# Décomposition en ligne de tenseurs

Abraham Traoré, Maxime Berar, Alain Rakotomamonjy

# Décomposition en ligne de tenseurs

Abraham Traoré[1], Maxime Berar [1], et Alain Rakotomamonjy[1]

[1]Université de Rouen, LITIS

76800 Saint-Etienne du Rouvray, FRANCE

## Résumé

Nous présentons un nouvel algorithme de décomposition de tenseurs (données multimodales) pour l'inférence de facteurs latents dans un environnement dynamique (les données sont acquises de manière séquentielle au fil du temps). Plus précisément, étant donné un échantillon de tenseurs (défini dans notre cadre comme un ensemble composé de tenseurs de taille égale), l'approche développée permet d'inférer une base génératrice de facteurs latents en traitant les données au fur et à mesure qu'elles sont acquises. L'algorithme proposé repose sur la descente de coordonnées par blocs et des approximations probabilistes. Des expériences numériques sur des données aussi bien réelles que synthétiques donnent des résultats prometteurs.

**Mots clé** : données multimodales, décomposition en ligne, approximations probabilistes.

## 1 Introduction

Le problème d'inférence de facteurs latents pour des données multimodales communément appelées tenseurs a suscité récemment un grand intérêt et les techniques de décomposition de tenseurs ont été appliquées avec succès dans diverses applications en traitement du signal [Ga16], apprentissage automatique [AF16], etc.. Les deux décompositions de tenseurs les plus communément utilisées pour l'analyse exploratoire de données multimodales sont la décomposition de Tucker, introduite par Tucker en 1963 [Tuc63] et la décomposition CP introduite indépendamment par Hitchcock dans [Hit27] et Cattell dans [Cat44].

Dans de nombreuses applications réelles, les données multimodales sont acquises au fil du temps (par exemple les données climatiques [XZT$^+$16]). S'il est toujours possible, après l'acquisition de nouvelles observations, de ré-appliquer les décompositions standard utilisant tout l'échantillon en une seule fois, cette approche peut conduire à une impasse en terme de ressources computationnelles car on doit stocker toutes les données en mémoire et les problèmes intermédiaires associés aux décompositions standard peuvent avoir une complexité en espace élevée (par exemple, certaines approches pour la décomposition de Tucker avec contrainte de positivité matérialisent le produit de Kronecker des facteurs latents).

Dans la littérature, les approches de décomposition en ligne peuvent être subdivisées en deux catégories. La première consiste à inférer les facteurs latents pour un tenseur dont l'une des dimensions croît au fil du temps (ajout d'une nouvelle observation) sans nécessité de stocker, à un pas de temps t, toutes les observations antérieures à t. La seconde consiste à stocker aussi bien les observations antérieures que celles nouvellement acquises avec la mise à jour des facteurs s'effectuant de manière récursive.

Dans ce travail, le problème que l'on considère est celui de l'apprentissage en ligne de facteurs latents, appelés aussi dictionnaires multimodaux, sans nécessité de stockage des observations. Deux méthodes présentées dans [KM16] et [YCL15] effectuent la décomposition de tenseurs en ligne sous des contraintes de faible rang et d'orthogonalité des facteurs, la première résolvant un problème d'optimisation sur une variété de Stiefel et la seconde utilisant des projections aléatoires. Les autres approches existantes reposent sur des contraintes similaires (voir [LHWG11], [STF06]) Elles ne sont donc pas adaptées pour des tâches telles que l'apprentissage de dictionnaires redondants ou l'inférence de facteurs positifs. L'ensemble de ces méthodes manque de flexibilité quant aux contraintes que l'on peut imposer aux facteurs latents : la non-négativité, la parcimonie,..

Dans ce papier, on introduit un nouvel algorithme de décomposition de tenseurs en ligne basé sur la des-

cente de coordonnées par blocs et des approximations probabilistes de fonctions de coût. Nos contributions sont les suivantes :

— Proposition d'un algorithme de décomposition en ligne de tenseurs qui peut facilement s'étendre pour incorporer tout type de contraintes (présentant une certaine régularité) et ne présentant aucune contrainte sur les tailles des facteurs latents ;

— La comparaison avec des méthodes de l'état de l'art donne des résultats prometteurs.

# 2 Notations et rappels sur les tenseurs

Un tenseur d'ordre N $\mathcal{X} \in \mathbb{R}^{I_1 \times .. \times I_N}$ est un tableau multidimensionnel défini sur le produit tensoriel de N espaces vectoriels de dimensions respectives $I_n, 1 \leq n \leq N$. Les matrices sont notées par des lettres majuscules en gras (par exemple $\mathbf{A}$). Les colonnes d'une matrices $\mathbf{A}$ sont désignées par $\mathbf{A}_{:,j}$. La matricisation d'un tenseur est l'opération consistant à réarranger ses éléments dans une matrice. La matricisation par rapport au mode n d'un tenseur $\mathcal{X}$ donne une matrice $\mathbf{X}^{(n)} \in \mathbb{R}^{I_n \times \prod_{k \neq n} I_k}$.

La multiplication par rapport au mode-n d'un tenseur $\mathcal{X} \in \mathbb{R}^{I_1 \times ... \times I_N}$ avec une matrice $\mathbf{B}^{(n)} \in \mathbb{R}^{J_n \times I_n}$, notée $\mathcal{X} \times_n \mathbf{B}^{(n)}$ donne un tenseur $\mathcal{Y} \in \mathbb{R}^{I_1 \times .. \times I_{n-1} \times J_n \times I_{n+1} .. \times I_N}$ vérifiant $\mathbf{Y}^{(n)} = \mathbf{B}^{(n)} \mathbf{X}^{(n)}$. Les normes de Frobenius et $\ell_1$ pour les tenseurs, notées respectivement $\|\|_F$ et $\|\|_1$ sont définies par :
$\|\mathcal{X}\|_F = \left( \sum_{i_1,..,i_N} \mathcal{X}^2_{i_1,..,i_N} \right)^{\frac{1}{2}}$,
$\|\mathcal{X}\|_1 = \sum_{i_1,..,i_N} |\mathcal{X}_{i_1,..,i_N}|$.
Les définitions sont identiques pour les normes de Frobenius et $\ell_1$ des matrices. On notera par $\mathcal{I}$ le tenseur dont toutes les entrées sont égales à 1.

Pour une raison de facilité d'écriture, nous considérons les ensembles d'entiers suivants :

— $I_N^n = \{n, .., N\}$ : ensemble des entiers consécutifs de n à N (n et N inclus). Si n est égal à 1, on note simplement cet ensemble $I_N$, c'est-à-dire : $I_N = \{1, .., N\}$.

— $I_{N \neq n} = \{1, .., n-1, n+1, .., N\}$ : ensemble des entiers consécutifs de 1 à N avec n exclu.

Pour deux ensembles $I_n = \{1, ..., n\}, I_N^{n+1} = \{n+1, ..., N\}$, un tenseur d'ordre N $\mathcal{Y} \in \mathbb{R}^{I_1 \times ... \times I_N}$ et N matrices $\left\{\mathbf{A}^{(n)}\right\}_{1 \leq n \leq N}$, on définit également des formes contractées de la multiplication par rapport à plusieurs modes :

$$\mathcal{Y} \times_{p \in I_n} \mathbf{A}^{(p)} \times_{q \in I_N^{n+1}} \mathbf{A}^{(q)} = \mathcal{Y} \times_1 \underbrace{\mathbf{A}^{(1)} .. \times_n \mathbf{A}^{(n)}}_{I_n} \times_{n+1} \underbrace{\mathbf{A}^{(n+1)} .... \times_N \mathbf{A}^{(N)}}_{I_N^{n+1}}$$

$$\mathcal{Y} \times_{m \in I_{N \neq n}} \mathbf{A}^{(m)} = \mathcal{Y} \times_1 .. \times_{n-1} \mathbf{A}^{(n-1)} \times_{n+1} \mathbf{A}^{(n+1)} .. \mathbf{A}^{(N)}$$

# 3 Présentation de la décomposition de Tucker en ligne

## 3.1 Cadre de décomposition Batch

La décomposition de Tucker est l'une des décompositions de tenseur les plus couramment utilisées. Pour un tenseur $\mathcal{X} \in \mathbb{R}^{I_1 \times .. \times \mathbf{I}_N}$ d'ordre N, elle consiste à approximer $\mathcal{X}$ par le produit d'un tenseur $\mathcal{G} \in \mathbb{R}^{J_1 \times .. \times J_N}$ avec N matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times J_n}$, c'est-à-dire : $\mathcal{X} \approx \mathcal{G} \times_{n \in I_N} \mathbf{A}^{(n)}$. Cette approximation s'effectue en résolvant le problème d'optimisation suivant :

$$\min_{\mathcal{G}, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}} \|\mathcal{X} - \mathcal{G} \times_{n \in I_N} \mathbf{A}^{(n)}\|_F^2$$

Le tenseur $\mathcal{G}$ est appelé tenseur noyau et les matrices $\mathbf{A}^{(n)}$ sont désignées par matrices de saturation. Cette décomposition peut s'interpréter en terme d'atomes de dictionnaires (voir [QHJ15]) : les matrices $\mathbf{A}^{(n)}$ représentent alors des dictionnaires associés aux différents modes et les composantes de $\mathcal{G}$ sont des coefficients d'activation. Le cadre qui nous intéresse est un cas particulier de ce problème où on vise à décomposer un tenseur sur l'ensemble des modes sauf un. Cette situation apparaît par exemple lorsque le tenseur peut être interprété comme l'observation de T tenseurs d'ordre N suivant une certaine distribution. Le problème auquel on s'intéresse peut donc s'écrire sous la forme :

$$\min_{\mathcal{G}, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}} \|\mathcal{X} - \mathcal{G} \times_1 I \times_2 \mathbf{A}^{(1)} \times_3 \mathbf{A}^{(2)} \times ... \times_{N+1} \mathbf{A}^{(N)}\|_F^2 \quad (1)$$

avec $\mathcal{X} \in \mathbb{R}^{T \times I_1 .. \times I_N}$ étant un tenseur dont la première dimension croit au fil du temps avec l'ajout d'un nouveau tenseur $\mathcal{X}_n \in \mathbb{R}^{I_1 .. \times I_N}$, $\mathcal{G} \in \mathbb{R}^{T \times J_1 .. \times J_N}$, $I$ étant la matrice identité. Si les observations $\mathcal{X}_i$ sont des matrices, ce problème correspond à la décomposition de Tucker2 [KB09]. En remarquant que le carré de la norme de Frobenius d'un tenseur est égale à la somme des carrés des normes de Frobenius de ses coupes frontales (tenseur déduit en fixant le premier indice), le problème (1) peut se réécrire sous la forme :

$$\min_{\mathbf{A}^{(1)}, .., \mathbf{A}^{(n)}, \{\mathcal{G}_i\}_{1 \leq i \leq T}} \sum_{i=1}^{T} \frac{1}{2} \|\mathcal{X}_i - \mathcal{G}_i \times_{n \in I_N} \mathbf{A}^{(n)}\|_F^2 \quad (2)$$

avec $\mathcal{X}_i \in \mathbb{R}^{I_1 \times \ldots \times I_N}, \mathcal{G}_i \in \mathbb{R}^{J_1 \times \ldots \times J_N}$ étant les coupes i des tenseurs $\mathcal{X}$ et $\mathcal{G}$ définis au niveau du problème (1). On notera que les ordres de $\mathcal{X}_i$ et $\mathcal{G}_i$ sont plus petits d'une unité que ceux de $\mathcal{X}$ et $\mathcal{G}$ par définition d'une coupe.

Enfin, similairement au problème d'apprentissage de dictionnaire, on peut y joindre à ce problème un terme de pénalité $\Omega_1$ forçant $\mathcal{G}$ à être parcimonieux et un terme de pénalité $\Omega_2$ contraignant les atomes de dictionnaires à ne pas diverger :

$$\min_{\mathbf{A}^{(n)}, \{\mathcal{G}_i\}_{1 \leq i \leq T}} \sum_{i=1}^{T} \frac{1}{2} \|\mathcal{X}_i - \mathcal{G}_i \times_{n \in I_N} \mathbf{A}^{(n)}\|_F^2 \qquad (3)$$

$$+\Omega_1(\mathcal{G}_i) + \Omega_2(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)})$$

Le problème (3) étant non convexe et n'admettant de solutions analytiques, des minimas globaux sont difficiles à obtenir. Pour obtenir des minimas locaux, une manière standard de résoudre ce problème est d'utiliser une minimisation alternée[FC00]. Cependant, lorsque le nombre d'observations T est grand, la résolution de ce problème peut devenir très coûteux en temps de calcul. Dans ce contexte, il devient nécessaire de pouvoir inférer les matrices de saturation $\mathbf{A}^{(n)}$ via une approche en ligne dont le principe est de traiter les observations de manière séquentielle.

## 3.2 Cadre de décomposition en ligne

Dans le problème d'apprentissage en ligne d'atomes de dictionnaires multi-modaux, on s'intéresse à l'inférence d'une base de facteurs latents pour des tenseurs acquis séquentiellement sans nécessité de les stocker au fil du temps avec un nombre de tenseurs potentiellement infini. Cela revient à déterminer les matrices $\mathbf{A}^{(n)}$ à partir du problème (3) pour $T \to \infty$. Pour résoudre ce problème pour une large séquence de tenseurs tout en contournant les contraintes liées au stockage de toute la séquence, nous proposons une approche probabiliste [KY06], qui est utilisée pour optimiser une fonction de coût exprimée en terme d'espérance. Cette approche, déjà utilisée dans le cadre de la décomposition en ligne de matrices, notamment en apprentissage de dictionnaires [MBPS09], a prouvé son efficacité dans le cadre de minimisation de problèmes non convexes [SG14].

Formellement, supposons qu'on observe des tenseurs, acquis au fil du temps, suivant un tirage aléatoire indépendant et identiquement distribué selon une loi de probabilité inconnue $\mathbb{P}$ sur l'espace des tenseurs de taille $I_1 \times .. \times I_N$. Notons $\mathcal{X}_t$ l'observation acquise au pas de temps $t$. Les facteurs latents sont mis à jour en utilisant une fonction de perte $l$ de telle sorte que l'écart entre toutes les observations $\mathcal{X}_t$ et leurs approximations données par ces facteurs soit faible en espérance.

Un problème d'optimisation pertinent est donc :

$$\min_{\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}} f(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) \qquad (4)$$

avec : $f(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) = \mathbb{E}_{\mathbb{P}}(l(\mathcal{X}, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)})$ où $\mathcal{X}$ est distribué suivant la loi de probabilité $\mathbb{P}$.

Pour notre problème, l'écart d'approximation $l$ est mesuré via la décomposition de Tucker. Ce choix est motivé par le fait que cette décomposition peut facilement être adaptée pour mesurer la perte via d'autres types de décomposition standard comme la décomposition canonique. On définit donc $l$ par :

$$l(\mathcal{X}, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) = \min_{\mathcal{G}} \frac{1}{2} \|\mathcal{X} - \mathcal{G} \times_{n \in I_N} \mathbf{A}^{(n)}\|_F^2$$

$$+\Omega_1(\mathcal{G}) + \Omega_2(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)})$$

Dans la suite, on supposera que $\Omega_2$ différentiable et que l'opérateur proximal de $\Omega_1$ est connu.

## 3.3 Algorithme

Comme la loi $\mathbb{P}$ est inconnue et les observations acquises séquentiellement au fil du temps, il est difficile de résoudre le problème (4). On choisit donc de minimiser l'approximation de l'espérance (définie par la loi forte des grands nombres) donnée par :

$$\widehat{f}_t(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) = \frac{1}{t} \sum_{i=1}^{t} l(\mathcal{X}_i, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)})$$

Ce raisonnement conduit donc au problème de minimisation suivant :

$$\min_{\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}} \widehat{f}_t(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) \qquad (5)$$

Notre approche consiste à mettre à jour les facteurs latents en minimisant $\widehat{f}_t$ et en utilisant uniquement l'observation $\mathcal{X}_t$, les facteurs inférés du traitement de $\mathcal{X}_{t-1}$ et un certain nombre de statistiques (fonction de l'échantillon). Notons $\mathbf{A}_t^{(n)}$ les dictionnaires obtenus après la résolution numérique du problème de minimisation de $\widehat{f}_t$.

Pour le calcul de la fonction $l(\mathcal{X}_t, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)})$, on utilise le code parcimonieux $\mathcal{G}_t$ associé à $\mathcal{X}_t$, obtenu par projections sur les facteurs $\left\{\mathbf{A}_{t-1}^{(n)}\right\}$, comme intermédiaire de calcul, c'est-à-dire :

$$l(\mathcal{X}_t, \mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) = \frac{1}{2} \|\mathcal{X}_t - \mathcal{G}_t \times_{n \in I_N} \mathbf{A}^{(n)}\|_F^2$$
$$+ \Omega_1(\mathcal{G}_t) + \Omega_2(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}),$$

avec $\mathcal{G}_t = \arg\min_{\mathcal{G}} \frac{1}{2} \|\mathcal{X}_t - \mathcal{G} \times_{n \in I_N} \mathbf{A}_{t-1}^{(n)}\|_F^2 + \Omega_1(\mathcal{G})$. Supposons l'acquisition d'une nouvelle observation $\mathcal{X}_t$ à un temps $t$. Pour déterminer les matrices $\mathbf{A}_t^{(n)}$, on procédera en 2 étapes :

— la détermination du code parcimonieux $\mathcal{G}_t$ associé à $\mathcal{X}_t$. Dans la suite, on désignera cette étape par *codage parcimonieux* ;

— la résolution du problème (5) par une *descente de coordonnées par blocs*, c'est-à-dire des minimisations successives de la fonction par rapport à une variable en gardant toutes les autres figées (dans l'ordre suivant : $1 \rightarrow 2 \rightarrow ..... \rightarrow N$), jusqu'à ce qu'un critère d'arrêt préalablement défini soit vérifié.

### 3.3.1 Première étape : codage parcimonieux

Dans cette partie, on s'intéresse à la minimisation d'une fonction définie par :

$$\min_{\mathcal{G}} \left(\mathcal{O}(\mathcal{G}) + \Omega_1(\mathcal{G})\right)$$

avec :

$$\mathcal{O}(\mathcal{G}) = \frac{1}{2}\|\mathcal{X}_t - \mathcal{G} \times_{n\in I_N} \mathbf{A}_{t-1}^{(n)}\|_F^2$$

Plutôt que de travailler sur la forme vectorisée et d'introduire une complexité en espace trop élevée dû au produit de Kronecker des matrices de dictionnaires, nous proposons d'utiliser directement une technique de minimisation proximale [BST14] dans l'espace des tenseurs. Les techniques de minimisation proximales sont utilisées pour minimiser la somme de deux fonctions, une différentiable (dans notre cas $\mathcal{O}$) et l'autre ne présentant pas de contrainte de régularité, mais dont l'opérateur proximal est connu ($\Omega_1$ dans notre cas). La différentielle de $\mathcal{O}$ est donnée par :

$$\frac{\partial \mathcal{O}}{\partial \mathcal{G}}(\mathcal{G}) = -\mathcal{X}_t \times_{n\in I_N} \mathbf{A}_{t-1}^{(n)T} + \mathcal{G} \times_{p\in I_N} \mathbf{A}_{t-1}^{(n)T}\mathbf{A}_{t-1}^{(n)} \quad (6)$$

Étant donné qu'on a supposé dans la section 3.2 que l'opérateur proximal de $\Omega_1$ était connu, le codage parcimonieux peut donc s'effectuer numériquement via l'algorithme du gradient proximal résumé comme suit :

*Codage parcimonieux*

**Entrées** : nouvelle observation $\mathcal{X}_t$, les matrices de dictionnaires $\mathbf{A}_{t-1}^{(n)}$, pas de descente de gradient $\eta$, valeur initial de $\mathcal{G}$,
iter $\leftarrow 0$ ;
$\mathcal{G}_{t,iter} \leftarrow \mathcal{G}_{init}$
**Répéter**
  $\mathcal{G}_{iter+1}$=$\mathbf{prox}_{\eta\Omega_1}(\mathcal{G}_{iter} - \eta\frac{\partial \mathcal{O}}{\partial \mathcal{G}}(\mathcal{G}_{iter}))$ défini par (6),
  iter $\leftarrow$ iter+1
**jusqu'à ce que** (un critère d'arrêt soit vérifié)

### 3.3.2 Descente de coordonnées par blocs

Considérons maintenant le problème de descente de coordonnées par blocs. Supposons que les $(n-1)$ premières matrices de saturation ont déjà été mises à jour (c'est-à-dire les matrices $\mathbf{A}_{t,k+1}^{(p)}$ sont connues avec $1 \leq p \leq n-1$, k faisant référence au numéro de l'itération) et qu'on s'intéresse à la mise à jour de $\mathbf{A}_{t,k+1}^{(n)}$. Ce problème est défini par :

$$\mathbf{A}_{t,k+1}^{(n)} \leftarrow \arg\min_{\mathbf{A}^{(n)}} \widehat{f}_{n,t}(.., \mathbf{A}_{t,k+1}^{(n-1)}, \mathbf{A}^{(n)}, \mathbf{A}_{t,k}^{(n+1)}, ..) \quad (7)$$

avec $\widehat{f}_{n,t}$ étant la fonction obtenue en considérant $\widehat{f}_t$ uniquement comme une fonction de $\mathbf{A}^{(n)}$.

Le problème (7) n'admettant pas de solution analytique et étant donné l'hypothèse de différentiabilité par rapport à $\mathbf{A}^{(n)}$ imposée sur la pénalité $\Omega_2$ assurant ainsi la différentiabilité de la fonction de coût comme somme de deux fonctions différentiables, on le résout par descente de gradient. Ce choix permet de mettre à jour $\mathbf{A}^{(n)}$ en utilisant uniquement l'observation au pas de temps $t$ $\mathcal{X}_t$ (à condition de mettre à jour régulièrement un certain nombre de statistiques de l'échantillon). En effet, la différentielle de $\widehat{f}_{n,t}$ est donnée par :

$$\frac{\partial \widehat{f}_{n,t}}{\partial \mathbf{A}^{(n)}}\left(\mathbf{A}^{(n)}\right) = -\frac{1}{t}\sum_{i=1}^{t}\left(\widehat{\mathbf{X}}_i^{(n)}\mathbf{G}_i^{(n)T} - \mathbf{A}^{(n)}\mathbf{B}_i^{(n)}\mathbf{B}_i^{(n)T}\right) + \frac{\partial \Omega_2}{\partial \mathbf{A}^{(n)}}(\mathbf{A}_{k+1}^{(1)}, ., \mathbf{A}_{k+1}^{(n-1)}, \mathbf{A}^{(n)}, \mathbf{A}_k^{(n+1)}, ., \mathbf{A}_k^{(N)}) \quad (8)$$

où la matrice $\mathbf{G}_i^{(n)}$ est la forme matricisée par rapport au mode $n$ de $\mathcal{G}_i$, $\mathbf{B}_i^{(n)}$ celle du tenseur $\mathcal{B}_i$ et $\widehat{\mathbf{X}}_i^{(n)}$ celle du tenseur $\widehat{\mathcal{X}}_i$ définis par :

$$\mathcal{B}_i = \mathcal{G}_i \times_{p\in I_{n-1}} \mathbf{A}_{k+1}^{(p)} \times_n \mathbf{I} \times_{q\in I_N^{n+1}} \mathbf{A}_k^{(q)}, \quad (9)$$

$$\widehat{\mathcal{X}}_i = \mathcal{X}_i \times_{p\in I_{n-1}} \mathbf{A}_{k+1}^{(p)T} \times_n \mathbf{I} \times_{q\in I_N^{n+1}} \mathbf{A}_k^{(q)T}. \quad (10)$$

La dérivée de $\widehat{f}_{n,t}$ peut donc se réécrire :

$$\frac{\partial \widehat{f}_{n,t}}{\partial \mathbf{A}^{(n)}}(\mathbf{A}^{(n)}) = -\frac{P_t}{t} + \frac{\mathbf{A}^{(n)}Q_t}{t} + \frac{\partial \Omega_2}{\partial \mathbf{A}^{(n)}}(\mathbf{A}_{k+1}^{(1)}, ., \mathbf{A}_{k+1}^{(n-1)}, \mathbf{A}^{(n)}, \mathbf{A}_k^{(n+1)}, ., \mathbf{A}_k^{(N)}) \quad (11)$$

avec $\mathbf{P}_t = \sum_{i=1}^{t}\widehat{\mathbf{X}}_i^{(n)}\mathbf{G}_i^{(n)T}$, $\mathbf{Q}_t = \sum_{i=1}^{t}\mathbf{B}_i^{(n)}\mathbf{B}_i^{(n)T}$.
On peut alors remarquer que :

$$\mathbf{P}_t = \mathbf{P}_{t-1} + \widehat{\mathbf{X}}_t^{(n)}\mathbf{G}_t^{(n)T} \quad (12)$$

$$\mathbf{Q}_t = \mathbf{Q}_{t-1} + \mathbf{B}_t^{(n)}\mathbf{B}_t^{(n)T} \quad (13)$$

Puisque les suites $\mathbf{P}_t$ et $\mathbf{Q}_t$ vérifient les équations récursives (12) et (13), le calcul du gradient de $\widehat{f}_{n,t}$ nécessite uniquement l'observation $\mathcal{X}_t$, les matrices $\left\{\mathbf{A}_{k+1}^{(p)}\right\}_{1\leq p\leq n-1}, \left\{\mathbf{A}_k^{(q)}\right\}_{n+1\leq q\leq N}$ et non le stockage des données $\{\mathcal{X}_i\}_{1\leq i\leq t-1}$.Cela nous permet de calculer la dérivée de $\widehat{f}_{n,t}$ utilisant uniquement $\mathcal{X}_t$, des matrices de saturation $\mathbf{A}^{(n)}$ et les statistiques $\mathbf{P}_{t-1}$ et $\mathbf{Q}_{t-1}$. Le schéma de mise-à-jour des $\mathbf{A}^{(n)}$ est décrit ci-après :

*Mise à jour de* $\mathbf{A}^{(n)}$

**Entrées** : nouvelle observation $\mathcal{X}_t$, code parcimonieux $\mathcal{G}_t$ associé à $\mathcal{X}_t$, les paramètres de pénalité $\alpha$ et $\theta$, les statistiques $\mathbf{P}_{t-1}$ et $\mathbf{Q}_{t-1}$, pas de descente de gradient $\eta$, valeur initiale de la matrice de saturation $\mathbf{A}^{(n)}$.
iter $\leftarrow 0$ ;
$\mathbf{A}^{(n)}_{iter} \leftarrow \mathbf{A}^{(n)}_{init}$
**Répéter**
  $\mathbf{A}^{(n)}_{iter+1} = \left(\mathbf{A}^{(n)}_{iter} - \eta \frac{\partial \widehat{f}_{n,t}}{\partial \mathbf{A}^{(n)}}(\mathbf{A}^{(n)}_{iter})\right)$, $\frac{\partial \widehat{f}_{n,t}}{\partial \mathbf{A}^{(n)}}(\mathbf{A}^{(n)}_{iter})$ défini par (11), (12) et (13)
  iter $\leftarrow$ iter+1 ;
**jusqu'à ce que** (un critère d'arrêt soit vérifié)

# 4 Extensions

## 4.1 Extension minibatch

Dans un contexte où $\rho$ tenseurs $\{\mathcal{X}_1, .., \mathcal{X}_\rho\}$ sont acquis simultanément au pas de temps $t$, on propose la modification suivante du schéma de l'algorithme :

- Effectuer le codage parcimonieux des $\rho$ tenseurs ;
- Mettre à jour $\mathbf{P}$ et $\mathbf{Q}$ de la manière suivante :
  $\mathbf{P}_t = \mathbf{P}_{t-1} + \sum_{r=1}^{\rho} \widehat{\mathbf{X}}_r^{(n)} \mathbf{G}_r^{(n)T}$, $\widehat{\mathbf{X}}_r$ étant la forme matricisée par rapport au mode n du tenseur défini par l'équation (10)
  $\mathbf{Q}_t = \mathbf{Q}_{t-1} + \sum_{r=1}^{\rho} \mathbf{B}_r^{(n)} \mathbf{B}_r^{(n)T}, \mathbf{B}_r^{(n)}$ étant la forme matricisée par rapport au mode n du tenseur défini par (9).

Cette extension réduit le coût des problèmes intermédiaires résultant de la descente de coordonnées par blocs.

## 4.2 Incorporation de contraintes

L'approche proposée peut facilement s'étendre à toutes contraintes pour lesquelles les pénalités associées ont un opérateur proximal ayant une formule analytique (voir [BST14]) ou présentant une certaine régularité (différentiabilité). Plus précisément, cela signifie que le raisonnement proposé peut marcher soit uniquement avec des pénalités pour lesquelles l'opérateur proximal admet une formule analytique, soit uniquement avec des pénalités différentiables, soit un mixte des deux. L'hypothèse minimale pour mettre en œuvre l'approche est l'existence de formule analytique de l'opérateur proximal, ce qui est moins contraignant que la régularité.

Un cas important est l'inférence de facteurs latents positifs, qui est une contrainte naturelle quand la donnée que l'on traite est positive par nature (par exemple des spectrogrammes). Pour inférer des facteurs positifs, il suffit de remplacer les étapes de mise à jour par :

- $\mathcal{G}_{iter+1} = \max(\mathcal{G}_{iter} - \eta(\frac{\partial \mathcal{C}}{\partial \mathcal{G}}(\mathcal{G}_{iter}) + \alpha\theta\mathcal{I}), 0)$ dans l'algorithme du *codage parcimonieux* ;
- $\mathbf{A}^{(n)}_{iter+1} = \max(\mathbf{A}^{(n)}_{iter} - \eta \frac{\partial f_{n,t}}{\mathbf{A}^{(n)}}(\mathbf{A}^{(n)}_{iter}), 0)$ dans l'algorithme de *mise à jour de* $\mathbf{A}^{(n)}$.

# 5 Expériences

L'objectif des expériences menées dans cette section est de montrer que d'une part, notre méthode aboutit à des facteurs latents pour lesquels les performances sont équivalentes à ceux obtenus en décomposition *Batch* et d'autre part, qu'elle donne des résultats compétitifs aussi bien dans un cadre favorable que défavorable à une méthode de décomposition en ligne de l'état de l'art. Plus précisément, on comparera la méthode *TuckerBatch* qui consiste à inférer la base de facteurs latents en appliquant une décomposition de Tucker standard utilisant toutes les données d'entraînement en une seule fois à l'approche *DLTsingle* introduite dans ce papier, ainsi qu'à la méthode *DLTminibatch* correspondant à l'extension de *DLTsingle* présentée dans la section précédente. Enfin, *ALTO* une autre méthode de décomposition en ligne sera comparée à ces méthodes. La différence fondamentale entre cette méthode et la nôtre se situe au niveau de la stratégie de mise à jour des matrices de dictionnaires : projections aléatoires pour *ALTO* et résolution numérique de problèmes de minimisation pour l'approche présentée *DTLsingle* et *DTLminibatch*. Cette méthode donne une erreur d'approximation petite pour $\mathcal{X}_t$ si une condition suffisante dite de rang faible (c'est-à-dire toutes les formes matricisées de tous les tenseurs de la séquence ont un rang inférieur ou égal à $R$, $(R, \cdots, R)$ étant la dimension du tenseur noyau) est vérifiée pour $\mathcal{X}_{t-1}$.

Dans le cadre de ces expériences, on considérera deux des pénalités les plus couramment utilisées dans l'apprentissage de dictionnaires : $\Omega_1(\mathcal{G}) = \alpha\theta\|\mathcal{G}\|_1$,
$\Omega_2(\mathbf{A}^{(1)}, .., \mathbf{A}^{(N)}) = \frac{\alpha(1-\theta)}{2} \sum_{n=1}^{N} \|\mathbf{A}^{(n)}\|_F^2$ avec $\alpha > 0, 0 \leq \theta \leq 1$. On notera que $\Omega_2$ est bien différentiable et que $\Omega_1$ admet un opérateur proximal qui est l'opérateur de *Soft-thresholding* [PSW15].

## 5.1 Expérience sur données simulées

### 5.1.1 Cadre expérimental

Pour l'évaluation du modèle, on considère 2 situations pour lesquelles on génère 2 bases de données. On génère :

- un premier échantillon $\mathbf{S}$ de 2000 tenseurs qu'on divise en deux sous-échantillons $\mathbf{S}_{train}$ et $\mathbf{S}_{test}$ suivant

un ratio de 4 :1 ($\mathbf{S}_{train}$ contient 1600 tenseurs tandis que $\mathbf{S}_{test}$ en contient 400).
— un second échantillon $\mathbf{S}$ de 6000 tenseurs qu'on divise suivant un ratio de 1 :2 (2000 tenseurs sont dans $\mathbf{S}_{train}$ et 4000 dans $\mathbf{S}_{test}$).

Lors de leur génération, chaque tenseur est défini par : $\mathcal{X}_t = \mathcal{G}_t \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}$ avec $\mathcal{G}_t \in \mathbb{R}^{R\times R\times R}$, tenseur d'ordre 3 dont les entrées sont tirées d'une gaussienne centrée d'écart type $\frac{1}{5}$ et $\mathbf{A}^{(1)} \in \mathbb{R}^{30\times R}, \mathbf{A}^{(2)} \in \mathbb{R}^{40\times R}, \mathbf{A}^{(3)} \in \mathbb{R}^{50\times R}$, matrices avec des entrées tirées d'une gaussienne centrée d'écart type $\frac{1}{10}$. Pour l'estimation des dictionnaires (matrices de saturation), on choisit des dictionnaires contenant $R$ atomes (taille avec laquelle les séquences ont été générées) pour ne pas biaiser les résultats. Les tenseurs noyaux sont initialisés en tirant des gaussiennes centrés d'écart type $\frac{1}{10}$ aussi bien durant la phase de test que la phase d'entraînement. Les matrices de saturation initiales sont tirés d'une gaussienne centrée d'écart type $\frac{1}{100}$.

Le pas de descente de gradient est fixé à $\eta = 10^{-5}$ et les valeurs des hyperparamètres sont fixés à $\alpha = 10^2$ et $\theta = 10^{-2}$. L'algorithme de descente de coordonnées par blocs est arrêté quand l'erreur d'ajustement est inférieure à un seul fixé ($10^{-5}$) ou quand vingt itérations sont atteintes (une itération dans ce cadre étant la résolution du problème de mise à jour de toutes les matrices $\mathbf{A}^{(n)}$). Chaque problème intervenant au niveau de la descente de coordonnées par blocs, qui correspond à la mise à jour d'une matrice $\mathbf{A}^{(n)}$ est aussi arrêté quand l'erreur relative d'ajustement est inférieure à $10^{-5}$ ou quand vingt itérations ont été effectuées (une itération est une mise à jour au niveau de la descente de gradient).

Enfin, pour l'extension *DTLminibatch*, on a choisi comme tailles des mini-batch $\{400, 400, 400, 400\}$, c'est-à-dire qu'au lieu de mettre à jour les matrices de dictionnaires en traitant les observations une par une, on les met à jour en traitant 4 fois 400 observations. Pour le deuxième scénario, les tailles de mini-batch choisies sont $\{400, 800, 800\}$.

### 5.1.2 Critère d'évaluation

Le critère de comparaison est l'erreur d'ajustement moyenne sur l'ensemble de test $\mathbf{S}_{test}$ définie par :

$$RMSE = \frac{1}{N}\sum_{t=1}^{N} \|\mathcal{X}_t - \widehat{\mathcal{G}}_t \times_1 \mathbf{A}_s^{(1)} \times_2 \mathbf{A}_s^{(2)} \times_3 \mathbf{A}_s^{(3)}\|_F^2$$

avec :

$$\widehat{\mathcal{G}}_t = \arg\min_{\mathcal{G}} \|\mathcal{X}_t - \mathcal{G} \times_1 \mathbf{A}_s^{(1)} \times_2 \mathbf{A}_s^{(2)} \times_3 \mathbf{A}_s^{(3)}\|_F^2 + \alpha\theta\|\mathcal{G}\|_1 \tag{14}$$

et où $N$ correspond au cardinal de $\mathbf{S}_{test}$, les matrices $\left\{\mathbf{A}_s^{(n)}\right\}_{1\leq n\leq 3}$ aux facteurs inférés en utilisant uniquement les observations de $\mathbf{S}_{train}$ et les observations $\mathcal{X}_t$ sont celles de $\mathbf{S}_{test}$. Pour chaque scénario, l'erreur d'ajustement moyenne est calculée de la manière suivante :

— On titre 3 fois $\mathbf{S}_{train}$ et $\mathbf{S}_{test}$ ;
— Pour chaque tirage, on calcule l'erreur d'ajustement ;
— On fait la moyenne des trois erreurs d'ajustement

### 5.1.3 Résultats

Les résultats que nous avons obtenus sont présentés dans le tableau 1. On constate que les méthodes *TuckerBatch*, *DLTsingle* et *DLTminibatch* rendent des erreurs d'approximation similaires et que *ALTO*, dans ce scénario considéré ne converge pas vers de bons minimas locaux. La performance de *ALTO* peut s'expliquer par la violation (vérifiée) de la condition de convergence qui est l'hypothèse de faible rang que doivent vérifier tous les tenseurs dans la séquence considérée. Ceci prouve bien que notre méthode est une bonne alternative à *ALTO*. Aussi, on observe des résultats similaires à ceux de la méthode *TuckerBatch* tout en contournant le coût du stockage de l'échantillon entier.

## 5.2 Expérience sur données réelles

Nous considérons une base de donnée réelle nommée **Foursquare** qui, dans la zone de Pittsburgh, contient le registre (ensemble de scores représentant une préférence, par exemple un score évaluant une impression par rapport à un lieu public) des utilisateurs dans plusieurs lieux de divertissement ainsi que des collèges et universités. Cette base de donnée est donc un tenseur $\mathcal{X} \in \mathbb{R}^{P\times T\times M}$ où P est égal à 56 et représente le nombre de lieux, T à 1200 et représente le nombre de points dans le temps choisis sur une période de 4 mois, et M à 15 et correspond au nombre de variables mesurées.

Pour cette expérience, notre objectif est prédire les valeurs des séries temporelles (série de tenseurs $P \times M$) représentant des observations historiques. Un des modèles couramment utilisés pour ce problème est un modèle de "vecteurs" auto-régressifs VAR(L) (voir [YCL15]), $L$ étant la taille de la fenêtre temporelle considérée. Étant donné un tenseur $\mathcal{X}^L \in \mathbb{R}^{P\times L\times M}$ d'ordre 3 (dont les modes sont l'emplacement, le temps et la variable) contenant les $L$ dernières observations, ce modèle cherche à déterminer le prochain tenseur de taille $(P \times 1 \times M)$ et pour cela s'appuie sur un modèle reposant sur un tenseur $\mathcal{W} \in \mathbb{R}^{P\times PL\times M}$ tel que :

$$\mathcal{X}_{:,t,m} = \mathcal{W}_{:,:,m}\mathbf{X}^L_{:,t-1:t-L,m} + \mathcal{E}$$

avec $1 \leq m \leq M$, $\mathbf{X}^L_{:,t-1:t-L,m} = [\mathcal{X}^T_{:,t-1,m}, .., \mathcal{X}^T_{:,t-L,m}]^T$. Le bruit $\mathcal{E}$ est supposé être distribué suivant une loi normale.

Pour déterminer $\mathcal{W}$, on considère le problème d'optimisation proposé dans [YCL15] se définissant comme suit :

$$\min_{\mathcal{W}} \|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2 + \mu\sum_{m=1}^{M} \text{Trace}(\widehat{\mathcal{X}}^T_{:,:,m}\mathbf{S}\widehat{\mathcal{X}}_{:,:,m}) \tag{15}$$

sous les contraintes : $\widehat{\mathcal{X}}^T_{:,t,m} = \mathcal{W}_{:,:,m}\mathbf{X}^L_{:,t,m}$, $\sum_{n=1}^{N} \text{rang}(\mathbf{W}^{(n)}) \leq R$, $\mathbf{X}^L_{:,t,m} = [\mathcal{X}^T_{:,t,m}, ....., \mathcal{X}^T_{:,t-L,m}]^T$ où $\mathbf{W}^{(n)}$ est la forme matricisée du tenseur $\mathcal{W}$ par rapport

| Méthodes / Taille noyau R | *TuckerBatch* | *ALTO* | *DLTsingle* | *DLTminibatch* |
|---|---|---|---|---|
| Scénario 1 : #($\mathbf{S}_{train}$)=1600, #($\mathbf{S}_{test}$)=400 | | | | |
| R=5 | 0.3188±0.01 | 1.5502±0.01 | **0.3189±0.01** | **0.3189± 0.01** |
| R=10 | 2.6520±0.2 | 12.6234±0.20 | **2.6537±0.20** | **2.6537±0.20** |
| R=15 | 7.5418±0.89 | 41.1823±0.88 | **7.5414± 0.89** | **7.5414±0.89** |
| R=20 | 20.1638±1.70 | 99.9323±1.73 | **20.1639±1.70** | **20.1639±1.70** |
| Scénario 2 : #($\mathbf{S}_{train}$)=2000, #($\mathbf{S}_{test}$)=4000 | | | | |
| R=5 | 0.2722 ±0.02 | 1.5076±0.02 | **0.2723±0.02** | **0.2723± 0.02** |
| R=10 | 2.4770±0.06 | 12.4428±0.07 | **2.4767±0.06** | **2.4767±0.06** |
| R=15 | 8.2009 ±0.71 | 41.8419±0.71 | **8.2010± 0.71** | **8.2010±0.71** |
| R=20 | 17.9004±0.73 | 97.6760±0.73 | **17.9009±0.73** | **17.9009±0.73** |

TABLE 1 – Erreur d'ajustement moyenne en fonction de la taille du tenseur noyau R pour les 2 scénarios

au mode $n$, **rang** et **Trace** correspondant respectivement au rang et à la trace d'une matrice. La matrice $\mathbf{S}$ est une matrice de similarité assurant la consistance locale qui traduit le fait que, pour un pas de temps donné, les paramètres ne doivent pas significativement varier pour des emplacements qui sont voisins.

Pour résoudre le problème (15) une approche (en ligne) en deux étapes a été introduite dans [YCL15] que l'on reprend et dont le principe est le suivant. Premièrement, mettre à jour le tenseur $\mathcal{W}$ de manière séquentielle et en traitant uniquement un sous échantillon (tiré dans le mode du temps qui correspond au second mode dans notre cas) du tenseur $\mathcal{X}$. Puis projeter le tenseur mis-à-jour $\mathcal{W}$ dans un espace de faible rang via des projections aléatoires et en utilisant uniquement le tenseur $\mathcal{W}$ obtenu au temps précédent et les matrices de saturation inférées précédemment. Cette deuxième étape suit un principe similaire à notre approche de décomposition en ligne. Nous proposons donc pour ce problème de prédiction de comparer l'influence de la méthode de décomposition en ligne en gardant la première étape identique à celle de *ALTO*, mais en variant la seconde étape. On comparera la proposition originale d'*ALTO* à une seconde étape où l'approximation de faible rang sera réalisée via *DLTsingle*. On considérera également l'approche consistant à remplacer la seconde étape par une décomposition de Tucker standard. Ce schéma sera désigné par *Tucker*.

### 5.2.1 Cadre expérimental

On divise le tenseur $\mathcal{X}$ dans le sens du deuxième mode (temps) en deux tenseurs d'ordre 3 $\mathcal{X}_{train} \in \mathbb{R}^{P\times T_1\times M}$ et $\mathcal{X}_{test} \in \mathbb{R}^{P\times T_2\times M}$ avec $T_1 = 0.9 \times T$ et $T_2 = 0.1 \times T$. Le tenseur $\mathcal{W}$ sera déterminé en utilisant $\mathcal{X}_{train}$ et l'erreur moyenne quadratique calculée sur $\mathcal{X}_{test}$.

Les valeurs des hyperparamètres sont fixés à $\alpha = 10^2$ et $\theta = 10^{-2}$. Le pas de descente de gradient est fixé à $\eta = 10^{-18}$. Le paramètre $\mu$ est fixé à $10^{-2}$. Les critères d'arrêt, aussi bien pour la descente de coordonnées par blocs que pour les problèmes de mise à jour des facteurs sont identiques à ceux de l'expérience sur les données simulées.

### 5.2.2 Critère d'évaluation

Le critère d'évaluation est l'erreur moyenne quadratique normalisée définie par : $RMSE = \left(\frac{1}{P\times T_1\times M}\sum_{m=1}^{M}\sum_{t=1}^{T_1}\|(\mathcal{X}_{test})^T_{:,t,m} - \mathcal{W}_{:,:,m}(\mathcal{X}_{test})^L_{:,t,m}\|_F^2\right)^{\frac{1}{2}}$, $(\mathcal{X}_{test})^L$ défini de manière similaire à $\mathcal{X}^L$.
Dans cette expression, le tenseur $\mathcal{W}$ est déterminé en résolvant le problème de régression (15) pour $\mathcal{X} = \mathcal{X}_{train}$.

### 5.2.3 Résultats

Les résultats de cette comparaison en fonction de la taille de la fenêtre temporelle L sont rassemblés dans la table 2. On observe que les trois méthodes donnent des erreurs relativement proches avec un léger avantage pour notre approche *DTLsingle*. L'écart entre *ALTO* et les deux autres méthodes peut s'expliquer par le fait que le biais induit via les projections aléatoires est plus important que celui induit en résolvant les problèmes de minimisation.
La réduction des écarts comparée à la première expérience résulte du fait que cette base de données vérifie bien la contrainte de faible rang (voir [YCL15])), assurant ainsi l'inférence de facteurs induisant une faible erreur d'approximation pour *ALTO*.

# 6 Conclusion

Dans ce papier, on a mis en place une technique d'apprentissage de dictionnaires multimodaux reposant sur la décomposition de tenseurs en ligne. Cette technique inspirée de l'apprentissage en ligne de dictionnaires dans le cadre matriciel repose sur l'alternance d'étapes de codages parcimonieux et de descente de coordonnées par bloc. Les comparaisons expérimentales réalisées sur des données jouées et dans le cadre d'un problème réel montrent que la technique en ligne obtient des résultats très similaires

| Méthodes / L | *DTLsingle* (notre méthode) | *ALTO* | *Tucker* |
|---|---|---|---|
| **1** | 0.1249 | 0.1253 | **0.1248** |
| **2** | **0.1249** | 0.1251 | 0.1249 |
| **3** | **0.1249** | 0.1250 | 0.1249 |

TABLE 2 – Errer moyenne quadratique normalisée pour différentes valeurs du paramètre de retard L, valeur du rang=(5,5,5)

à ceux de la décomposition utilisant toutes les données en une seule fois et des résultats pouvant s'avérer supérieurs à ceux d'une méthode de l'état de l'art quand l'hypothèse de rang faible n'est pas satisfaite.

Dans le futur, nous fournirons une analyse théorique de l'approche proposée. Nous prévoyons également l'extension de l'approche au cas où les données grandissent couramment dans plusieurs modes et ainsi nous approcher de la résolution d'une décomposition tensorielle sur de très grandes données.

# Références


[AF16] Muhammad Ali and Hassan Foroosh. Character recognition in natural scene images using rank-1 tensor decomposition. *ICIP*, 2016.

[BST14] Jerome Bolte, Shoham Sabach, and Marc Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1–2,) :459–494, 2014.

[Cat44] Raymond B. Cattell. Parallel proportional profiles" and other principles for determining the choice of factors by rotation. *Psychometrika*, 9(4) :267–283, 1944.

[FC00] Tony F.Chan and C.K.Wong. Convergence of the alternating minimization algorithm for blind deconvolution. *Linear Algebra and its Applications*, 316 :259–285, 2000.

[Ga16] Xian Guo and al. Support Tensor Machines for Classification of Hyperspectral Remote Sensing Imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 54(6) :3248 – 3264, 2016.

[Hit27] F. L. Hitchcock. The expression of a tensor or a polyadic as a sum of products. *J. Math.Phys.*, 6(1) :164–189, 1927.

[KB09] Tamara G. Kolda and Brett W. Bader. Tensor Decompositions and Applications. *SIAM REVIEW*, 51(3) :455–500, 2009.

[KM16] Hiroyuki Kasai and Bamdev Mishra. Low-rank tensor completion :a riemannian manifold preconditioning approach. *ICML*, 48 :1012–1021, 2016.

[KY06] H. J. Kushner and G. G. Yin. Stochastic approximation and recursive algorithms with applications. *Journal of the Royal Statistical Society Series A,*, 169(3) :654–654, 2006.

[LHWG11] Jie Li, Guan Han, Jing Wen, and Xinbo Gao. Robust tensor subspace learning for anomaly detection. *International Journal of Machine Learning and Cybernetics*, 2(2) :89–98, 2011.

[MBPS09] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. ICML '09, pages 689–696, 2009.

[PSW15] Nicholas G. Polson, James G. Scott, and Brandon T. Willard. Proximal Algorithms in Statistics and Machine Learning. 2015.

[QHJ15] Yuhui Quan, Yan Huang, and Hui Ji. Dynamic Texture Recognition via Orthogonal Tensor Dictionary Learning. *ICCV*, 2015.

[SG14] Konstantinos Slavakis and Georgios B. Giannakis. Online dictionary learning from big data using accelerated stochastic approximation algorithms. *ICASSP*, pages 16–20, 2014.

[STF06] Jimeng Sun, Dacheng Tao, and Christos Faloutsos. Beyond streams and graphs : Dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 374–383, 2006.

[Tuc63] L. R. Tucker. Implications of factor analysis of three-way matrices for measurement of change. *C.W. Harris (Ed.), Problems in Measuring Change, University of Wisconsin Press*, page 122–137, 1963.

[XZT+16] Jianpeng Xu, Jiayu Zhou, Pang-Ning Tan, Xi Liu, and Lifeng Luo. Wisdom : Weighted incremental spatio-temporal multi-task learning via tensor decomposition. *International Conference on Big Data*, 2016.

[YCL15] Rose Yu, Dehua Cheng, and Yan Liu. Accelerated online low-rank tensor learning for multivariate spatio-temporal streams. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning*, pages 238–247, 2015.