coût (1), nous adoptons une approche de type ADMM. À cette fin, nous introduisons deux variables auxiliaires $\tilde{\mathbf{U}}$ et $\tilde{\mathbf{V}}$ et considérons le problème équivalent à (1) suivant :

$$\begin{aligned}&\underset{\tilde{\mathbf{S}}^{(k+1)},\tilde{\mathbf{A}}^{(k+1)},\tilde{\mathbf{V}}^{(k+1)},\tilde{\mathbf{U}}^{(k+1)}}{\text{minimiser}}\ \frac{1}{2}\alpha \sum_{\ell=1}^{k} \left\|\tilde{\mathbf{X}}^{(\ell)} - \tilde{\mathbf{S}}^{(k+1)}\tilde{\mathbf{A}}^{(\ell)}\right\|_F^2 \\ &+\frac{1}{2}(1-\alpha)\left\|\tilde{\mathbf{X}}^{(k+1)} - \tilde{\mathbf{S}}^{(k+1)}\tilde{\mathbf{A}}^{(k+1)}\right\|_F^2 + \mu\ \mathrm{trace}\left(\tilde{\mathbf{S}}^{(k+1)}\mathbf{P}\tilde{\mathbf{S}}^{(k+1)T}\right) \\ &+\mathbb{I}_{R^+}\left(\tilde{\mathbf{V}}^{(k+1)}\right) + \mathbb{I}_{R^+}\left(\tilde{\mathbf{U}}^{(k+1)}\right), \\ &\text{sous contraintes } \tilde{\mathbf{S}}^{(k+1)} = \tilde{\mathbf{U}}^{(k+1)} \text{ et } \tilde{\mathbf{A}}^{(k+1)} = \tilde{\mathbf{V}}^{(k+1)}. \end{aligned} \quad (2)$$

$\mathbb{I}_{\mathbb{R}_+}$ représente la fonction indicatrice de $\mathbb{R}_+$ assurant la positivité des abondances et des *endmembers*. Par convenance algorithmique, nous utilisons la version normalisée de l'optimisation ADMM [2] dans laquelle les termes linéaires et quadratiques sont combinés dans le Lagrangien augmenté et les variables duales sont mises à l'échelle. L'expression du Lagrangien augmenté $\mathcal{L}$ correspondant au problème (2) est donnée par l'équation (3), où $\rho > 0$ est un paramètre qui contrôle la vitesse de convergence de la méthode. $\tilde{\mathbf{\Pi}}^{(k+1)}$ et $\tilde{\mathbf{\Lambda}}^{(k+1)}$ sont les versions mises à l'échelle des variables duales.

$$\begin{aligned}&\mathcal{L}\left(\tilde{\mathbf{A}}^{(k+1)}, \tilde{\mathbf{S}}^{(k+1)}, \tilde{\mathbf{V}}^{(k+1)}, \tilde{\mathbf{U}}^{(k+1)}, \tilde{\mathbf{\Pi}}^{(k+1)}, \tilde{\mathbf{\Lambda}}^{(k+1)}\right) = \frac{1}{2}\alpha \sum_{\ell=1}^{k} \left\|\tilde{\mathbf{X}}^{(\ell)} - \tilde{\mathbf{S}}^{(k+1)}\tilde{\mathbf{A}}^{(\ell)}\right\|_F^2 + \frac{1}{2}(1-\alpha)\left\|\tilde{\mathbf{X}}^{(k+1)} - \tilde{\mathbf{S}}^{(k+1)}\tilde{\mathbf{A}}^{(k+1)}\right\|_F^2 \\ &+\mu\ \mathrm{trace}\left(\tilde{\mathbf{S}}^{(k+1)}\mathbf{P}\tilde{\mathbf{S}}^{(k+1)T}\right) + \frac{\rho}{2}\left\|\tilde{\mathbf{A}}^{(k+1)} - \tilde{\mathbf{V}}^{(k+1)} + \tilde{\mathbf{\Pi}}^{(k+1)}\right\|_F^2 - \frac{\rho}{2}\left\|\tilde{\mathbf{\Pi}}^{(k+1)}\right\|_F^2 + \frac{\rho}{2}\left\|\tilde{\mathbf{S}}^{(k+1)} - \tilde{\mathbf{U}}^{(k+1)} + \tilde{\mathbf{\Lambda}}^{(k+1)}\right\|_F^2 - \frac{\rho}{2}\left\|\tilde{\mathbf{\Lambda}}^{(k+1)}\right\|_F^2 \\ &+\mathbb{I}_{\mathbb{R}_+}\left(\tilde{\mathbf{V}}^{(k+1)}\right) + \mathbb{I}_{\mathbb{R}_+}\left(\tilde{\mathbf{U}}^{(k+1)}\right). \end{aligned} \quad (3)$$

L'optimisation ADMM minimise alternativement le Lagrangien augmenté (3) par rapport à $\tilde{\mathbf{A}}^{(k+1)}$, $\tilde{\mathbf{S}}^{(k+1)}$, $\tilde{\mathbf{V}}^{(k+1)}$ et $\tilde{\mathbf{U}}^{(k+1)}$ et met ensuite à jour les variables duales $\tilde{\mathbf{\Pi}}^{(k+1)}$ et $\tilde{\mathbf{\Lambda}}^{(k+1)}$. L'**Algorithme 1** résume l'approche proposée. L'algorithme comprend deux boucles principales : une boucle externe qui produit des estimations de l'ensemble des paramètres pour chaque nouvel échantillon et une boucle interne qui permet d'affiner itérativement les estimations en utilisant un nombre d'itérations fixe (N_{iter}).

Algorithme 1 MDC-ADMM en-ligne

Entrées : $\mathbf{X}$; $\mathbf{P}$; r ; α ; μ ; ρ ; N_{iter} ;
Initialisation : $k = 0$; $\mathbf{N} = \text{zeros}(n_\lambda, r)$; $\mathbf{M} = \text{zeros}(r, r)$; $\tilde{\mathbf{S}} = \text{rand}(n_\lambda, r)$; $\tilde{\mathbf{V}} = \text{zeros}(r, n_x)$; $\tilde{\mathbf{U}} = \text{zeros}(n_\lambda, r)$; $\tilde{\mathbf{\Pi}} = \text{zeros}(r, n_x)$; $\tilde{\mathbf{\Lambda}} = \text{zeros}(n_\lambda, r)$; $\mathbf{A} = [\,]$; $\mathbf{S} = [\,]$;
Sorties : $\mathbf{A}$; $\mathbf{S}$;
Tant que Nouvel échantillon $k+1$ disponible **faire**
  $\tilde{\mathbf{X}} = \tilde{\mathbf{X}}^{(k+1)}$;
  $t = 1$;
  Tant que $t < N_{iter}$ **faire**
    $\tilde{\mathbf{A}} = \left((1-\alpha)\tilde{\mathbf{S}}^T\tilde{\mathbf{S}} + \rho\mathbf{I}\right)^{-1}\left((1-\alpha)\tilde{\mathbf{S}}^T\tilde{\mathbf{X}} + \rho\left(\tilde{\mathbf{V}} - \tilde{\mathbf{\Pi}}\right)\right)$;
    $\tilde{\mathbf{V}} = \max\left(\mathbf{0}, \tilde{\mathbf{A}} + \tilde{\mathbf{\Pi}}\right)$;
    $\tilde{\mathbf{\Pi}} \leftarrow \tilde{\mathbf{\Pi}} + \tilde{\mathbf{A}} - \tilde{\mathbf{V}}$;
    $\tilde{\mathbf{N}} = \alpha\mathbf{N} + (1-\alpha)\left(\tilde{\mathbf{X}}\tilde{\mathbf{A}}^T\right)$;
    $\tilde{\mathbf{M}} = \alpha\mathbf{M} + (1-\alpha)\left(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\right)$;
    $\tilde{\mathbf{S}} = \left(\tilde{\mathbf{N}} + \rho\left(\tilde{\mathbf{U}} - \tilde{\mathbf{\Lambda}}\right)\right)\left(\tilde{\mathbf{M}} + \rho\mathbf{I} + 2\mu\mathbf{P}\right)^{-1}$;
    $\tilde{\mathbf{U}} = \max\left(\mathbf{0}, \tilde{\mathbf{S}} + \tilde{\mathbf{\Lambda}}\right)$;
    $\tilde{\mathbf{\Lambda}} \leftarrow \tilde{\mathbf{\Lambda}} + \tilde{\mathbf{S}} - \tilde{\mathbf{U}}$;
    $t \leftarrow t + 1$;
  Fin tant que
  $\mathbf{N} = \tilde{\mathbf{N}}$;
  $\mathbf{M} = \tilde{\mathbf{M}}$;
  $\mathbf{A} = \left[\mathbf{A}\ \tilde{\mathbf{A}}\right]$; $\mathbf{S} = \left[\mathbf{S}\ \tilde{\mathbf{S}}\right]$;
Fin tant que

3 Analyse de l'algorithme

3.1 Convergence

En raison de la non-convexité du problème (2), nous fournissons dans cette section un résultat partiel sur la convergence de l'algorithme proposé, en montrant que tout point généré par une séquence d'itérations est un point stationnaire qui satisfait les conditions de Karush-Kuhn-Tucker (KKT) [3]. Pour cette analyse, nous considérons le comportement de l'algorithme pour un seul échantillon de l'image ; en théorie, sous l'hypothèse que $\tilde{\mathbf{S}}^{(k+1)} \approx \tilde{\mathbf{S}}^{(k)}$, le problème de convergence ne se pose plus lors de l'arrivée d'un nouvel échantillon. Pour simplifier les notations, nous regroupons toutes les variables à estimer dans une matrice $\mathbf{W} := (\tilde{\mathbf{A}}, \tilde{\mathbf{S}}, \tilde{\mathbf{V}}, \tilde{\mathbf{U}}, \tilde{\mathbf{\Pi}}, \tilde{\mathbf{\Lambda}})$. En se référant à [3], un point $\mathbf{W}$ est un point KKT du problème (2) si :

$$(1-\alpha)\tilde{\mathbf{S}}^T\tilde{\mathbf{S}}\tilde{\mathbf{A}} - (1-\alpha)\tilde{\mathbf{S}}^T\tilde{\mathbf{X}} + \tilde{\mathbf{\Pi}} = \mathbf{0}, \tag{4a}$$

$$\tilde{\mathbf{S}}\tilde{\mathbf{M}} - \tilde{\mathbf{N}} + 2\mu\tilde{\mathbf{S}}\mathbf{P} + \tilde{\mathbf{\Lambda}} = \mathbf{0}, \tag{4b}$$

$$\tilde{\mathbf{A}} - \tilde{\mathbf{V}} = \mathbf{0}, \tag{4c}$$

$$\tilde{\mathbf{S}} - \tilde{\mathbf{U}} = \mathbf{0}, \tag{4d}$$

$$\tilde{\mathbf{\Pi}} \leq \mathbf{0} \leq \tilde{\mathbf{V}}, \tilde{\mathbf{\Pi}} \odot \tilde{\mathbf{V}} = \mathbf{0}, \tag{4e}$$

$$\tilde{\mathbf{\Lambda}} \leq \mathbf{0} \leq \tilde{\mathbf{U}}, \tilde{\mathbf{\Lambda}} \odot \tilde{\mathbf{U}} = \mathbf{0}, \tag{4f}$$

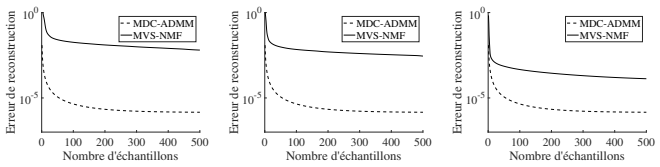
où $\odot$ représente le produit matriciel de Hadamard. Soit $\mathbf{W}_t := (\tilde{\mathbf{A}}_t, \tilde{\mathbf{S}}_t, \tilde{\mathbf{V}}_t, \tilde{\mathbf{U}}_t, \tilde{\mathbf{\Pi}}_t, \tilde{\mathbf{\Lambda}}_t)$, l'estimation de $\mathbf{W}$ à l'itération t dans l'**Algorithme 1**. Alors, la relation suivante peut s'écrire entre $\tilde{\mathbf{A}}_t$ et $\tilde{\mathbf{A}}_{t+1}$:

$$\begin{aligned}&\left((1-\alpha)\tilde{\mathbf{S}}_t^T\tilde{\mathbf{S}}_t + \rho\mathbf{I}\right)\left(\tilde{\mathbf{A}}_{t+1} - \tilde{\mathbf{A}}_t\right)\\ &= -\left((1-\alpha)\tilde{\mathbf{S}}_t^T\tilde{\mathbf{S}}_t\tilde{\mathbf{A}}_t - (1-\alpha)\tilde{\mathbf{S}}_t^T\tilde{\mathbf{X}} + \rho\left(\tilde{\mathbf{A}}_t - \tilde{\mathbf{V}}_t\right) + \rho\tilde{\mathbf{\Pi}}_t\right).\end{aligned} \tag{5}$$

Supposons que l'algorithme atteint un point stationnaire, *i.e.,* $\mathbf{W}_{t+1} = \mathbf{W}_t = \mathbf{W}^\star \Rightarrow \tilde{\mathbf{A}}_{t+1} = \tilde{\mathbf{A}}_t = \tilde{\mathbf{V}}_t = \tilde{\mathbf{A}}^\star$. En remplaçant dans (5), on obtient $(1-\alpha)\tilde{\mathbf{S}}^{\star T}\tilde{\mathbf{S}}^\star\tilde{\mathbf{A}}^\star - (1-\alpha)\tilde{\mathbf{S}}^{\star T}\tilde{\mathbf{X}} + \rho\tilde{\mathbf{\Pi}}^\star = \mathbf{0}$. En utilisant un raisonnement similaire pour les autres paramètres de $\mathbf{W}$, on montre que les quatre premières équations relatives aux conditions KKT (4) sont satisfaites pour tout point limite : $\mathbf{W}^\star := \left(\tilde{\mathbf{A}}^\star, \tilde{\mathbf{S}}^\star, \tilde{\mathbf{V}}^\star, \tilde{\mathbf{U}}^\star, \tilde{\mathbf{\Pi}}^\star, \tilde{\mathbf{\Lambda}}^\star\right)$. Pour démontrer la condition (4e), nous pouvons écrire : $\max\left(\mathbf{0}, \tilde{\mathbf{A}}^\star + \tilde{\mathbf{\Pi}}^\star\right) = \tilde{\mathbf{V}}^\star$. Si $\tilde{\mathbf{A}}^\star = \tilde{\mathbf{V}}^\star = \mathbf{0}$, alors $\max\left(\mathbf{0}, \tilde{\mathbf{\Pi}}^\star\right) = \mathbf{0}$, ce qui mène à $\tilde{\mathbf{\Pi}}^\star < \mathbf{0}$. Si $\tilde{\mathbf{A}}^\star = \tilde{\mathbf{V}}^\star > \mathbf{0}$, alors $\tilde{\mathbf{\Pi}}^\star = \mathbf{0}$. Le même type de raisonnement s'applique à (4f). Nous avons donc montré que pour le problème (2), tout point stationnaire $\mathbf{W}^\star$ est un point qui satisfait les conditions de KKT. Dans notre cas, il s'agit seulement d'une condition nécessaire d'optimalité ; néanmoins, cela constitue un résultat partiel de convergence.

3.2 Comportement transitoire

Les vitesses de convergence de l'algorithme MDC-ADMM et MVS-NMF en-ligne sont comparées sur des données hyperspectrales simulées de dimension $119 \times 40 \times 500$. 119 correspond au nombre de longueurs d'onde et 40×500 aux dimensions spatiales. L'image est composée de trois *endmembers* qui ne varient pas dans le temps. Les deux algorithmes sont initialisés avec la même matrice des *endmembers* ; l'initialisation est aléatoire et choisie de manière à ce que les deux algorithmes convergent vers la même solution. La valeur du paramètre α est fixée à 0.99. Les valeurs des paramètres de réglage μ et ρ sont déterminées de manière empirique. L'erreur de reconstruction en fonction du nombre d'échantillons est étudiée pour différentes valeurs de N_{iter} et les résultas sont présentés sur la figure 3. Pour toutes valeurs de N_{iter}, l'algorithme MDC-ADMM en-ligne converge plus rapidement que l'algorithme MVS-NMF en-ligne. À partir de 50 itérations environ, l'algorithme génère déjà la bonne solution. Ces résultats montrent qu'il y a un réel intérêt à utiliser l'optimisation ADMM dans le cas de processus en-ligne.



(a) $N_{iter} = 50$ (b) $N_{iter} = 100$ (c) $N_{iter} = 200$

FIGURE 3 – Vitesses de convergence

3.3 Complexité de calcul

Nous évaluons la complexité de calcul de l'algorithme pour une seule tranche de l'image hyperspectrale et une seule itération et nous considérons uniquement les opérations de multiplication. La complexité de l'algorithme MDC-ADMM en-ligne est de l'ordre de $2(rn_xn_\lambda + (n_\lambda + n_x)r^2 + r^3)$; elle est plus faible que celle de l'algorithme MVS-NMF en-ligne [6] qui est de l'ordre de $2rn_xn_\lambda + (6n_\lambda + 2n_x)r^2 + (n_\lambda + n_x)r$.

4 Résultats expérimentaux

Afin d'évaluer les performances de démélange de notre algorithme sur des données réelles, nous avons utilisé l'image hyperspectrale (avec vérité terrain) *Jasper Ridge* et comparé les résultats avec ceux de la méthode MVS-NMF en-ligne [6]. Dans cette image, quatre principaux composants sont présents : arbre, eau, sol et route. Pour la comparison, deux critères de performance ont été utilisés : *Spectral Angle Distance (SAD)* pour les *endmembers*, et *Root Mean Square Error (RMSE)* pour les abondances. Pour les deux méthodes, les initialisations ont été générées aléatoirement et chaque expérience a été répétée 50 fois. Le tableau 1 regroupe les valeurs moyennes obtenues pour chaque méthode et la figure 4 représente les cartes d'abondances réelles et estimées (le « meilleur » résultat) par les deux algorithmes et pour chaque source.

	MDC-ADMM		MVS-NMF	
Endmembers	SAD	RMSE	SAD	RMSE
Arbre	0.0629	0.0384	0.0475	0.1328
Eau	0.2397	0.0344	0.6613	0.1142
Sol	0.0953	0.0851	0.1307	0.2128
Route	0.0535	0.0737	0.2201	0.2019
Moyenne	0.1128	0.0579	0.2649	0.1654

TABLE 1 – SAD et RMSE

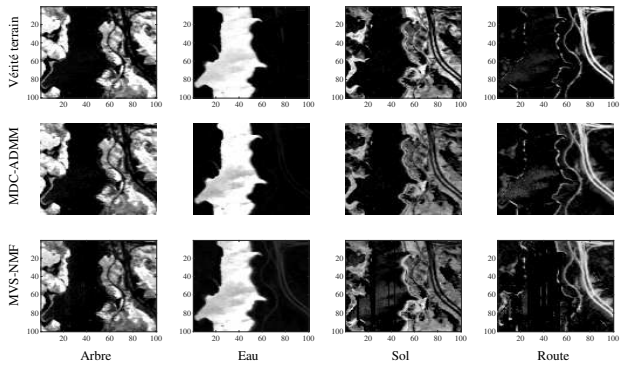


FIGURE 4 – Cartes d'abondances

Ces résultats montrent que l'algorithme MVS-NMF en-ligne génère des estimations de moins bonne qualité que la méthode MDC-ADMM en-ligne ; cela s'explique par le fait que les règles de mise à jour multiplicatives sont plus sensibles aux initialisations aléatoires, qui dégradent fortement la vitesse de convergence et la précision des solutions. En effet, cet algorithme requiert approximativement 4000 itérations pour converger vers une solution qui s'approche de la vérité terrain, lorsque l'initialisation le permet, ce qui représente un temps de traitement de l'ordre de 50 secondes. Pour la méthode MDC-ADMM en-ligne, 500 itérations suffisent pour un temps de traitement de l'ordre de 6 secondes.

5 Conclusions et perspectives

Nous avons proposé un nouvel algorithme (MDC-ADMM en-ligne) spécialement conçu pour le démélange de données hyperspectrales *pushbroom*. Les tests sur des données réelles ont montré que ce nouvel algorithme surpasse les méthodes de l'état de l'art fondées sur des règles de mise à jour multiplicatives, en ce qui concerne la vitesse de convergence, la robustesse face aux initialisations aléatoires et la qualité de l'estimation. L'ajout de la contrainte de dispersion minimale sur la matrice des *endmembers* permet de régulariser le problème et de stabiliser la solution. Néanmoins, les performances de cette méthode peuvent être sensibles au paramètre de réglage ρ. Ainsi, les travaux futurs s'orienteront vers le choix optimal de ce paramètre.

Références

[1] José M BIOUCAS-DIAS, Antonio PLAZA, Nicolas DOBIGEON, Mario PARENTE, Qian DU, Paul GADER et Jocelyn CHANUSSOT : Hyperspectral unmixing overview : Geometrical, statistical, and sparse regression-based approaches. *IEEE journal of selected topics in applied earth observations and remote sensing*, 5(2):354–379, 2012.

[2] Stephen BOYD, Neal PARIKH, Eric CHU, Borja PELEATO, Jonathan ECKSTEIN *et al.* : Distributed optimization and statistical learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[3] Stephen BOYD et Lieven VANDENBERGHE : *Convex optimization*. Cambridge university press, 2004.

[4] Serhat S BUCAK et Bilge GUNSEL : Incremental subspace learning via Non-negative Matrix Factorization. *Pattern recognition*, 42(5):788–797, 2009.

[5] Daniel D LEE et H Sebastian SEUNG : Algorithms for Non-negative Matrix Factorization. *In Advances in neural information processing systems*, pages 556–562, 2001.

[6] Ludivine NUS, Sebastian MIRON et David BRIE : On-line blind unmixing for hyperspectral pushbroom imaging systems. *In 2018 IEEE Statistical Signal Processing Workshop (SSP)*, pages 418–422. IEEE, 2018.

[7] Yi WU, Bin SHEN et Haibin LING : Visual tracking via on-line Non-negative Matrix Factorization. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(3):374–383, 2014.

[8] Yue YU, Shan GUO et Weidong SUN : Minimum distance constrained Non-negative Matrix Factorization for the endmember extraction of hyperspectral images. *In International Symposium on Multispectral Image Processing and Pattern Recognition*, pages 679015–679015. International Society for Optics and Photonics, 2007.

[9] Renbo ZHAO et Vincent YF TAN : On-line Non-negative Matrix Factorization with outliers. *In Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 2662–2666. IEEE, 2016.

[10] Guoxu ZHOU, Zuyuan YANG, Shengli XIE et Jun-Mei YANG : On-line blind source separation using incremental Non-negative Matrix Factorization with volume constraint. *IEEE transactions on neural networks*, 22(4):550–560, 2011.