Mises à jour multiplicatives dépliées pour la Factorisation en Matrices Non-négatives appliquée au démélange spectral

Christophe KERVAZO[1] Jérémy COHEN[2]

[1]LTCI, Télécom Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France

[2] CNRS, Univ Lyon, INSA-Lyon, Université Claude Bernard Lyon 1, UJM-Saint Etienne, Inserm, CREATIS UMR 5220, U1294, F-69621, LYON, France

Résumé – Le démélange hyperspectral (HSU), problème qui consiste à séparer des spectres de matériaux superposés dans une image hyperspectrale, a motivé le développement de nombreux algorithmes dans les deux dernières décennies. Ceux-ci peuvent être catégorisés en "basés modèles" et "apprentissage profond". Dans ce travail, nous proposons de marier les deux approches en explorant le paradigme du dépliement d'algorithmes. Notre contribution est double. Premièrement, nous proposons de revisiter l'algorithme NALMU que nous avons récemment proposé et qui déplie les mises à jour multiplicatives pour la factorisation en matrices non-négatives. Nous montrons que sous certaines hypothèses simplificatrices, NALMU minimise une fonction de coût explicite. Deuxièmement, nous proposons un nouvel algorithme déplié adaptatif, RALMU, qui améliore significativement les performances empiriques de NALMU. Les performances de ces méthodes sont comparées sur des jeux de données en astrophysique et en télédétection, et nous montrons que les méthodes dépliées proposées offrent de meilleures performances que les algorithmes classiques ainsi que d'autres méthodes dépliées de la littérature.

Abstract – HyperSpectral Unmixing (HSU), the problem of separating mixed spectra of overlapping materials in a hyperspectral image, has motivated dedicated algorithmic developments in the last two decades, which can be roughly categorized as "model-based" and "deep learning". This work utilizes the strengths of both approaches by exploring the deep unrolling paradigm. Our contribution is twofold. First, we revisit the Non-Adaptive Learned Multiplicative Updates (NALMU) algorithm, which is based on deep unrolling of the well-known Multiplicative Updates. We relate the NALMU to the minimization of an explicit cost function under some assumptions. Such guarantees are unique in the HSU field. Second, we propose a new unrolling-based algorithm coined RALMU, which overcomes NALMU shortcomings and considerably improves its practical performance. RALMU is tested on astrophysics and remote sensing datasets, exhibiting superior performances compared to other unrolled HSU algorithms and vanilla Multiplicative Updates.

1 Démélange hyperspectral

1.1 Contexte

La séparation aveugle de sources est une technique qui vise à extraire des signaux sources à partir de leurs mélanges, l'opérateur de mélange étant inconnu. Elle trouve des applications très diverses comme l'extraction d'information musicale et le réhaussement de la parole, l'imagerie médicale ainsi que le démélange spectral d'images hyperspectrales (HSU) [1] qui nous intéresse dans ce travail. La taille des pixels des images hyperspectrales forme un compromis avec la résolution spectrale de ces images ; par conséquent ils sont souvent suffisamment grands pour contenir plusieurs matériaux, et les spectres observés pour chaque pixels constituent un mélange des spectres individuels de chaque matériau.

La plupart des algorithmes de HSU reposent sur le modèle de mélange linéaire. Pour chaque pixel $\mathbf{V}[:,n] \in \mathbb{R}^M$ de l'image hyperspectrale, on suppose que

$$\mathbf{V}[:,n] \approx \mathbf{WH}[:,n]$$

où M est le nombre de bandes spectrales et $N = N_1 \times N_2$ est le nombre total de pixels. Les colonnes de la matrice $\mathbf{W} \in \mathbb{R}^{M\times R}$ sont appelées les signatures spectrales, et R représente le nombre de matériaux. Les colonnes de la matrice $\mathbf{H} \in \mathbb{R}^{R\times N}$ sont appelées les abondances et correspondent aux concentrations relatives (ici non normalisées) de chaque matériau pour chaque pixel. On peut également écrire ce modèle plus simplement comme $\mathbf{V} \approx \mathbf{WH}$. Des modèles non-linéaires plus complexes existent également, voir [8] et les références incluses. Nous supposerons dans la suite que l'erreur de modélisation suit une distribution normale centrée réduite élément par élément.

Le problème du HSU revient donc à estimer les matrices $\mathbf{W}$ et $\mathbf{H}$ à partir uniquement des données $\mathbf{V}$ et de la connaissance du nombre de composantes R. On utilise souvent l'hypothèse que les abondances et les signatures spectrales sont non-négatives, auquel cas le HSU devient un problème de factorisation en matrices non-négatives (NMF). Même avec ces contraintes, le HSU est cependant un problème inverse mal posé, qui a conduit au développement de nombreux algorithmes dans la littérature.

1.2 État de l'art

On peut résumer les approches pour le HSU en deux familles, les approches basées modèle, et les approches basées données. La première famille regroupe des méthodes géométriques telles, notamment les méthodes de recherche de pixels purs comme VCA [14, 13], et des approches génériques de NMF reposant sur une hypothèse de parcimonie [10] ou de volume minimal [11].

Les méthodes basées données reposent essentiellement sur l'utilisation de réseaux de neurones tels que des auto-encoders [16] mais mettent en oeuvre des décodeurs linéaire pour suivre le modèle de mélange linéaire. D'autres architectures ont été employées comme des transformers. Un des problèmes principaux de ces approches est le manque de données d'apprentissage étiquetées, et des modèles auto-supervisés ont donc été proposés [7, 5].

Afin de bénéficier de l'interprétabilité des méthodes basées modèle, mais de pouvoir malgré tout utiliser l'information contenue dans des données d'entrainement, [4] a proposé le paradigme des algorithmes dépliés. Un algorithme d'optimisation itératif est vu comme une cascade d'opérateurs non-linéaires ; on peut garder en mémoire les gradients d'une fonction de coût d'entrainement par rapport à certains hyperparamètres de cet algorithme afin de les optimiser [12]. Dans le contexte de la séparation de sources, il existe quelques instances du dépliement d'algorithmes dont [17, 2] basés sur un a priori de parcimonie et [6, 15, 9] basés sur la nonnégativité.

1.3 Contributions

Nous avons récemment proposé l'algorithme NALMU [9] qui déplie les itérations des mises à jour multiplicatives pour la NMF. Après une courte présentation de cet algorithme, nous montrons que sous certaines hypothèses, NALMU minimise une fonction de coût explicite. De plus, nous proposons un algorithme plus performant, RALMU, dont les hyperparamètres s'adaptent aux données d'entrée et prend d'avantage en compte la structure spatiale des images hyperspectrales. Nous comparons enfin ces méthodes à d'autres de l'état de l'art sur des données issues de la télédétection.

2 Algorithme NALMU

Dans cette section nous présentons l'algorithme «Non-Adaptative Learned Multiplicative Updates» (NALMU) proposé précédemment [9]. Après un court rappel sur les mises à jour multiplicatives (MU) et sur l'algorithme NALMU, nous présentons de nouvelles garanties théoriques sur la convergence de NALMU.

2.1 Mises à jour multiplicatives MU

L'algorithme MU a été proposé par Lee et Seung [10] il y a plus de vingt ans et reste un algorithme très populaire pour calculer des solutions à la NMF. Chaque matrice $\mathbf{W}$ et $\mathbf{H}$ est multipliée par un terme nonnégatif qui garantit la décroissance de la fonction de coût ainsi que la satisfaction des contraintes. Dans le cas de la norme de Frobenius comme fonction de perte, ces mises à jour prennent la forme

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{V}\mathbf{H}^T}{\mathbf{W}\mathbf{H}\mathbf{H}^T} \tag{1}$$

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\mathbf{V}}{\mathbf{W}^T\mathbf{W}\mathbf{H}}, \tag{2}$$

où $\odot$ est le produit de Hadamard (élément par élément). Notons que cet algorithme n'est pas état de l'art dans le cas Euclidien. De plus on peut aisément améliorer sa vitesse de convergence en répétant plusieurs fois chaque opération avant d'alterner la mise à jour des matrices.

2.2 MU déplié : NALMU

Intéressons nous ici uniquement au sous-problème qui consiste à estimer $\mathbf{W}$ à $\mathbf{H}$ fixé. Afin d'utiliser des données d'entraînement, NALMU introduit des hyperparamètres à apprendre dans l'étape (1) de l'algorithme MU. L'algorithme NALMU pour estimer $\mathbf{H}$ itère (1) et, à l'itération l, prend alors la forme

$$\mathbf{W}^{(l+1)} \leftarrow \mathbf{W}^{(l)} \odot \underline{\mathbf{A}_{\mathbf{W}}} \odot \frac{\mathbf{V}\mathbf{H}^T}{\mathbf{W}^{(l)}\mathbf{H}\mathbf{H}^T}. \tag{3}$$

La matrice $\underline{\mathbf{A}_{\mathbf{W}}}$ est fixe en fonction de l'itération l et ne dépend pas de l'image d'entrée $\mathbf{V}$. On définit la matrice $\underline{\mathbf{A}_{\mathbf{H}}}$ de façon similaire.

Remarque : Sans rentrer dans les détails de [9] par manque de place, dans notre proposition originale, NALMU est appliqué au problème NMF au sein d'un algorithme alterné mais sans apprentissage pour la matrice $\mathbf{H}$. La matrice $\underline{\mathbf{A}_{\mathbf{W}}}$ peut de plus dépendre de l'itération courante. Par ailleurs, une version adaptative ALMU est proposée qui paramètre $\underline{\mathbf{A}_{\mathbf{W}}}$ comme un réseau prenant en entrée une première estimation de $\mathbf{W}$ et $\mathbf{H}$. Nous proposons dans la section 3 une amélioration de ALMU.

2.3 NALMU minimise une fonction de perte explicite

Une des forces des algorithmes dépliés est leur interprétabilité, de par leur architecture comparable à celle des algorithmes classiques (comme ceux de NMF). Pourtant dans le contexte de la NMF et notamment des algorithmes MU dépliés, à notre connaissance il n'est pas évident que l'algorithme déplié minimise vraiment une fonction de coût ; il n'est pas évident non plus de savoir quelle fonction de coût serait minimisée. La base du problème provient du fait que l'algorithme MU n'a pas d'hyperparamètre que l'on pourrait apprendre et que l'on pourrait directement relier à une fonction de coût. La proposition suivante montre que, après apprentissage, NALMU revient à appliquer MU à une fonction de coût régularisée faisant intervenir les données d'apprentissage.

Proposition 2.1. *Soit* $\mathbf{v} := \mathbf{V}[m,:] \in \mathbb{R}^{1\times N}$ *la* m*-ième ligne de l'image* $\mathbf{V}$ *et* $\mathbf{w} := \mathbf{W}[m,:] \in \mathbb{R}^{1\times R}$ *la ligne correspondante dans* $\mathbf{W}$*. Pour une matrice diagonale* $\mathbf{\Lambda}_m \in \mathbb{R}^{R\times R}$ *on définit*

$$C(\mathbf{w};\mathbf{\Lambda}_m) = \frac{1}{2}\|\mathbf{w}\mathbf{H} - \mathbf{v}\|_2^2 - \mathbf{w}\mathbf{\Lambda}_m\mathbf{H}\mathbf{v}^T. \tag{4}$$

Alors les mises à jour de NALMU pour la matrice $\mathbf{W}$ *font toujours décroître le coût* $\sum_{m=1}^{M} C(\mathbf{W}[m,:];\mathbf{\Lambda}_m)$ *dans* $\mathbb{R}_+^{M\times R}$*, avec* $\mathbf{\Lambda}_m = \mathbf{Diag}(\underline{\mathbf{A}_{\mathbf{W}}}[m,:]) - \mathbf{I}$*, où* $\mathbf{Diag}(\underline{\mathbf{A}_{\mathbf{W}}}[m,:])$ *est une matrice diagonale dont les éléments diagonaux sont les coefficients du vecteur* $(\underline{\mathbf{A}_{\mathbf{W}}}[m,:])$.

La preuve de ce résultat est une application directe de la preuve de Févotte et Idier [3] pour la convergence de MU basée sur une procédure de majoration minimisation, la fonction de coût C étant celle de la NMF avec un terme linéaire séparable en $\mathbf{W}$. Ce terme linéaire peut être majoré par lui-même pour construire une majorante séparable du coût global C dont la minimisation est donné par la mise à jour de NALMU.

Interpétation : Lorsque la matrice $\underline{\mathbf{A}_{\mathbf{W}}}$ ne contient que des entrée à un, NALMU se réduit à MU avec une attache aux données euclidienne. Dans le cas géneral, puisque le terme de

régularisation $\mathbf{w}\mathbf{H}\mathbf{v}^T$ correspond à la corrélation entre une ligne de l'image hyperspectrale et sa reconstruction, NALMU a une reconstruction biaisée vers certaines données en fonction des poids $\mathbf{\Lambda}_m$; NALMU se comporte comme un MU pondéré avec des poids appris à partir de données d'entrainement.

Remarque : Ce résultat s'étend sans problème au cas de l'algorithme ALMU [9] qui satisfait les mêmes conditions.

3 NALMU adaptatif : RALMU

L'algorithme NALMU souffre de plusieurs défauts : 1) les poids $\underline{\mathbf{A}_\mathbf{W}}$ sont identiques quel que soit le jeu de données utilisé à l'inférence, 2) la matrice $\underline{\mathbf{A}_\mathbf{H}}$ est grande en général car elle contient autant d'éléments que les cartes d'abondances $\mathbf{H}$, ce qui ralentit la phase d'entrainement et pose des problèmes de stockage. Dans les faits, seule la matrice $\underline{\mathbf{A}_\mathbf{W}}$ a été entraînée dans nos travaux précédents. 3) Le fait que les colonnes de $\mathbf{H}$ soient des images n'est pas pris en compte, la méthode n'inclus aucune régularisation spatiale.

Afin de résoudre ces problèmes et d'améliorer les performances de NALMU en pratique, nous proposons dans ce travail l'algorithme «recursive ALMU» (RALMU), que l'on définit ici afin d'estimer les deux matrices $\mathbf{W}$ et $\mathbf{H}$:

$$\mathbf{W}^{(l+1)} \leftarrow \mathbf{W}^{(l+1)} \odot \underline{\mathbf{A}_\mathbf{W}}^{(l)}(\mathbf{W}^{(l)}) \odot \frac{\mathbf{V}\mathbf{H}^{(l)^T}}{\mathbf{W}^{(l)}\mathbf{H}^{(l)}\mathbf{H}^{(l)^T}} \quad (5)$$

$$\mathbf{H}^{(l+1)} \leftarrow \mathbf{H}^{(l)} \odot \underline{\mathbf{A}_\mathbf{H}}^{(l)}(\mathbf{H}^{(l)}) \odot \frac{\mathbf{W}^{(l+1)^T}\mathbf{V}}{\mathbf{W}^{(l+1)^T}\mathbf{W}^{(l+1)}\mathbf{H}^{(l)}}. \quad (6)$$

La différence principale entre RALMU et NALMU est que les poids $\underline{\mathbf{A}_\mathbf{W}}^{(l)}$ et $\underline{\mathbf{A}_\mathbf{H}}^{(l)}$ sont les sorties de fonctions des itérées précédentes, définies comme des petits réseaux de neurones, et dont les paramètres changent à chaque itération. Plus précisément, $\underline{\mathbf{A}_\mathbf{W}}^{(l)}(\cdot)$ est un perceptron multicouche tandis que $\underline{\mathbf{A}_\mathbf{H}}^{(l)}$ est un réseau de neurone convolutif sur les cartes d'abondances, permettant donc une prise en compte de leur cohérence spatiale. Afin de conserver la positivité des estimés, ces réseaux renvoient des valeurs positives en utilisant des non-linéarités ReLU.

3.1 Entraînement de RALMU

L'entrainement de RALMU s'effectue à partir de S images hyperspectrales $\mathbf{V}_S$ dont on connaît les abondances $\mathbf{H}_S$ et les spectres $\mathbf{W}_S$. Il n'est pas nécessaire que le modèle linéaire soit vérifié, l'algorithme RALMU une fois entraîné ne calculant pas exactement une NMF. La fonction de coût pour l'entrainement de RALMU est

$$\sum_{s=1}^{S}\sum_{l=1}^{L_{RALMU}} \alpha^{(l)} \left(\text{SAD}\left(\mathbf{W}_s^{(l)}, \mathbf{W}_s^*\right) + \text{NMSE}\left(\mathbf{H}_s^{(l)}, \mathbf{H}_s^*\right)\right) \quad (7)$$

où SAD et NMSE signifient respectivement «Spectral Angular Deviation» et «Normalized Mean Squared Error» et sont définis par

$$\text{SAD}\left(\mathbf{W}_s^{(l)}, \mathbf{W}_s^*\right) = \frac{1}{R}\sum_{r=1}^{R}\left\langle \mathbf{W}_s^{(l)}[:,r] \mid \mathbf{W}_s^*[:,r]\right\rangle,$$

$$\text{NMSE}\left(\mathbf{H}_s^{(l)}, \mathbf{H}_s^*\right) = \frac{\sum_{r=1}^{R}\sum_{n=1}^{N}\left(\mathbf{H}_s^{(l)}[r,n] - \mathbf{H}_s^*[r,n]\right)^2}{\sum_{r=1}^{R}\sum_{n=1}^{N}\mathbf{H}_s^*[r,n]^2}.$$

Dans la fonction de coût (7), les matrices $\mathbf{W}_s^{(l)}$ et $\mathbf{H}_s^{(l)}$ sont les estimées de RALMU après l itérations à partir de l'entrée $\mathbf{V}_s$. Les coefficients $\alpha^{(l)}$ pondèrent les estimations à chaque itération avec une importance qui augmente linéairement. Cette technique améliore les résultats et permet parfois d'éviter empiriquement le problème des gradients évanescents [17, 2].

4 Expériences numériques

Dans cette section nous comparons les performances des algorithmes NALMU et RALMU avec d'autres algorithmes de la littérature pour le HSU. Nous réalisons également une étude ablative afin de montrer l'impact de chaque élément nouveau dans RALMU sur les performances en démélange spectral. Pour toutes les expériences, RALMU est entraîné avec un pas de 10^{-5} et l'algorithme ADAM, les autres méthodes sont entraînées comme spécifiées dans les papiers d'origine.

4.1 Comparaisons sur le dataset Chandra

Nous avons construit les bases de donnée d'entraînement et de test à partir de spectres simulés réalistes correspondant à quatre composants : une émission synchrotron, une émission thermale, et deux émissions de fer avec du décalage dans le rouge dans le rouge. Nous référons à [2] pour plus détail sur le processus de simulation.

RALMU, entrainé avec 1000 époques, est comparé aux algorithmes suivants : deux algorithmes pour la NMF (l'implémentation de la descente par coordonnée (CD) de scikit-learn avec 10 000 itérations, et l'algorithme MU avec 100 000 itérations) et quatres algorithmes dépliés : Deep unrolled NMF (DNMF [15]), SNMF-Net [17], ainsi que NALMU et ALMU que nous avons proposés précedemment [9]. DNMF et SNMF-Net sont des algorithmes non-supervisés et ne sont donc pas entrainés à partir de la base de donnée d'entraînement.

La table 1 contient les résultats quantitatifs tandis que la figure 1 montre les résultats qualitatifs obtenus pour chaque méthode ; les temps de calculs rapportés sont obtenus avec un NVIDIA V100 GPU 32GB, ou bien pour les méthodes n'utilisant pas d'accélération graphique (MU, CD, SNMF), avec un 2,6 GHz Intel Core i7 CPU.

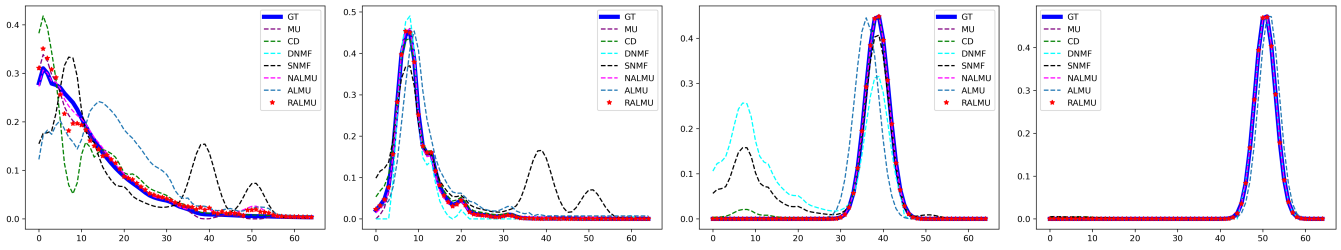
Résultats : L'algorithme RALMU obtient globalement de meilleurs résultats tout en fournissant un temps de calcul à l'inférence bas par rapport aux méthodes traditionnelles. Notons en particulier les bonnes performances d'estimation de RALMU pour les cartes d'abondances vis à vis de NALMU et RALMU. Ces algorithmes n'utilisent pas de poids entraînés sur les abondances.

4.2 Étude ablative

Nous testons d'abord l'impact de la matrice des poids $\underline{\mathbf{A}_\mathbf{H}}^{(l)}$ dans RALMU sur les performances. Pour ce faire, la mise à jour des abondances dans RALMU est réduite à une mise à jour MU classique. Les résultats sont présentés dans la table 1 sous la colonne «$\underline{\mathbf{A}_\mathbf{H}}$ abl». On observe que l'erreur d'estimation sur les abondances double, la placant à un niveau comparable à ce qu'obtient CD. L'erreur d'estimation sur les spectres diminue aussi, montrant l'importance d'apprendre des poids pour les spectres ainsi que les abondances.

TABLE 1 : Evaluations quantitatives sur les datasets Chandra (en haut) et Samson (en bas). Pour Chandra, les résultats sont moyennés sur 150 images hyperspectrales de l'ensemble de test. Le temps de calcul est présenté en seconde par HSI.

	MU	CD	DNMF	SNMF	NALMU	ALMU	RALMU	$\mathbf{A_H}$ abl.
SAD($\mathbf{W}^*, \mathbf{W}$)	0.989	0.989	0.701	0.777	0.994	0.995	**0.997**	0.992
NMSE($\mathbf{H}^*, \mathbf{H}$)	0.0853	0.0138	3.638	1.53×10^8	0.169	0.889	**0.0126**	0.0250
Time	235.87	103.27	426.17	5380.58	**0.118**	0.173	0.181	0.138
SAD($\mathbf{W}^*, \mathbf{W}$)	0.984	0.861	0.933	0.932	**0.999**	**0.999**	**0.999**	
NMSE($\mathbf{H}^*, \mathbf{H}$)	0.0703	0.328	340.915	0.648	0.422	0.687	**0.0172**	



(a) Synchrotron emission (b) Thermal emission (c) Iron 1 emission (d) Iron 2 emission

FIGURE 1 : Exemple de signatures spectrales normalisées obtenues pour le dataset Chandra. La majorité des algorithmes obtiennent de bons résultats pour l'émission fer 2, une différence plus significative est observable pour les émissions synchrotron et thermale.

4.3 Dataset Samson

Les performances de RALMU sont également évaluées sur l'image hyperspectrale Samson, qui contient 156 bandes spectrales et 95 pixels. On suppose en général que cette image contient trois signatures spectrales (terre, arbres et eau). Cette image dispose d'une pseudo-vérité terrain pour les abondances et signatures spectrales, que nous utilisons ici. RALMU est entrainé avec la procédure auto-supervisée décrite dans [5]. Globalement, RALMU obtient de meilleurs résultats que les autres méthodes dépliées, notamment pour l'estimation des cartes d'abondance.

Remerciements : Ce travail est financé partiellement par l'ANR JCJC LoRAiA ANR-20-CE23-0010, et par l'Agence de l'Innovation de Défense - AID - via le Centre Interdisciplinaire d'Etudes pour la Défense et la Sécurité - CIEDS (projet 2023 ALIA).

Références

[1] BIOUCAS-DIAS et AL. : Hyperspectral unmixing overview : Geometrical, statistical, and sparse regression-based approaches. *IEEE JSTAR*, 2012.

[2] Fahes et AL. : Unrolling palm for sparse semi-blind source separation. *In ICLR*, 2022.

[3] Févotte et AL. : Algorithms for nonnegative matrix factorization with the β-divergence. *Neural computation*, 2011.

[4] Gregor et AL. : Learning fast approximations of sparse coding. *In ICML*, 2010.

[5] Hadjeres et AL. : Generating synthetic data to train a deep unrolled network for hyperspectral unmixing. *In EUSIPCO*, 2024.

[6] Hershey et AL. : Deep Unfolding : Model-Based Inspiration of Novel Deep Architectures, 2014.

[7] Hong et AL. : Endmember-guided unmixing network (egu-net) : A general deep learning framework for self-supervised hyperspectral unmixing. *Transactions on Neural Networks and Learning Systems*, 2021.

[8] Kervazo et AL. : Provably robust blind source separation of linear-quadratic near-separable mixtures. *SIAM Journal on Imaging Sciences*, 2021.

[9] Kervazo et AL. : Deep unrolling of the multiplicative updates algorithm for blind source separation, with application to hyperspectral unmixing. *In EUSIPCO*, 2024.

[10] Lee et AL. : Learning the parts of objects by non-negative matrix factorization. *nature*, 1999.

[11] Li et AL. : Minimum volume simplex analysis : A fast algorithm to unmix hyperspectral data. *In IGARSS*, 2008.

[12] Monga et AL. : Algorithm unrolling : Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[13] Nadisic et AL. : Smoothed separable nonnegative matrix factorization. *Linear Algebra and its Applications*, 2023.

[14] Nascimento et AL. : Vertex component analysis : A fast algorithm to unmix hyperspectral data. *TGRS*, 2005.

[15] Nasser et AL. : Deep unfolding for non-negative matrix factorization with application to mutational signature analysis. *Journal of computational biology*, 2022.

[16] Palsson et AL. : Blind hyperspectral unmixing using autoencoders : A critical comparison. *JSTAR*, 2022.

[17] Xiong et AL. : Snmf-net : Learning a deep alternating neural network for hyperspectral unmixing. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–16, 2021.