Identification aveugle de mélanges et décomposition canonique de tenseurs : application à l'analyse de l'eau

Jean-Philip Royer

▶ **To cite this version:**

Jean-Philip Royer. Identification aveugle de mélanges et décomposition canonique de tenseurs : application à l'analyse de l'eau. Autre. Université Nice Sophia Antipolis, 2013. Français. NNT : 2013NICE4073 . tel-00933819

HAL Id: tel-00933819

https://theses.hal.science/tel-00933819

Submitted on 21 Jan 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS & UNIVERSITÉ DU SUD
TOULON-VAR
I3S, UMR CNRS 7271 & LSIS, UMR CNRS 7296

ECOLE DOCTORALE STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

THÈSE

pour l'obtention du grade de

Docteur en Sciences

de l'Université de Nice-Sophia Antipolis

Mention : Automatique, Traitement du Signal et des Images (ATSI)

présentée et soutenue par

Jean-Philip ROYER

IDENTIFICATION AVEUGLE DE MÉLANGES ET DÉCOMPOSITION CANONIQUE DE TENSEURS : APPLICATION À L'ANALYSE DE L'EAU

Thèse dirigée par Pierre COMON et Nadège THIRION-MOREAU

soutenue le 4 octobre 2013

Jury :

M. Gérard FAVIER	Directeur de recherche CNRS, I3S, UNS	Président
M. Jérôme MARS	Professeur à l'INP Grenoble	Rapporteur
M. David BRIE	Professeur à l'Université de Lorraine	Rapporteur
M. Stéphane MOUNIER	Maître de conférences HDR à l'Université du Sud-Toulon Var	Examinateur
M. Pierre COMON	Directeur de Recherche CNRS, GIPSA-LAB, INP Grenoble	Directeur de Thèse
Mme. Nadège THIRION-MOREAU	Professeur des Universités à l'Université du Sud-Toulon	Co-directrice de Thèse

Remerciements

Arrivant à l'issue de travaux de thèse, j'ai eu l'occasion tout au long de ces années de rencontrer un certain nombre de personnes qui m'ont permis, directement ou indirectement, de mener à bien mon projet. C'est l'occasion pour moi de leur rendre hommage.

Tout d'abord, je suis très reconnaissant aux membres du jury d'avoir accepté d'évaluer mon travail. Merci donc à Jérome Mars et David Brie pour leur relecture en tant que rapporteurs et merci à Gérard Favier et Stéphane Mounier pour leur présence en tant qu'examinateurs.

J'adresse de vifs remerciements à mes deux encadrants de thèse, Pierre Comon et Nadège Thirion-Moreau. Pierre est à la fois quelqu'un de très humain et un chercheur passionné qui a su me prodiguer des conseils très judicieux et me faire profiter de sa rigueur scientifique. J'ai pu compter sur la grande disponibilité et sur l'encadrement très maternel de Nadège, qui m'a épaulé tout au long de mes travaux de thèse et fait bénéficier de ses idées éclairées qui m'ont souvent permis de résoudre les problèmes auxquels j'étais confronté. Pour leur soutien indéfectible tant au niveau scientique, qu'humain ou administratif, je leur dois beaucoup.

Je remercie les membres de l'équipe PROTEE pour la collaboration qu'on a pu avoir ensemble. De nouveau, je cite Stéphane Mounier, pour les longues heures qu'il a dû passer à nous préparer des échantillons de travail. Je remercie Roland Redon pour les échanges scientifiques que nous avons eus ensemble, ainsi qu'Annaëlle Zhao pour les données qu'elle a pu me procurer.

Je remercie ensuite toute l'équipe de l'aile Télécom pour la bonne ambiance et l'accueil qui m'a été réservé au sein de ses locaux, que j'avais déjà fréquentés (assidument !) étant étudiant : Eric Moreau, qui m'a permis de travailler avec lui l'année précédent ma thèse, Laurent Enel, Christophe De Luigi, Olivier Derrien, Bernard Xerri, Jean-Marc Stamegna, Audrey Minghelli-Roman, Cyril Prissette, Sylvain Maire, avec lequel j'ai entretenu des discussions intéressantes, à la fois sur les bienfaits des fast-foods mais surtout sur de nouvelles approches qui se sont avérées utiles à mon travail, et Xavier Luciani, pour son aide et les réponses qu'il a pu m'apporter.

Bien entendu, je n'oublie pas la clique de doctorants et stagiaires que j'ai croisés tout au long de ces années. Tual, mon collègue de promo. Diogone, le plus sérieux d'entre tous (je vois d'ici sa réaction...). Manchun, mon partenaire de badminton, qui sait choisir le moment pour nous faire rire. Victor, mon partenaire de pétanque et de tennis. On aura au moins gagné le tournoi intra-LSIS ensemble ! Giang, la plus marseillaise des vietnamiennes. Justine, la globe-

trotter (oui, c'est loin la Corse et moi j'ai pas fait de faute sur ton prénom !). Bon courage aux nouvelles arrivées, Xuan (que j'ai embrigadée pour le badminton !) et Cécile (courage pour nous supporter). Weili (encore une qui fait du badminton !), tu seras bientôt française, il ne te reste plus que la langue ! Merci également à tous ceux avec qui j'ai pu partager des activités ludiques. Il y a la partie bio et chimie, avec Florence et Cynthia, les non-permanents du LSIS avec entre autres, Régis, Thibaut, Vincent, Alain, Vincente notre présidente, la partie chimie et signal avec Cheikh, la partie culture avec Florian et la partie maritime avec Nore, Jenna et Karen.

Enfin, mes remerciements les plus chaleureux vont à ma famille, pour leur soutien et leurs encouragements. Ils ont su m'insuffler le goût de la connaissance et je n'en serais pas là sans eux.

Résumé

Dans cette thèse, nous nous focalisons sur le problème de la décomposition polyadique minimale de tenseurs de dimension trois, problème auquel on se réfère généralement sous différentes terminologies : “Polyadique Canonique” (CP en anglais), “CanDecomp”, ou encore“Parafac”. Cette décomposition s'avère très utile dans un très large panel d'applications. Cependant, nous nous concentrons ici sur la spectroscopie de fluorescence appliquée à des données environnementales particulières de type échantillons d'eau qui pourront avoir être collectés en divers endroits ou différents lieux. Ils contiennent un mélange de plusieurs molécules organiques et l'objectif des traitement numériques mis en oeuvre est de parvenir à séparer et à ré-estimer les composés de matières organiques présents dans les échantillons étudiés. Par ailleurs, dans plusieurs applications comme l'imagerie hyperspectrale ou justement, la chimiométrie, il est intéressant de contraindre les matrices de facteurs recherchées à être réelles et non négatives car elles traduisent des quantités physiques réelles non négatives (spectres, fractions d'abondance, concentrations, etc...). C'est pourquoi tous les algorithmes développés durant cette thèse l'ont été dans ce cadre (l'avantage majeur de cette contrainte étant de rendre le problème d'approximation considéré bien posé). Certains reposent sur l'utilisation de fonctions barrières, d'autres approches consistent à paramétriser directement les matrices de facteurs recherchées par des carrés. Divers algorithmes d'optimisation ont été étudiés : approches de type gradient, gradient conjugué non linéaire, bien adapté à des problèmes en grande dimension, Quasi-Newton (BGF et DFP) et enfin Levenberg-Marquardt. Deux versions de chacun de ces algorithmes ont systématiquement été considérées : la version “recherche linéaire améliorée” (Enhanced Line Search ou ELS, permettant de s'échapper de minima locaux) et la version “recherche linéaire par marche arrière” (backtracking en alternance avec l'ELS). De plus, des versions plus générales de ces algorithmes ont également été développées afin de pouvoir prendre en compte d'autres types de problèmes et/ou contraintes susceptibles d'apparaître : données manquantes, parcimonie des données. Différentes optimisations algorithmiques ont enfin été testées afin d'accélérer les vitesses de convergence des algorithmes. A des fins d'évaluation et de calibration, les différentes solutions proposées ont ensuite été testées sur des mélanges de composés connus effectués en laboratoire et comparées aux méthodes de “l'état de l'art”.

Mots clefs

Non négativité ; tenseurs d'ordre trois ; analyse multi-variée ou multi-linéaire ; décomposition canonique polyadique ; décomposition CP ; CanDecomp ; Parafac ; optimisation ; gradient conjuqué non linéaire ; algorithmes de gradient ; Quasi-Newton ; BFGS ; DFP ; méthodes de

préconditionnement ; données manquantes ; parcimonie ; données creuses ; chimiométrie ; analyse de données ; fouille de données ; spectroscopie de fluorescence ; séparation aveugle de composés de matière organique dissous (MOD).

Abstract

In this thesis, we focus on the problem of the minimal polyadic decomposition of a three-way tensor, sometimes referred to as "Canonical Polyadic" (CP), or also called "CanDecomp", "CanD", or "Parafac". This decomposition turns out to be very useful in a wide panel of applications. Yet, in this thesis, we will concentrate on 3D fluorescence spectroscopy of water samples collected in sea, estuaries, rivers or in water engineering process that contain a mixture of organic molecules. The aim is to recover the contribution of each organic compound present in the samples.

Even if an exact fit exists with a known number of terms, the calculation of the CP consists of finding the zeros of a polynomial of degree six or larger, in a very large number of variables. This problem is numerically very difficult to solve, even if the number of zeros is finite. Second, if the model is subject to errors, an approximate fit is wished to be computed. However, it is now well known that a best approximate may not always exist. Third, in several applications such as hyperspectral imaging or chemometrics, the loading matrices need to be constrained to be real and nonnegative. We shall subsequently concentrate on this framework. Fortunately, one advantage of the latter constraint is that the approximation problem becomes well posed.

We present the different new algorithms that have been developed to tackle the problem of the nonnegative polyadic decomposition of a three-way tensor : some of them are based on the use of barrier functions, other approaches consist of explicitly taking into account the nonnegative nature of the loading matrices by direct parameterization (square and thus Hadamard products) instead of enforcing positive entries by projection. Different optimization algorithms have been studied too : non linear conjugate gradient, well matched to large dimensions, combined with a global search in a chosen dimension. The latter combination permits to escape from local minima, gradient, Quasi-Newton (BGF and DFP) and Levenberg-Marquardt approaches. Two versions of each algorithms are considered : the Enhanced Line Search version (ELS) and the backtracking version (alternating with ELS). Moreover, generalizations of these solutions have also been considered in order to take into account possible missing data, sparsity, and different algorithmic optimizations have been suggested to increase the convergence speed. In order to evaluate their performances, algorithms have been tested on mixtures of known organic compounds carried out in laboratory and compared to the "state of the art" methods.

Key words Nonnegativity ; 3-way array / third order tensors ; tensor factorization ; multi-way analysis ; canonical polyadic decomposition ; CP decomposition ; canonical decomposition ; CanDecomp ; Parafac ; optimization ; non linear conjugate gradient ; gradient algorithms ; Quasi-Newton ; BFGS ; DFP ; preconditioning ; missing data ; sparsity ; chemometrics ; data analysis ; data mining ; fluorescence spectroscopy ; blind separation of dissolved organic matter (DOM).

Table des matières

Table des figures

Liste des tableaux

Liste des acronymes

ALS	Alternating Least Squares
BFGS	Broyden-Fletcher-Goldfarb-Shanno
c.à.d.	c'est-à-dire
Candecomp	Canonical decomposition
CP	Candecomp/Parafac ou Canonical Polyadic decomposition
CP – WOPT	CP Weighted OPTimization
DFP	Davidon-Fletcher-Powell
DMN	Décomposition matricielle non négative
DOM	Dissolved Organic Matter
DTN	Décomposition tensorielle non négative
EEG	Électro-EncéphaloGraphie
ELS	Enhanced Line Search
FEEM	Fluorescence Emission-Excitation Matrix
HALS	Hierarchical Alternating Least Squares
i.e.	id est ou c'est-à-dire
K – L	Kullback-Leibler divergence
LS	Line Search
MEEF	Matrice d'Emission-Excitation de Fluorescence
MIMO	Multiple Inputs Multiple Outputs
MOD	Matière organique dissoute
NMF	Nonnegative Matrix Factorization
NNLS	NonNegative Least Squares
NTD	Nonnegative Tucker decomposition
NTF	Nonnegative Tensor Factorization
PARAFAC	PARAllel FACtor analysis
RSB	Rapport Signal à Bruit
VW	Varying Weights

Notations

Dans toute la suite du manuscrit, nous utiliserons les notations suivantes :

$\mathbb{R}$	corps des réels.
$\mathbb{R}^+$	corps des réels non négatifs.
a	Une lettre en minuscule désigne une variable scalaire.
$\mathbf{a}$	Une lettre en gras et en minuscule désigne une variable vectorielle.
$\mathbf{A}$	Une lettre en majuscule et en gras désigne une matrice ou un tenseur.
$\mathbf{A} = (a_{ij}) \in \mathbb{R}^{I \times J}$	matrice de taille $I \times J$.
a_{ij}	Elément de coordonnées (i, j) d'une matrice $\mathbf{A} \in \mathbb{R}^{I \times J}$.
a_{ijk}	Elément (i, j, k) d'un tenseur $\mathbf{A} \in \mathbb{R}^{I \times J \times K}$ du $3^{\text{ème}}$ ordre.
$\mathbf{I}_N$	matrice identité de dimension $N \times N$.
$\mathbf{A}^T$	Opérateur de transposition d'une matrice, tel que $a_{ij} = a_{ji}$.
$\mathbf{A}^{-1}$	Opérateur d'inversion dans le cas d'une matrice carrée.
$\mathbf{A}^\dagger$	Pseudo-inverse de Moore-Penrose d'une matrice $\mathbf{A}$.
$\|.\|_1$	Norme L_1.
$\|.\|_F$	Norme de Frobenius.
diag	Si cet opérateur est appliqué à un vecteur, il retourne une matrice carrée contenant les éléments du vecteur sur la diagonale. Si cet opérateur est appliqué à une matrice, il retourne un vecteur contenant la diagonale de la matrice.
$\odot$	Produit de Khatri-Rao.
$\otimes$	Produit de Kronecker.
$\circledast$	Produit tensoriel.
$\boxdot$	Produit de Hadamard.
$\mathsf{exp}\{\cdot\}$	fonction exponentielle.
$\langle \cdot, \cdot \rangle$	Opérateur de produit scalaire.
∂	Opérateur de dérivation partielle.
d	Opérateur différentiel.
$\mathsf{vec}\,(.)$	Opérateur de vectorisation.

$\mathsf{trace}\,\{.\}$	Trace d'une matrice.
$\mathbf{T}^{I,KJ}_{(1)}$	Dépliement du tenseur $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$ dans le mode 1.
$\mathbf{T}^{J,KI}_{(2)}$	Dépliement du tenseur $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$ dans le mode 2.
$\mathbf{T}^{K,JI}_{(3)}$	Dépliement du tenseur $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$ dans le mode 3.
log	logarithme népérien ou naturel.
$\log_{10}$	logarithme à base 10 ou logarithme décimal.
$\mathbf{1}_{K,F}$	matrice de taille $K \times F$ ne contenant que des 1.
$\mathbf{0}_{I,1}$	vecteur de taille $I \times 1$ ne contenant que des éléments nuls.
$\mathsf{max}\{\cdot\}$	opérateur qui retourne la plus grande des valeurs passées en paramètre.
$\mathsf{min}\{\cdot\}$	opérateur qui retourne la plus petite des valeurs passées en paramètre.

Chapitre 1

Introduction générale

Dans cette thèse, nous nous concentrons sur le problème de la décomposition polyadique minimale de tenseurs de dimension trois. Selon les communautés scientifiques, on se réfère à ce problème sous différentes terminologies : “Polyadique Canonique” (PC ou CP en anglais), ou encore “CanDecomp”, “CanD” ou “Parafac”. Même si un modèle exact existe caractérisé par un nombre connu de paramètres, le calcul de la décomposition CP consiste à trouver les zéros d'un polynôme de degré six voire plus élevé, en présence d'un très grand nombre de variables. Il s'agit donc d'un problème numériquement très difficile à résoudre et ce en dépit du fait que le nombre de zéros reste fini. En outre, le modèle est sujet à des erreurs, c'est pourquoi on en cherche la meilleure approximation au sens d'un certain critère. Il est toutefois acquis aujourd'hui que la meilleure approximation peut ne pas toujours exister... Par ailleurs, dans plusieurs applications telles que l'imagerie hyperspectrale ou la chimiométrie par exemple, il est intéressant de contraindre les matrices de facteurs recherchées à être réelles et non négatives car elles sont représentatives des quantités physiques réelles et non négatives. Tel est le cas lorsque l'on cherche à estimer des spectres, des fractions d'abondance, des concentrations, ...etc. Si une telle contrainte rend le problème d'approximation considéré encore plus compliqué, son avantage majeur est de le ramener à un problème bien posé [LC09].
Le cadre applicatif considéré ici est celui de l'analyse de données environnementales pour des données particulières de type échantillons d'eau. Ces échantillons pourront avoir été collectés en différents endroits (mer, estuaires, rivières ou encore au niveau de processus industriels liés au traitement de l'eau) ou à différents instants. Ils contiennent un mélange de plusieurs molécules organiques et le but des traitements numériques qui seront mis en oeuvre sera de parvenir à séparer et à ré-estimer les composés de matières organiques présents dans les solutions considérées. Pour étudier ces échantillons d'eau prélevés, la technique classiquement utilisée depuis les travaux de Stedmon [SMB03a][SM03][SM05] consiste à coupler l'analyse de la luminescence totale ou spectroscopie de fluorescence à des algorithmes de décompositions multi-linéaires CP. Les spectres de luminescence totale sont également appelés “spectroscopie de fluorescence 3D” ou Matrices d'Emission-Excitation de Fluorescence (MEEF). Dans notre cas, ces signaux nous ont été fournis par le laboratoire PROTEE de l'USTV. Ils ont été mesurés à l'aide d'un spectrofluorimètre équipé d'une source d'excitation continue en fonction du temps. Un ensemble de MEEF constitue alors un tenseur d'ordre trois que l'on peut modéliser par des décompositions multi-linéaires (Candecomp/Parafac ou décompositions canoniques polyadiques (CP)). Dans cette application, les matrices de facteurs recherchées correspondent à de quantités physiques réelles et non négatives : spectres d'émission, spectres d'excitation et

concentrations des différents composés présents au niveau des divers échantillons de solutions.

C'est la raison pour laquelle, nous nous sommes donc intéressés au développement de nouveaux algorithmes de décomposition CP non négative pour des tenseurs d'ordre trois. La plupart des algorithmes existants assurent la contrainte de non négativité au moyen d'une projection. Nous avons donc exploré d'autres voies : la première approche que nous avons considérée consiste à utiliser des fonctions barrières (écrites sous forme de fonctions exponentielles dans notre cas) lesquelles jouent alors le rôle de terme de pénalisation en cas de violation de la contrainte de non négativité. Nous nous sommes également tournés vers d'autres approches inspirées des solutions proposées dans le cadre des décompositions matricielles non négatives. Elles consistent à prendre en compte explicitement la nature non négative des matrices de facteurs recherchées à travers la paramétrisation même du problème. Nous modélisons les quantités recherchées au moyen de "carrés" ce qui nous amène tout naturellement à introduire des produits de Hadamard. Une fois la fonction de coût ré-écrite et des quantités telles que les gradients matriciels voire les matrices Hessiennes à nouveau calculées, divers algorithmes d'optimisation ont alors pu être testés : tout d'abord le gradient conjugué non linéaire, bien adapté à des problèmes en grande dimension, combiné avec une recherche linéaire globale dans une direction (Enhanced Line Search). Cette combinaison permet de s'échapper de minima locaux. Mais d'autres algorithmes d'optimisation ont également été étudiés : approches de type gradient, Quasi-Newton (BGF et DFP) et enfin Levenberg-Marquardt. Deux versions de chacun de ces algorithmes ont systématiquement été considérées : la version "recherche linéaire globale" (Enhanced Line Search ou ELS) et la version "méthode par marche arrière" (ou backtracking, en alternance avec l'ELS). De plus, des versions plus générales de ces algorithmes ont également été développées afin de pouvoir tenir compte d'autres types de problèmes et/ou contraintes propres à l'application visée : prise en compte d'éventuelles données manquantes, parcimonie ou aspect creux des données. Différentes optimisations algorithmiques ont enfin été testées afin d'accélérer les vitesses de convergence des algorithmes (découpage du tenseur selon une direction, exploitation des matrices creuses sous Matlab, etc...). A des fins d'évaluation et de calibration, les différentes solutions proposées ont ensuite été testées sur des mélanges de composés connus effectués en laboratoire et comparées aux résultats obtenus au moyen des méthodes de "l'état de l'art".

Le manuscrit est donc organisé de la façon suivante. Après cette introduction générale, dans le chapitre deux, nous introduirons les outils mathématiques et les notions sur les tenseurs qui sont nécessaires aux calculs des gradients matriciels utilisés dans l'élaboration des nouveaux algorithmes de décomposition CP non négative. Nous y présenterons également le principe de la décomposition CP et de quelques algorithmes existants (sans contrainte de non négativité). Nous passerons ensuite à l'application considérée à savoir la spectroscopie de fluorescence 3D pour l'analyse de la composition de l'eau et à ses liens avec la décomposition CP. Dans le troisième chapitre, après quelques rappels sur l'état de l'art en matière de contrainte de non négativité, qu'il s'agisse de décompositions matricielles ou de tensorielles, nous présenterons les nouveaux algorithmes de décomposition CP que nous avons développés lesquels sont fondés sur des modifications de la fonction de coût considérée. Plusieurs approches possibles sont détaillées : introduction de fonctions barrières, modélisation au moyen de "carrés" d'où l'introduction de produits de Hadamard, éventuel ajout de termes de pénalisation. Dans chacun de ces cas, les gradients matriciels seront calculés, ce qui nous permettra de considérer en-

suite différents algorithmes d'optimisation itératifs (gradient, gradient conjugué non linéaire, Quasi-Newton, Levenberg-Marquardt). La question du choix du pas d'adaptation sera également traitée. Enfin, un algorithme de Tucker 3 généralisant l'un des algorithmes précédents sera également présenté. Toutes les méthodes détaillées dans ce chapitre seront ensuite comparées sur des exemples synthétiques de mélanges tri-linéaires. Le chapitre quatre sera quant à lui consacré au problème des données manquantes, qui pourrait se poser dans un contexte de traitement quasi temps-réel et in-situ des données. Là encore deux solutions seront proposées : l'une généralise un algorithme existant qui ne prenait en compte aucune contrainte de non négativité et considérait des poids binaires affectés à chaque entrée du tenseur de données. La seconde, plus robuste, considère des poids variables au cours des itérations. Dans le chapitre cinq, nous considérerons différentes optimisations algorithmiques afin d'accélérer les vitesses de convergence. Enfin dans le chapitre six, nous testerons les différents algorithmes proposés sur des mélanges de composés organiques connus réalisés en laboratoire . Nous montrerons l'apport des solutions proposées et leur robustesse vis-à-vis d'erreurs au niveau de l'estimation du nombre de composés présents dans les solutions notamment. Afin d'éliminer l'effet des diffusions Raman et Rayleigh, une nouvelle méthode de pré-traitement sera également introduite. Dans un dernier chapitre, enfin, nous discuterons des conclusions et perspectives de cette étude.

Chapitre 2

Décompositions tensorielles : état de l'art et applications en fluorimétrie

2.1 Introduction

Ce chapitre poursuit le double objectif de présenter à la fois les outils mathématiques exploités dans la suite du manuscrit et l'application à laquelle celui-ci est principalement dévolu. Ainsi, nous introduirons les outils et notations utilisées dans le cadre de l'algèbre tensorielle, dont nous rappellerons les principes et les propriétés qui seront exploitées par la suite. Ces notions permettront d'aborder la spectroscopie de fluorescence 3D et son lien avec la décomposition CP. Cette application en chimiométrie et analyse de données environnementales servira de base de tests et de comparaisons à tous les algorithmes que nous serons amenés à développer dans les prochains chapitres.

2.2 Outils

2.2.1 Produits usuels et outils de calcul

2.2.1.1 Produit de Kronecker

Le produit de Kronecker [Bre78] entre deux matrices $\mathbf{A} = (a_{ij}) = [\mathbf{a}_1, \mathbf{a}_2, \dots \mathbf{a}_F] \in \mathbb{R}^{I \times F}$ et $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_G] \in \mathbb{R}^{J \times G}$ est une matrice notée $\mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{IJ \times FG}$ et définie comme suit :

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots & a_{1F}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots & a_{2F}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \dots & a_{IF}\mathbf{B} \end{pmatrix} \tag{2.1}$$

Propriétés - Le produit de Kronecker présente les propriétés suivantes si l'on considère 3 matrices $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$, dont deux (les matrices $\mathbf{B}$ et $\mathbf{C}$) sont de la même taille, a désignant un scalaire :
$\succ$ Associatif *i.e.* $\mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C}$

≻ Distributif *i.e.* $\mathbf{A} \otimes (\mathbf{B} + \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) + (\mathbf{A} \otimes \mathbf{C})$ et $(\mathbf{B} + \mathbf{C}) \otimes \mathbf{A} = (\mathbf{B} \otimes \mathbf{A}) + (\mathbf{C} \otimes \mathbf{A})$
≻ Non commutatif *i.e.* $\mathbf{A} \otimes \mathbf{B} \neq \mathbf{B} \otimes \mathbf{A}$
≻ $(\mathbf{A} \otimes \mathbf{B})^T = \mathbf{A}^T \otimes \mathbf{B}^T$
≻ Si $\mathbf{A}$ et $\mathbf{B}$ sont non singulières, $(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$
≻ $(\mathbf{A} \otimes \mathbf{B})^{\dagger} = \mathbf{A}^{\dagger} \otimes \mathbf{B}^{\dagger}$
≻ $a\,(\mathbf{A} \otimes \mathbf{B}) = (a\mathbf{A}) \otimes \mathbf{B} = \mathbf{A} \otimes (a\mathbf{B})$

2.2.1.2 Produit de Khatri-Rao

Le produit de Khatri-Rao entre deux matrices possédant le même nombre de colonnes, $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_F] \in \mathbb{R}^{I \times F}$ et $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_F] \in \mathbb{R}^{J \times F}$, est défini comme le produit de Kronecker selon les colonnes :

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \quad \mathbf{a}_2 \otimes \mathbf{b}_2, \quad \mathbf{a}_F \otimes \mathbf{b}_F] \in \mathbb{R}^{IJ \times F}. \tag{2.2}$$

Propriétés - Il présente les propriétés suivantes si l'on considère 2 matrices $\mathbf{A}$ et $\mathbf{B}$ possédant le même nombre de colonnes et une troisième matrice $\mathbf{C}$ de la même taille que $\mathbf{B}$ ainsi qu'un scalaire a :
≻ Associatif *i.e.* $\mathbf{A} \odot (\mathbf{B} \odot \mathbf{C}) = (\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C}$
≻ Distributif *i.e.* $\mathbf{A} \odot (\mathbf{B} + \mathbf{C}) = (\mathbf{A} \odot \mathbf{B}) + (\mathbf{A} \odot \mathbf{C})$
≻ Non commutatif *i.e.* $\mathbf{A} \odot \mathbf{B} \neq \mathbf{B} \odot \mathbf{A}$
≻ $a\,(\mathbf{A} \odot \mathbf{B}) = (a\mathbf{A}) \odot \mathbf{B} = \mathbf{A} \odot (a\mathbf{B})$

2.2.1.3 Produit de Hadamard

Le produit de Hadamard entre 2 matrices $\mathbf{A}$ et $\mathbf{B}$ de même taille est le produit terme à terme entre chaque élément des matrices :

$$(\mathbf{A} \boxdot \mathbf{B})_{ij} = a_{ij} b_{ij} \tag{2.3}$$

Propriétés - Ce produit présente les propriétés suivantes si l'on considère 3 matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ de la même taille et un scalaire a :
≻ Associatif *i.e.* $\mathbf{A} \boxdot (\mathbf{B} \boxdot \mathbf{C}) = (\mathbf{A} \boxdot \mathbf{B}) \boxdot \mathbf{C}$
≻ Distributif *i.e.* $\mathbf{A} \boxdot (\mathbf{B} + \mathbf{C}) = (\mathbf{A} \boxdot \mathbf{B}) + (\mathbf{A} \boxdot \mathbf{C})$
≻ Commutatif *i.e.* $\mathbf{A} \boxdot \mathbf{B} = \mathbf{B} \boxdot \mathbf{A}$
≻ $(\mathbf{A} \boxdot \mathbf{B})^T = \mathbf{A}^T \boxdot \mathbf{B}^T = \mathbf{B}^T \boxdot \mathbf{A}^T$
≻ $a\,(\mathbf{A} \boxdot \mathbf{B}) = (a\mathbf{A}) \boxdot \mathbf{B} = \mathbf{A} \boxdot (a\mathbf{B})$
≻ $(\mathbf{A} \odot \mathbf{B})^T (\mathbf{A} \odot \mathbf{B}) = \mathbf{A}^T\mathbf{A} \boxdot \mathbf{B}^T\mathbf{B}$

2.2.1.4 Trace d'une matrice

La trace d'une matrice est simplement la somme de ses éléments diagonaux. Pour une matrice $\mathbf{A}$ carrée de dimension $N \times N$:

$$\mathsf{trace}\{\mathbf{A}\} = \sum_{i=1}^{N} a_{ii} \tag{2.4}$$

Propriétés - Elle présente des propriétés qui seront exploitées par la suite. Considérons trois matrices carrées $\mathbf{D}_1$, $\mathbf{D}_2$ et $\mathbf{D}_3$ de dimension $M \times M$ et quatre matrices rectangulaires $\mathbf{D}_4$, $\mathbf{D}_5$, $\mathbf{D}_6$ et $\mathbf{D}_7$ (de taille resp. $M \times N$, $N \times M$, $M \times N$ et $M \times N$), on a alors les propriétés suivantes [MN07] :

- $\mathsf{trace}\{\mathbf{D}_1 + \mathbf{D}_2\} = \mathsf{trace}\{\mathbf{D}_1\} + \mathsf{trace}\{\mathbf{D}_2\}$.
- $\mathsf{trace}\{\mathbf{D}_1\mathbf{D}_2\mathbf{D}_3\} = \mathsf{trace}\{\mathbf{D}_3\mathbf{D}_1\mathbf{D}_2\} = \mathsf{trace}\{\mathbf{D}_2\mathbf{D}_3\mathbf{D}_1\}$
  $\Rightarrow \mathsf{trace}\{\mathbf{D}_1\mathbf{D}_2\} = \mathsf{trace}\{\mathbf{D}_2\mathbf{D}_1\}$.
- $\mathsf{trace}\{\mathbf{D}_4\mathbf{D}_5\} = \mathsf{trace}\{\mathbf{D}_5\mathbf{D}_4\}$.
- $\mathsf{d}(\mathbf{D}_1^T) = (\mathsf{d}\mathbf{D}_1)^T$.
- $\mathsf{trace}\{\mathbf{D}_4^T(\mathbf{D}_6 \boxdot \mathbf{D}_7)\} = \mathsf{trace}\{(\mathbf{D}_4^T \boxdot \mathbf{D}_6^T)\mathbf{D}_7\}$. (2.5)

2.2.1.5 Opérateur de vectorisation

L'opérateur $\mathsf{vec}\,(.)$ permet de transformer une matrice en un vecteur. La convention choisie ici est d'empiler les colonnes de la matrice. Si $\mathbf{A} \in \mathbb{R}^{M \times N}$:

$$\mathsf{vec}\,(\mathbf{A}) = \begin{pmatrix} a_{11} \\ \dots \\ a_{M1} \\ \hline a_{12} \\ \dots \\ a_{M2} \\ \hline \dots \\ \hline a_{1N} \\ \dots \\ a_{MN} \end{pmatrix} \tag{2.6}$$

Propriétés La vectorisation permet de simplifier certains calculs en offrant des relations avec le produit de Kronecker ou encore la trace par exemple [Bre78, MN07]. Si on considère 4 matrices $\mathbf{A} \in M \times N$, $\mathbf{B} \in M \times N$, $\mathbf{C} \in M \times M$, $\mathbf{D} \in M \times M$ et 2 vecteurs $\mathbf{a}$ et $\mathbf{b}$ de taille quelconque :

- $\mathsf{vec}\left(\mathsf{a}\mathsf{b}^T\right) = \mathsf{b} \otimes \mathsf{a}$.
- $(\mathsf{vec}\,\mathbf{A})^T\,\mathsf{vec}\,\mathbf{B} = \mathsf{trace}\left(\mathbf{A}^T\mathbf{B}\right)$.
- $\mathsf{vec}(\mathbf{ABC}) = \left(\mathbf{C}^T \otimes \mathbf{A}\right)\mathsf{vec}(\mathbf{B})$ (N.B : cette relation reste vraie pour n'importe quelles matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, dès lors que le produit $\mathbf{ABC}$ est défini et carré).
- $\mathsf{trace}\,(\mathbf{ABCD}) = \left(\mathsf{vec}\,\mathbf{D}^T\right)^T \left(\mathbf{A} \otimes \mathbf{C}^T\right)\mathsf{vec}\left(\mathbf{B}^T\right)$. (N.B : cette relation reste vraie pour n'importe quelles matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, dès lors que le produit $\mathbf{ABCD}$ est défini et carré).

2.2.1.6 Produit scalaire et norme de matrices

Le produit scalaire de Frobenius est défini de la façon suivante : $\langle \mathbf{A}, \mathbf{B} \rangle = \mathsf{trace}\{\mathbf{A}^T\mathbf{B}\}$. Ce qui implique également que : $\langle \mathbf{A}, \mathbf{A} \rangle = \|\mathbf{A}\|_F^2 = \mathsf{trace}\{\mathbf{A}^T\mathbf{A}\}$
Nous serons amenés par la suite à utiliser les deux normes suivantes :

Norme L_2 : norme de Frobenius - Elle est définie de la façon suivante :

$$\|\mathbf{A}\|_F = \sqrt{\mathsf{trace}\,\{\mathbf{A}\mathbf{A}^T\}}. \tag{2.7}$$

Norme L_1 - On considère ici $\mathbf{A}$ comme élément d'espace vectoriel :

$$\|\mathbf{A}\|_1 = \sum_{i,j} |a_{ij}|. \tag{2.8}$$

2.2.1.7 Dérivée et gradient matriciel

On définit la différentielle d'une fonction f à variables matricielles $\mathbf{X}^{(1)}$, $\mathbf{X}^{(2)}$, $\ldots, \mathbf{X}^{(N)}$ comme :

$$\mathsf{d}f = \langle \frac{\partial f(\mathbf{X}^{(1)}, \ldots \mathbf{X}^{(N)}}{\partial \mathbf{X}^{(1)}}, \mathsf{d}\mathbf{X}^{(1)} \rangle + \ldots + \langle \frac{\partial f(\mathbf{X}^{(1)}, \ldots \mathbf{X}^{(N)})}{\partial \mathbf{X}^{(N)}}, \mathsf{d}\mathbf{X}^{(N)} \rangle \tag{2.9}$$

où $\frac{\partial f(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)}}{\partial \mathbf{X}^{(i)}}$, avec i variant de 1 à N, est la dérivée partielle de f par rapport à $\mathbf{X}^{(i)}$.
On définit les N gradients matriciels de cette fonction f par rapport à ses variables $\mathbf{X}^{(i)}$, pour i variant de 1 à N, comme :

$$\nabla_{\mathbf{X}^{(i)}} f(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)}) = \frac{\partial f(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)})}{\partial \mathbf{X}^{(i)}} \tag{2.10}$$

2.3 Décompositions tensorielles

2.3.1 Tenseurs

Un tenseur définit une application multilinéaire, et lorsque les bases des espaces sont fixées, il est associé à un tableau multi-dimensionnel.
Ainsi, un tenseur d'ordre trois peut être représenté par une somme de produits tensoriels entre trois vecteurs. Un tenseur d'ordre trois (ou du troisième ordre) peut ainsi être généré au moyen de 3 vecteurs et représenté par un tableau à 3 dimensions. On parle aussi de modes [SBG04]. L'ordre d'un tenseur correspond donc au nombre d'indices nécessaires pour identifier un de ses éléments.
Le produit tensoriel entre deux tenseurs $\mathbf{X}$ et $\mathbf{Y}$ de tailles $\mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ et $\mathbf{Y} \in \mathbb{R}^{J_1 \times J_2 \times \ldots \times J_M}$ est noté :

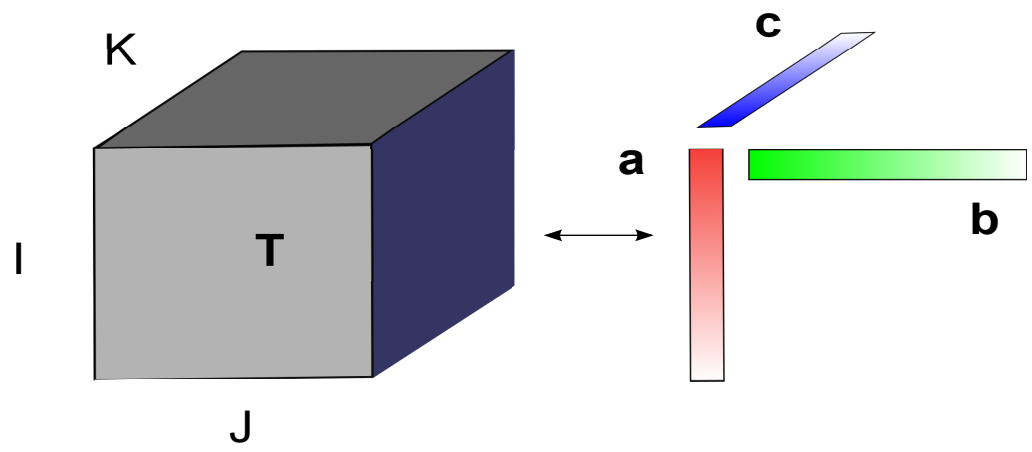


FIGURE 2.1 – Représentation visuelle de la génération d'un tenseur de rang 1.

$$\mathbf{Z} = \mathbf{X} \circledast \mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N \times J_1 \times J_2 \times \ldots \times J_M}, \tag{2.11}$$

dont chaque élément est alors défini par :

$$z_{i_1 i_2 \ldots i_N j_1 j_2 \ldots j_M} = x_{i_1 i_2 \ldots i_N} y_{j_1 j_2 \ldots j_M}. \tag{2.12}$$

Dans le cas particulier de trois vecteurs $\mathbf{a}$, $\mathbf{b}$ et $\mathbf{c}$ de taille respective $I \times 1$, $J \times 1$ et $K \times 1$, alors le produit tensoriel entre $\mathbf{a}$ et $\mathbf{b}$ donne naissance à une matrice $\mathbf{M} \in \mathbb{R}^{I \times J}$ de rang 1 telle que :

$$\mathbf{M} = \mathbf{a} \circledast \mathbf{b} = \mathbf{a}\mathbf{b}^T. \tag{2.13}$$

Le produit tensoriel entre les trois vecteurs $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ engendre le tenseur $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$ de rang 1 tel que :

$$\mathbf{T} = \mathbf{a} \circledast \mathbf{b} \circledast \mathbf{c}, \tag{2.14}$$

impliquant que les éléments de $\mathbf{T}$ sont données par la relation $t_{ijk} = a_i b_j c_k$ pour tout $i = 1, \ldots I$, pour tout $j = 1, \ldots J$ et pour tout $k = 1, \ldots K$. On peut en voir une représentation schématique sur la figure 2.1.

2.3.2 Dépliement

Un tenseur d'ordre trois peut être partitionné en tranches. Il existe alors trois types de tranches possibles : les tranches frontales, horizontales et verticales, comme le montre la figure 2.3. En juxtaposant ces tranches, on déplie le tenseur en matrice, comme on peut le voir sur la figure 2.2. Il existe 3 types généraux de dépliements, selon les 3 modes considérés. Si l'on considère

un tenseur $\mathbf{T}$ de taille $I \times J \times K$, le dépliement dans le premier mode mène à une matrice de taille $\mathbf{T}_{(1)}^{I,KJ}$. Les dépliements dans le mode deux et trois mènent respectivement à une matrice de taille $\mathbf{T}_{(2)}^{J,KI}$ et $\mathbf{T}_{(3)}^{K,JI}$. Notons que pour chaque mode, on peut modifier la direction selon laquelle on juxtapose les tranches. Les matrices résultantes sont toujours de la même taille que celles sus-citées, mais l'ordre dans lequel apparaissent les valeurs diffère.

2.3.3 Décomposition CP

2.3.3.1 Généralités

Si on considère un tenseur d'ordre trois, il admet une décomposition sous forme d'une somme de tenseurs de rang 1 (ceci est généralisable à n'importe quel tenseur d'ordre quelconque). Ainsi pour un tenseur $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$:

$$\mathbf{T} = \sum_{f=1}^{F} \mathbf{a}_f \circledast \mathbf{b}_f \circledast \mathbf{c}_f, \tag{2.15}$$

où les 3 matrices impliquées $\mathbf{A} = (a_{if}) = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_F] \in \mathbb{R}^{I \times F}$, $\mathbf{B} = (b_{jf}) = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_F] \in \mathbb{R}^{J \times F}$, $\mathbf{C} = (c_{kf}) = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_F]$ sont les matrices de facteurs, dont les colonnes sont appelées les facteurs. F est un nombre entier. Lorsque le nombre F de tenseurs de rang 1 nécessaires au maintien de cette égalité est minimal, on parle de décomposition canonique polyadique (CP), F représentant alors le rang du tenseur.
On peut réécrire de manière similaire cette relation en faisant intervenir les composantes des matrices de facteurs :

$$t_{ijk} = \sum_{f=1}^{F} a_{if} b_{jf} c_{kf}, \tag{2.16}$$

Cette décomposition est aussi parfois écrite en terme de tranches frontales. Pour les K tranches frontales :

$$\mathbf{T}_k = \mathbf{A}\mathbf{D}^{(k)}\mathbf{B}^T \tag{2.17}$$

où $\mathbf{D}^{(k)}$ est une matrice diagonale qui contient la $k^{\text{ième}}$ ligne de $\mathbf{C}$ sur sa diagonale.
La décomposition CP a été introduite en premier par F. L. Hitchcock dès 1927 sous l'appellation "décomposition polyadique" [Hit27]. Elle a été ensuite reprise indépendamment par J. Carroll et J-J. Chang sous la terminologie "Canonical Decomposition" (CanDecomp) [CC70] ou par Harshman sous la dénomination Parallel Factors (PARAFAC [Har70]. On utilise également l'acronyme équivalent CP.
Il est parfois plus pratique de supposer que les vecteurs sont de norme unité, et d'appliquer un facteur d'échelle λ_f à tous les tenseurs de rang 1, ce qui modifie de la manière suivante le modèle (2.15) :

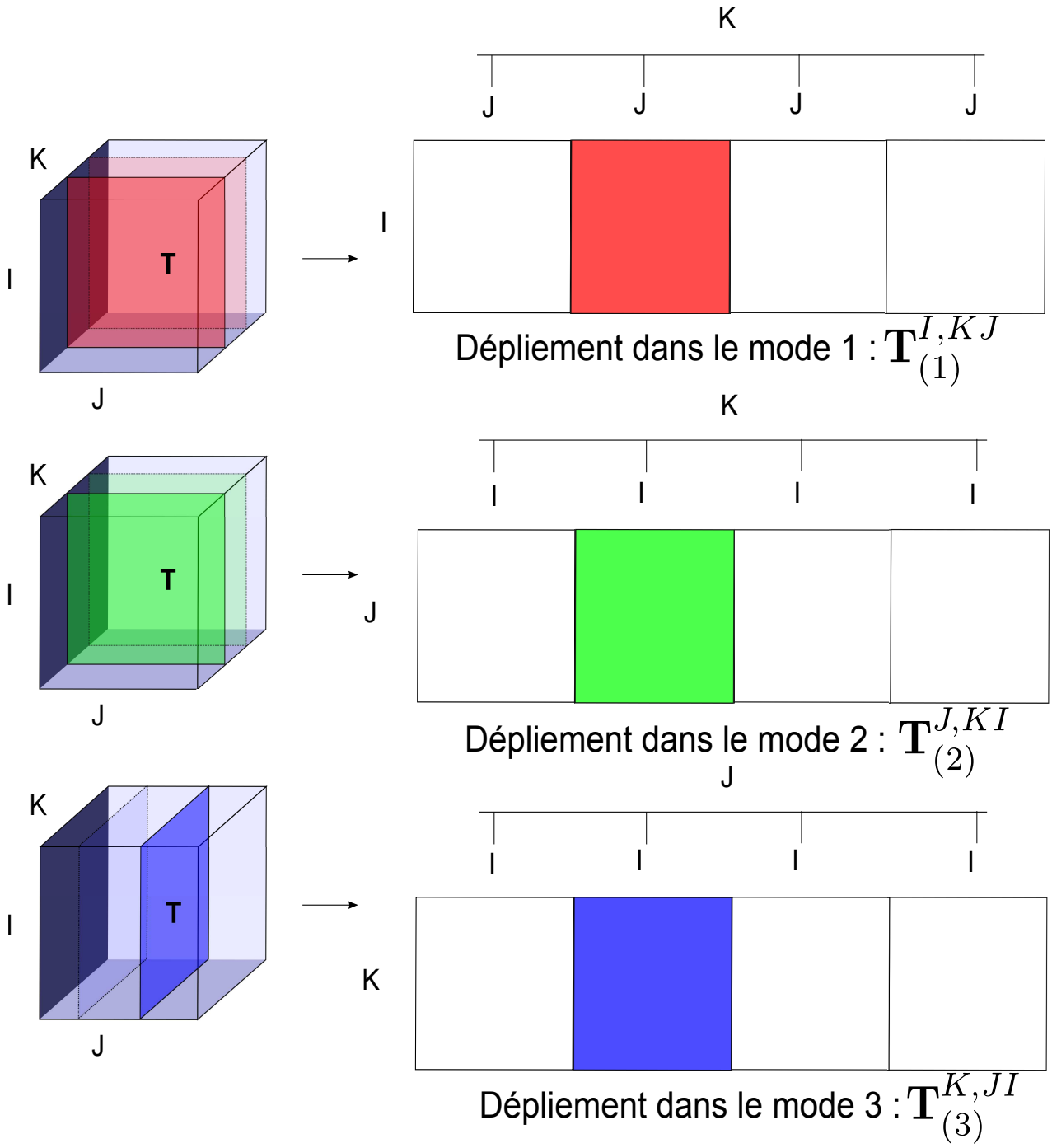


FIGURE 2.2 – Représentation visuelle du dépliement d'un tenseur d'ordre 3 en matrice.

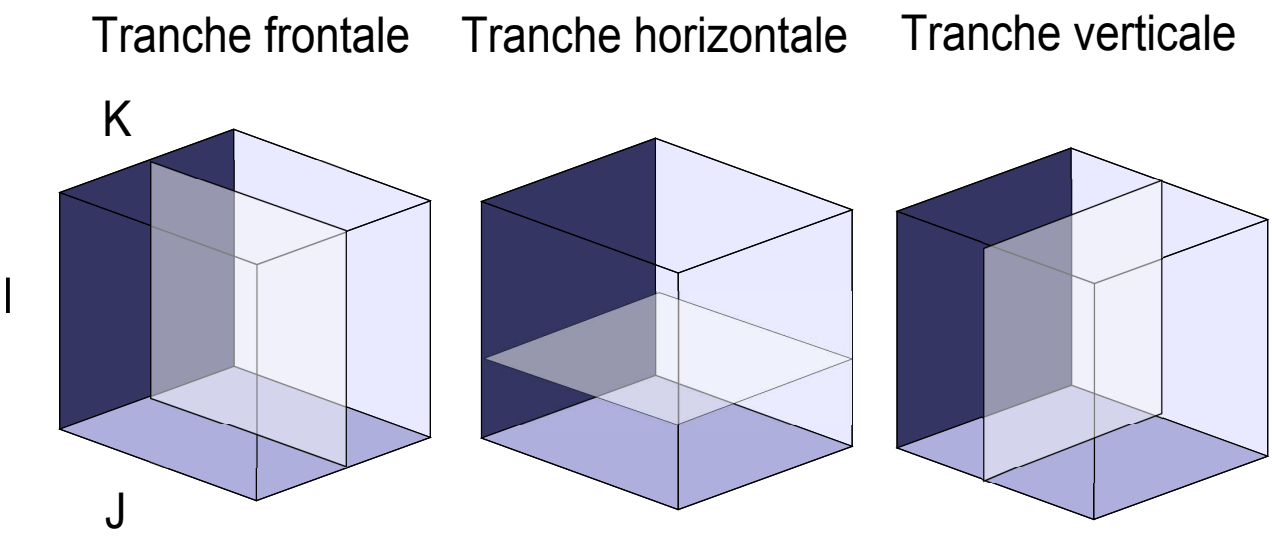


FIGURE 2.3 – Représentation des tranches d'un tenseur du 3ème ordre ($I \times J \times K$).

$$\mathbf{T} = \sum_{f=1}^{F} \lambda_f \, \mathbf{a}_f \circledast \mathbf{b}_f \circledast \mathbf{c}_f \tag{2.18}$$

où $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_F]^T$.
Les applications de cette décomposition sont très diverses [KB09, Com09, CLDA09, SBG04]. Elle a été utilisée en spectroscopie de fluorescence [SBG04, Luc07], dans l'industrie agro-alimentaire [PBdJ+02], en biomedical [FG91], en détection de cibles via radar MIMO [NS09], en traitement d'antennes [SMBG00, GMB+11], en séparation de signaux de parole [NMSP10], pour l'identification de signatures spectrales de matériaux en imagerie hyperspectrale [ZWPP08], en télécommunications [LdAC11, dAFM07], ...etc.

2.3.3.2 Estimation des matrices de facteurs

Le problème ici posé consiste donc à estimer les 3 matrices de facteurs $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$, en supposant que le rang F du tenseur est connu. Un moyen classique de modéliser ce problème consiste à se ramener à un problème d'optimisation en cherchant alors à minimiser une fonction de coût judicieusement choisie ($\mathcal{F}(\mathbf{A}, \mathbf{B}, \mathbf{C}; \boldsymbol{\Lambda})$). On écrit généralement ce problème comme un problème d'ajustement de modèle en choisissant alors de minimiser l'erreur quadratique :

$$\mathcal{F}(\mathbf{A}, \mathbf{B}, \mathbf{C}; \boldsymbol{\Lambda}) = \|\mathbf{T}_{(1)}^{I,KJ} - \mathbf{A}\boldsymbol{\Lambda}(\mathbf{C} \odot \mathbf{B})^T\|_F^2 \tag{2.19}$$
$$= \|\mathbf{T}_{(2)}^{J,KI} - \mathbf{B}\boldsymbol{\Lambda}(\mathbf{C} \odot \mathbf{A})^T\|_F^2 \tag{2.20}$$
$$= \|\mathbf{T}_{(3)}^{K,JI} - \mathbf{C}\boldsymbol{\Lambda}(\mathbf{B} \odot \mathbf{A})^T\|_F^2, \tag{2.21}$$

$\boldsymbol{\Lambda}$ est une matrice d'échelle de taille $F \times F$ dont la diagonale contient les λ_f pour $f = 1, \ldots, F$ intervenant au niveau de l'équation (2.18), elle s'écrit donc $\boldsymbol{\Lambda} = \mathsf{diag}\,(\lambda)$.

Notons qu'il est possible de permuter les matrices de facteurs du produit de Khatri-Rao. Comme évoqué dans la section 2.3.2, il faut alors modifier la direction de dépliement au sein du mode considéré.
Dans la littérature, on peut trouver plusieurs solutions pour résoudre ce problème d'optimisation (voir par exemple [TB06] pour une étude et une comparaison de plusieurs méthodes standard existantes). L'approche la plus populaire est d'appliquer la technique de des moindres carrés alternés (Alternating Least Squares ou ALS) [Bro98, CC70, Har70, JWLY99]. L'ALS est un algorithme assez lent à converger dans sa version de base. C'est pourquoi des améliorations de type recherche linéaire ont vu le jour dans [Bro97] et même recherche linéaire améliorée (ELS) [RCH08]. Le principe de la méthode est d'optimiser alternativement une fonction de coût par rapport à une des matrices de facteurs, les deux autres étant alors considérées comme fixes et indépendantes, ce qui est clairement sous-optimal.
Pour pouvoir utiliser d'autres types d'algorithmes d'optimisation, la différentielle $\mathsf{d}\mathcal{F}$ of $\mathcal{F}$ doit être calculée, et finalement, les gradients matriciels (la matrice $I \times F$ $\nabla_{\mathbf{A}}\mathcal{F}$, la matrice $J \times F$ $\nabla_{\mathbf{B}}\mathcal{F}$ et la matrice $K \times F$ $\nabla_{\mathbf{C}}\mathcal{F}$) peuvent être évalués.
On a (le cas $\mathbf{\Lambda} = \mathbf{I}_F$, où $\mathbf{I}_F$ est la matrice identité de taille $F \times F$, a été abondamment traité dans la littérature [CZPA09, Fra92]) :

$$\begin{aligned}
\nabla_{\mathbf{A}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= 2\left[-\mathbf{T}^{I,KJ}_{(1)} + \mathbf{A}\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{B})^T\right](\mathbf{C}\odot\mathbf{B})\mathbf{\Lambda} \\
&= 2\left(-\mathbf{T}^{I,KJ}_{(1)}(\mathbf{C}\odot\mathbf{B})\mathbf{\Lambda} + \mathbf{A}\mathbf{\Lambda}(\mathbf{C}^T\mathbf{C})\boxdot(\mathbf{B}^T\mathbf{B})\mathbf{\Lambda}\right), \qquad (2.22)\\
\nabla_{\mathbf{B}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= 2\left[-\mathbf{T}^{J,KI}_{(2)} + \mathbf{B}\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{A})^T\right](\mathbf{C}\odot\mathbf{A})\mathbf{\Lambda}, \\
&= 2\left(-\mathbf{T}^{J,KI}_{(2)}(\mathbf{C}\odot\mathbf{A})\mathbf{\Lambda} + \mathbf{B}\mathbf{\Lambda}(\mathbf{C}^T\mathbf{C})\boxdot(\mathbf{A}^T\mathbf{A})\mathbf{\Lambda}\right) \qquad (2.23)\\
\nabla_{\mathbf{C}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= 2\left[-\mathbf{T}^{K,JI}_{(3)} + \mathbf{C}\mathbf{\Lambda}(\mathbf{B}\odot\mathbf{A})^T\right](\mathbf{B}\odot\mathbf{A})\mathbf{\Lambda} \\
&= 2\left(-\mathbf{T}^{K,JI}_{(3)}(\mathbf{B}\odot\mathbf{A})\mathbf{\Lambda} + \mathbf{C}\mathbf{\Lambda}(\mathbf{B}^T\mathbf{B})\boxdot(\mathbf{A}^T\mathbf{A})\mathbf{\Lambda}\right), \qquad (2.24)
\end{aligned}$$

En égalisant les composantes du gradient à 0, on obtient la solution suivante :

$$\widehat{\mathbf{A}} = \mathbf{T}^{I,JK}_{(1)}(\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{B})^T)^\dagger, \qquad (2.25)$$
$$\widehat{\mathbf{B}} = \mathbf{T}^{J,KI}_{(2)}(\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{A})^T)^\dagger, \qquad (2.26)$$
$$\widehat{\mathbf{C}} = \mathbf{T}^{K,JI}_{(3)}(\mathbf{\Lambda}(\mathbf{B}\odot\mathbf{A})^T)^\dagger. \qquad (2.27)$$

2.3.3.3 Problèmes d'unicité

Un avantage de la décomposition CP est qu'elle garantit sous certaines conditions que la solution du problème de minimisation soit unique. Par unique, on sous-entend dans toute la suite "essentiellement unique", c'est à dire que la solution est trouvée à une permutation près sur les colonnes des matrices de facteurs de même qu'à un facteur d'échelle près (qui se réduit à un signe si les colonnes sont de norme unité).

Rang de Kruskal Le rang de Kruskal, encore appelé k-rang, est un nouveau concept de rang nommé dans [HL84], suite à une publication de Kruskal qui en fait mention dans [Kru77] sans le définir.
On définit le k-rang de $\mathbf{A}$ comme *le nombre maximum k tel que chaque ensemble de k colonnes de* $\mathbf{A}$ *est linéairement indépendant.*

Exemple 1 :

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 5 \\ 2 & 4 & 7 \\ 3 & 6 & 5 \end{pmatrix} \tag{2.28}$$

$\mathbf{A}$ est de rang 2. Tous les ensembles de 2 colonnes ne sont pas linéairement indépendants (les vecteurs associés à la $1^{\text{ère}}$ et à la $2^{\text{ème}}$ colonne sont colinéaires). Il n'y a que les ensembles de 1 colonne qui sont tous linéairement indépendants. Donc le k-rang de $\mathbf{A}$ est 1.
Exemple 2 :

$$\mathbf{A} = \begin{pmatrix} 2 & 1 & 1 \\ 5 & 2 & 3 \\ 8 & 3 & 5 \end{pmatrix} \tag{2.29}$$

$\mathbf{A}$ est de rang 2 (la $1^{\text{ère}}$ colonne est la somme de la $2^{\text{ème}}$ et de la $3^{\text{ème}}$ colonne). Il n'y a qu'un ensemble de 3 colonnes (la matrice elle-même, et elle n'est pas de rang plein). Par contre, tous les ensembles de 2 colonnes sont linéairement indépendants. Donc son k-rang est 2.

Condition de Kruskal - Kruskal a établi en 1977 une condition suffisante qui garantit l'unicité de la décomposition CP pour les tenseurs d'ordre 3 [Kru77] laquelle s'écrit :

$$2F + 2 \leq \mathsf{min}(I, F) + \mathsf{min}(J, F) + \mathsf{min}(K, F) \tag{2.30}$$

En effet, pour des matrices tirées aléatoirement selon une distribution continue, le k-rang et le rang coincident avec probabilité 1, et le rang de $\mathbf{A}$ par exemple vaut $\mathsf{min}(I, F)$.
La condition (2.30) est souvent réécrite de la manière suivante [HL84, SB00, SS07] :

$$2F + 2 \leq k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \tag{2.31}$$

où $k_{\mathbf{A}}$, $k_{\mathbf{B}}$ et $k_{\mathbf{C}}$ représentent les k-rangs des matrices de facteurs.
Cette preuve de l'unicité de la solution a par la suite été étendue à des tenseurs d'ordre supérieur à 3 [SB00].
De façon intuitive, on peut démontrer que la décomposition CP ne souffre pas de certaines ambiguïtés, comme l'ambiguïté de rotation que présente l'ACP si on n'impose pas de contraintes particulières, comme l'orthogonalité des axes. Comme expliqué dans [Bro98], si on considère un modèle bilinéaire à F composants :

$$\mathbf{X} = \mathbf{A}\mathbf{B}^T, \tag{2.32}$$

alors, n'importe quelle matrice $\mathbf{T}$ de taille $F \times F$ non singulière appliquée à (2.32) de la façon suivante

$$\mathbf{X} = \mathbf{A}\mathbf{T}\mathbf{T}^{-1}\mathbf{B}^T \tag{2.33}$$

reconstruit la matrice $\mathbf{X}$. Donc, les *scores* $\mathbf{AT}$ et *loadings* $\mathbf{B}\left(\mathbf{T}^{-1}\right)^T$ sont un choix tout autant justifiable que $\mathbf{A}$ et $\mathbf{B}$.
Considérons maintenant le modèle CP écrit en (2.17). En prenant des matrices $\mathbf{T}$ et $\mathbf{P}$, toutes deux de taille $F \times F$ et non singulières, on peut écrire la relation suivante sans changer la matrice $\mathbf{X}$:

$$\mathbf{X} = \mathbf{A}\mathbf{T}\mathbf{T}^{-1}\mathbf{D}^{(k)}\mathbf{P}\mathbf{P}^{-1}\mathbf{B}^T. \tag{2.34}$$

On se retrouverait donc avec les matrices de facteurs $\mathbf{AT}$, $\mathbf{T}^{-1}\mathbf{D}^{(k)}\mathbf{P}$ et $\mathbf{P}^{-1}\mathbf{B}^T$.
Or, $\mathbf{T}^{-1}\mathbf{D}^{(k)}\mathbf{P}$ doit être diagonale, ce qui implique que $\mathbf{T}$ et $\mathbf{P}$ doivent être choisies comme matrices de permutations ou de facteurs d'échelle. De cette manière, on ne peut donc être confronté qu'à des indéterminations de permutations ou d'échelle, qui, comme dit précédemment, ne sont pas un obstacle à la détermination d'une solution unique.

2.3.3.4 Problèmes de convergence

Le comportement d'algorithmes numériques a été classifié dans [KHL89] puis à nouveau considéré dans [CLDA09]. En effet, il arrive parfois que l'on observe au niveau des courbes de convergence des ralentissements ou des stagnations, qui sont attribuées à la dégénérescence du tenseur :
- ≻ Un *bottleneck* : lorsque 2 facteurs ou plus d'un mode sont pratiquement colinéaires.
- ≻ Un *swamp* : lorsque qu'un *bottleneck* existe dans tous les modes.
- ≻ *CP-degeneracies* : cas particulier des *swamps*, lorsque certains facteurs divergent mais tendent à s'annuler entre leurs contributions par le biais de signes opposés.

2.4 Spectroscopie de fluorescence

La spectroscopie de fluorescence est un type particulier de spectroscopie dont l'objet est d'analyser le comportement et les propriétés de fluorescence d'un échantillon.
La phénomène de fluorescence se produit lors de l'excitation d'une molécule par une onde électromagnétique. Cette énergie est d'abord absorbée par la molécule, qui sort alors de son état fondamental pour passer dans un état excité instable. Le retour à l'état initial se fait par émission d'un photon. On peut en voir une représentation sur la figure 2.4.
La fluorescence est à distinguer de la phosphorescence par le fait qu'il s'agit d'un processus à durée courte, le retour à l'état fondamental ne s'effectuant pas de la même manière.

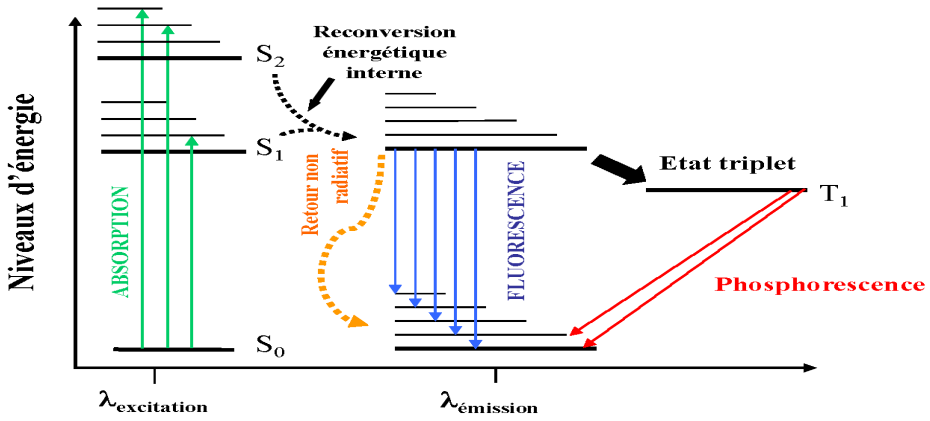


FIGURE 2.4 – Diagramme de Jablonski. Image tirée de [Mau].

Dans toute la suite de ce manuscrit, les échantillons analysés seront constitués de plusieurs composés de matière organique naturelle dissoute dans un solvant. Ces composés fluorescents sont aussi appelés fluorophores. Une étude plus poussée de l'origine de ces éléments ainsi que de leur évolution peut être trouvée dans [Zha11].

2.4.1 Fluorescence

Quand une solution contenant des échantillons de matière fluorescente est éclairée par une source lumineuse, il y a aussi production de deux effets de diffusion Rayleigh et Raman, qui sont des perturbations inévitables, dues aux chocs entre les photons incidents et le milieu, qui sont élastiques pour la première (conservation de la longueur d'onde incidente), et inélastiques pour la deuxième (non conservation de la longueur d'onde). Les phénomènes de diffusion masquent souvent la fluorescence que l'on veut mesurer. L'objectif des travaux présentés dans ce manuscrit est d'analyser des images de fluorescence. Dans notre cas, ces images nous ont été fournis par le laboratoire PROTEE de l'USTV. Elles ont été acquises à l'aide d'un spectrofluorimètre équipé d'une source d'excitation continue en fonction du temps dont le schéma de principe est donné au niveau de la figure 2.5. Préalablement à tout autre traitement, les raies de diffusion forcément présentes dans toutes les images de fluorescence acquises (voir figure 2.6) devront être éliminées. Pour ce faire, plusieurs techniques existent, la plus utilisée habituellement restant celle présentée dans [ZSM04]. Nous la détaillerons un peu plus loin.
La relation entre l'intensité de fluorescence, la longueur d'onde d'excitation, et la longueur d'onde d'émission est exprimée par la loi de Beer-Lambert. Pour de faibles concentrations, cette loi peut être linéarisée, et s'écrit alors :

$$I(\lambda_e, \lambda_f) = I_0 \epsilon(\lambda_e) \gamma(\lambda_f) c, \tag{2.35}$$

où I_0 est une constante. λ_e est la longueur d'onde dite d'excitation, c'est à dire celle transmise à l'échantillon lors de l'éclairement. λ_f est la longueur d'onde d'émission. ϵ est le spectre

d'excitation relatif. γ est le spectre d'émission relatif. c représente la concentration du composé organique.
Quand plusieurs composés sont présents dans une solution, cette loi est modifiée comme suit :

$$I(\lambda_e, \lambda_f) = I_0 \sum_{l=0}^{L} \epsilon_l(\lambda_e)\gamma_l(\lambda_f)c_l. \tag{2.36}$$

Ici, l est l'indice du composé considéré. On peut également étendre cette relation en ajoutant une dimension temporelle supplémentaire (ou spatiale). En effet, si on dispose de plusieurs échantillons d'eau prélevés à différents moments ou à différents endroits, le spectre 3D est alors défini comme :

$$I(\lambda_e, \lambda_f, k) = I_0 \sum_{l=0}^{L} \epsilon_l(\lambda_e)\gamma_l(\lambda_f)c_{k,l}, \tag{2.37}$$

avec k représentant le numéro de l'échantillon.

2.4.2 MEEF

Exciter une solution contenant des fluorophores à l'aide d'une longueur d'onde incidente permet de mesurer un spectre émis en retour par fluorescence. Balayer une plage de N longueurs d'onde d'excitation permet de construire une matrice qu'on appelle Matrice d'Emission-Excitation de Fluorescence ou MEEF. Elle correspond aux spectres de luminescence totale encore appelés "spectroscopie de fluorescence 3D". L'image de fluorescence résultante provient donc de la combinaison d'un mélange de plusieurs fluorophores présents dans l'échantillon considéré.
La génération de ces MEEF se fait à l'aide d'un spectrofluorimètre (cf. figure 2.5), dont le principe simplifié est le suivant : un faisceau incident est généralement issu d'une lampe au xénon (il existe d'autres dispositifs d'émission), traversant un monochromateur sélectionnant finement la longueur d'onde excitatrice. L'intensité de la lampe est correcte à partir de 250 nm. En dessous, le rapport signal à bruit se dégrade, comme on pourra le voir sur certaines figures de la section 4. L'onde excitatrice λ_e atteint l'échantillon à observer, qui fluoresce en réaction. Sa lumière émise est captée par un monochromateur d'émission, semblable à celui d'excitation, et dont la fente ne laisse passer que la longueur d'onde λ_f choisie. Pour récupérer un spectre d'émission, il faut donc faire varier les longueurs d'ondes acceptées en modifiant la position angulaire du réseau de diffraction du monochromateur. La valeur du spectre d'émission pour chaque λ_f est mesurée à partir du photomultiplicateur, qui reçoit l'onde issue du monochromateur, normalisée par la tension délivrée par la photodiode.

2.4.3 Diffusion

Comme énoncé dans la section 2.4.1, les raies de diffusion Raman et Rayleigh sont des perturbations qui surviennent lors de l'acquisition de l'image de fluorescence. En effet, elles ne résultent pas de l'absorption de photons, mais des collisions de ces derniers avec les molécules présentes dans le milieu.

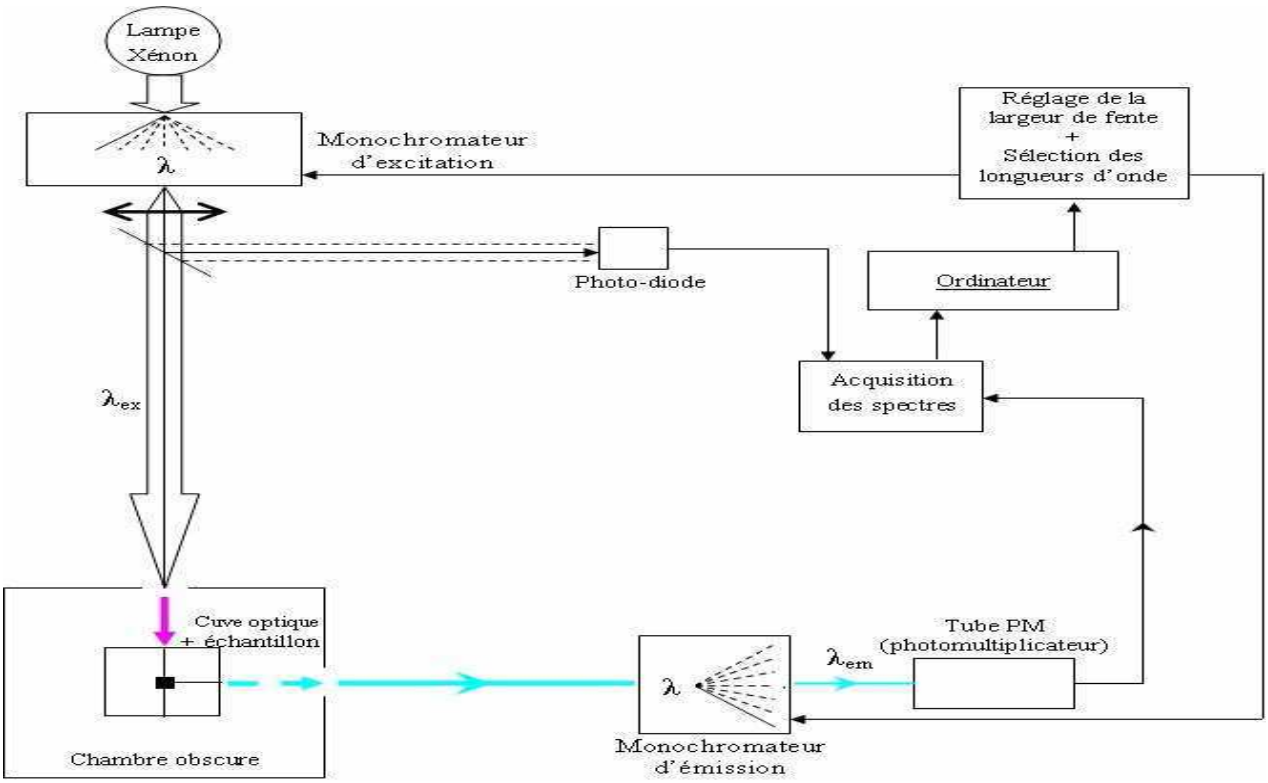


FIGURE 2.5 – Spectrofluorimètre. Image tirée de [Zha11].

La diffusion Rayleigh apparaît dans les zones où $\lambda_e \approx \lambda_f$, celle de Raman dans les zones où $\lambda_f \approx 2\lambda_e$. La forte intensité du Rayleigh, et celle moindre du Raman étant nuisibles aux méthodes d'estimation des composés organiques dissous, puisqu'elles masquent le signal utile, il est nécessaire de pré-traiter les MEEF avant de pouvoir les utiliser. Plusieurs méthodes de traitements existent dans la littérature. Nous détaillons ici celle énoncée dans [ZSM04]. Une méthode inédite par filtrage morphologique d'images avec un élément structurant judicieusement choisi sera introduite dans le chapitre 6. Nous la comparerons à la méthode précédente sur des exemples réels.

Méthode de Zepp

1. La première étape consiste à modéliser l'emplacement et le comportement des raies de diffusion à l'aide d'un système d 4 équations du second degré, comme on peut le voir sur l'image de gauche de la figure 2.7.

2. Dans un deuxième temps, on élimine tous les pixels présents aux emplacements marqués.

3. Pour finir, on reconstruit par interpolation (triangulation de Delaunay) les pixels manquants.

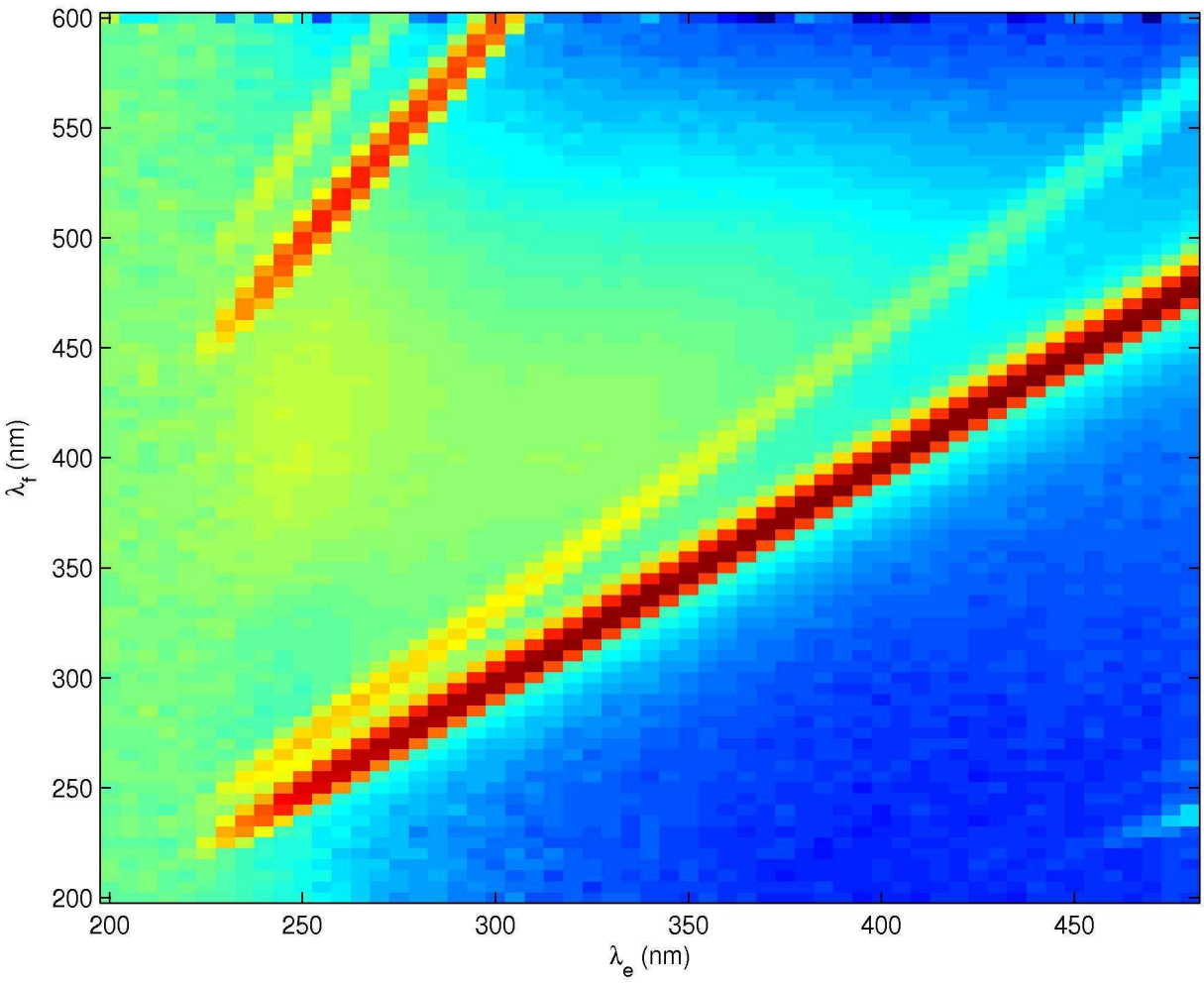


FIGURE 2.6 – Diffusions Rayleigh et Raman. Le signal d'intérêt est masqué du fait de la forte intensité de ces deux raies.

2.4.4 Liens entre écriture tensorielle et spectroscopie de fluorescence

Rappelons ici l'expression du modèle trilinéaire donné par la décomposition CP, pour un tenseur du 3ème ordre $\mathbf{X}$:

$$x_{ijk} = \sum_{f=0}^{F} a_{if} b_{jf} c_{kf}, \tag{2.38}$$

Si on compare cette relation avec celle donnée au niveau de l'équation (2.37), et grâce à l'unicité de la décomposition CP , on peut alors identifier, par analogie :

- F équivaut au nombre de composés L.
- $\mathbf{a}_f$ représente le spectre d'excitation (ϵ_l) relatif d'un composé, et donc, $\mathbf{A} \in \mathbb{R}^{I \times F}$ est la matrice qui contient les spectres d'excitation relatifs de tous les composés, rangés en

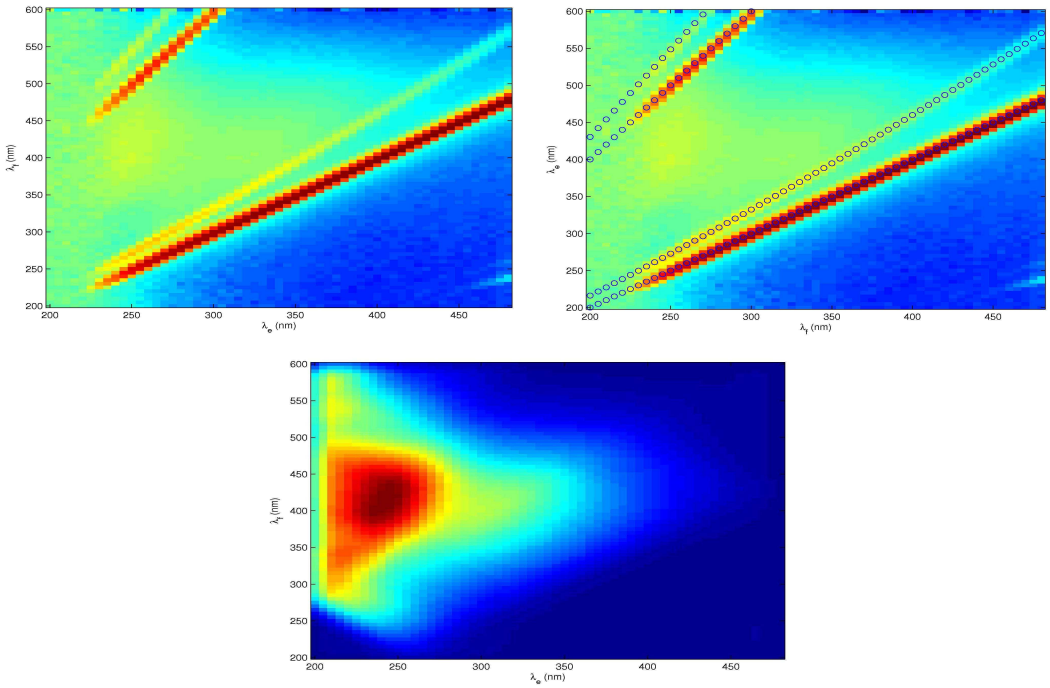


FIGURE 2.7 – Présentation des différentes étapes de l'algorithme de Zepp. En haut à gauche, l'image d'origine, à droite, l'image avec les zones à corriger. En bas, l'image résultante, après suppression des raies de diffusion, et interpolation des pixels manquants.

colonnes.

- ≻ $\mathbf{b}_f$ représente le spectre d'émission relatif (γ_l) d'un composé, et donc, $\mathbf{B} \in \mathbb{R}^{J \times F}$ est la matrice qui contient les spectres d'excitation relatifs de tous les composés, rangés en colonnes.
- ≻ $\mathbf{c}_f$ représente le vecteur de concentrations d'un composé à travers les k échantillons du tenseur de données, à une valeur d'échelle près. Donc, $\mathbf{C} \in \mathbb{R}^{K \times F}$ est la matrice qui contient les vecteurs de concentrations de tous les composés, rangés en colonnes.
- ≻ La longueur d'onde d'excitation λ_e correspond à la variable i.
- ≻ La longueur d'onde d'émission λ_f correspond à la variable j.

De manière équivalente, pour chaque tranche $\mathbf{X}_k$ du tenseur $\mathbf{X}$:

$$\mathbf{X}_k = \sum_{f=0}^{F} \mathbf{S}_f c_{kf}, \tag{2.39}$$

où $\mathbf{S}_f$ représente la MEEF de chacun des F composés. Après décomposition CP, ces dernières peuvent ainsi être reconstruites à partir du produit $\mathbf{a}_f \mathbf{b}_f^T$ où les F $\mathbf{a}_f$ et $\mathbf{b}_f$ représentent les colonnes des matrices de facteurs $\mathbf{A}$ et $\mathbf{B}$.

On peut également reconstruire directement les F MEEF par la relation :

$$\mathbf{M} = (\mathbf{B} \odot \mathbf{A})^T \tag{2.40}$$

M est alors une matrice de taille $F \times IJ$. Chaque ligne de **M** contient l'ensemble des pixels de la MEEF d'un composé, dont l'image a été vectorisée, et qu'il convient de réagencer pour récupérer la matrice souhaitée.

2.4.5 Effet d'écran

Afin de s'assurer du fait que le jeu de données expérimentales suive bien un modèle trilinéaire, il est utile voire nécessaire d'appliquer des pré-traitements. Nous avons vu dans la section 2.4.3 qu'il était nécessaire d'éliminer au préalable les diffusions Raman et Rayleigh des MEEF .

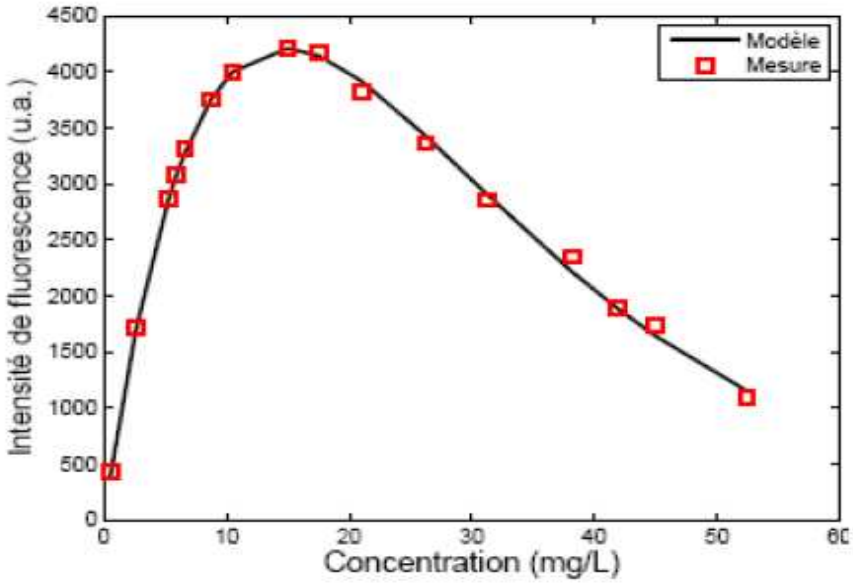


FIGURE 2.8 – Effet de la concentration sur l'intensité de fluorescence. La zone linéaire n'existe que pour des valeurs inférieures à environ 10 mg.L^{-1}. Image tirée de [Luc07].

Il faut également envisager le cas où les fluorophores ne sont pas en faible concentration. Dans ce cas, la loi de Beer-Lambert ne peut pas être linéarisée, et le modèle CP n'est plus valide. De plus, d'autres phénomènes, comme la réabsorption du spectre de fluorescence émis par la solution analysée ne sont pas à négliger. On peut voir un tel exemple au niveau de la figure 2.8. Quand la concentration dépasse un certain seuil dépendant du fluorophore, l'intensité de fluorescence n'est plus une fonction linéaire de la concentration.
Plusieurs méthodes ont été développées pour corriger l'effet d'écran. Citons par exemple, la correction par absorbance [SMB03b], et plus récemment, la correction par dilution [Luc07, LMRB09].
Cette dernière méthode donne de bons résultats, est simple à mettre en oeuvre et élimine l'effet d'écran de façon plus efficace que la correction par absorbance dans le cas de solutions soumises à un fort effet d'écran. Elle nécessite d'utiliser 2 MEEF d'un même échantillon à deux dilutions différentes, ce qui permet alors de calculer une MEEF corrigée de façon analytique.
Si on modélise la cuve contenant la solution en volumes élémentaires, on peut établir une expression liant l'intensité de fluorescence émise en fonction d'une partie linéaire constituée

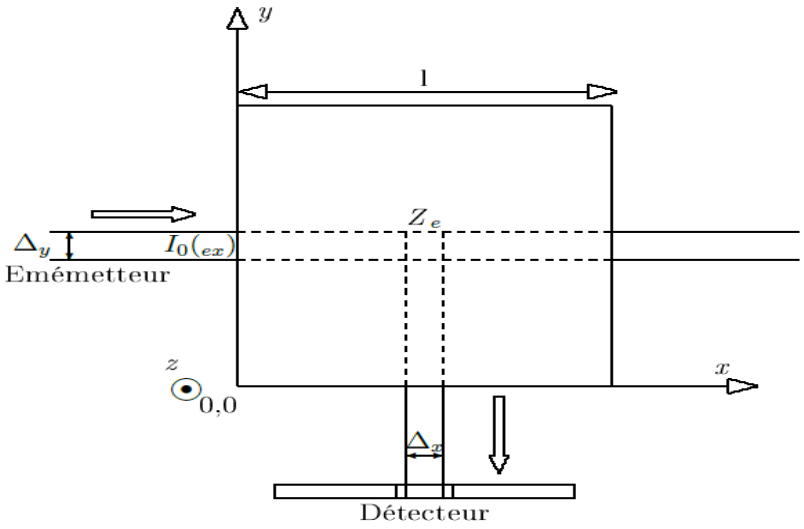


FIGURE 2.9 – Modélisation de la vue de dessus de la cuve contenant l'échantillon à analyser. Image tirée de [Luc07].

du coefficient d'absorption du composé n, α_n, du rendement quantique de fluorescence Φ_n du composé n, du spectre d'émission normé γ_n du composé n, d'une partie non linéaire et d'autres variables modélisant la géométrie de la cuve (cf. figure 2.9) :

$$I_{3D}(\lambda_e, \lambda_f) = I_0 \Delta_x \left(\sum_{n=1}^{N} \alpha_n(\lambda_e) \Phi_n \gamma_n(\lambda_f) \right) e^{-\frac{\alpha(\lambda_e) l}{2}} e^{-\frac{\alpha(\lambda_f) l}{2}} \tag{2.41}$$

Si on pose $g = \dfrac{l}{2}$ et $G_n = I_0 \Delta_x \Phi_n$, l'expression devient :

$$I_{3D}(\lambda_e, \lambda_f) = \left(\sum_{n=1}^{N} G_n c_n \epsilon_n(\lambda_e) \gamma_n(\lambda_f) \right) \prod_{n=1}^{N} e^{-g\left(c_n \epsilon(\lambda_e) + c_n \epsilon_n(\lambda_f)\right)}. \tag{2.42}$$

ϵ_n représente alors le coefficient d'absorption molaire normé, et c_n la concentration du composé n. Ce modèle mathématique caractérisant l'intensité de fluorescence d'une solution quelconque peut être réécrit de la manière suivante :

$$I_{3D}(\lambda_e, \lambda_f) = L(\lambda_e, \lambda_f) H(\lambda_e, \lambda_f), \tag{2.43}$$

avec $L(\lambda_e, \lambda_f)$ correspondant à la partie linéaire, et $H(\lambda_e, \lambda_f)$ à la déviation non linéaire due à l'effet d'écran. L'objectif est alors de pouvoir déterminer cette partie linéaire, ce qui permet d'obtenir une MEEF corrigée.
En notant I_{3D} l'échantillon d'origine, et I_{3D_p} le même échantillon dilué d'un facteur p (souvent choisi faible, de sorte à conserver le rapport signal à bruit), on peut écrire :

$$I_{3D}\left(\lambda_e,\lambda_f\right)=\left(\sum_{n=1}^{N}G_n c_n \epsilon_n\left(\lambda_e\right)\gamma_n\left(\lambda_f\right)\right)\prod_{n=1}^{N}e^{-g\left(c_n\epsilon(\lambda_e)+c_n\epsilon_n\left(\lambda_f\right)\right)} \tag{2.44}$$

$$I_{3D_p}\left(\lambda_e,\lambda_f\right)=\frac{1}{p}\left(\sum_{n=1}^{N}G_n c_n \epsilon_n\left(\lambda_e\right)\gamma_n\left(\lambda_f\right)\right)\left(\prod_{n=1}^{N}e^{-g\left(c_n\epsilon(\lambda_e)+c_n\epsilon_n\left(\lambda_f\right)\right)}\right)^{\frac{1}{p}}, \tag{2.45}$$

ce qui se simplifie en :

$$I_{3D}\left(\lambda_e,\lambda_f\right)=L\left(\lambda_e,\lambda_f\right)H\left(\lambda_e,\lambda_f\right) \tag{2.46}$$

$$I_{3D_p}\left(\lambda_e,\lambda_f\right)=\frac{1}{p}L\left(\lambda_e,\lambda_f\right)H^{\frac{1}{p}}\left(\lambda_e,\lambda_f\right) \tag{2.47}$$

On obtient donc un système d'équations dont on tire les valeurs de H, et surtout de L :

$$H\left(\lambda_e,\lambda_f\right)=\left(\frac{I_{3D}\left(\lambda_e,\lambda_f\right)}{pI_{3D_p}\left(\lambda_e,\lambda_f\right)}\right)^{\frac{p}{p-1}} \tag{2.48}$$

$$L\left(\lambda_e,\lambda_f\right)=\left(\frac{\left(pI_{3D_p}\left(\lambda_e,\lambda_f\right)\right)^{p}}{I_{3D}\left(\lambda_e,\lambda_f\right)}\right)^{\frac{1}{p-1}} \tag{2.49}$$

Dans l'équation (2.49), L permet alors de reconstruire la MEEF linéarisée. Notons que si I_{3D}, et par conséquent I_{3D_p}, sont choisies dans la zone linéaire, alors les termes de déviation $H\left(\lambda_e,\lambda_f\right)$ sont idéalement égaux à 1.

Chapitre 3

Algorithmes de factorisation tensorielle sous contrainte de non négativité

3.1 Introduction

Il est utile de considérer dans certaines applications des algorithmes prenant en compte la nature non négative des données. En effet, comme cela est dit dans [LC09] la décomposition CP mène à un problème mal posé (nous rappelons que l'acronyme CP est utilisé pour désigner aussi bien la décomposition **C**andecomp/**P**arafac qu'en remplacement de la terminologie anglaise **C**anonical **P**olyadic decomposition). L'estimation des matrices de facteurs peut devenir alors instable. Les symptômes d'un tenseur dégénéré (*degenerate array*) apparaissent quand deux colonnes colinéaires sont calculées avec des signes opposés, amenant les contributions à s'annuler entre elles [CLDA09, Paa00]. Un tel phénomène ne peut apparaître lors de l'estimation de matrices de facteurs non négatives, transformant par la même le problème sus-cité en un problème bien posé.
Il est également pertinent de considérer la nature non négative des quantités que l'on cherche à estimer. Certaines applications comme la spectroscopie de fluorescence [Bro97, Bro98, SBG04], ou l'imagerie multispectrale et hyperspectrale [KC09, ZWPP08], nécessitent de manipuler des spectres ou des concentrations (respectivement des spectres et des fractions d'abondance) qui sont par nature des grandeurs positives ou nulles. Un livre, paru récemment [CZPA09], est d'ailleurs entièrement consacré à une grande partie des approches matricielles et tensorielles permettant la prise en compte de cet aspect.
Diverses approches ont été proposées afin de tenir compte de la non négativité des données. Nous allons maintenant rappeler les plus connues d'entre elles, avant d'attaquer la présentation des solutions que nous avons développées.

3.2 Quelques rappels sur les approches existantes

3.2.1 Approches pour les matrices

Il peut être intéressant à ce stade d'évoquer les méthodes existant pour la factorisation de matrices non négatives, couramment appelée NMF (Nonnegative Matrix Factorization). La factorisation de matrices trouve de nombreuses applications, par exemple en reconnaissance

de caractéristiques de visages [LS99, HD04], en séparation de sources audio [FBD09], en séparation de spectres pour identifier des objets présents dans une image, ou encore dans l'analyse de la composition de l'air [CZPA09, BBL+07].
La factorisation d'une matrice $\mathbf{X}$ de taille $I \times J$ sous la forme d'un produit de deux matrices $\mathbf{W}$ et $\mathbf{H}$ s'écrit de la manière suivante :

$$\mathbf{X} = \mathbf{WH}, \tag{3.1}$$

avec $\mathbf{W}$ une matrice de taille $I \times F$ et $\mathbf{H}$ une matrice de taille $F \times J$. F est un paramètre choisi arbitrairement. Il en résulte que si F est choisi suffisamment petit, tout en permettant de modéliser les données $\mathbf{X}$ sans erreur, $\mathbf{W}$ et $\mathbf{H}$ permettent de compresser l'information contenue dans $\mathbf{X}$.

NMF : estimation avec des règles de mise à jour multiplicatives - Dans [LS00], Lee et Seung ont proposé un algorithme d'estimation de ces deux matrices $\mathbf{W}$ et $\mathbf{H}$, fondé non pas sur des algorithmes d'optimisation de type gradient, mais en utilisant des règles de mise à jour multiplicatives. Cet algorithme est itératif et à chaque étape les valeurs des matrices $\mathbf{W}$ et $\mathbf{H}$ à l'itération précédente sont mises à jour en les multipliant par un facteur calculé dans l'itération en cours pour chaque matrice. D'après les auteurs, les règles de mise à jour additives utilisées dans les algorithmes de type gradient, sont :

- ≻ relativement lentes à converger (des méthodes plus efficaces, comme le gradient conjugué, existent, mais ne sont pas toujours simples à mettre en place),
- ≻ présentent l'inconvénient de nécessiter le choix d'un pas d'adaptation à chaque itération. Le choisir de façon optimale se révèle souvent coûteux en terme de temps de calcul/coût algorithmique. Par ailleurs si ce pas est mal adapté, alors l'algorithme peut ne pas converger.

Lee et Seung ont proposé des algorithmes afin de minimiser l'une des deux fonctions de coût suivantes. Toutes deux reposent sur un critère d'ajustement de modèle consistant à minimiser l'erreur entre l'observation et le modèle. Toutefois, alors que la première est basée sur le carré de la distance Euclidienne :

$$\mathcal{D}_F\left(\mathbf{X}||\mathbf{WH}\right) = \frac{1}{2}\|\mathbf{X} - \mathbf{WH}\|^2 = \frac{1}{2}\sum_{ij}\left(x_{ij} - (\mathbf{WH})_{ij}\right)^2, \tag{3.2}$$

la seconde est fondée sur la divergence de Kullback-Leibler :

$$\mathcal{D}_{KL}\left(\mathbf{X}||\mathbf{WH}\right) = \sum_{ij}\left(x_{ij}\log\frac{x_{ij}}{(\mathbf{WH})_{ij}} - x_{ij} + (\mathbf{WH})_{ij}\right) \tag{3.3}$$

où log désigne le log népérien. Dans le cas de la distance euclidienne, les règles de mise à jour sont les suivantes :

$$h_{fj} \leftarrow h_{fj} \frac{\left(\mathbf{W}^T\mathbf{X}\right)_{fj}}{\left(\mathbf{W}^T\mathbf{W}\mathbf{H}\right)_{fj}} \qquad w_{if} \leftarrow w_{if} \frac{\left(\mathbf{X}\mathbf{H}^T\right)_{if}}{\left(\mathbf{W}\mathbf{H}\mathbf{H}^T\right)_{if}} \tag{3.4}$$

Dans le cas de la divergence de Kullback-Leibler, les règles de mise à jour deviennent :

$$h_{fj} \leftarrow h_{fj} \frac{\sum_i w_{if} x_{ij} / \left(\mathbf{W}\mathbf{H}\right)_{ij}}{\sum_k w_{kf}} \qquad w_{if} \leftarrow w_{if} \frac{\sum_j h_{fj} x_{ij} / \left(\mathbf{W}\mathbf{H}\right)_{ij}}{\sum_m h_{fm}} \tag{3.5}$$

NMF : estimation par ALS et projection - Dans [PCB10], il est proposé d'estimer les matrices recherchées en utilisant l'ALS , et en rendant les quantités positives par le biais d'un opérateur de projection.
En repartant de (3.2), on peut calculer les gradients de $\mathbf{W}$ et $\mathbf{H}$. La différentielle de $\mathcal{D}_F$ donne :

$$\begin{aligned} \mathsf{d}\mathcal{D}_F &= \frac{1}{2}\mathsf{trace}\left\{\mathsf{d}\left(\left(\mathbf{X}-\mathbf{W}\mathbf{H}\right)^T\left(\mathbf{X}-\mathbf{W}\mathbf{H}\right)\right)\right\} \\ &= \mathsf{trace}\left\{\left(\mathbf{X}-\mathbf{W}\mathbf{H}\right)^T \mathsf{d}\left(\mathbf{X}-\mathbf{W}\mathbf{H}\right)\right\} \\ &= \mathsf{trace}\left\{-\left(\mathbf{X}-\mathbf{W}\mathbf{H}\right)^T\left(\left(\mathsf{d}\mathbf{W}\right)\mathbf{H}+\mathbf{W}\mathsf{d}\mathbf{H}\right)\right\} \end{aligned} \tag{3.6}$$

De (3.6), on tire directement le gradient $\nabla_{\mathbf{H}}\mathcal{D}_F$ de $\mathbf{H}$:

$$\nabla_{\mathbf{H}}\mathcal{D}_F = -\left(\mathbf{W}^T\mathbf{X} - \mathbf{W}^T\mathbf{W}\mathbf{H}\right) \tag{3.7}$$

et par symétrie entre $\mathbf{W}$ et $\mathbf{H}$, le gradient $\nabla_{\mathbf{W}}\mathcal{D}_F$ de $\mathbf{W}$:

$$\nabla_{\mathbf{W}}\mathcal{D}_F = -\left(\mathbf{X}\mathbf{H}^T - \mathbf{W}\mathbf{H}\mathbf{H}^T\right) \tag{3.8}$$

En égalant (3.7) et (3.8) à zéro, on obtient les règles de mise à jour via l'ALS de $\mathbf{W}$ et $\mathbf{H}$:

$$\mathbf{H} = \left(\mathbf{W}^T\mathbf{W}\right)^{-1}\left(\mathbf{W}^T\mathbf{X}\right) \tag{3.9}$$

$$\mathbf{W} = \left(\mathbf{X}\mathbf{H}^T\right)\left(\mathbf{H}\mathbf{H}^T\right)^{-1} \tag{3.10}$$

Suite à cela, on applique un opérateur de projection $[\cdot]_+$ dans le but d'assurer la positivité des matrices estimées.

$$\widehat{\mathbf{W}} \leftarrow \left[\widehat{\mathbf{W}}\right]_+, \quad \widehat{\mathbf{H}} \leftarrow \left[\widehat{\mathbf{H}}\right]_+. \tag{3.11}$$

$[\mathbf{M} = (m_{ij})]_+$ retourne une matrice de la même taille que $\mathbf{M}$, et dont la (i,j)-ème valeur vaut $\mathsf{max}\{\epsilon, m_{ij}\}$, où ϵ est une constante de faible valeur (typiquement 10^{-16}) et $\mathsf{max}\{\cdot\}$ est l'opérateur qui retourne la plus grande des valeurs passées en paramètre.

NMF : estimation basée sur des produits de Hadamard - En utilisant la fonction de coût donnée par l'Eq. (3.2) mais en modélisant les quantités recherchées comme des carrés au moyen d'un produit de Hadamard noté ici $\boxdot$, la fonction de coût précédente s'écrit cette fois :

$$\mathcal{C}_F\left(\mathbf{W},\mathbf{H}\right)=\frac{1}{2}\|\mathbf{X}-\left(\mathbf{W}\boxdot\mathbf{W}\right)\left(\mathbf{H}\boxdot\mathbf{H}\right)\|^2, \tag{3.12}$$

En posant $\boldsymbol{\eta}=\mathbf{X}-\left(\mathbf{W}\boxdot\mathbf{W}\right)\left(\mathbf{H}\boxdot\mathbf{H}\right)$, le calcul de la différentiel de la fonction $\mathcal{C}_F$ conduit à :

$$\begin{aligned}\mathsf{d}\mathcal{C}_F(\mathbf{W},\mathbf{H}) &= \langle 2\left(\mathbf{W}\boxdot\mathbf{W}\right)^T\left(-\boldsymbol{\eta}\right)\boxdot\mathbf{H},\mathsf{d}\mathbf{H}\rangle \\ &\quad+\langle 2\left(-\boldsymbol{\eta}(\mathbf{H}\boxdot\mathbf{H})^T\boxdot\mathbf{W}\right),\mathsf{d}\mathbf{W}\rangle,\end{aligned} \tag{3.13}$$

On peut ensuite en déduire les gradients matriciels :

$$\begin{aligned}\nabla_{\mathbf{W}}\mathcal{C}_F(\mathbf{W},\mathbf{H}) &= \frac{\partial\mathcal{C}_F(\mathbf{W},\mathbf{H})}{\partial\mathbf{W}} \\ &= 2\mathbf{W}\boxdot\left(\left(-\boldsymbol{\eta}\right)\left[\left(\mathbf{H}\boxdot\mathbf{H}\right)\right]^T\right) \\ &= -2\mathbf{W}\boxdot\left(\mathbf{X}-\left(\mathbf{W}\boxdot\mathbf{W}\right)\left(\mathbf{H}\boxdot\mathbf{H}\right)\right)\left[\left(\mathbf{H}\boxdot\mathbf{H}\right)^T\right],\end{aligned} \tag{3.14}$$

$$\begin{aligned}\nabla_{\mathbf{H}}\mathcal{C}_F(\mathbf{W},\mathbf{H}) &= \frac{\partial\mathcal{C}_F(\mathbf{W},\mathbf{H})}{\partial\mathbf{H}} \\ &= 2\mathbf{H}\boxdot\left(\mathbf{W}\boxdot\mathbf{W}\right)^T\left(-\boldsymbol{\eta}\right) \\ &= -2\mathbf{H}\boxdot\left(\mathbf{W}\boxdot\mathbf{W}\right)^T\left(\mathbf{X}-\left(\mathbf{W}\boxdot\mathbf{W}\right)\left(\mathbf{H}\boxdot\mathbf{H}\right)\right),\end{aligned} \tag{3.15}$$

C'est de cette approche que nous nous inspirerons pour dériver une décomposition CP non négative. Cependant, le calcul des matrices de gradients dans le cas des NMF est plus simple et ne fait pas par exemple appel à la propriété donnée en (2.5). D'autres méthodes permettant de résoudre le problème posé par les NMF sont détaillées dans [CZPA09].

3.2.2 Approches pour les tenseurs

Commençons par un rapide état de l'art des méthodes qui ont été suggérées dans le cas des tenseurs d'ordre trois. On peut en trouver par exemple dans [PTC11, CZPA09], mais nous en détaillerons deux. La première proposée dans [CZPA09], consiste à commencer par pénaliser la fonction de coût puis à utiliser un algorithme d'optimisation itératif de type ALS , auquel après chaque itération, un opérateur de projection destiné à assurer la positivité des quantités trouvées est appliqué. Le second, baptisé algorithme NNLS (pour NonNegative Least Squares), a été quant à lui suggéré dans [BD97].

Méthode combinant pénalisation et projection (NTF-ALS) [CZPA09] - Cette méthode consiste à utiliser la même fonction de coût que celle fournie dans l'équation (2.21), mais y ajoutant des termes de pénalité qui ont pour objectif de renforcer certaines propriétés de la solution recherchée, comme par exemple l'aspect parcimonieux des matrices de facteurs, ou

encore l'aspect "continu" ou "lissé" de leurs valeurs. Puis dans un second temps, un opérateur de projection, noté $[\cdot]_+$ dont le but est d'imposer la non négativité des entrées est appliqué, cette propriété n'étant bien évidemment pas assurée par l'ajout des termes de pénalité. Enfin, en ce qui concerne l'algorithme d'optimisation, les auteurs suggèrent d'utiliser l'algorithme ALS ou HALS (présenté ci-après). Le principe de l'algorithme ALS est d'optimiser la fonction de coût alternativement par rapport à l'une des trois matrices de facteurs, les deux autres étant maintenues fixes [BA, Bro97, NL08, RCH08] . Cet algorithme sera appelé NTF-ALS dans la suite.
En conséquence, il est suggéré d'opter plutôt pour une fonction de coût de la forme suivante :

$$\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}), +\alpha_A\|\mathbf{A}\|_F^2 + \alpha_B\|\mathbf{B}\|_F^2 + \alpha_C\|\mathbf{C}\|_F^2 \tag{3.16a}$$

$$\mathcal{G}_1(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + \alpha_A\|\mathbf{A}\|_1 + \alpha_B\|\mathbf{B}\|_1 + \alpha_C\|\mathbf{C}\|_1, \tag{3.16b}$$

où α_A, α_B et α_C sont des scalaires positifs ou nuls appelés paramètres de régularisation. Dans (3.16a), la norme L_2 est destinée à assurer le lissage de la solution obtenue, tandis que la norme L_1 (ici ($\|\mathbf{A}\|_1 = \sum_{i,j}|a_{ij}|$) utilisée dans (3.16b) permet de contraindre la solution à être plus parcimonieuse.
On peut calculer les nouvelles composantes des gradients des matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ prenant en compte ces termes de pénalité. Pour la pénalité à base de norme L_2, on obtient ainsi :

$$\nabla_{\mathbf{A}}\mathcal{G}(\cdot) = \nabla_{\mathbf{A}}\mathcal{F}(\cdot) + 2\alpha_A\mathbf{A} \tag{3.17}$$

$$\nabla_{\mathbf{B}}\mathcal{G}(\cdot) = \nabla_{\mathbf{B}}\mathcal{F}(\cdot) + 2\alpha_B\mathbf{B} \tag{3.18}$$

$$\nabla_{\mathbf{C}}\mathcal{G}(\cdot) = \nabla_{\mathbf{C}}\mathcal{F}(\cdot) + 2\alpha_C\mathbf{C} \tag{3.19}$$

Les gradients matriciels liés à une pénalisation de type norme L_1 sont quant à eux définis de la manière suivante :

$$\nabla_{\mathbf{A}}\mathcal{G}_1(\cdot) = \nabla_{\mathbf{A}}\mathcal{F}(\cdot) + \alpha_A\mathbf{1}_{I,F}, \tag{3.20}$$

$$\nabla_{\mathbf{B}}\mathcal{G}_1(\cdot) = \nabla_{\mathbf{B}}\mathcal{F}(\cdot) + \alpha_B\mathbf{1}_{J,F}, \tag{3.21}$$

$$\nabla_{\mathbf{C}}\mathcal{G}_1(\cdot) = \nabla_{\mathbf{C}}\mathcal{F}(\cdot) + \alpha_C\mathbf{1}_{K,F} \tag{3.22}$$

$\mathbf{1}_{K,F}$ représente la matrice de taille $K \times F$ ne contenant que des 1.
En rendant les composantes du gradient égales à zéro, il est possible d'en déduire les estimées des trois matrices recherchées $\widehat{\mathbf{A}}$, $\widehat{\mathbf{B}}$ et $\widehat{\mathbf{C}}$ dans les deux cas précédents. D'abord pour la pénalisation de type norme L_2 :

$$\widehat{\mathbf{A}} = \mathbf{T}_{(1)}^{I,KJ}(\mathbf{C}\odot\mathbf{B})\mathbf{\Lambda}\left[\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{B})^T)(\mathbf{C}\odot\mathbf{B})\mathbf{\Lambda} + 2\alpha_A\mathbf{I}_F\right]^\dagger, \tag{3.23}$$

$$\widehat{\mathbf{B}} = \mathbf{T}_{(2)}^{J,KI}(\mathbf{C}\odot\mathbf{A})\mathbf{\Lambda}\left[\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{A})^T)(\mathbf{C}\odot\mathbf{A})\mathbf{\Lambda} + 2\alpha_B\mathbf{I}_F\right]^\dagger, \tag{3.24}$$

$$\widehat{\mathbf{C}} = \mathbf{T}_{(3)}^{K,JI}(\mathbf{B}\odot\mathbf{A})\mathbf{\Lambda}\left[\mathbf{\Lambda}(\mathbf{B}\odot\mathbf{A})^T)(\mathbf{B}\odot\mathbf{A})\mathbf{\Lambda} + 2\alpha_C\mathbf{I}_F\right]^\dagger. \tag{3.25}$$

Puis pour la pénalisation de type norme L_1, on obtient alors :

$$\widehat{\mathbf{A}} = \left[\mathbf{T}_{(1)}^{I,KJ}(\mathbf{C}\odot\mathbf{B})\mathbf{\Lambda} - \alpha_A\mathbf{1}_{I,F}\right]\left[\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{B})^T)(\mathbf{C}\odot\mathbf{B})\mathbf{\Lambda}\right]^\dagger, \tag{3.26}$$

$$\widehat{\mathbf{B}} = \left[\mathbf{T}_{(2)}^{J,KI}(\mathbf{C}\odot\mathbf{A})\mathbf{\Lambda} - \alpha_B\mathbf{1}_{J,F}\right]\left[\mathbf{\Lambda}(\mathbf{C}\odot\mathbf{A})^T)(\mathbf{C}\odot\mathbf{A})\mathbf{\Lambda}\right]^\dagger, \tag{3.27}$$

$$\widehat{\mathbf{C}} = \left[\mathbf{T}_{(3)}^{K,JI}(\mathbf{B}\odot\mathbf{A})\mathbf{\Lambda} - \alpha_C\mathbf{1}_{K,F}\right]\left[\mathbf{\Lambda}(\mathbf{B}\odot\mathbf{A})^T)(\mathbf{B}\odot\mathbf{A})\mathbf{\Lambda}\right]^\dagger. \tag{3.28}$$

où $[\cdot]_+$ est un opérateur de projection assurant la positivité des entrées, cette propriété n'étant évidemment pas garantie par l'ajout des termes de pénalité.

$$\widehat{\mathbf{A}} \leftarrow \left[\widehat{\mathbf{A}}\right]_+, \quad \widehat{\mathbf{B}} \leftarrow \left[\widehat{\mathbf{B}}\right]_+, \quad \widehat{\mathbf{C}} \leftarrow \left[\widehat{\mathbf{C}}\right]_+. \tag{3.29}$$

Algorithme HALS (NTF-HALS) - Le HALS (ALS hiérarchique), appelé NTF-HALS dans la suite, minimise un ensemble de fonctions de coût possédant le même minimum global, pour lesquelles on essaie d'approximer un tenseur de rang 1. On estime alors un facteur (colonne) de chaque matrice de facteurs de manière successive. Pour i variant de 1 à F (nombre de colonnes des matrices de facteurs), ces fonctions de coût s'écrivent :

$$\mathcal{G}^{(i)}_{\mathsf{HALS}}\left(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i\right) = \|\mathbf{T}^{(i)} - \mathbf{a}_i \circledast \mathbf{b}_i \circledast \mathbf{c}_i\|_F \tag{3.30}$$

où $\mathbf{T}^{(i)}$ est le tenseur suivant :

$$\mathbf{T}^{(i)} = \mathbf{T} - \sum_{j \neq i} \mathbf{a}_j \circledast \mathbf{b}_j \circledast \mathbf{c}_j \tag{3.31}$$

$$= \mathbf{T} - \sum_{j=1}^{F} \mathbf{a}_j \circledast \mathbf{b}_j \circledast \mathbf{c}_j + \mathbf{a}_i \circledast \mathbf{b}_i \circledast \mathbf{c}_i \tag{3.32}$$

avec $\mathbf{T}$ qui est le tenseur de données. On pose $\mathbf{E} = \mathbf{T} - \sum_{j=1}^{F} \mathbf{a}_j \circledast \mathbf{b}_j \circledast \mathbf{c}_j$ pour la suite.
Le calcul des gradients de (3.30) donne :

$$\nabla_{\mathbf{a}_i} \mathcal{G}^{(i)}_{\mathsf{HALS}} = -\mathbf{T}^{(i)}\left(\mathbf{c}_i \odot \mathbf{b}_i\right) + \mathbf{a}_i \gamma_{\mathbf{a}_i} \tag{3.33}$$

$$\nabla_{\mathbf{b}_i} \mathcal{G}^{(i)}_{\mathsf{HALS}} = -\mathbf{T}^{(i)}\left(\mathbf{c}_i \odot \mathbf{a}_i\right) + \mathbf{b}_i \gamma_{\mathbf{b}_i} \tag{3.34}$$

$$\nabla_{\mathbf{c}_i} \mathcal{G}^{(i)}_{\mathsf{HALS}} = -\mathbf{T}^{(i)}\left(\mathbf{b}_i \odot \mathbf{a}_i\right) + \mathbf{c}_i \gamma_{\mathbf{c}_i}. \tag{3.35}$$

où $\gamma_{\mathbf{a}_i}$, $\gamma_{\mathbf{b}_i}$, $\gamma_{\mathbf{c}_i}$ sont des coefficients d'échelle dont la valeur est donnée dans [CZPA09]. Ceux-ci seront omis par la suite étant donné qu'on procèdera à la place à une normalisation des facteurs estimés.
(3.35) permet d'en déduire les règles de mise à jour des facteurs :

$$\widehat{\mathbf{a}}_i = \left[\mathbf{T}^{(i)}\left(\mathbf{c}_i \odot \mathbf{b}_i\right)\right]_+ \tag{3.36}$$

$$\widehat{\mathbf{b}}_i = \left[\mathbf{T}^{(i)}\left(\mathbf{c}_i \odot \mathbf{a}_i\right)\right]_+ \tag{3.37}$$

$$\widehat{\mathbf{c}}_i = \left[\mathbf{T}^{(i)}\left(\mathbf{b}_i \odot \mathbf{a}_i\right)\right]_+ \tag{3.38}$$

On peut donc synthétiser le HALS de la manière suivante :

Déroulement du HALS

1. Initialiser les matrices de facteurs.
2. Calculer $\mathbf{E}$.
3. Tant que le critère d'arrêt n'est pas atteint, pour $i = 1$ à F :
   (a) $\mathbf{T}^{(i)} = \mathbf{E} + \widehat{\mathbf{a}}_i \circledast \widehat{\mathbf{b}}_i \circledast \widehat{\mathbf{c}}_i$
   (b) Calculer les estimées des facteurs :
   $\widehat{\mathbf{a}}_i = \left[\mathbf{T}^{(i)} \left(\mathbf{c}_i \odot \mathbf{b}_i\right)\right]_+$
   $\widehat{\mathbf{b}}_i = \left[\mathbf{T}^{(i)} \left(\mathbf{c}_i \odot \mathbf{a}_i\right)\right]_+$
   $\widehat{\mathbf{c}}_i = \left[\mathbf{T}^{(i)} \left(\mathbf{b}_i \odot \mathbf{a}_i\right)\right]_+$
   (c) Etape de normalisation des facteurs :
   $\widehat{\mathbf{a}}_i = \frac{\widehat{\mathbf{a}}_i}{\|\widehat{\mathbf{a}}_i\|_2}$ et $\widehat{\mathbf{b}}_i = \frac{\widehat{\mathbf{b}}_i}{\|\widehat{\mathbf{b}}_i\|_2}$
   (d) $\mathbf{E} = \mathbf{T}^{(i)} - \widehat{\mathbf{a}}_i \circledast \widehat{\mathbf{b}}_i \circledast \widehat{\mathbf{c}}_i$

A noter qu'une version rapide du HALS a été développée et est présentée dans [PC08, CZPA09].

Algorithme NNLS - A l'origine, le NNLS (NonNegative Least Squares) est un algorithme itératif permettant de résoudre et minimiser le problème suivant [BD97, LH87] :

$$\|\mathbf{x} - \mathbf{Z}\mathbf{d}\|^2 \tag{3.39}$$

où $\mathbf{x} \in \mathbb{R}^{M\times 1}$, $\mathbf{Z} \in \mathbb{R}^{M\times F}$ et $\mathbf{d} \in \mathbb{R}^{+F\times 1}$ est le vecteur à déterminer, dont les F éléments sont positifs ou nuls.
L'algorithme NNLS est basé sur la constitution de deux ensembles complémentaires, l'un dit passif, P, qui contient les indices des F variables libres, et un ensemble actif contenant les indices des coefficients du vecteur de résultat mis actuellement à zéro. Quand tous les éléments de l'ensemble actif sont connus, ce vecteur de résultat $\mathbf{d}$ est alors simplement déterminé en utilisant l'estimateur des moindres carrés sans contraintes, pour les valeurs des indices de l'ensemble passif.
A chaque étape de l'algorithme, on calcule un vecteur $\mathbf{w} = \mathbf{Z}^T(\mathbf{x} - \mathbf{Z}\mathbf{d})$, qui est égal à la moitié de l'opposée de la dérivée de (3.39). Si l'ensemble actif n'est pas vide, on détermine le $w_f > 0$ de plus grande valeur, et on transfère l'indice de sa position dans l'ensemble passif. Similairement, on retire cet indice de l'ensemble actif.
On peut alors calculer le vecteur de regression $\mathbf{s} = \left(\left(\mathbf{Z}^P\right)^T \mathbf{Z}^P\right)^{-1} \left(\mathbf{Z}^P\right)^T \mathbf{x}$, qui est l'estimateur des moindres carrés, avec $\mathbf{Z}^P$ représentant une matrice ne contenant que les élements de l'ensemble passif. On affecte ensuite la valeur de $\mathbf{s}$ à $\mathbf{d}$, on recalcule $\mathbf{w}$, et on s'arrête si l'ensemble actif est vide ou si aucun w_f n'est positif.
Dans [BD97, Paa97, TB04, TB06], il est suggéré une amélioration de cette méthode afin de pouvoir l'appliquer au cas des décompositions trilinéaires. La deuxième contribution est une amélioration algorithmique permettant de diminuer les temps de calcul. L'objectif est alors de résoudre le système vectorisé suivant :

$$\mathsf{vec}\{\mathbf{T}_{(1)}^{I,KJ} - \mathbf{A}\mathbf{\Lambda}(\mathbf{C} \odot \mathbf{B})^T\} = \mathbf{0}_{IJK,1} \tag{3.40}$$

où l'opérateur $\mathsf{vec}\{\cdot\}$ appliqué à une matrice permet d'agréger ses colonnes en un vecteur colonne et $\mathbf{0}_{IJK,1}$ est un vecteur de taille $IJK \times 1$ qui ne contient que des éléments nuls.
On estime donc chaque matrice de facteurs ligne par ligne (soit un vecteur $\in \mathbb{R}_+^{1\times F}$).
Ecrit sous forme matricielle et pour établir une analogie avec le NNLS classique, on aurait :

$$\mathbf{T}_{(1)}^{I,KJ} = \mathbf{Z}\mathbf{A}^T, \tag{3.41}$$

où $\mathbf{Z} = \mathbf{B} \odot \mathbf{C}$ dans le cas du premier dépliement du tenseur $\mathbf{T} \in I \times J \times K$. On trouve les autres matrices $\mathbf{Z}$ correspondant aux deux autres dépliements par permutations.
Sachant qu'on peut calculer très efficacement les produits suivants :

$$\mathbf{Z}^T\mathbf{Z} = \left(\mathbf{C}^T\mathbf{C}\right) \boxdot \left(\mathbf{B}^T\mathbf{B}\right) \tag{3.42}$$

$$\left(\mathbf{T}_{(1)}^{I,KJ}\right)^T \mathbf{Z} = \mathbf{T}_1\mathbf{B}\mathbf{D}_1 + \mathbf{T}_1\mathbf{B}\mathbf{D}_1 + ... + \mathbf{T}_K\mathbf{B}\mathbf{D}_K, \tag{3.43}$$

où $\mathbf{D}_k$ est la matrice diagonale contenant les éléments de la k ième ligne de $\mathbf{C}$.
On peut alors se contenter de calculer $\mathbf{Z}^T\mathbf{Z}$ et $\left(\mathbf{T}_{(1)}^{I,KJ}\right)^T \mathbf{Z}$ sans passer par le produit de Khatri-Rhao permettant d'estimer $\mathbf{Z}$. Dans le cas où le nombre de lignes des matrices de facteurs est supérieur à leur nombre de colonnes (cas le plus fréquent), on a un gain de temps en termes de complexité algorithmique.
Il est alors nécessaire de procéder à quelques modifications sur l'algorithme NNLS pour ne considérer que $\mathbf{Z}^T\mathbf{Z}$ et $\left(\mathbf{T}_{(1)}^{I,KJ}\right)^T \mathbf{Z}$. Considérant que le vecteur $\mathbf{x}$ donné en (3.39) est remplacé ici par $\left(\mathbf{T}_{(1)}^{I,KJ}\right)_i$ où i est la i ième ligne de $\mathbf{T}$, l'estimation du vecteur $\mathbf{w}$ devient :

$$\mathbf{w} = \mathbf{Z}^T\mathbf{x} - \left(\mathbf{Z}^T\mathbf{Z}\right)\mathbf{d}. \tag{3.44}$$

L'estimation du vecteur de régression $\mathbf{s}$ devient :

$$\mathbf{s} = \left(\left(\mathbf{Z}^T\mathbf{Z}\right)^P\right)^{-1} \left(\mathbf{Z}^T\mathbf{x}\right)^P \tag{3.45}$$

$\left(\mathbf{Z}^T\mathbf{Z}\right)^P$ *et* $\left(\mathbf{Z}^T\mathbf{x}\right)^P$ ne contiennent alors que les éléments de l'ensemble P.
Un rapide état de l'art des méthodes existantes les plus utilisées ayant été effectué, nous allons maintenant présenter dans la partie suivante les nouveaux algorithmes que nous avons développés pour la décomposition CP de tenseurs d'ordre trois sous contrainte de non négativité. Ces approches sont bien évidemment toutes généralisables à des tenseurs d'ordre plus élevé.

3.3 Nouvelles approches pour la décomposition CP d'ordre 3 non négative

3.3.1 Approche par paramétrisation par produits de Hadamard

La première idée que nous ayons eu est comme dans le cas des NMF d'essayer de prendre en compte l'aspect non négatif sans faire appel à des pénalisations qui, en plus de complexifier la fonction de coût, nécessitent souvent d'ajuster un ou plusieurs paramètres scalaires. La question de l'ajout d'une pénalisation de ce type sera traitée dans la section 3.3.2.
Un moyen simple de définir un élément positif consiste à l'écrire comme le carré d'un autre. Ainsi, pour considérer qu'une matrice ne contient que des valeurs positives, on peut simplement poser que $a'_{ij} = a^2_{ij}$. En utilisant le produit de Hadamard, cela implique que $\mathbf{A}' = \mathbf{A} \boxdot \mathbf{A}$, pour n'importe quelle matrice $\mathbf{A}$. On peut alors réécrire la fonction de coût correspondant à ce changement de variable :

$$\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \mathcal{F}(\mathbf{A}\boxdot\mathbf{A},\mathbf{B}\boxdot\mathbf{B},\mathbf{C}\boxdot\mathbf{C})$$
$$\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \|\mathbf{T}^{I,KJ}_{(1)} - (\mathbf{A}\boxdot\mathbf{A})\mathbf{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]^T\|^2_F \tag{3.46}$$
$$\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \|\mathbf{T}^{J,KI}_{(2)} - (\mathbf{B}\boxdot\mathbf{B})\mathbf{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T\|^2_F \tag{3.47}$$
$$\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \|\mathbf{T}^{K,JI}_{(3)} - (\mathbf{C}\boxdot\mathbf{C})\mathbf{\Lambda}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T\|^2_F, \tag{3.48}$$

Pour simplifier les expressions, posons :

$$\boldsymbol{\delta}_{(1)} = \mathbf{T}^{I,KJ}_{(1)} - (\mathbf{A}\boxdot\mathbf{A})\mathbf{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]^T \tag{3.49}$$
$$\boldsymbol{\delta}_{(2)} = \mathbf{T}^{J,KI}_{(2)} - (\mathbf{B}\boxdot\mathbf{B})\mathbf{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T \tag{3.50}$$
$$\boldsymbol{\delta}_{(3)} = \mathbf{T}^{K,JI}_{(3)} - (\mathbf{C}\boxdot\mathbf{C})\mathbf{\Lambda}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T \tag{3.51}$$

En calculant la différentielle $\mathsf{d}\mathcal{H}$ de $\mathcal{H}$, on est alors à même de calculer les composantes des gradients, à savoir la matrice $\nabla_{\mathbf{A}}\mathcal{H}$ de taille $I \times F$, la matrice $\nabla_{\mathbf{B}}\mathcal{H}$ de taille $J \times F$, et la matrice $\nabla_{\mathbf{C}}\mathcal{H}$ de taille $K \times F$. Il serait également possible de poursuivre le calcul analytique jusqu'à la matrice Hessienne (il faudrait alors calculer les dérivées secondes), de taille $(I+J+K)F \times (I+J+K)F$ dans le cas où l'on voudrait pouvoir ensuite utiliser des algorithmes d'optimisation de type Gauss-Newton.
Le produit scalaire de Frobenius [AMDDM60] est défini comme :

$$\langle \mathbf{A},\mathbf{B}\rangle = \mathsf{trace}\{\mathbf{A}^T\mathbf{B}\}. \tag{3.52}$$

Nous avons aussi :

$$\langle \mathbf{A},\mathbf{A}\rangle = \|\mathbf{A}\|^2_F = \mathsf{trace}\{\mathbf{A}^T\mathbf{A}\}. \tag{3.53}$$

On peut alors réécrire la fonction de coût $\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})$ de façon simplifiée :

$$\begin{aligned}\langle \boldsymbol{\delta}_{(1)},\boldsymbol{\delta}_{(1)}\rangle &= \mathsf{trace}\left\{\boldsymbol{\delta}^T_{(1)}\boldsymbol{\delta}_{(1)}\right\}\\ &= \mathsf{trace}\left\{\left(\mathbf{T}^{I,KJ}_{(1)} - (\mathbf{A}\boxdot\mathbf{A})\boldsymbol{\delta}_{(1)}[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})]^T\right)^T\cdot\right.\\ &\qquad\left.\left(\mathbf{T}^{I,KJ}_{(1)} - (\mathbf{A}\boxdot\mathbf{A})\boldsymbol{\delta}_{(1)}[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})]^T\right)\right\}.\end{aligned}$$

Le calcul détaillé de $\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})$ est réalisé dans l'annexe 6.4 où nous avons monté qu'il valait finalement :

$$\begin{aligned}\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \langle 4\left[\mathbf{A}\boxdot\left((-\boldsymbol{\delta}_{(1)})\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]\boldsymbol{\Lambda}\right)\right],\mathsf{d}\mathbf{A}\rangle \\ &+ \langle 4\left[\mathbf{B}\boxdot\left((-\boldsymbol{\delta}_{(2)})\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]\boldsymbol{\Lambda}\right)\right],\mathsf{d}\mathbf{B}\rangle \\ &+ \langle 4\left[\mathbf{C}\boxdot\left((-\boldsymbol{\delta}_{(3)})\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]\boldsymbol{\Lambda}\right)\right],\mathsf{d}\mathbf{C}\rangle \qquad (3.54)\end{aligned}$$

En utilisant la différentielle donnée en (3.54), les matrices de gradients peuvent alors être déterminées simplement :

$$\begin{aligned}\nabla_{\mathbf{A}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}} \\ &= 4\mathbf{A}\boxdot\left((-\boldsymbol{\delta}_{(1)})\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]\boldsymbol{\Lambda}\right) \qquad (3.55)\\ &= 4\mathbf{A}\boxdot\left((\mathbf{T}_{(1)}^{I,KJ}-(\mathbf{A}\boxdot\mathbf{A})\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]^{T})\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]\boldsymbol{\Lambda}\right), \\ \nabla_{\mathbf{B}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}} \\ &= 4\mathbf{B}\boxdot\left((-\boldsymbol{\delta}_{(2)})[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})]\boldsymbol{\Lambda}\right) \qquad (3.56)\\ &= 4\mathbf{B}\boxdot\left((\mathbf{T}_{(2)}^{J,KI}-(\mathbf{B}\boxdot\mathbf{B})\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^{T})[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})]\boldsymbol{\Lambda}\right), \\ \nabla_{\mathbf{C}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}} \\ &= 4\mathbf{C}\boxdot\left((-\boldsymbol{\delta}_{(3)})[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})]\boldsymbol{\Lambda}\right) \qquad (3.57)\\ &= 4\mathbf{C}\boxdot\left((\mathbf{T}_{(3)}^{K,JI}-(\mathbf{C}\boxdot\mathbf{C})\boldsymbol{\Lambda}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^{T})[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})]\boldsymbol{\Lambda}\right).\end{aligned}$$

On peut maintenant construire les matrices de taille $(I+J+K)\times F$ suivantes :
$\mathbf{G}^{(k)}$ et $\mathbf{X}^{(k)}$:

$$\mathbf{G}^{(k)}=\begin{pmatrix}\nabla_{\mathbf{A}}\mathcal{H}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\\ \nabla_{\mathbf{B}}\mathcal{H}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\\ \nabla_{\mathbf{C}}\mathcal{H}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\end{pmatrix},\quad \mathbf{X}^{(k)}=\begin{pmatrix}\mathbf{A}^{(k)}\\ \mathbf{B}^{(k)}\\ \mathbf{C}^{(k)}\end{pmatrix} \qquad (3.58)$$

ou encore les vecteurs $(I+J+K)F\times 1$ suivants :

$$\mathbf{g}^{(k)}=\begin{pmatrix}\mathsf{vec}\{\nabla_{\mathbf{A}}\mathcal{H}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\}\\ \mathsf{vec}\{\nabla_{\mathbf{B}}\mathcal{H}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\}\\ \mathsf{vec}\{\nabla_{\mathbf{C}}\mathcal{H}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\}\end{pmatrix},\quad \mathbf{x}^{(k)}=\begin{pmatrix}\mathsf{vec}\{\mathbf{A}^{(k)}\}\\ \mathsf{vec}\{\mathbf{B}^{(k)}\}\\ \mathsf{vec}\{\mathbf{C}^{(k)}\}\end{pmatrix} \qquad (3.59)$$

Une fois ces termes calculés, il est possible d'en dériver les schémas d'algorithmes d'optimisation "classiques" de type descente. C'est ce que nous ferons au paragraphe 3.4.

3.3.2 Approche par ajout d'un terme de pénalisation exponentielle

Nous venons de présenter une manière intuitive de considérer la non négativité des matrices de facteurs. Mais il aurait également été possible d'opter pour une autre approche consistant

à faire en sorte que la fonction de coût prenne des valeurs très grandes dès lors que les matrices de facteurs $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$ contiennent des éléments négatifs, tandis que sa valeur restera inchangée et sera petite si les matrices de facteurs $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$ ne contiennent que des éléments non négatifs. Pour cela, on peut ajouter des termes de pénalisation, et modifier la fonction de coût proposée en (2.21) par une fonction de coût de la forme suivante :

$$\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \alpha f(\mathbf{A}) + \beta f(\mathbf{B}) + \gamma f(\mathbf{C}), \tag{3.60}$$

avec α, β, γ des scalaires positifs qui jouent le rôle de poids sur les matrices de facteurs. $f(\cdot)$ est choisie de telle sorte que

(i) $f(\cdot)$ est continue.
(ii) $f(\mathbf{A}) \geq 0$ pour tout $\mathbf{A} \in \mathbb{R}^{I\times F}$.
(iii) $f(\mathbf{A}) = 0$ si et seulement si $\mathbf{A} \in \mathbb{R}^{+I\times F}$.

Des travaux ont déjà eu lieu sur les *fonctions barrière* [CMI12, NW00] avec des applications dans les problèmes inverses liés à la restauration d'images notamment. Nous proposons ici d'ajouter des termes de pénalisation en $e^{-\alpha\mathbf{X}}$, faciles à dériver et qui remplissent les conditions recherchées. Cela nous amène donc à utiliser la fonction de coût suivante :

$$\begin{aligned}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) &+ \alpha\|e^{-\gamma_A \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 \\ &+ \beta\|e^{-\gamma_B \mathsf{diag}\{\mathsf{vec}\{\mathbf{B}\}\}}\|_F^2 + \gamma\|e^{-\gamma_C \mathsf{diag}\{\mathsf{vec}\{\mathbf{C}\}\}}\|_F^2,\end{aligned} \tag{3.61}$$

Où $\gamma_A, \gamma_B, \gamma_C$ sont des constantes positives qu'il nous faudra ajuster, ce qui complique bien évidemment le problème. L'opérateur $\mathsf{diag}\{\cdot\}$ retourne une matrice carrée diagonale qui contient dans sa diagonale les éléments du vecteur donné en argument.
A nouveau le calcul de la différentielle $\mathsf{d}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C})$ doit être réalisé afin d'en déduire les gradients matriciels. Le calcul détaillé a été déporté à l'annexe 6.4.2 où nous avons montré que la différentielle des termes de pénalisation valait :

$$\mathsf{d}\|e^{-\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 = 4\gamma^2\left[\mathbf{A} - \frac{\mathbf{1}_{I,F}}{2\gamma}\right]\mathsf{d}\mathbf{A}. \tag{3.62}$$

Ce qui conduit donc aux gradients matriciels suivants :

$$\nabla_{\mathbf{A}}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{A}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + 4\alpha\gamma_A^2\left[\mathbf{A} - \frac{\mathbf{1}_{I,F}}{2\gamma_A}\right], \tag{3.63}$$

$$\nabla_{\mathbf{B}}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{B}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + 4\beta\gamma_B^2\left[\mathbf{B} - \frac{\mathbf{1}_{J,F}}{2\gamma_B}\right], \tag{3.64}$$

$$\nabla_{\mathbf{C}}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{C}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + 4\gamma\gamma_C^2\left[\mathbf{C} - \frac{\mathbf{1}_{K,F}}{2\gamma_C}\right] \tag{3.65}$$

(où $\mathbf{1}_{I,F}$ est une matrice de taille $I \times F$ remplie uniquement de 1). Comme précédemment, il est ensuite possible de construire les matrices $\mathbf{G}^{(k)}$ et $\mathbf{X}^{(k)}$ ou encore les vecteurs $\mathbf{g}^{(k)}$ et $\mathbf{g}^{(k)}$ qui ont été définis par les équations (3.58) et (3.59).
Une fois les gradients matriciels calculés, il devient possible de minimiser les différentes fonctions de coût qui viennent d'être introduites. Pour ce faire, nous allons donc appliquer l'un des différents algorithmes d'optimisation dont nous allons maintenant rappeler le principe dans le but de parvenir à estimer les trois matrices de facteurs $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$.

3.4 Algorithmes d'optimisation de type descente

Pour estimer les matrices de facteurs $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, les fonctions de coûts introduites au paragraphe précédent (par exemple en (3.46 pour une paramétrisation au moyen de produits de Hadamard) doivent être minimisées. Dans cette optique, nous suggérons d'optimiser les fonctions de coût considérées de façon simultanée vis à vis des trois matrices de facteurs et non pas alternée comme cela est fait dans l'ALS - cette approche étant réputée sous optimale -. Nous allons donc maintenant rappeler le principe de différents algorithmes d'optimisation parmi lesquels la méthode du gradient conjugué préconditionné [She94, Pol97].
Dans l'approche classique du gradient, la variable $\mathbf{X}$ donnée en (3.58) est mise à jour à chaque itération suivant la règle d'adaptation suivante :

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \mu^{(k)}\mathbf{G}^{(k)}, \tag{3.66}$$

où $\mathbf{G}$ est la matrice de gradient donnée en (3.58), en utilisant les valeurs analytiques calculées en (3.55) - (3.57). μ est le pas d'adaptation scalaire, qui peut être soit fixe tout au long de l'algorithme, soit calculé, manière exacte ou approchée, à chaque itération. Ce point sera étudié en détail au niveau de la section 3.6. On peut également utiliser la règle suivante écrite sur des vecteurs et non plus des matrices :

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mu^{(k)}\mathbf{g}^{(k)}, \tag{3.67}$$

où $\mathbf{x}$ et $\mathbf{g}$ sont donnés en (3.59). Notons que dans le cas où la contrainte de non négativité n'est plus requise, les gradients contenus dans $\mathbf{G}$ et $\mathbf{g}$ sont simplement remplacés par leur équivalents sans contrainte dont nous avions rappelé la valeur au niveau des équations (2.22), (2.23) et (2.24).

3.4.1 Gradient conjugué préconditionné

Dans le gradient conjugué préconditionné, la règle d'adapation est modifiée ainsi :

$$\left\{ \begin{array}{ll} \mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} + \mu^{(k)}\mathbf{d}^{(k)} \\ \mathbf{d}^{(k+1)} & = -(\mathbf{M}^{(k+1)})^{-1}\mathbf{g}^{(k+1)} + \beta^{(k)}\mathbf{d}^{(k)} \end{array} \right. \tag{3.68}$$

Le vecteur $\mathbf{d}$ contient les directions de descente. On peut l'initialiser de telle sorte que $\mathbf{d}^{(1)} = -\mathbf{g}^{(1)}$. $\mathbf{M}$ est une matrice carrée qui tient lieu de préconditionneur, destiné à accélérer la convergence de l'algorithme. Il est conseillé dans [She94, Pol97] d'utiliser pour préconditionneur la matrice Hessienne, voire juste sa diagonale.
β est un scalaire qui peut être calculé de deux façons [Pol97] : soit par la formule de Fletcher-Reeves (β_{FR}), soit par celle de Polak-Ribière (β_{PR}). Dans chacun de ces deux cas, on a alors :

$$\beta_{\mathsf{FR}}^{(k+1)} = \frac{{\mathbf{g}^{(k+1)}}^T\mathbf{g}^{(k+1)}}{{\mathbf{g}^{(k)}}^T, \mathbf{g}^{(k)}} \tag{3.69}$$

$$\beta_{\mathsf{PR}}^{(k+1)} = \frac{{\mathbf{g}^{(k+1)}}^T(\mathbf{g}^{(k+1)} - \mathbf{g}^{(k)})}{{\mathbf{g}^{(k)}}^T\mathbf{g}^{(k)}}. \tag{3.70}$$

3.4.2 Un cas particulier

En considérant que $\beta = 0$ et $\mathbf{M} = \mathbf{I}$ dans la règle d'adaptation précédente, on retrouve la règle d'adaptation du gradient simple, à savoir, pour le cas matriciel :

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + \mu^{(k)}\mathbf{D}^{(k)}, \tag{3.71}$$

ou encore pour le cas vectoriel :

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mu^{(k)}\mathbf{d}^{(k)}, \tag{3.72}$$

donnée en (3.66).
$\mathbf{D}$ est la matrice contenant les directions de descente. Elle est définie de la manière suivante :

$$\mathbf{D^{(k)}} = \begin{pmatrix} \mathbf{D_A^{(k)}} \\ \mathbf{D_B^{(k)}} \\ \mathbf{D_C^{(k)}} \end{pmatrix}, \mathbf{d^{(k)}} = \begin{pmatrix} \mathsf{vec}\{\mathbf{D_A^{(k)}}\} \\ \mathsf{vec}\{\mathbf{D_B^{(k)}}\} \\ \mathsf{vec}\{\mathbf{D_C^{(k)}}\} \end{pmatrix} = \begin{pmatrix} \mathbf{d_A^{(k)}} \\ \mathbf{d_B^{(k)}} \\ \mathbf{d_C^{(k)}} \end{pmatrix} \tag{3.73}$$

Notons qu'ici, nous normalisons les colonnes de $\mathbf{B}$ et $\mathbf{C}$ par la norme L_1, et on peut n'utiliser que $J-1$ lignes (resp. $K-1$) pour la construction des matrices $\mathbf{D_B^{(k)}}$ et $\mathbf{D_C^{(k)}}$, la $J^{\text{ème}}$ ligne de $\mathbf{B}$ et la $K^{\text{ème}}$ ligne de $\mathbf{C}$ se déduisant directement des autres (à condition de vérifier que le terme obtenu est bien positif).
Si on pose $\mathbf{d^{(k)}} = -\mathbf{g^{(k)}}$ ou $\mathbf{D^{(k)}} = -\mathbf{G^{(k)}}$, on retombe alors bien sur la règle d'adaptation donnée en (3.66), à savoir :

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \mu^{(k)}\mathbf{G}^{(k)}, \tag{3.74}$$

ou pour le cas vectoriel

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mu^{(k)}\mathbf{g}^{(k)}. \tag{3.75}$$

3.4.3 Gradient conjugué

En considérant $\mathbf{M} = \mathbf{I}$ dans l'équation (3.68), on retombe sur la méthode du gradient conjugué :

$$\begin{cases} \mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} + \mu^{(k)}\mathbf{d}^{(k)} \\ \mathbf{d}^{(k+1)} & = -\mathbf{g}^{(k+1)} + \beta^{(k)}\mathbf{d}^{(k)} \end{cases} \tag{3.76}$$

On peut l'écrire de façon équivalente sous forme matricielle :

$$\begin{cases} \mathbf{X}^{(k+1)} & = \mathbf{X}^{(k)} + \mu^{(k)}\mathbf{D}^{(k)} \\ \mathbf{D}^{(k+1)} & = -\mathbf{G}^{(k+1)} + \beta^{(k)}\mathbf{D}^{(k)}, \end{cases} \tag{3.77}$$

Les deux expressions classiquement utilisées pour la valeur de β restent les mêmes que celles présentées précédemment, à savoir : les formules de Fletcher-Reeves (β_{FR}) et Polak-Ribière (β_{PR}) [Pol97], qu'on réécrit ici sous forme matricielle :

$$\beta_{\mathsf{FR}}^{(k+1)} = \frac{\langle \mathbf{G}^{(k+1)}, \mathbf{G}^{(k+1)} \rangle}{\langle \mathbf{G}^{(k)}, \mathbf{G}^{(k)} \rangle} = \frac{\|\mathbf{G}^{(k+1)}\|_F^2}{\|\mathbf{G}^{(k)}\|_F^2}, \tag{3.78}$$

$$\beta_{\mathsf{PR}}^{(k+1)} = \frac{\langle \mathbf{G}^{(k+1)}, \mathbf{G}^{(k+1)} - \mathbf{G}^{(k)} \rangle}{\langle \mathbf{G}^{(k)}, \mathbf{G}^{(k)} \rangle} = \frac{\langle \mathbf{G}^{(k+1)}, \mathbf{G}^{(k+1)} - \mathbf{G}^{(k)} \rangle}{\|\mathbf{G}^{(k)}\|_F^2}. \tag{3.79}$$

Ici encore, la direction de descente initiale est donnée par l'une ou l'autre des deux formules suivantes :

$$\mathbf{D}^{(1)} = -\mathbf{G}^{(1)} \tag{3.80}$$

ou sous forme vectorisée :

$$\mathbf{d}^{(1)} = -\mathbf{g}^{(1)}. \tag{3.81}$$

Dernière amélioration possible enfin : nous choisissons de réinitialiser l'algorithme ponctuellement, par exemple toutes les $(I+J+K)F$ itérations à l'aide de la relation (3.82) ou (3.83) :

$$\mathbf{D}^{(i)} = -\mathbf{G}^{(i)} \tag{3.82}$$

ou :

$$\mathbf{d}^{(i)} = -\mathbf{g}^{(i)} \tag{3.83}$$

L'objet de ce "restart" périodique est là encore l'accélération de la vitesse de convergence de l'algorithme.

3.4.4 Méthodes de Quasi-Newton : BFGS et DFP

BFGS - Si l'on considère cette fois que $\beta = 0$ au niveau de l'équation (3.68), et que le préconditionneur $\mathbf{M}$ est égal à la matrice Hessienne, de taille $(I+J+K)F \times (I+J+K)F$ (ou à une approximation de celle-ci, voire uniquement ses termes diagonaux), on obtient les approches de type Gauss-Newton ou Quasi-Newton.
Si on considère l'approximation de la matrice Hessienne donnée par la règle suivante :

$$\mathbf{M}^{(k+1)} = \mathbf{M}^{(k)} + \frac{\Delta\mathbf{g}^{(k)}(\Delta\mathbf{g}^{(k)})^T}{\langle \Delta\mathbf{g}^{(k)}, \Delta\mathbf{x}^{(k)} \rangle} - \frac{(\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)})(\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)})^T}{\langle \mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)}, \Delta\mathbf{x}^{(k)} \rangle}, \tag{3.84}$$

on peut alors définir la règle d'adaptation suivante, telle qu'elle est donnée par l'algorithme de Broyden-Fletcher-Goldfarb-Shanno (BFGS) :

$$\left\{\begin{array}{ll}\mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} - \mu^{(k)}(\mathbf{M}^{(k)})^{-1}\mathbf{g}^{(k)} \\ \Delta\mathbf{x}^{(k)} & = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} \\ \Delta\mathbf{g}^{(k)} & = \mathbf{g}^{(k+1)} - \mathbf{g}^{(k)} \\ \mathbf{M}^{(k+1)} & = \mathbf{M}^{(k)} + \frac{\Delta\mathbf{g}^{(k)}(\Delta\mathbf{g}^{(k)})^T}{\langle\Delta\mathbf{g}^{(k)},\Delta\mathbf{x}^{(k)}\rangle} - \frac{(\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)})(\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)})^T}{\langle\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)},\Delta\mathbf{x}^{(k)}\rangle}\end{array}\right. \tag{3.85}$$

En utilisant le lemme d'inversion, et en écrivant $\rho = \frac{1}{(\Delta\mathbf{g}^{(k)})^T\Delta\mathbf{x}^{(k)}}$, on peut alors estimer directement l'inverse de l'approximée de la matrice Hessienne $\mathbf{M}^{(k)}$. On réécrit ainsi l'algorithme donné en (3.85) :

$$\left\{\begin{array}{ll}\mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} - \mu^{(k)}(\mathbf{M}^{(k)})^{-1}\mathbf{g}^{(k)} \\ \Delta\mathbf{x}^{(k)} & = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} \\ \Delta\mathbf{g}^{(k)} & = \mathbf{g}^{(k+1)} - \mathbf{g}^{(k)} \\ (\mathbf{M}^{(k+1)})^{-1} & = (\mathbf{M}^{(k)})^{-1} + \rho\left[1 + \rho(\Delta\mathbf{g}^{(k)})^T(\mathbf{M}^{(k)})^{-1}\Delta\mathbf{g}^{(k)}\right]\Delta\mathbf{x}^{(k)}(\Delta\mathbf{x}^{(k)})^T \\ & \quad -\rho\Delta\mathbf{x}^{(k)}(\Delta\mathbf{g}^{(k)})^T(\mathbf{M}^{(k)})^{-1} - \rho(\mathbf{M}^{(k)})^{-1}\Delta\mathbf{g}^{(k)}(\Delta\mathbf{x}^{(k)})^T\end{array}\right. \tag{3.86}$$

DFP - Si on considère $\beta = 0$ dans l'équation (3.68), et que le préconditionneur $\mathbf{M}$ est directement égal à l'inverse de l'approximée de la matrice Hessienne, de taille $(I+J+K)F \times (I+J+K)F$, donnée ci-dessous :

$$\mathbf{M}^{(k+1)} = \mathbf{M}^{(k)} + \frac{\Delta\mathbf{x}^{(k)}(\Delta\mathbf{x}^{(k)})^T}{\langle\Delta\mathbf{g}^{(k)},\Delta\mathbf{x}^{(k)}\rangle} - \frac{(\mathbf{M}^{(k)}\Delta\mathbf{g}^{(k)})(\mathbf{M}^{(k)}\Delta\mathbf{g}^{(k)})^T}{\langle\Delta\mathbf{g}^{(k)},\mathbf{M}^{(k)}\Delta\mathbf{g}^{(k)}\rangle}, \tag{3.87}$$

on obtient cette fois la règle d'adaptation de l'algorithme de Davidon-Fletcher-Powell (DFP) :

$$\left\{\begin{array}{ll}\mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} - \mu^{(k)}\mathbf{M}^{(k)}\mathbf{g}^{(k)} \\ \Delta\mathbf{x}^{(k)} & = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} \\ \Delta\mathbf{g}^{(k)} & = \mathbf{g}^{(k+1)} - \mathbf{g}^{(k)} \\ \mathbf{M}^{(k+1)} & = \mathbf{M}^{(k)} + \frac{\Delta\mathbf{x}^{(k)}(\Delta\mathbf{x}^{(k)})^T}{\langle\Delta\mathbf{g}^{(k)},\Delta\mathbf{x}^{(k)}\rangle} - \frac{(\mathbf{M}^{(k)}\Delta\mathbf{g}^{(k)})(\mathbf{M}^{(k)}\Delta\mathbf{g}^{(k)})^T}{\langle\Delta\mathbf{g}^{(k)},\mathbf{M}^{(k)}\Delta\mathbf{g}^{(k)}\rangle}\end{array}\right. \tag{3.88}$$

Les deux algorithmes doivent être initialisés en utilisant une matrice de taille $(I+J+K)F \times (I+J+K)F$ qui est symétrique définie-positive pour $\mathbf{M}^{(1)}$ (ou $(\mathbf{M}^{(1)})^{-1}$). On peut ainsi tout à fait prendre la matrice identité. L'algorithme BFGS garantit alors la conservation de la propriété *symétrique définie-positive*. Il faut cependant noter que la matrice $\mathbf{M}^{(i)}$ tend à être de moins en moins bien conditionnée au fur et à mesure des itérations, le rang de la matrice Hessienne étant égal au nombre de ses paramètres libres, soit $(I+J+K-2)F$. La figure 3.1 montre l'évolution de ce conditionnement sur un exemple synthétique, avec un tenseur de taille $71 \times 47 \times 15$.

3.4.5 Algorithme de Levenberg-Marquardt

On peut encourager la stabilité du préconditionneur quand ce dernier tend à perdre son caractère matrice "définie-positive" au fil des itérations et/ou n'est plus inversible, en ajoutant

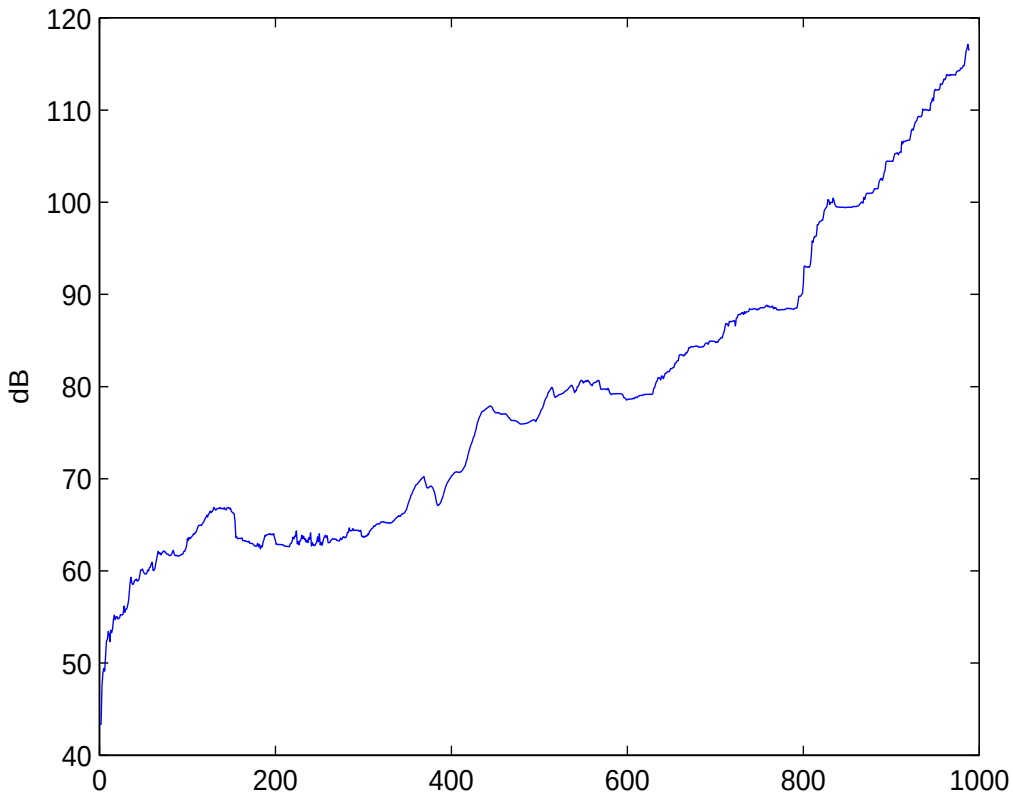


FIGURE 3.1 – Conditionnement de la matrice Hessienne au fil des itérations.

un multiple de la matrice identité à la matrice $\mathbf{M}$ avant son inversion. Ceci est connu comme l'approche Levenberg-Marquardt [Lue69] :

$$\begin{cases} \mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} - \mu^{(k)}(\mathbf{M}^{(k)} + \alpha \mathbf{I}_{(\mathbf{I+J+K})\mathbf{F}})^{-1}\mathbf{g}^{(k)} \\ \Delta\mathbf{x}^{(k)} & = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} \\ \Delta\mathbf{g}^{(k)} & = \mathbf{g}^{(k+1)} - \mathbf{g}^{(k)} \\ \mathbf{M}^{(k+1)} & = \mathbf{M}^{(k)} + \frac{\Delta\mathbf{g}^{(k)}(\Delta\mathbf{g}^{(k)})^T}{\langle \Delta\mathbf{g}^{(k)}, \Delta\mathbf{x}^{(k)} \rangle} - \frac{(\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)})(\mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)})^T}{\langle \mathbf{M}^{(k)}\Delta\mathbf{x}^{(k)}, \Delta\mathbf{x}^{(k)} \rangle} \end{cases} \tag{3.89}$$

où α est un coefficient de relaxation. Notons qu'en posant $\alpha = 0$ dans (3.89), l'algorithme (3.85) est retrouvé. De même, en posant $\mathbf{M} = \mathbf{I}_{(I+J+K)F}$ ou en considérant que α est choisi suffisamment grand par rapport aux valeurs de $\mathbf{M}$, on retombe sur l'algorithme du gradient qui avait été donné dans l'équation (3.71).

3.5 Ajout de termes de pénalisation

On peut renforcer certaines propriétés de la solution recherchée en ajoutant des termes de pénalisation à la fonction de coût (cf. la méthode combinant pénalisation et projection suggérée

dans [CZPA09] évoquée au niveau de la section 3.2.2). Nous proposons ici de donner les expressions des termes de pénalisation dans le cas de l'utilisation d'une norme L_1, qui ont vocation à aiguiller vers une solution parcimonieuse, ainsi que celles des termes sous pénalisation de type norme L_2, qui visent à lisser la solution. Nous les avions déjà donnés respectivement dans les équations (3.20)-(3.22) et (3.17)-(3.19) dans le cas où aucune contrainte de non négativité n'était imposée.
Voici l'expression des fonctions de coût pénalisées. D'abord dans le cas de la norme L_1 :

$$\mathcal{H}_{L_1}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + \alpha_A\|\mathbf{A}\boxdot\mathbf{A}\|_1 + \alpha_B\|\mathbf{B}\boxdot\mathbf{B}\|_1 + \alpha_C\|\mathbf{C}\boxdot\mathbf{C}\|_1, \quad (3.90)$$

et dans le cas de la norme L_2 :

$$\mathcal{H}_{L_2}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}), +\alpha_A\|\mathbf{A}\boxdot\mathbf{A}\|_F^2 + \alpha_B\|\mathbf{B}\boxdot\mathbf{B}\|_F^2 + \alpha_C\|\mathbf{C}\boxdot\mathbf{C}\|_F^2 \quad (3.91)$$
$$(3.92)$$

Les gradients associés sont, dans le cas de la norme L_1 :

$$\nabla_{\mathbf{A}}(\mathcal{H}_{L_1}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \nabla_{\mathbf{A}}(H(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + 2\alpha_A\mathbf{A} \quad (3.93)$$
$$\nabla_{\mathbf{B}}(\mathcal{H}_{L_1}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \nabla_{\mathbf{B}}(H(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + 2\alpha_B\mathbf{B} \quad (3.94)$$
$$\nabla_{\mathbf{C}}(\mathcal{H}_{L_1}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \nabla_{\mathbf{C}}(H(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + 2\alpha_C\mathbf{C}, \quad (3.95)$$
$$(3.96)$$

et dans le cas de la norme L_2 :

$$\nabla_{\mathbf{A}}(\mathcal{H}_{L_2}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \nabla_{\mathbf{A}}(H(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + 4\alpha_A\mathbf{A}\boxdot\mathbf{A}\boxdot\mathbf{A} \quad (3.97)$$
$$\nabla_{\mathbf{B}}(\mathcal{H}_{L_2}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \nabla_{\mathbf{B}}(H(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + 4\alpha_B\mathbf{B}\boxdot\mathbf{B}\boxdot\mathbf{B} \quad (3.98)$$
$$\nabla_{\mathbf{C}}(\mathcal{H}_{L_2}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \nabla_{\mathbf{C}}(H(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + 4\alpha_C\mathbf{C}\boxdot\mathbf{C}\boxdot\mathbf{C}. \quad (3.99)$$

Les détails des calculs sont donnés en annexe 6.4.1.2.

3.6 Méthodes de détermination du pas d'adaptation

Les algorithmes de descente présentés à partir du paragraphe (3.68) requièrent tous l'estimation d'un pas d'adaptation μ scalaire qui varie à chaque itération. Il reste bien sûr toujours possible d'utiliser un pas fixe tout au long des itérations, mais cela risque d'être synonyme d'une faible vitesse de convergence si le pas est choisi trop petit. Dans le cas contraire, càd si on choisit un pas trop grand, l'algorithme risque de diverger, et on se trouvera alors dans l'incapacité d'estimer correctement les matrices de facteurs. Le pas choisi peut également s'avérer correctement dimensionné au niveau des premières itérations, puis se révéler trop grand par la suite. Il semble alors judicieux de choisir un pas suffisamment faible pour ne pas tomber dans ce type de problème, mais cela porte inévitablement préjudice à la vitesse de convergence.

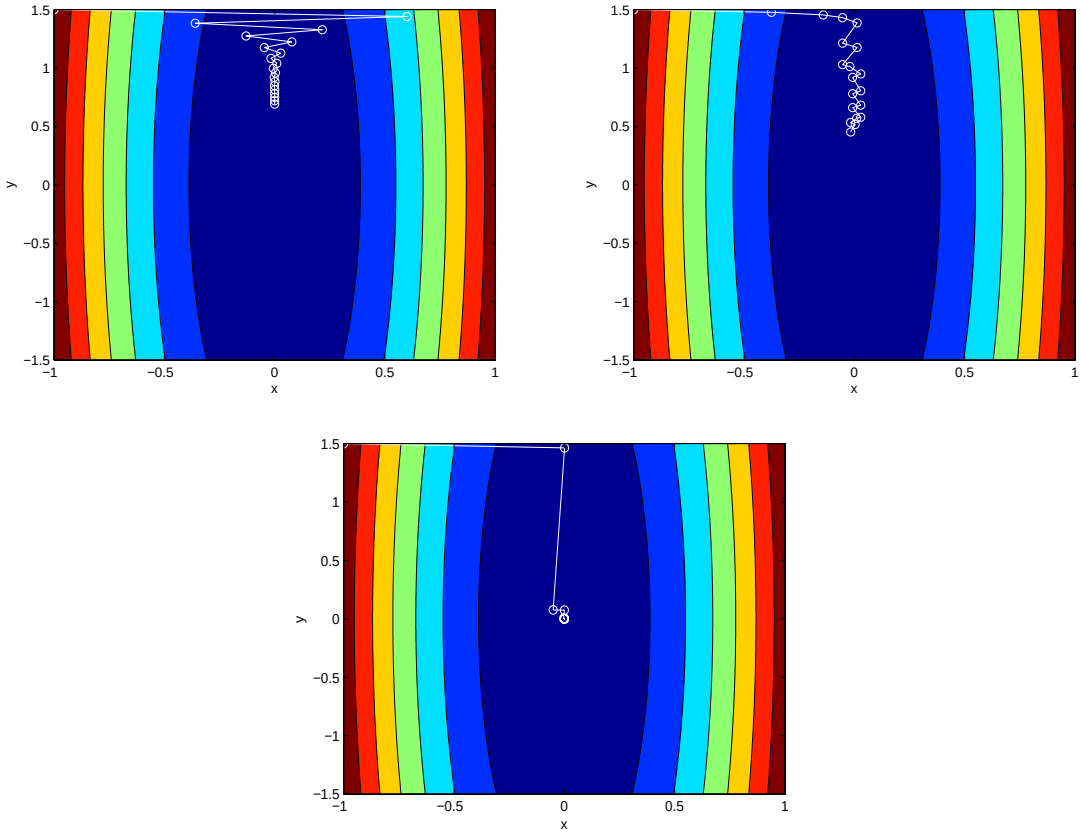


FIGURE 3.2 – Comparaison de la vitesse de convergence en fonction du pas d'adaptation choisi, pour les 20 premières itérations de l'algorithme du gradient. En haut à gauche, le pas fixe ($\mu = 0.04$). En haut à droite, le pas approché. En bas, le pas globalement optimal.

Plusieurs solutions existent pour palier ce problème. Les plus connues sont le calcul du pas optimal à chaque itération, et le calcul d'un pas approximé par marche arrière.
Avant d'aborder le principe du calcul du pas optimal et du pas approché par marche arrière, voici sur un exemple simple, une illlustration/comparaison du comportement du même algorithme d'optimisation dans chacune de ses trois versions à pas fixe, à pas approché par marche arrière ou enfin à pas optimal. Considérons la fonction ci-dessous :

$$f(x, y) = \frac{1}{2}(40x^2 + y^2) \tag{3.100}$$

Les résultats sont donnés sur la figure 3.2. On a y a tracé les 20 premières itérations pour chaque cas de figure (pas fixe, pas approché par marche arrière et pas globalement optimal). On vérifie bien que l'algorithme à pas fixe (avec un pas $\mu = 0.04$) est le plus lent à atteindre le minimum de la fonction donnée au niveau de l'équation (3.100). La recherche par marche arrière donne des résultats plus satisfaisants. Le pas optimal est de loin le plus rapide pour converger.

3.6.1 Pas optimal

Le calcul du pas optimal permet, comme son nom l'indique, de déterminer le pas d'adapatation qui minimisera au maximum la fonction de coût à l'étape k, en fonction de la direction de descente calculée. L'objectif est calculer $\mu^{(k)}$ tel que pour une fonction $f(\cdot)$ dépendant d'un paramètre x :

$$\mu^{(k)} = \underset{\mu^{(k)} \geq 0}{\arg\min}\, f(x + \mu^{(k)} d) \tag{3.101}$$

Dans le cas de la factorisation de tenseurs du 3ème ordre, il s'agit de minimiser la fonction :

$$\mathcal{H}(\mathbf{A}^{(k+1)}, \mathbf{B}^{(k+1)}, \mathbf{C}^{(k+1)}) = \mathcal{H}\left[(\mathbf{A}^{(k)} + \mu {\mathbf{D}_\mathbf{A}}^{(k)}), (\mathbf{B}^{(k)} + \mu {\mathbf{D}_\mathbf{B}}^{(k)}), (\mathbf{C}^{(k)} + \mu {\mathbf{D}_\mathbf{C}}^{(k)})\right]. \tag{3.102}$$

De façon similaire, dans le cas de la factorisation de tenseurs du 3ème ordre sous la contrainte de non négativité imposée par produits de Hadamard, en posant $\mathbf{P_A} = \mathbf{A}^{(k)} + \mu\mathbf{D_A}$, $\mathbf{P_B} = \mathbf{B}^{(k)} + \mu\mathbf{D_B}$, $\mathbf{P_C} = \mathbf{C}^{(k)} + \mu\mathbf{D_C}$, on doit trouver $\mu(k)$ qui minimise :

$$\mathcal{H}(\mathbf{A}^{(k+1)}, \mathbf{B}^{(k+1)}, \mathbf{C}^{(k+1)}) = \mathcal{H}\left[\mathbf{P_A} \boxdot \mathbf{P_A}, \mathbf{P_B} \boxdot \mathbf{P_B}, \mathbf{P_C} \boxdot \mathbf{P_C}\right]. \tag{3.103}$$

Le détail des calculs est donné au niveau de l'annexe 6.4.3. Il s'agit en fait d'un polynôme en μ de degré 12, dont l'expression est donnée par (nous avons choisi d'omettre la dépendance vis à vis des paramètres de $\mathcal{H}$ pour simplifier les expressions) :

$$\mathcal{H}(.) = \sum_{i=0}^{12} a_i \mu^i, \tag{3.104}$$

où les 13 coefficients a_i, pour $i = 0, \ldots, 12$ valent (la définition de $\mathbf{K}_i$, où i varie de 1 à 6 est donnée en annexe 6.4.3) :

$$a_0 = \mathsf{trace}\left[\mathbf{K_0}{\mathbf{K_0}}^T\right] \tag{3.105a}$$
$$a_1 = \mathsf{trace}\left[2\mathbf{K_1}{\mathbf{K_0}}^T\right] \tag{3.105b}$$
$$a_2 = \mathsf{trace}\left[2\mathbf{K_2}{\mathbf{K_0}}^T + \mathbf{K_1}{\mathbf{K_1}}^T\right] \tag{3.105c}$$
$$a_3 = \mathsf{trace}\left[2\left(\mathbf{K_3}{\mathbf{K_0}}^T + \mathbf{K_2}{\mathbf{K_1}}^T\right)\right] \tag{3.105d}$$
$$a_4 = \mathsf{trace}\left[2\left(\mathbf{K_4}{\mathbf{K_0}}^T + \mathbf{K_3}{\mathbf{K_1}}^T\right) + \mathbf{K_2}{\mathbf{K_2}}^T\right] \tag{3.105e}$$
$$a_5 = \mathsf{trace}\left[2\left(\mathbf{K_5}{\mathbf{K_0}}^T + \mathbf{K_4}{\mathbf{K_1}}^T + \mathbf{K_3}{\mathbf{K_2}}^T + \mathbf{K_3}{\mathbf{K_2}}^T\right)\right] \tag{3.105f}$$
$$a_6 = \mathsf{trace}\left[2\left(\mathbf{K_6}{\mathbf{K_0}}^T + \mathbf{K_5}{\mathbf{K_1}}^T + \mathbf{K_4}{\mathbf{K_2}}^T\right) + \mathbf{K_3}{\mathbf{K_3}}^T\right] \tag{3.105g}$$
$$a_7 = \mathsf{trace}\left[2\left(\mathbf{K_6}{\mathbf{K_1}}^T + \mathbf{K_5}{\mathbf{K_2}}^T + \mathbf{K_4}{\mathbf{K_3}}^T\right)\right] \tag{3.105h}$$
$$a_8 = \mathsf{trace}\left[2\left(\mathbf{K_6}{\mathbf{K_2}}^T + \mathbf{K_5}{\mathbf{K_3}}^T\right) + \mathbf{K_4}\mathbf{K_4}\right] \tag{3.105i}$$
$$a_9 = \mathsf{trace}\left[2\left(\mathbf{K_6}{\mathbf{K_3}}^T + \mathbf{K_5}{\mathbf{K_4}}^T\right)\right] \tag{3.105j}$$
$$a_{10} = \mathsf{trace}\left[2\mathbf{K_6}{\mathbf{K_4}}^T + \mathbf{K_5}{\mathbf{K_5}}^T\right] \tag{3.105k}$$
$$a_{11} = \mathsf{trace}\left[2\mathbf{K_6}{\mathbf{K_5}}^T\right] \tag{3.105l}$$
$$a_{12} = \mathsf{trace}\left[\mathbf{K_6}{\mathbf{K_6}}^T\right] \tag{3.105m}$$

En dérivant l'expression de $\mathcal{H}$ par rapport à μ, on obtient un polynôme de degré 11 donné par l'équation (3.106) :

$$\mathsf{d}\mathcal{H}(.) = \sum_{i=0}^{11}(i+1)a_{i+1}\mu^i, \tag{3.106}$$

Le pas optimal $\mu_{\mathsf{opt}}^{(k)}$ correspond alors à la racine réelle et positive du polynôme (3.106) menant au minimum du critère donné en (3.104). Ces racines sont estimées numériquement car pour des degrés aussi élevés, il n'existe généralement pas de formule algébrique pour le calcul des racines.

3.6.2 Une solution inexacte, mais rapide : méthode par marche arrière

On peut se demander pourquoi il est utile de rechercher une valeur approchée du pas d'adaptation, quand il est possible comme ici d'en calculer analytiquement la valeur optimale. Le calcul du pas optimal est une solution coûteuse en terme de temps de calcul qui consiste à calculer les racines d'un polynôme de degré élevé dans notre cas. Nous montrerons d'ailleurs au chapitre 5, que c'est le calcul des coefficients du polynôme qui est le plus coûteux en terme de temps de calcul. Les coefficients donnés dans le paragraphe précédent nécessitent le calcul de produits scalaires de grosses matrices. Dans des problèmes de grandes dimensions (lorsque le tenseur des observations est constitué de larges images par exemple), le calcul du pas optimal représente la majeure partie de la complexité algorithmique totale à chaque itération.
Il vaut mieux, alors, lui préférer une version approchée, comme la méthode par marche arrière (*backtracking*, qui permet à moindre coût d'obtenir une bonne approximation de $\mu^{(k)}$. L'idéal étant de combiner une méthode par marche arrière avec le calcul du pas optimal seulement toutes les 10 ou 20 (voire plus) itérations.
$\mu^{(k)}$ est toujours choisi de sorte à minimiser (approximativement cette fois) une fonction $f(\cdot)$, comme celle donnée dans l'exemple (3.101), le long de la ligne $x + \mu^{(k)}d$, avec $\mu^{(k)} \geq 0$.
μ est choisi suffisamment grand au début, par exemple un pas unité, et est décrémenté d'un facteur β, soit $\mu = \beta\mu$ jusqu'à ce que la condition d'Armijo (3.107) [BV04, LY08], soit satisfaite. Le pas μ résultant est celui retenu pour le $\mu^{(k)}$ utilisé dans la règle d'adaptation de l'algorithme de descente considéré. En reprenant nos notations précédentes, et en considérant une fonction de coût notée $\mathcal{H}(\cdot)$, la condition d'Armijo s'écrit :

$$\mathcal{H}(\mathbf{A}+\mu\mathbf{D_A}, \mathbf{B}+\mu\mathbf{D_B}, \mathbf{C}+\mu\mathbf{D_C}) < \mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) + \alpha\ \mu\ \mathbf{g}^T\mathbf{d} \tag{3.107}$$

où α est un paramètre constant choisi dans l'intervalle $[10^{-4},\ 10^{-1}]$, $\mathbf{d}$ est la direction de descente donnée en (3.73) et $\mathbf{g}$ est le gradient donné en (3.59). Notons que puisque $\mathbf{d}$ est une direction de descente, on a $\mathbf{g}^T\mathbf{d} < 0$ (ce n'est autre que la dérivée directionnelle au point considéré). Dans le cas de l'algorithme du gradient, $\mathbf{d} = -\mathbf{g}$, alors que $\mathbf{d} = -\mathbf{M}^{-1}\mathbf{g}$ pour les algorithmes de type Quasi-Newton. La condition d'Armijo indique que le pas doit être tel que sa valeur est celle qui fait se couper la fonction de coût et la droite de pente γ plus faible que la droite ayant pour pente la dérivée directionnelle. Le principe de cette méthode est illustré au niveau de la figure 3.3.

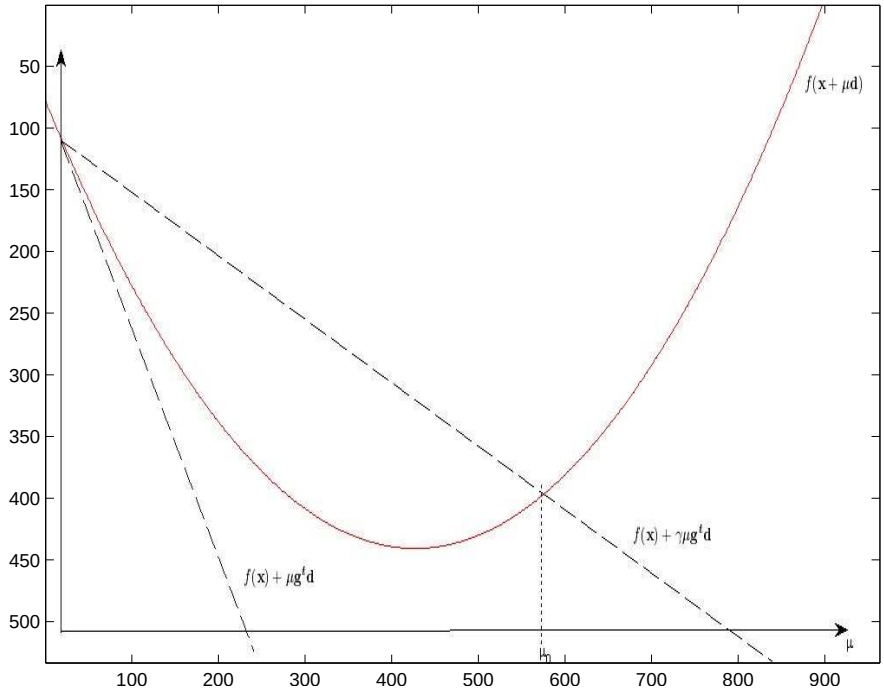


FIGURE 3.3 – Modélisation graphique de la condition d'Armijo. La fonction de coût est représentée en coupe. La condition devient valide dès que f passe en dessous de la ligne en tirets supérieure, i.e. $0 \leq \mu \leq \mu_0$.

3.7 Comparaison des différents algorithmes sur des mélanges synthétiques

Ce paragraphe est dévolu à la comparaison des différents algorithmes de décomposition de tenseurs d'ordre 3 faisant intervenir une contrainte de non négativité sur les matrices de facteurs recherchées. Nous appliquons les diverses méthodes à des mélanges synthétiques d'images de fluorescence de composés organiques classiquement rencontrés dans l'analyse d'échantillons d'eau.

Première série de simulations - Deux tenseurs T_1 and T_2 ont été simulés en utilisant les images de fluorescence de $F = 4$ composés dont les matrices d'émission excitation ($\mathbf{a}_i\mathbf{b}_i^T$, $\forall i = 1, \ldots, 4$) sont données au niveau de la figure 3.4. Ces images [MZBR10] nous ont été fournies par le laboratoire PROTEE-EA 3819 de l'Université du Sud Toulon Var (France). Deux matrices positives $\mathbf{C}$ ont été générées de manière aléatoire suivant une loi uniforme (une matrice 10×4 et une matrice 128×4). Disposant des images de fluorescence des composés, et des concentrations de chaque composé pour chaque échantillon, le tenseur peut alors être reconstruit tranche par tranche, une tranche étant une image de fluorescence du mélange. Le premier tenseur T_1 construit est donc dans cet exemple de taille $71 \times 47 \times 10$ et le second T_2 est de taille $71 \times 47 \times 128$.
Pour pouvoir établir une comparaison entre les différents algorithmes, nous avons besoin d'indices de performance judicieux ou plutôt ici une mesure de erreur commise. Nous choisissons

d'utiliser un premier indice d'erreur, défini de la manière suivante :

$$E = \|\mathbf{T} - \widehat{\mathbf{T}}\|_F^2 \quad \text{ou} \quad E_{\mathsf{dB}} = 10\log_{10}(E), \tag{3.108}$$

avec $\widehat{\mathbf{T}} = \sum_{f=1}^{F} \widehat{\mathbf{a}}_f \circledast \widehat{\mathbf{b}}_f \circledast \widehat{\mathbf{c}}_f$ et $\widehat{\mathbf{a}}_f$, $\widehat{\mathbf{b}}_f$ et $\widehat{\mathbf{c}}_f$ pour $f = 1, \ldots, F$ les estimés de la f-ème colonne des différentes matrices de facteurs.
Les meilleurs résultats sont obtenus quand l'indice d'erreur E est proche de 0 sur une échelle linéaire ($-\infty$ en échelle logarithmique).
Nous verrons par la suite que cet indice n'est pas toujours critère optimal de qualité. Voilà pourquoi nous en définissons un deuxième, qui ne peut pas être utilisé sur des données expérimentales. Il est basé sur l'erreur entre la MEEF de l'estimation et la MEEF de référence :

$$E_{2\mathsf{dB}} = 10\mathsf{log}_{10}(\sum_{i=1}^{F} \|\mathbf{a}_i\mathbf{b}_i^T - \widehat{\mathbf{a}}_i\widehat{\mathbf{b}}_i^T)\|_F) \tag{3.109}$$

En premier lieu, pour comparer les images correspondant au même composé organique, les MEEF de référence et estimées doivent être normalisées, puis triées.

Les simulations qui suivent mettent en confrontation les algorithmes de la littérature NTF-ALS (pénalisé avec une norme L_1 pour lisser les résultats et rappelé dans la section 3.2.2) et NTF-HALS (rappelé dans la section 3.2.2 et implémenté selon la description faite p. 357 de [CZPA09]) avec les algorithmes développés et présentés précédemment : gradient et gradient conjugué avec contrainte de non négativité par produits de Hadamard, BFGS et DFP .
Les 4 MEEF de référence sont montrées sur la figure 3.4. Sur la figure 3.5, on affiche les MEEF estimées par gradient conjugué avec contrainte de non négativité (ELS) et avec le BFGS (ELS aussi). Notons tout d'abord que les résultats donnés par ces deux algorithmes sont identiques ici. De la même façon, on observe que qu'on ne peut pas différencier les estimations des images de référence. Dans un cas idéal, sans bruit, et quand la décomposition se fait à la valeur du rang du tenseur, ces algorithmes se comportent donc d'excellente façon. Notons que le NTF-ALS et NTF-HALS donnent ici les mêmes résultats.
Attardons-nous maintenant sur divers points de comparaisons des performances des méthodes. Les figures 3.6 a), 3.6 b), 3.6 c), 3.6 d), 3.7 et 3.8 ont été obtenues en initialisant tous les algorithmes avec avec la même solution initiale calculée au moyen de l'algorithme DTLD présenté dans [SBG04] et proposé dans la toolbox de R. Bro et Claus A. Andersson [BA]. Pour l'algorithme ALS – Cichocki régularisé soit par une norme L_1 soit par une norme L_2, les coefficients de régularisation ont été choisis tels que $\alpha_A = \alpha_B = \alpha_C = 10^{-6}$ (c'est cette régularisation qui explique le fait que l'erreur soit bornée). Pour ces figures, quand la recherche linéaire globale (ELS) est effectuée, elle l'est à chaque itération. Notons toutefois que des versions de l'algorithme ALS faisant intervenir de l'ELS ont été suggérées dans ([RCH08]) par exemple, mais pour une version sans contrainte de non négativité.
Au niveau des figures 3.6 a) et 3.6 b), nous affichons E_{dB} qui mesure l'erreur entre le tenseur de départ et le tenseur reconstruit en échelle logarithmique (la légende pour ces deux courbes est affichée au niveau de la figure 3.6 a)). Pour ces deux figures, l'apparence des courbes est très similaires. Seul change le nombre d'itérations mis pour atteindre des performances similaires,

puisque la taille du tenseur n'est pas la même (dans le mode des concentrations). Le NTF-ALS (L_1 ou L_2) et son dérivé, le NTF-HALS , sont les plus rapides à atteindre la convergence. Du fait de sa régularisation, les performances du NTF-ALS sont cependant bornées, ce qui explique que l'erreur de reconstruction stagne à environ -40 dB sur les deux figures. Nous verrons par la suite qu'une faible erreur de reconstruction n'implique pas nécessairement une bonne approximation des MEEF. Les deux algorithmes de Quasi-Newton, à savoir, le BFGS et le DFP, donnent des résultats très proches et convergent rapidement en dessous des -140 dB. Le gradient conjugué met un peu plus d'itérations pour atteindre ce même résultat (7 fois plus sur le tenseur T_1, et 4 fois plus sur le tenseur T_2). Le gradient donne lui les plus mauvais résultats. Sa courbe de convergence est extrêmement lente, si bien qu'il ne passe sous les -45 dB qu'après, respectivement, 3500 et 7000 itérations pour les deux figures.

Sur les figures 3.6 c) et 3.6 d), l'erreur est présentée non plus en fonction du nombre d'itérations mais en fonction de la complexité algorithmique. Les courbes présentés ici gardent le même aspect que celles tracées sur les figures en fonction des itérations. En effet, les algorithmes basés sur l'ALS ont un coût algorithmique ridiculement faible. BFGS et DFP, bien que présentant un coût par itération supérieur aux algorithmiques basés sur le calcul du gradient seul, requièrent ici moins de temps pour descendre jusqu'à -140 dB. C'est dû au fait que le gradient conjugué présente des paliers de plus en plus marqués alors que l'erreur de reconstruction décroît, rallongeant ainsi la durée de calcul. Notons cependant qu'il n'est pas nécessaire d'atteindre des erreurs aussi basses pour avoir une bonne approximation de la décomposition. Pour finir, le gradient, bien qu'ayant une faible complexité algorithmique, ne converge pas suffisamment rapidement pour s'avérer être un choix intéressant.

Pour résumer sur ces 4 figures, les algorithmes les plus rapides à atteindre la convergence sont ceux basés sur l'ALS. Nous verrons toutefois que cet indice de performances n'est pas un gage de qualité. Les deux algorithmes de Quasi-Newton requièrent ici moins de temps que le gradient conjugué pour atteindre la convergence. Cependant, ce dernier offre le meilleur compromis entre vitesse et performances. Contrairement aux algorithmes de type Newton-Raphson ou Quasi-Newton, il ne demande pas l'estimation des matrices Hessiennes de taille $(I+J+K)F \times (I+J+K)F$ (ou de leur approximation), ce qui fait qu'il peut être appliqué même dans le cas de tenseurs de très grande taille.

Intéressons-nous maintenant à une comparaison des performances entre le backtracking et l'ELS . On affiche sur la figure 3.7 l'erreur de reconstruction en fonction du nombre d'itérations, et cela appliqué au tenseur $\mathbf{T}_2$ en utilisant le BFGS . La première courbe montre donc l'évolution de l'erreur E pour du BFGS faisant intervenir de l'ELS à chaque itération. La deuxième courbe montre la même chose mais l'ELS n'intervient que toutes les 10 itérations. Le backtracking est donc appliqué le reste du temps. L'ELS étant une recherche globale et optimale du pas d'adaptation, la première courbe met logiquement moins d'itérations pour atteindre les mêmes performances, mais l'écart avec la deuxième courbe reste faible. Si on regarde la figure 3.8 qui donne cette fois-ci l'erreur de reconstruction en fonction de la complexité algorithmique, on s'aperçoit qu'alterner du backtracking avec de l'ELS, demande en fin de compte moins de temps que faire uniquement de l'ELS. Sur des problèmes de taille importante, il est donc recommandé d'utiliser l'ELS de façon très parcimonieuse (exemple : toutes les 10 ou 20 itérations) et d'utiliser du backtracking le reste du temps.

Revenons à l'estimation des MEEF. Si ces dernières sont parfaitement estimées quel que soit l'algorithme mis en jeu à modèle exact, examinons le comportement des méthodes présentées dans le cas où on introduit des erreurs de modèle. Prenons le cas d'une surestimation du nombre de composés, (dans cet exemple $F_{\text{estimé}} = 5$ alors que $F_{\text{réel}} = 4$). On affiche les images reconstruites par le gradient conjugué sur la figure 3.9, le BFGS ($\alpha = 10^{-4}$) (le DFP donne ici la même chose) sur la figure 3.10 et on les confronte à celles reconstruites via le NTF-ALS (figure 3.11 et le NTF-HALS (figure 3.12). Tous ces algorithmes ont été initialisés à partir des mêmes matrices de facteurs aléatoires (loi uniforme sans valeurs négatives). Le gradient conjugué réestime très bien les 4 composés du mélange, la $5^{\text{ème}}$ image étant alors logiquement nulle, ou presque (pour parvenir à ce résultat, nous pénalisons avec une norme L_1 de coefficient $\alpha = 10^{-5}$). C'est un peu moins vrai pour le BFGS, dont la reconstruction est un peu moins bonne. Rappelons que la matrice Hessienne du BFGS tend à être de moins en moins bien conditionnée au fur et à mesure des itérations. C'est encore moins vrai pour NTF-ALS et NTF-HALS qui dispatchent de la même façon l'un des composés sur 2 images. Ces derniers ont donc tendance à forcer l'estimation d'un $5^{\text{ème}}$ fluorophore inexistant dans le mélange. Les algorithmes que nous avons développés précédemment se montrent donc pour cet exemple moins sensibles à ce type d'erreur sur le modèle. Ceci nous permet de faire intervenir notre deuxième indice de performances E_{2dB}. Comme nous l'avons dit, une faible erreur de reconstruction n'implique pas nécessairement une bonne réestimation des MEEF. Sur la figure 3.13, nous superposons les courbes des 3 algorithmes précédents, qui montrent l'évolution de l'indice E_2 en fonction du nombre d'itération. Cette fois-ci, le gradient conjugué présente des performances bien meilleures que le NTF-ALS et NTF-HALS , ce qui conforte ce que nous venons de dire quant à l'aspect visuel des MEEF estimées. Le BFGS montre une certaine instabilité, qui peut être liée à la dégradation du conditionnement de la matrice Hessienne approximée.

Pour terminer vérifions la sensibilité de ces algorithmes à l'initialisation. Sur les exemples précédents relatifs à la surestimation du rang, l'algorithme du gradient conjugué que nous avons développé ici donnait les meilleurs résultats. On repart de cet algorithme, qu'on confronte à ceux de la littérature et du tenseur précédent, toujours avec $F_{\text{estimé}} = 5$, tandis que $F_{\text{réel}} = 4$. On itère sur 100 réalisations, en générant pour chacune une initialisation basée sur des matrices de facteurs aléatoires. On récupère pour chaque réalisation l'indice de performances E_{2dB}. On trace en 3.14 la courbe moyennée point à point sur les 100 réalisations, tout au long des itérations de l'algorithme. On observe ainsi le bon comportement de l'algorithme du gradient conjugué sous contrainte, qui affiche en moyenne un indice E_{2dB} bien inférieur à ce qu'on peut obtenir avec les algorithmes basés sur l'ALS. Sur l'autre figure tracée en 3.14, on affiche la courbe représentant la dernière valeur (à la dernière itération) de E_{2dB} pour les 100 réalisations, triées par ordre croissant, ce qui nous donne un aperçu de la répartition des valeurs pour les 100 réalisations. Ainsi, 90% d'entre elles finissent sous les 10 dB, ce qui, logiquement en accord avec la courbe précédente, est meilleur que les autres algorithmes présentés sur la figure 3.13. Le HALS ne descend sous les 10 dB que dans environ 20% des cas environ, tandis que l'ALS dépasse les 20 dB dans 30% des cas.

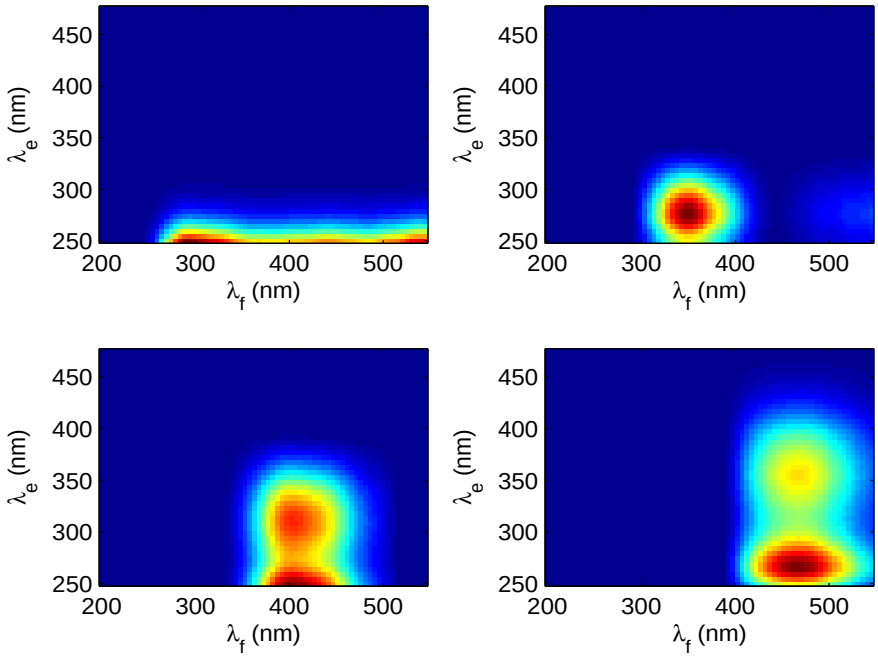


FIGURE 3.4 – Les MEEF des 4 composés de référence (avant mélange).

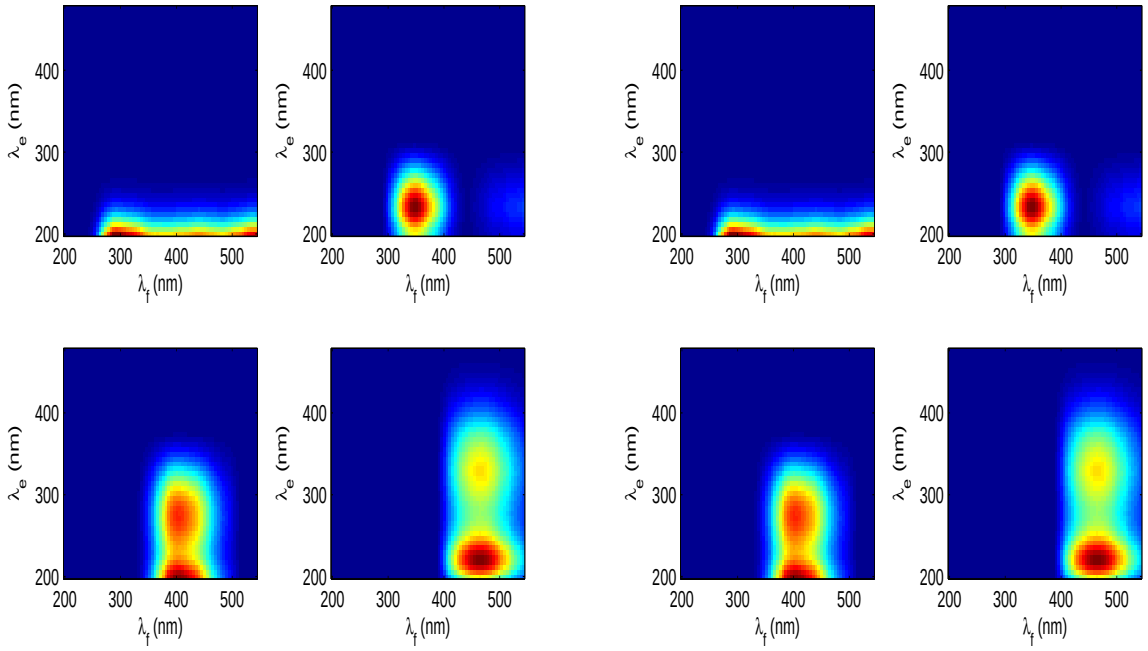
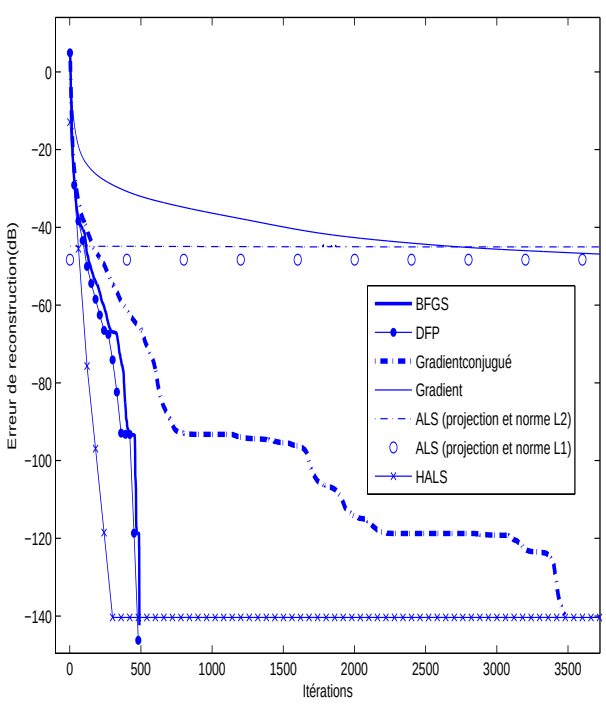
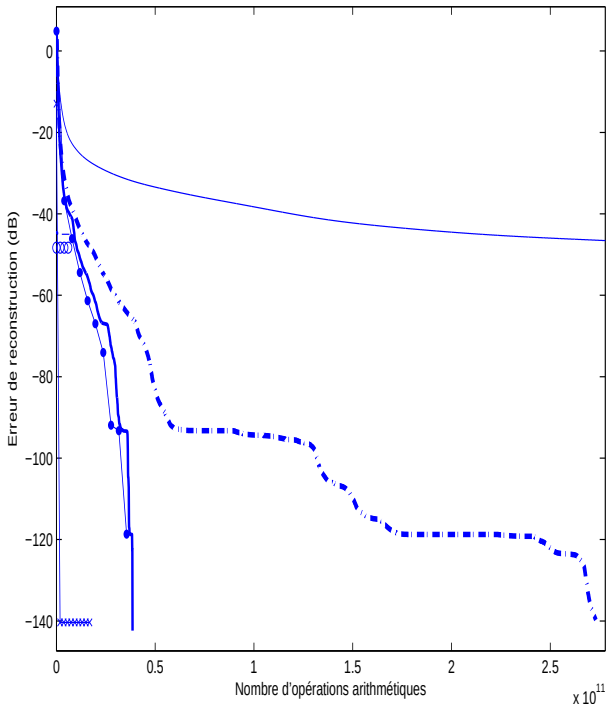


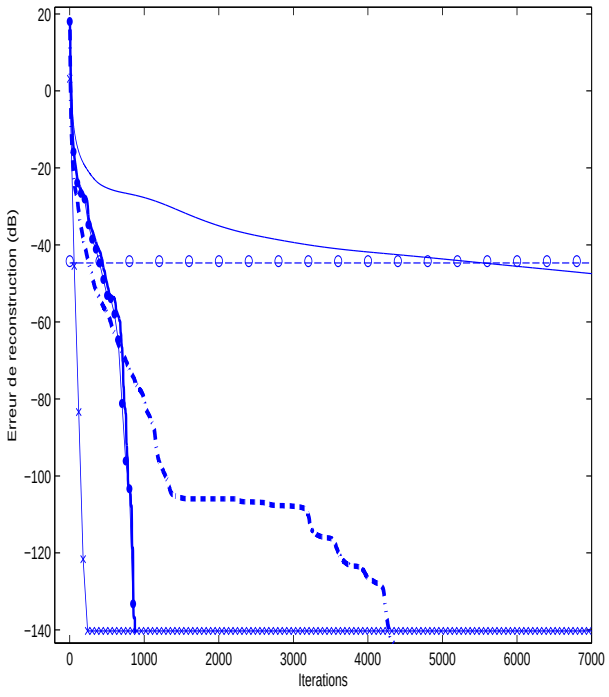
FIGURE 3.5 – Réestimation des 4 composés via l'algorithme du gradient conjugué (gauche) et BFGS (droite), tous deux avec contrainte de non négativité imposée par produits de Hadamard.



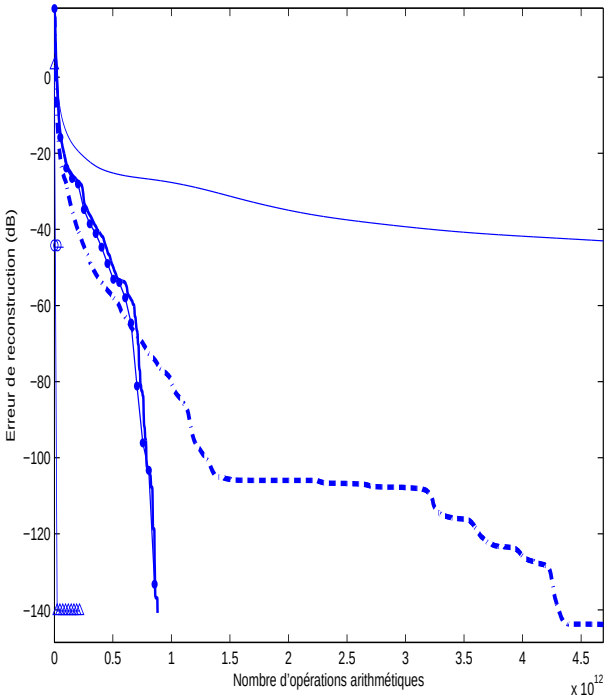
a) Erreur en fonction du nombre d'itérations.



b) Erreur en fonction de la complexité algorithmique.



c) Erreur en fonction du nombre d'itérations.



d) Erreur en fonction de la complexité algorithmique.

FIGURE 3.6 – Erreur de reconstruction (dB) en fonction du nombre d'itérations (gauche) pour un tenseur non négatif $71 \times 47 \times 10$ (haut gauche), un tenseur non négatif de taille $71 \times 47 \times 128$ (bas gauche). Erreur de reconstruction (dB) en fonction du nombre d'opérations arithmétiques (droite) pour un tenseur non négatif $71 \times 47 \times 10$ (haut droite), un tenseur non négatif de taille $71 \times 47 \times 128$ (bas droite). La même légende est utilisée pour les 4 sous-figures. Notons qu'une faible erreur de reconstruction n'implique pas que les matrices de facteurs soient correctement estimées. Il faut aussi que le nombre de composés soit correctement détecté (cf. figures 3.9 et 3.12).

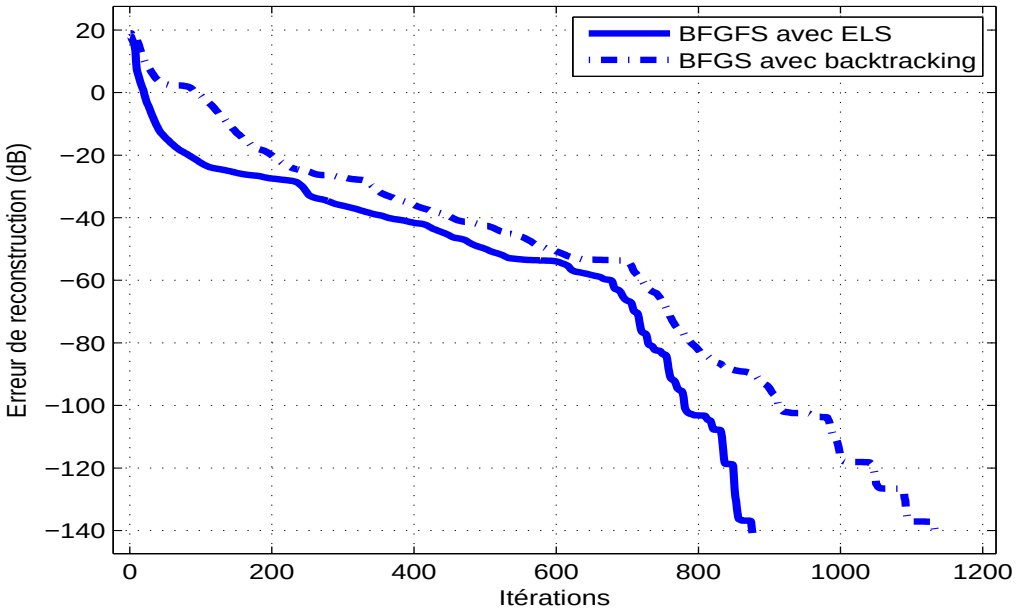


FIGURE 3.7 – Comparaison du BFGS avec le backtracking (alternant avec de l'ELS toutes les 10 itérations) et BFGS avec ELS à chaque itération. Erreur de reconstruction en fonction du nombre d'itérations.

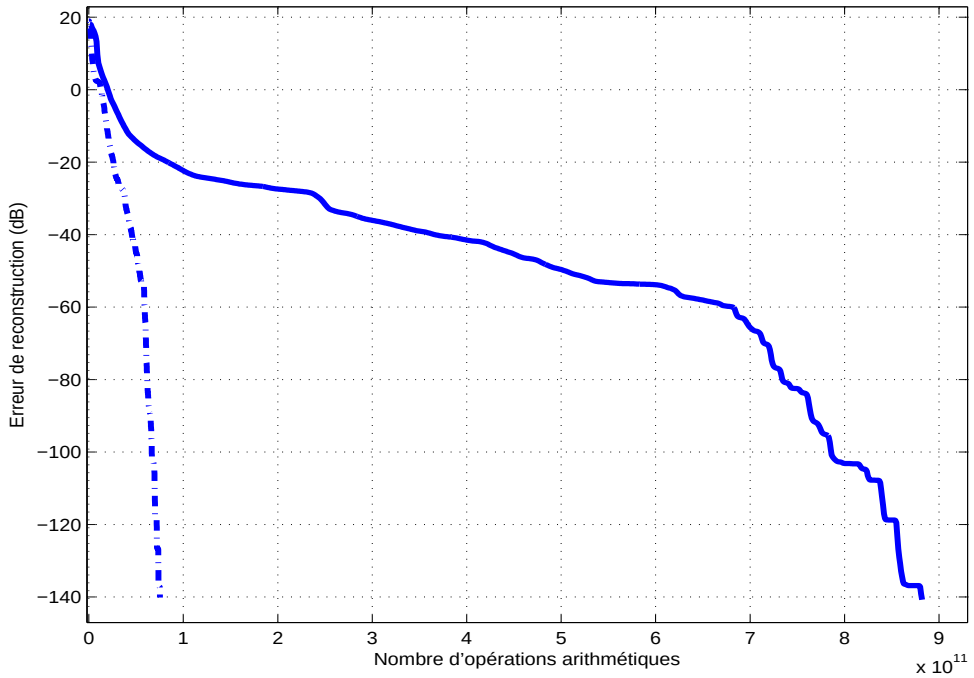


FIGURE 3.8 – Comparaison du BFGS avec backtracking (alternant avec de l'ELS toutes les 10 itérations) et du BFGS (avec ELS à chaque itération). Erreur de reconstruction en fonction de la complexité algorithmique.

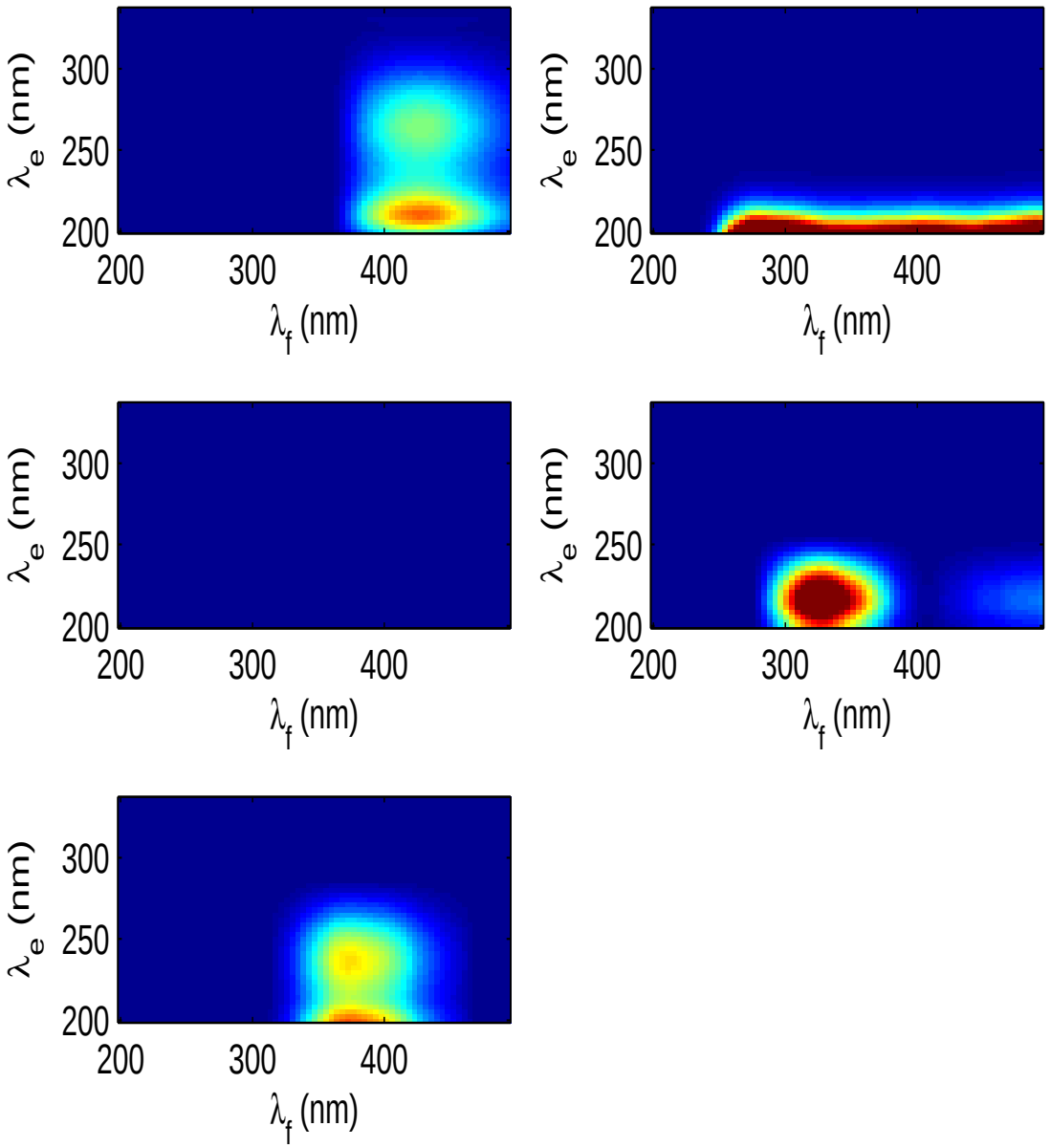


FIGURE 3.9 – Effet d'une surestimation du rang du tenseur : 5 composés estimés pour 4 réellement présents. Utilisation de l'algorithme du gradient conjugué avec non négativité imposée par produits de Hadamard.

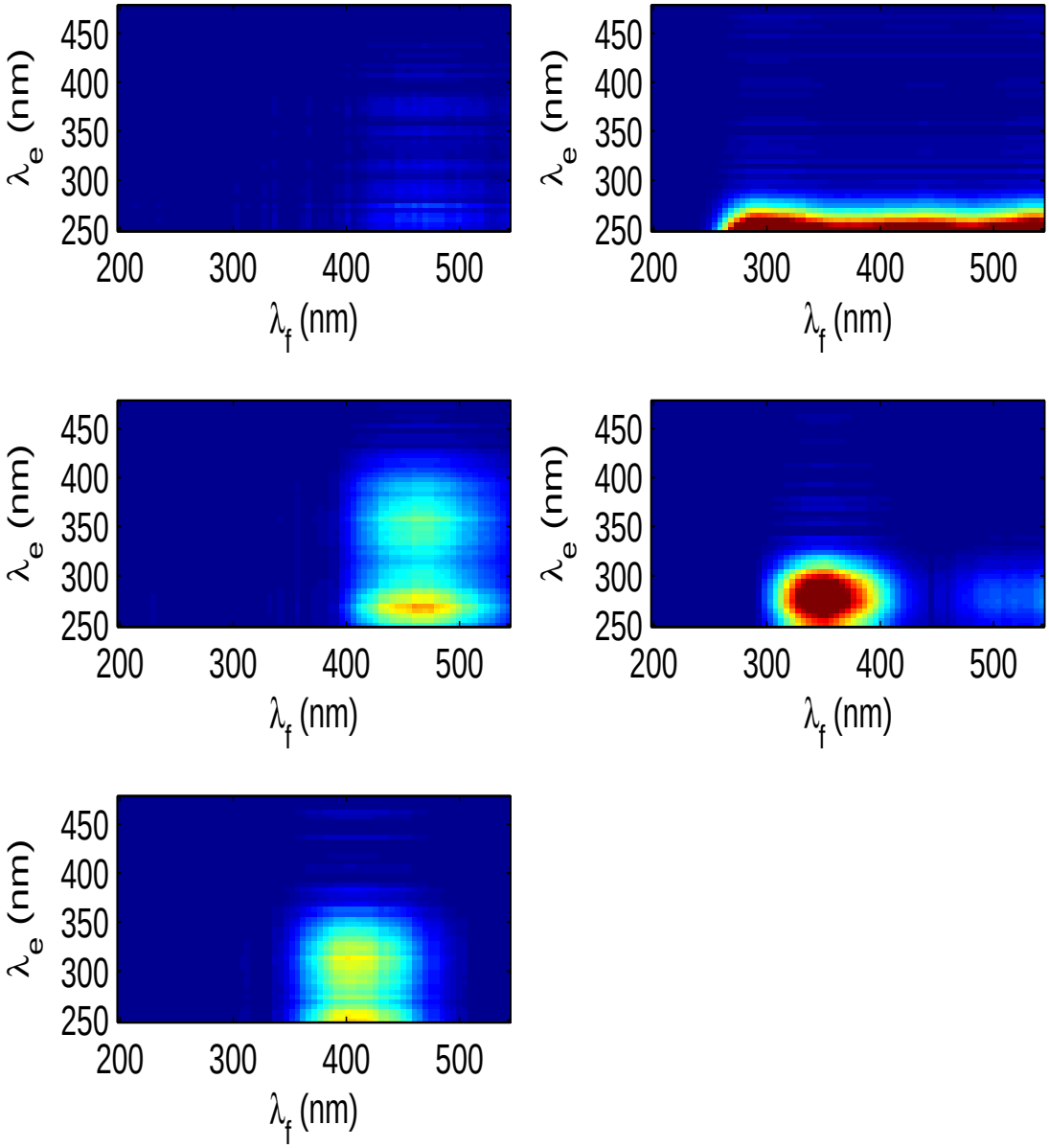


FIGURE 3.10 – Effet d'une surestimation du rang du tenseur : 5 composés estimés pour 4 réellement présents. Utilisation de l'algorithme BFGS avec non négativité imposée par produits de Hadamard.

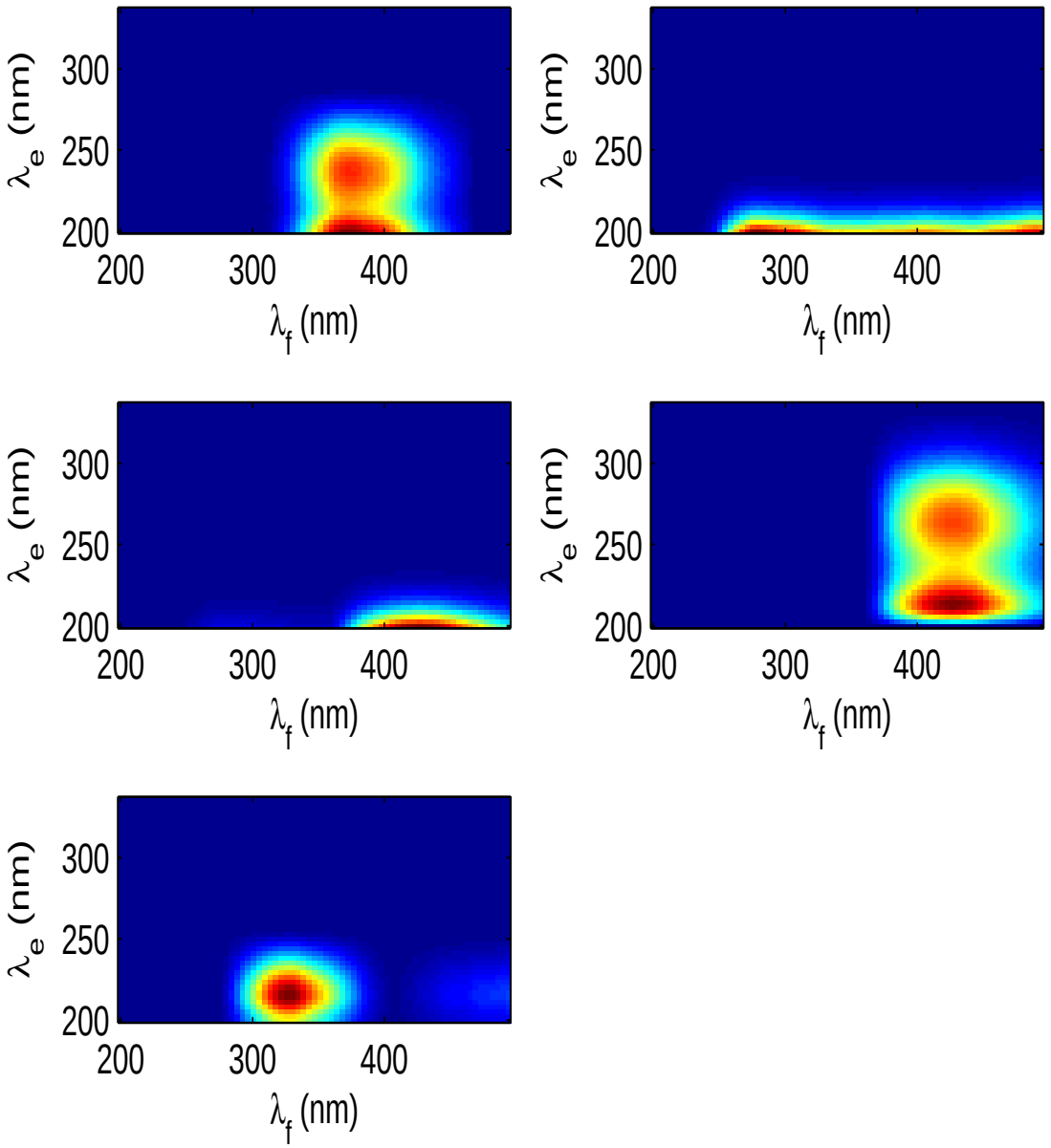


FIGURE 3.11 – Effet d'une surestimation du tenseur : 5 composés estimés pour 4 composés réellement présents. Utilisation de l'algorithme ALS décrit dans [CZPA09].

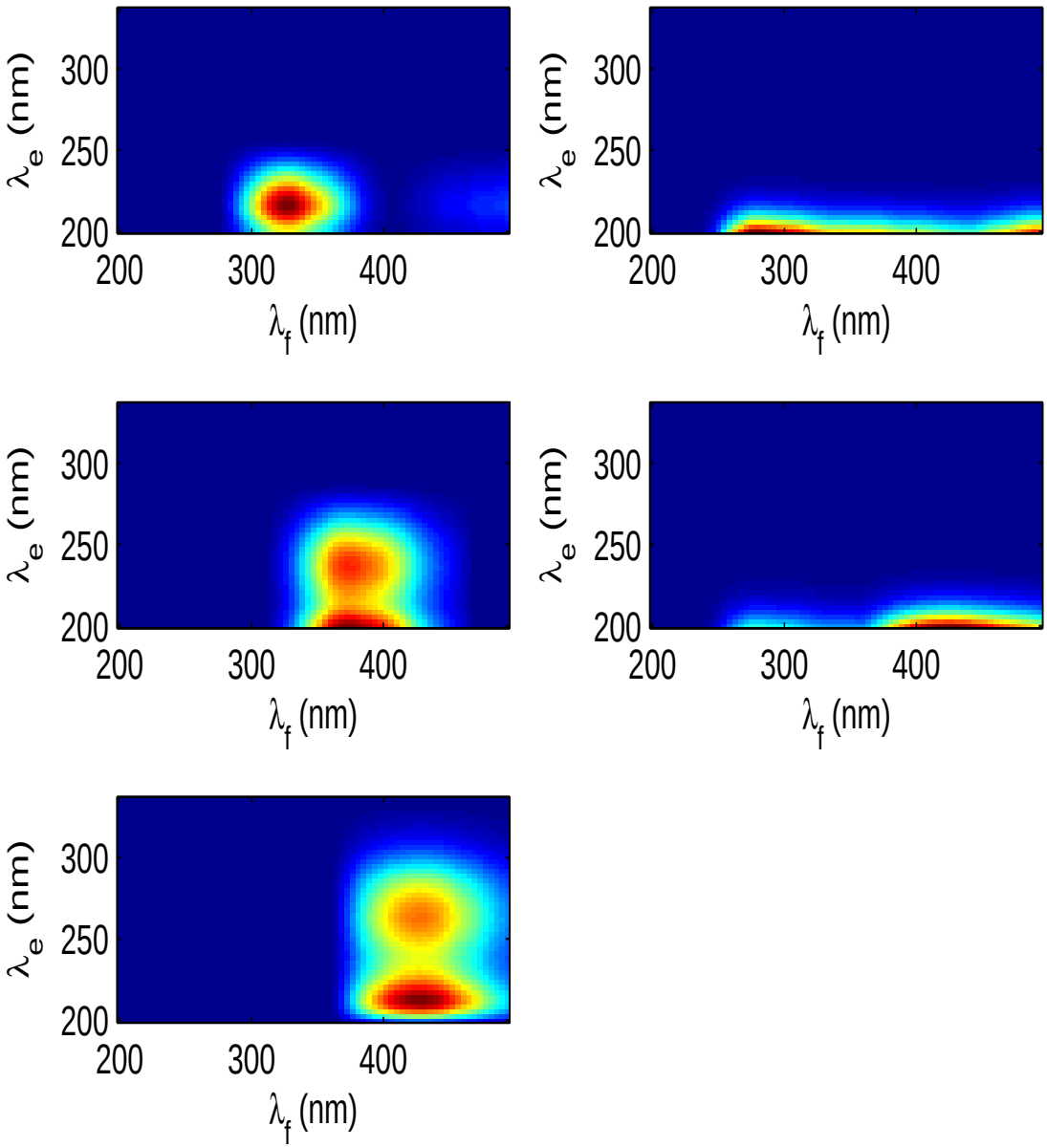


FIGURE 3.12 – Effet d'une surestimation du rang du tenseur : 5 composés estimés pour 4 réellement présents. Utilisation de l'algorithme HALS décrit en [CZPA09]

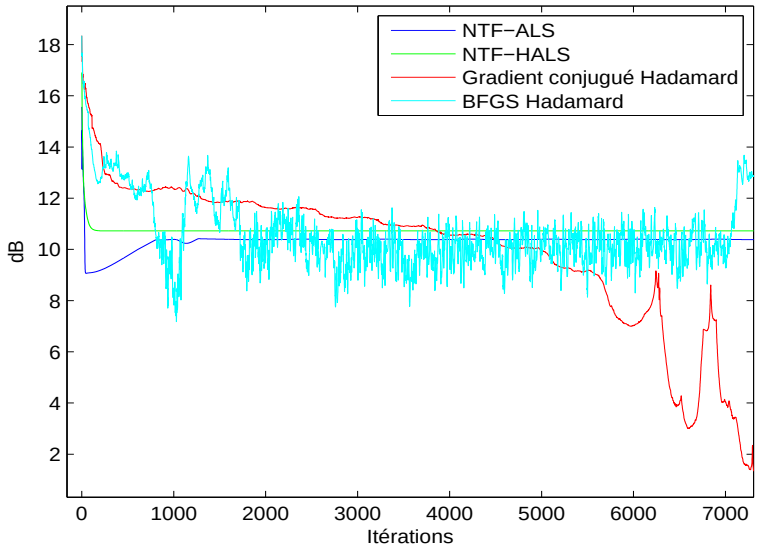


FIGURE 3.13 – Performances de la reconstruction des MEEF dans le cas surestimé pour différents algorithmes.

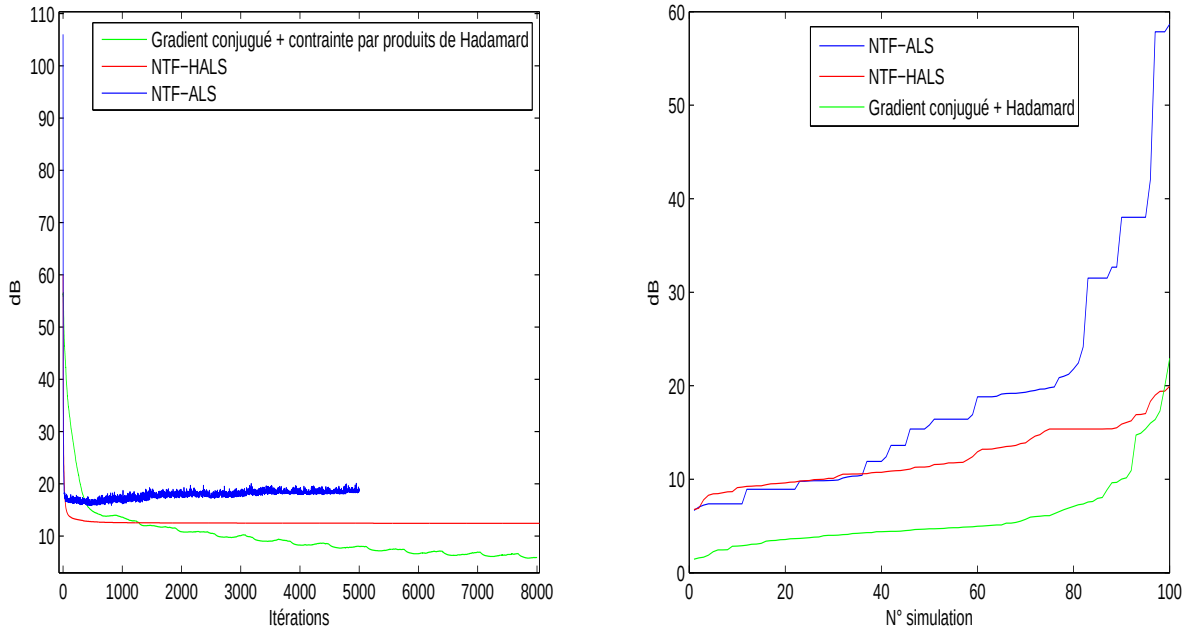


FIGURE 3.14 – Illustration du bon comportement du gradient conjugué paramétré par produits de Hadamard pour 100 initialisations différentes dans le cas surestimé. A gauche, on trace le moyennage point à point de $E_{2\text{dB}}$ au fil des itérations des 100 réalisations. A droite, on trace la courbe triée par ordre croissant représentant la dernière valeur (atteinte à la dernière itération) de $E_{2\text{dB}}$ pour les 100 réalisations (ce qui revient à tracer une fonction de répartition de la dernière valeur).

Deuxième série de simulations - Nous avons présenté ci-dessus les résultats de simulations en utilisant les approches basées sur des produits de Hadamard. Nous allons maintenant confronter la méthode faisant appel à une pénalisation exponentielle et la confronter à nos précédents résultats.

On génère un nouveau tenseur avec les mêmes MEEF de référence et paramètres aléatoires que précédemment, de sorte à ce que la matrice des concentrations soit de taille 50×4. Le tenseur ainsi créé est de taille $47 \times 71 \times 50$. Nous mettons en jeu le gradient conjugué pénalisé par des exponentielles (et non plus par une paramétrisation au moyen de produits de Hadamard), l'ALS pénalisé de la même façon, et le NTF-ALS. On compare sur la figure 3.16 le gradient conjugué présenté ci-dessus et l'ALS NTF-ALS avec $\alpha = \beta = \gamma = 10^{-6}$. Concernant notre algorithme, γ_A, γ_B et γ_C ont été choisis pour "normaliser" *i.e.* $\gamma_A = \frac{1}{\|\mathbf{A}\|}$, $\gamma_B = \frac{1}{\|\mathbf{B}\|}$, $\gamma_C = \frac{1}{\|\mathbf{C}\|}$. De plus, α, β, γ sont décrémentées à chaque itération : $\alpha(k+1) = \beta(k+1) = \gamma(k+1) = \frac{\alpha(k)}{m}$, avec m une constante et $\alpha(0) = 0.5$), de telle sorte que l'influence des termes de pénalisation devienne de plus en plus faible.

On peut observer que les 3 algorithmes sont assez rapides sur la figure 3.16(convergence atteinte en moins de 100 itérations). L'erreur de reconstruction est plus petite avec nos algorithmes qu'avec le NTF-HALS, même si les performances de ce dernier sont bornées du fait de la pénalisation. Un tel problème est éliminé dans notre cas en diminuant l'influence des termes de pénalisation au fur et à mesure des itérations. (Cependant, nous avons déjà noté qu'une faible erreur de reconstruction n'impliquait pas nécessairement une bonne estimation des matrices de facteurs). Les MEEF estimées par le gradient conjugué sont tracées sur la figure 3.15 et sont conformes à celles de référence.

Supposons maintenant que nous surestimions le rang en choisissant $F = 5$ pour ce mélange constitué de 4 composés. Les résultats obtenus sont affichés au niveau la figure 3.17. A gauche, les images ont été obtenues grâce à notre algorithme, tandis qu'à droite, elles ont été reconstruites grâce au NTF-ALS. Dans les deux cas, il est difficile d'affirmer qu'il y a 4 composés. Les artefacts restent néanmoins moins importants là encore avec notre approche qu'avec l'approche ALS. Cette méthode se révèle donc rapide, mais moins robuste que celle impliquant de paramétrer par des produits de Hadamard.

Ces algorithmes ainsi pénalisés restent assez sensibles à l'intialisation. On repart du même mélange, mais en initialisant les matrices de facteurs d'autres valeurs aléatoires. On retrace sur la figure 3.18 les performances de l'erreur de reconstruction. Là encore, les 3 algorithmes convergent très rapidement. Le gradient conjugué affiche les meilleures performances pour cet indice, tandis que l'ALS pénalisé donne les moins bonnes. Sur la figure 3.19, on trace les MEEF estimées par les 3 algorithmes mis en jeu ici. La reconstruction n'est pas très bonne avec le NTFALS. Elle est légèrement meilleure avec l'ALS et le gradient conjugué tous deux pénalisés par une exponentielle. Cela n'est toutefois pas aussi bon que dans le cas le cas d'une paramétrisation par produits de Hadamard.

Pour résumer les résultats obtenus à partir de cette pénalisation exponentielle, nous pouvons dire que c'est une méthode peu coûteuse numériquement, qui donne de bons résultats à modèle exact. Notons toutefois que cela n'impose pas la non négativité des matrices au même sens que la paramétrisation par produits de Hadamard. Cela aiguille la solution vers des matrices non négatives, mais il n'y a pas de garantie absolue de ne pas trouver quelques valeurs négatives. Par ailleurs, le choix des paramètres (α, γ_A, γ_B, γ_C) influence considérablement la solution

obtenue, et il n'est pas évident de les choisir judicieusement.

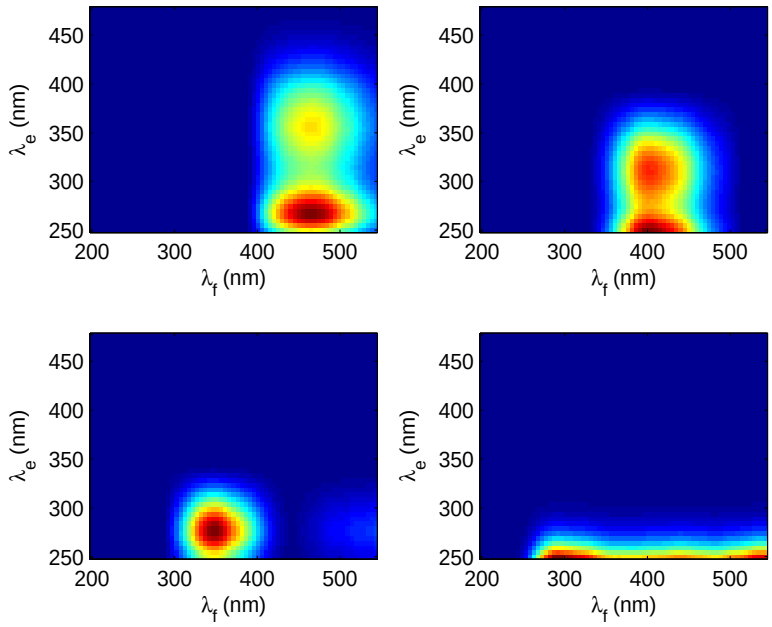


FIGURE 3.15 – MEEF estimées par gradient conjugué pénalisé par des exponentielles pour un tenseur non négatif de taille $71 \times 47 \times 50$.

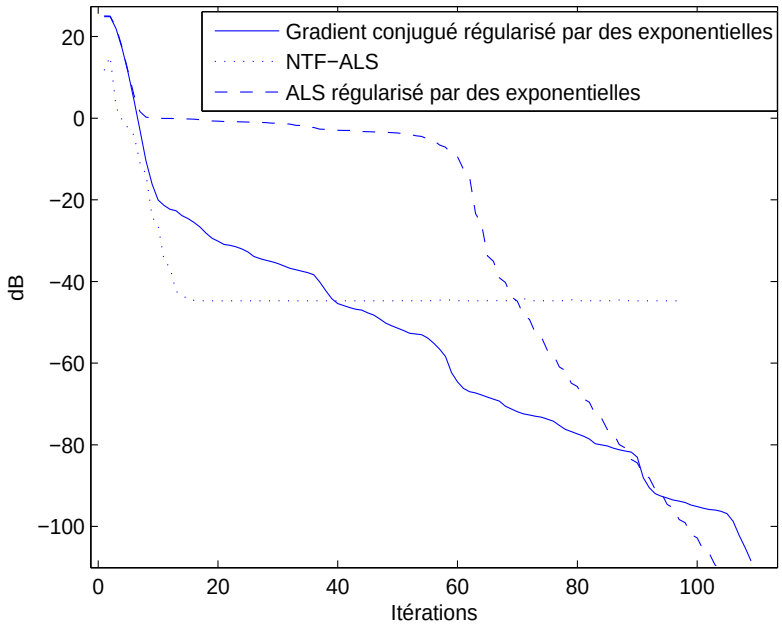


FIGURE 3.16 – Erreur de reconstruction (dB) en fonction du nombre d'itérations en utilisant un tenseur non négatif de taille $71 \times 47 \times 50$.

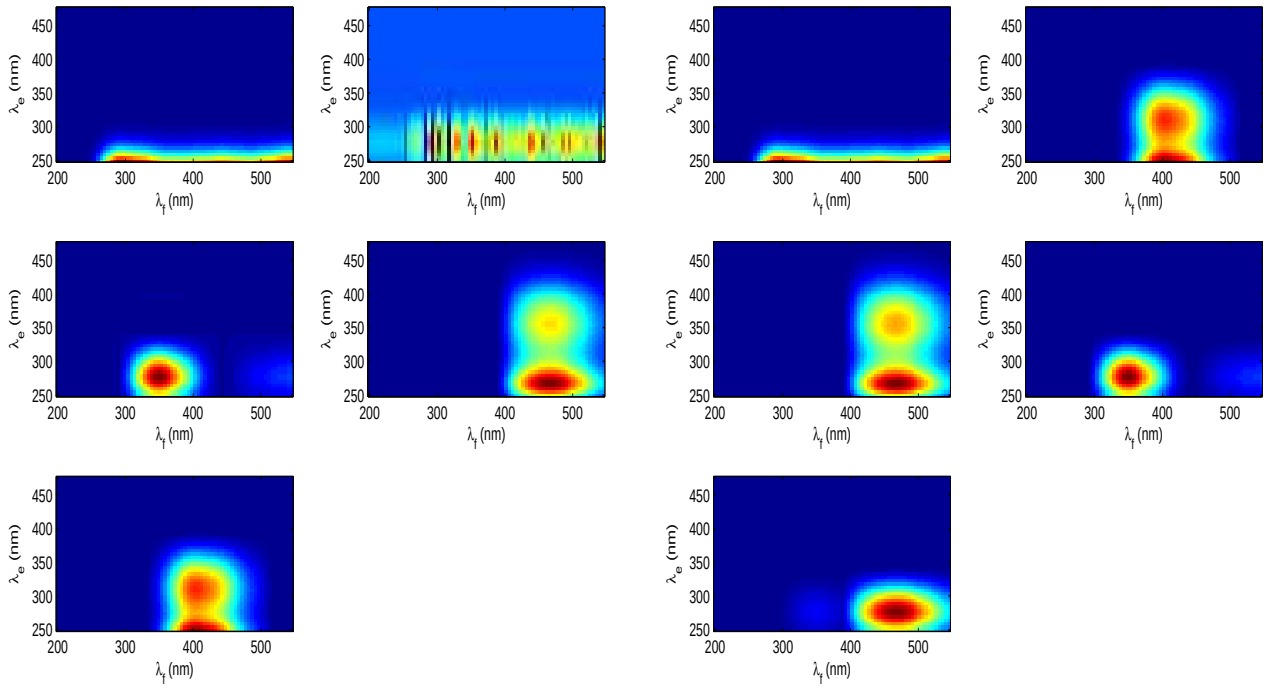


FIGURE 3.17 – Mélange de 4 facteurs, en supposant que $F = 5$; les 5 MEEF sont estimées en utilisant le gradient conjugué et une fonction de coût avec pénalisation exponentielle afin d'assurer la non-négativité (gauche) et NTF-ALS de [CZPA09].

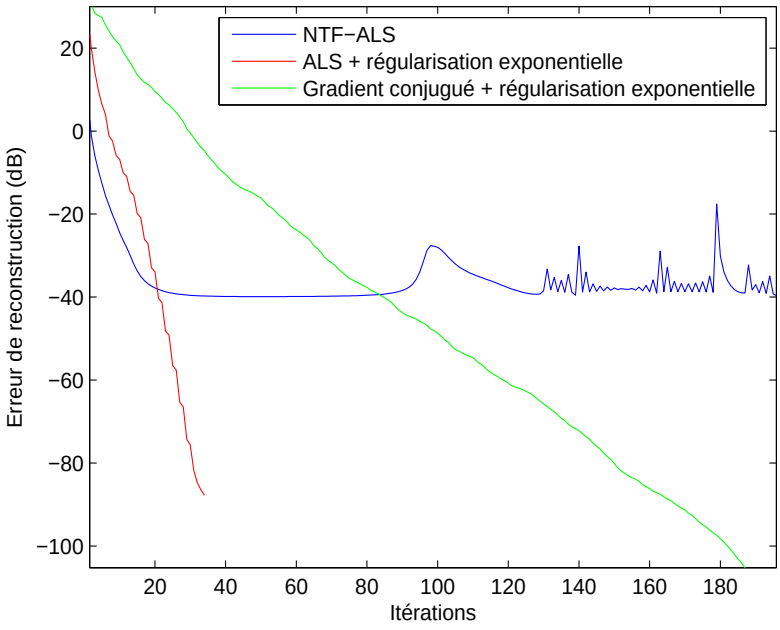


FIGURE 3.18 – Erreur de reconstruction (dB) en fonction du nombre d'itérations en utilisant un tenseur non négatif de taille $71 \times 47 \times 50$.

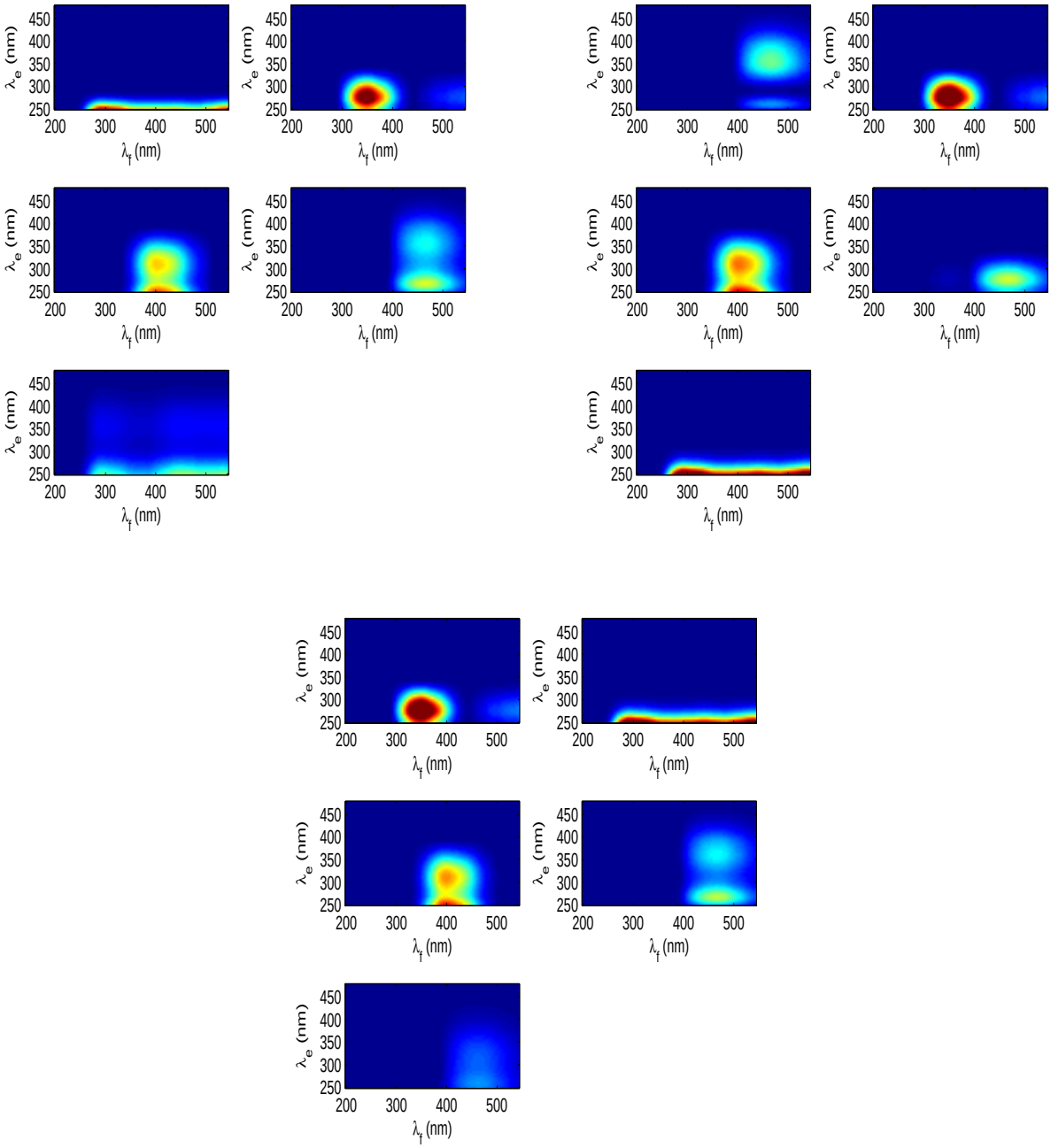


FIGURE 3.19 – Mélange de 4 facteurs, en supposant que $F = 5$; les 5 MEEF sont estimées en utilisant le gradient conjugué et une fonction de coût avec pénalisation exponentielle afin d'assurer la non-négativité (haut-gauche) et NTF-ALS de [CZPA09] (haut-droite), et de l'ALS pénalisé par des exponentielles (bas).

Conclusion - En conclusion de cette étude sur des mélanges synthétiques, nous avons mis en évidence le bon comportement des algorithmes présentés dans cette section. Gradient, gradient

conjugué et algorithmes de Quasi-Newton, dont la positivité des matrices de facteurs est assurée par la contrainte faisant intervenir le produit de Hadamard, donnent des résultats très comparables. Le gradient n'est pas recommandé du fait de sa lenteur de convergence. Il vaut mieux lui préférer le gradient conjugué, qui offre un très compromis performances / vitesse. Pour les problèmes de petite taille et à modèle exact, le BFGS et le DFP sont probablement le meilleur choix. Mais dès que la taille du tenseur considéré devient trop importante, l'estimation de la matrice Hessienne se révèle trop coûteuse. Similairement, l'ELS peut être employée pour ces algorithmes à chaque itération si la taille des donnée est suffisamment faible. C'est ce qui permet d'avoir les meilleures performances. Le coût algorithmique d'une étape d'ELS étant relativement élevé, il vaut mieux lui préférer une alternance avec du backtracking quand on s'oriente vers des problèmes de taille importante. Dans tous les cas, ces algorithmes présentent l'avantage de ne pas avoir de paramètres à régler de façon empirique.
A modèle exact, les algorithmes basés sur l'ALS donnent par ailleurs de très bons résultats ici en plus de s'avérer particullièrement rapides. Ils sont une bonne alternative à la paramétrisation par Hadamard lorsque le temps est un facteur décisif.

Lorsque l'on introduit des erreurs dans le modèle de type surestimation du nombre de composés, nos algorithmes montrent un comportement très appréciable. Notons qu'il peut alors être utile de pénaliser les méthodes. Une régularisation L_1, visant à aiguiller vers des solutions parcimonieuses, permet ici d'estimer comme nuls les composés correspondant à des erreurs de modèle. De plus, le gradient conjugué s'avère ici être le choix le plus adapté : bien que la fonction de coût que l'on optimise reste la même, les algorithmes de Quasi-Newton se révèlent ici moins stables avec l'introduction d'erreurs et pénalisation. Nous avions déjà vu que le conditionnement de la matrice Hessienne se révélait être de moins en moins bon au fil des itérations. Pour finir, bien que la pénalisation exponentielle s'avère très efficace pour faire converger rapidement les algorithmes en favorisant la positivité des matrices de facteurs, il faut noter que l'estimation des composés est moins bonne dans le cas surestimé. Dans ce cas là, la solution finale dépend ici beaucoup de l'initialisation pour les algorithmes basés sur l'ALS ou pénalisés par une exponentielle. C'est moins vrai pour les algorithmes paramétrés par produits de Hadamard, qui, dans le cas surestimé, convergent toujours vers les 4 bonnes MEEF avec la $5^{\text{ème}}$ d'intensité pratiquement nulle. Qui plus est, les algorithmes basés sur une paramétrisation par produits de Hadamard présentent l'avantage de ne pas avoir de paramètres à régler manuellement. L'ELS, quand on utilise un algorithme d'optimisation basé sur du gradient, est calculée automatiquement et idem pour le backtracking.

3.8 Vers une première généralisation : nouvel algorithme de Tucker3 non négatif

Dans cette dernière partie, nous allons reprendre la même démarche que celle qui avait été adoptée lors de la recherche d'une décomposition CP non négative, en utilisant une paramétrisation des matrices de facteurs recherchées au moyen de carrés, ce qui impliquait donc l'utilisation de produits de Hadamard. Nous allons maintenant chercher à la généraliser à un autre type de décomposition tensorielle : la décomposition de Tucker 3 non négative. Com-

mençons d'abord par quelques rappels sur la décomposition de Tucker et ses liens avec la décomposition CP.

3.8.1 Quelques rappels préliminaires

Considérons la décomposition en valeurs singulières d'une matrice $\mathbf{X}$ de taille $I \times J$:

$$\mathbf{X} = \mathbf{AGB}^T \tag{3.110}$$

Ici, $\mathbf{A}$ est de taille $I \times I$, $\mathbf{B}$ est de taille $J \times J$, et $\mathbf{G}$ est une matrice rectangulaire pseudo-diagonale de taille $I \times J$.
Considérons la décomposition en valeurs singulières réduite de cette matrice $\mathbf{X}$. En supposant que le rang de $\mathbf{X}$ est F, on se limite alors aux F valeurs singulières constituant $\mathbf{G}$, qui devient alors carrée ($F \times F$). De même, on ne considère que les F premiers vecteurs singuliers de $\mathbf{A}$ qui devient de taille $I \times F$, et même chose pour la matrice $\mathbf{B}$ qui devient alors $J \times F$. En décomposant élément par élément, cela se traduit par :

$$x_{ij} = \sum_{f=1}^{F} a_{if} g_{ff} b_{jf} \tag{3.111}$$

Réécrit sous une forme matricielle, cela conduit à la forme suivante :

$$\mathbf{X} = \mathbf{A}_F \mathbf{G}_F \mathbf{B}_F^T \tag{3.112}$$

En posant $\mathbf{W} = \mathbf{A}_F \mathbf{G}_F$ et $\mathbf{H} = \mathbf{B}_F$, on a :

$$\mathbf{X} = \mathbf{WH}^T \tag{3.113}$$

Notons que $\mathbf{W}$ et $\mathbf{H}$ sont communément appelées respectivement scores et loadings en analyse en composantes principales (l'équation (3.113) peut alors se généraliser pour le cas en trois dimensions vers la décomposition CP).
Toujours en dimension deux, on peut généraliser la décomposition donnée en (3.112) pour une matrice $\mathbf{G}$ non diagonale de taille quelconque. $\mathbf{A}$ et $\mathbf{B}$ n'auront plus alors le même nombre de colonnes.

$$x_{ij} = \sum_{m=1}^{M} \sum_{n=1}^{N} a_{im} g_{mn} b_{jn} \tag{3.114}$$

A. Tucker a proposé de nouveaux modèles en dimension N au milieu des années 1960 [Tuc66]. La décomposition de Tucker-3 peut être vue comme une généralisation à un ordre supérieur de l'analyse en composantes principales, ainsi qu'une généralisation de la décomposition CP vue jusque là.

Considérant un tenseur du troisième ordre, de taille $I \times J \times K$, celui-ci peut s'écrire :

$$x_{ijk} = \sum_{f_1} \sum_{f_2} \sum_{f_3} g_{f_1 f_2 f_3} a_{i f_1} b_{j f_2} c_{k f_3}. \tag{3.115}$$

Ici, $\mathbf{A}$ est une matrice de taille $I \times F_1$, $\mathbf{G}$ est un tenseur cœur de taille $F_1 \times F_2 \times F_3$, $\mathbf{B}$ est une matrice de taille $J \times F_2$, et $\mathbf{C}$ est une matrice de taille $K \times F_3$.
On peut également réécrire ce modèle en utilisant le produit $n-mode$:

$$\mathbf{X} = \mathbf{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} \tag{3.116}$$

On peut aussi le mettre sous forme matricielle, ce qui donne pour le premier mode :

$$\mathbf{X}_{(1)}^{I,KJ} = \mathbf{A}\mathbf{G}_{(1)}^{F_1,F_3F_2} \left(\mathbf{C} \otimes \mathbf{B}\right)^T. \tag{3.117}$$

On fait alors intervenir le produit de Kronecker, et on déplie le tenseur cœur $\mathbf{G}$ dans le premier mode. Il est bien sûr possible de faire la même chose pour les deux autres modes :

$$\mathbf{X}_{(2)}^{J,KI} = \mathbf{B}\mathbf{G}_{(2)}^{F_2,F_3F_1} \left(\mathbf{C} \otimes \mathbf{A}\right)^T \tag{3.118}$$

$$\mathbf{X}_{(3)}^{K,JI} = \mathbf{C}\mathbf{G}_{(3)}^{F_3,F_2F_1} \left(\mathbf{B} \otimes \mathbf{A}\right)^T. \tag{3.119}$$

Si on définit le tenseur cœur $\mathbf{G}$ de taille $F \times F \times F$ (càd que l'on suppose que $F_1 = F_2 = F_3 = F$) comme étant diagonal, soit $g_{ijk} = 1$ si $i = j = k$ et 0 sinon, on obtient alors la décomposition CP. En effet, si on déplie $\mathbf{G}$ dans le premier mode, on a une matrice de taille $F \times F^2$ telle que :

$$\mathbf{S} = \begin{array}{c} \begin{array}{ccccccccc} 1 & \ldots & F & F+1 & F+2 & \ldots & & & F^2 \end{array} \\ F \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 & & & \vdots \\ 0 & \ldots & & 0 & 1 & 0 & \ldots & 0 \\ \vdots & & & 0 & 0 & & & 1 \end{pmatrix} \end{array} \tag{3.120}$$

Il existe un lien entre une telle matrice, le produit de Kronecker et le produits de Khatri-Rao [LT08] :

$$\mathbf{S}\left(\mathbf{C} \otimes \mathbf{B}\right)^T = \left(\mathbf{C} \odot \mathbf{B}\right)^T \tag{3.121}$$

Ainsi, si $\mathbf{G}$ est diagonal, alors il vient :

$$\mathbf{A}\mathbf{G}_{(1)}^{F_1,F_3F_2} \left(\mathbf{C} \otimes \mathbf{B}\right)^T = \mathbf{A}\left(\mathbf{C} \odot \mathbf{B}\right)^T. \tag{3.122}$$

Comme précédemment, on peut imposer une contrainte de non-négativité sur les facteurs et sur le tenseur cœur à la décomposition régie par l'équation (3.115). Des méthodes prenant en compte cette contrainte existent déjà [KS98][MHA08][CZPA09][PTC][PC11] [ZAX12][ZC12]. Dans [MHA08], Mørup & *al* ont proposé une méthode basée sur une généralisation des NMF à l'ordre supérieur avec prise en compte de la parcimonie des données appliquée à la décomposition de Tucker-3. Dans [BA], il est proposé un algorithme avec semi-contrainte de non-négativité : les facteurs sont contraints, mais le tenseur cœur peut prendre des valeurs négatives.
La décomposition de Tucker trouve par exemple des applications dans l'imagerie hyperspectrale, la réduction de dimension, la réduction de bruit ou la déconvolution d'images [Ren08, KC09, Kop09]. Pour un panorama des applications possibles, on pourra aussi consulter [AY09].

3.8.2 Nouvel algorithme de décomposition de Tucker 3 non négatif

Nous proposons dans ce paragraphe d'expliquer comment il est possible d'écrire un algorithme fondé sur une paramétrisation par produits de Hadamard pour imposer les contraintes de non-négativité aux facteurs et au tenseur cœur recherchés. La minimisation d'un problème de Tucker-3 non contraint est basée sur la fonction de coût suivante :

$$\mathcal{T}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}) = \|\mathbf{T}^{I,KJ}_{(1)} - \mathbf{A}\mathbf{G}^{F_1,F_3F_2}_{(1)}(\mathbf{C}\otimes\mathbf{B})^T\|_F^2 = \|\boldsymbol{\delta}'_{(1)}\|_F^2 \tag{3.123}$$
$$= \|\mathbf{T}^{J,KI}_{(2)} - \mathbf{B}\mathbf{G}^{F_2,F_3F_1}_{(2)}(\mathbf{C}\otimes\mathbf{A})^T\|_F^2 = \|\boldsymbol{\delta}'_{(2)}\|_F^2 \tag{3.124}$$
$$= \|\mathbf{T}^{K,JI}_{(3)} - \mathbf{C}\mathbf{G}^{F_3,F_2F_1}_{(3)}(\mathbf{B}\otimes\mathbf{A})^T\|_F^2 = \|\boldsymbol{\delta}'_{(3)}\|_F^2, \tag{3.125}$$

En posant $\mathbf{A}' = \mathbf{A}\boxdot\mathbf{A}$, $\mathbf{B}' = \mathbf{B}\boxdot\mathbf{B}$, $\mathbf{C}' = \mathbf{C}\boxdot\mathbf{C}$, $\mathbf{G}' = \mathbf{G}\boxdot\mathbf{G}$, on peut réécrire le problème de la façon suivante :

$$\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}) = \|\mathbf{T}^{I,KJ}_{(1)} - (\mathbf{A}\boxdot\mathbf{A})\left(\mathbf{G}^{F_1,F_3F_2}_{(1)}\boxdot\mathbf{G}^{F_1,F_3F_2}_{(1)}\right)((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B}))^T\|_F^2$$
$$= \|\boldsymbol{\delta}_{(1)}\|_F^2 \tag{3.126}$$
$$= \|\mathbf{T}^{J,KI}_{(2)} - (\mathbf{B}\boxdot\mathbf{B})\left(\mathbf{G}^{F_2,F_3F_1}_{(2)}\boxdot\mathbf{G}^{F_2,F_3F_1}_{(2)}\right)((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A}))^T\|_F^2$$
$$= \|\boldsymbol{\delta}_{(2)}\|_F^2 \tag{3.127}$$
$$= \|\mathbf{T}^{K,JI}_{(3)} - (\mathbf{C}\boxdot\mathbf{C})\left(\mathbf{G}^{F_3,F_2F_1}_{(3)}\boxdot\mathbf{G}^{F_3,F_2F_1}_{(3)}\right)((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A}))^T\|_F^2$$
$$= \|\boldsymbol{\delta}_{(3)}\|_F^2, \tag{3.128}$$

Après des calculs dont le détail est donné au niveau de l'annexe 6.6.1, on obtient finalement les matrices de gradient $\nabla_{\mathbf{A}}(\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}))$, $\nabla_{\mathbf{B}}(\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}))$, $\nabla_{\mathbf{C}}(\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}))$, et $\nabla_{\mathbf{G}}(\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}))$ suivantes :

$$\nabla_{\mathbf{A}}\left(\mathcal{T}_c\left(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}\right)\right) = \frac{\partial \mathcal{T}_c}{\partial \mathbf{A}} = -4\mathbf{A} \boxdot \left[\boldsymbol{\delta}_{(1)}\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)^T\right] \tag{3.129}$$

$$\nabla_{\mathbf{B}}\left(\mathcal{T}_c\left(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}\right)\right) = \frac{\partial \mathcal{T}_c}{\partial \mathbf{B}} = -4\mathbf{B} \boxdot \left[\boldsymbol{\delta}_{(2)}\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)^T\right] \tag{3.130}$$

$$\nabla_{\mathbf{C}}\left(\mathcal{T}_c\left(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}\right)\right) = \frac{\partial \mathcal{T}_c}{\partial \mathbf{C}} = -4\mathbf{C} \boxdot \left[\boldsymbol{\delta}_{(3)}\left((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)^T\right] \tag{3.131}$$

$$\nabla_{\mathbf{G}}\left(\mathcal{T}_c\left(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{G}\right)\right) = \frac{\partial \mathcal{T}_c}{\partial \mathbf{G}_{(1)}^{F_1,F_3F_2}} = -4\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \left[\left(\mathbf{A}\boxdot\mathbf{A}\right)^T\boldsymbol{\delta}_{(1)}\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)\right] \tag{3.132}$$

Et comme précédemment nous pourrons utiliser ces expressions afin de construire un vecteur de gradient de taille $(IF_1 + JF_2 + KF_3 + F_1F_2F_3) \times 1$ et un vecteur $\mathbf{x}^{(k)}$:

$$\mathbf{g}^{(k)} = \begin{pmatrix} \mathsf{vec}\{\nabla_{\mathbf{A}}\mathcal{T}_c(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)};\mathbf{G}^{(k)})\} \\ \mathsf{vec}\{\nabla_{\mathbf{B}}\mathcal{T}_c(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)};\mathbf{G}^{(k)})\} \\ \mathsf{vec}\{\nabla_{\mathbf{C}}\mathcal{T}_c(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)};\mathbf{G}^{(k)})\} \\ \mathsf{vec}\{\nabla_{\mathbf{G}}\mathcal{T}_c(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)};\mathbf{G}^{(k)})\} \end{pmatrix}, \quad \mathbf{x}^{(k)} = \begin{pmatrix} \mathsf{vec}\{\mathbf{A}^{(k)}\} \\ \mathsf{vec}\{\mathbf{B}^{(k)}\} \\ \mathsf{vec}\{\mathbf{C}^{(k)}\} \\ \mathsf{vec}\{\mathbf{G}^{(k)}\} \end{pmatrix} \tag{3.133}$$

qui nous servira ensuite pour construire un algorithme d'optimisation de type descente comme cela a déjà été expliqué au niveau du paragraphe 3.4.

On peut également calculer les coefficients du polynôme utilisé dans le pas optimal (détails du calcul dans l'annexe 6.6.2. Nous procédons de manière identique à ce qui avait été fait au niveau du paragraphe 3.6.1.

Le calcul du pas optimal consiste à minimiser la fonction suivante :

$$\begin{aligned} &\mathcal{T}_c(\mathbf{A}^{(k+1)},\mathbf{B}^{(k+1)},\mathbf{C}^{(k+1)};\mathbf{G}^{(k+1)}) \\ &= \mathcal{T}_c\left[(\mathbf{A}^{(k)}+\mu\mathbf{D_A}^{(k)}),(\mathbf{B}^{(k)}+\mu\mathbf{D_B}^{(k)}),(\mathbf{C}^{(k)}+\mu\mathbf{D_C}^{(k)});(\mathbf{G}^{(k)}+\mu\mathbf{D_G}^{(k)})\right]. \end{aligned} \tag{3.134}$$

Le développement de ce polynôme d'ordre 16 (dans ce cas là) nous amène à devoir calculer la valeur de ses 17 coefficients (cf. Annexe 6.6.2, pour le détail des calculs) :

$$\mathsf{d}\mathcal{T}_c(.) = \sum_{i=0}^{16} a_i\mu^i, \tag{3.135}$$

avec :

$$
\begin{aligned}
a_{16} &= \mathsf{trace}\left(\mathbf{K_8}\mathbf{K_8}^T\right)\\
a_{15} &= \mathsf{trace}\left(2\mathbf{K_7}\mathbf{K_8}^T\right)\\
a_{14} &= \mathsf{trace}\left(2\mathbf{K_6}\mathbf{K_8}^T + \mathbf{K_7}\mathbf{K_7}^T\right)\\
a_{13} &= \mathsf{trace}\left(2\left(\mathbf{K_5}\mathbf{K_8}^T + \mathbf{K_6}\mathbf{K_7}^T\right)\right)\\
a_{12} &= \mathsf{trace}\left(2\left(\mathbf{K_4}\mathbf{K_8}^T + \mathbf{K_5}\mathbf{K_7}^T\right) + \mathbf{K_6}\mathbf{K_6}^T\right)\\
a_{11} &= \mathsf{trace}\left(2\left(\mathbf{K_3}\mathbf{K_8}^T + \mathbf{K_4}\mathbf{K_7}^T + \mathbf{K_5}\mathbf{K_6}^T+\right)\right)\\
a_{10} &= \mathsf{trace}\left(2\left(\mathbf{K_2}\mathbf{K_8}^T + \mathbf{K_3}\mathbf{K_7}^T + \mathbf{K_4}\mathbf{K_6}^T\right) + \mathbf{K_5}\mathbf{K_5}^T\right)\\
a_{9} &= \mathsf{trace}\left(2\left(\mathbf{K_1}\mathbf{K_8}^T + \mathbf{K_2}\mathbf{K_7}^T + \mathbf{K_3}\mathbf{K_6}^T + \mathbf{K_4}\mathbf{K_5}^T\right)\right)\\
a_{8} &= \mathsf{trace}\left(2\left(\mathbf{K_0}\mathbf{K_8}^T + \mathbf{K_1}\mathbf{K_7}^T + \mathbf{K_2}\mathbf{K_6}^T + \mathbf{K_3}\mathbf{K_5}^T\right) + \mathbf{K_4}\mathbf{K_4}^T\right)\\
a_{7} &= \mathsf{trace}\left(2\left(\mathbf{K_0}\mathbf{K_7}^T + \mathbf{K_1}\mathbf{K_6}^T + \mathbf{K_2}\mathbf{K_5}^T + \mathbf{K_3}\mathbf{K_4}^T\right)\right)\\
a_{6} &= \mathsf{trace}\left(2\left(\mathbf{K_2}\mathbf{K_4}^T + \mathbf{K_1}\mathbf{K_5}^T + \mathbf{K_0}\mathbf{K_6}^T\right) + \mathbf{K_3}\mathbf{K_3}^T\right)\\
a_{5} &= \mathsf{trace}\left(2\left(\mathbf{K_1}\mathbf{K_4}^T + \mathbf{K_2}\mathbf{K_3}^T\right)\right)\\
a_{4} &= \mathsf{trace}\left(2\left(\mathbf{K_4}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_3}^T\right) + \mathbf{K_2}\mathbf{K_2}^T\right)\\
a_{3} &= \mathsf{trace}\left(2\left(\mathbf{K_3}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_2}^T\right)\right)\\
a_{2} &= \mathsf{trace}\left(2\mathbf{K_2}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_1}^T\right)\\
a_{1} &= \mathsf{trace}\left(2\mathbf{K_1}\mathbf{K_0}^T\right)\\
a_{0} &= \mathsf{trace}\left(\mathbf{K_0}\mathbf{K_0^T}\right)
\end{aligned}
$$

A nouveau, il nous faudra dériver ce polynôme et par rapport à μ :

$$\mathsf{d}\mathcal{T}_c(.) = \sum_{i=0}^{15}(i+1)a_{i+1}\mu^i, \tag{3.136}$$

Le pas optimal $\mu_{\mathsf{opt}}^{(k)}$ correspond alors à la racine réelle et positive de ce polynôme de degré 15 menant au minimum du critère donné en (3.135). Les racines sont estimées là encore numériquement.

3.8.3 Un exemple numérique

On illustre les expressions ci-dessus permettant d'estimer les matrices de facteurs et le cœur en les contraignant à être non négatifs en reprenant l'exemple donné dans la partie 3.7. On dispose donc d'un tenseur de taille $47 \times 71 \times 10$.
Ce modèle était fait pour être estimé par décomposition CP. Rappelons en effet qu'il est mathématiquement défini par la loi de Beer-Lambert (2.37) (pour des concentrations suffisamment faibles, dans le cas de mélanges réels). Nous avons vu précédemment que pour retrouver l'équivalence entre la décomposition de Tucker et CP, il était nécessaire que le tenseur cœur $\mathbf{G}$ soit diagonal. Afin d'aiguiller l'algorithme vers les MEEF de référence, et du

fait que la décomposition de Tucker ne garantit pas l'unicité de la solution estimée contrairement à la décomposition CP , nous initialisons le tenseur cœur comme diagonal, en ajoutant sur chaque valeur un biais aléatoire issu d'une loi uniforme, d'amplitude maximale 30. Les 3 autres matrices de facteurs sont initialisées aléatoirement. Nous choisissons d'utiliser le gradient conjugué, qui était le meilleur compromis performances / temps par itérations sur les exemples précédents. Nous alternons également du backtracking avec l'ELS pour accélérer le calcul, dont la complexité algorithmique par étape devient importante.
On trace sur la figure 3.20 les MEEF estimées via cet algorithme de Tucker non négatif. Elles ne diffèrent pas de la référence donnée en 3.4. De plus, le tenseur cœur réestimé se rapproche d'un tenseur diagonal, dont voici, pour l'exemple, les 4 valeurs g_{iii}, d'abord pour les valeurs initiales :

$$\mathbf{g}_{\text{diagInitial}} = \begin{pmatrix} 26.0949 & 14.8458 & 26.1475 & 18.3471 \end{pmatrix} \tag{3.137}$$

Puis les valeurs estimées :

$$\mathbf{g}_{\text{diagEstimée}} = \begin{pmatrix} 65.2053 & 43.1615 & 53.6602 & 47.5976 \end{pmatrix} \tag{3.138}$$

En effet, si on normalise chacun de ces 2 vecteurs par leur valeur maximale pour les recadrer sur une même échelle, on constate que l'écart type de $\mathbf{g}_{\text{diagEstimée}}$ est inférieur à celui de $\mathbf{g}_{\text{diagInitial}}$.

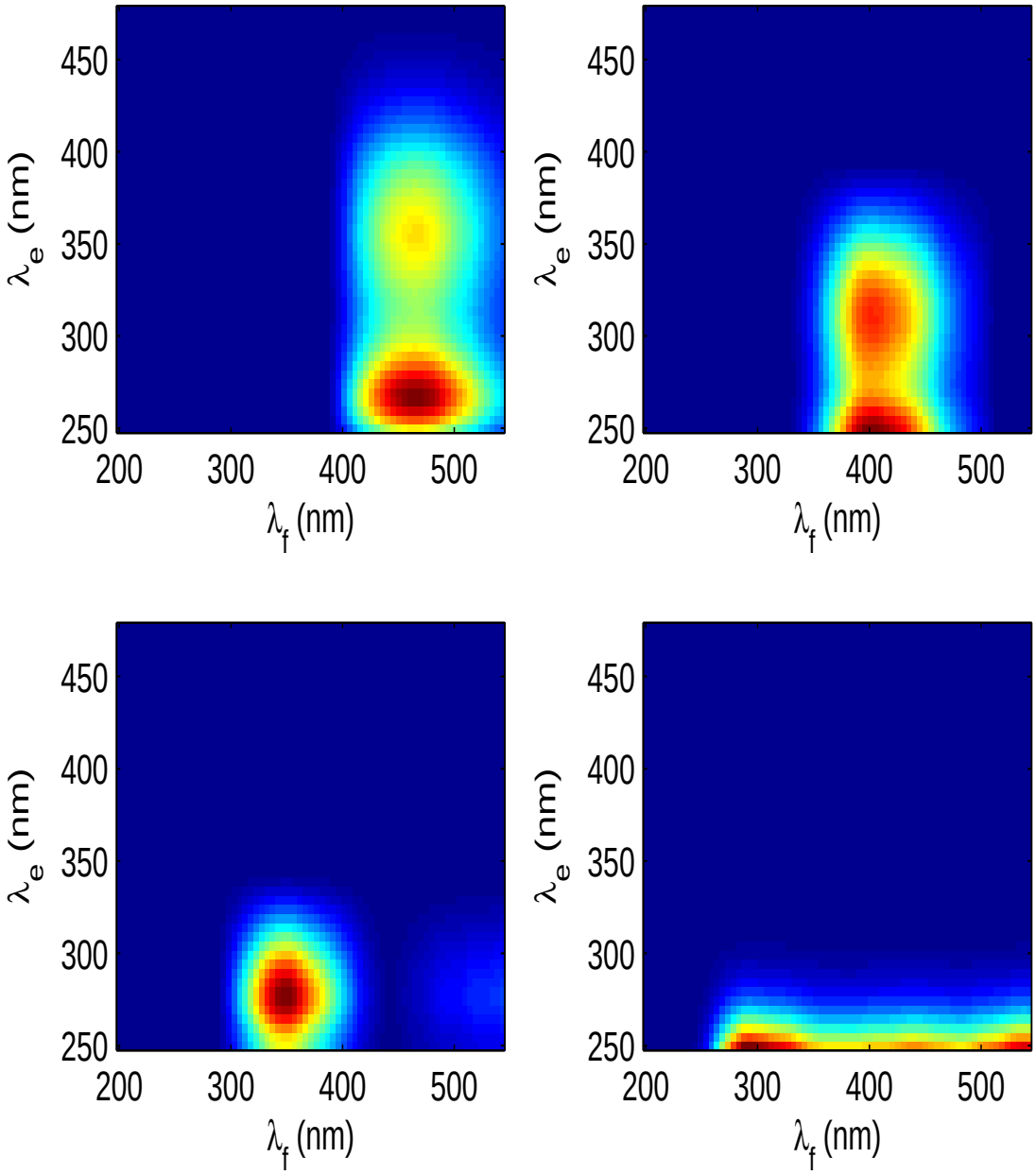


FIGURE 3.20 – MEEF estimées par gradient conjugué appliqué à la décomposition de Tucker non négative (produits de Hadamard).

Chapitre 4

Mesures manquantes : prise en compte du problème

4.1 Problématique

Au cours du processus d'acquisition de mesures, il peut s'avérer que certaines valeurs soient manquantes ou peu fiables. On peut prendre l'exemple du domaine bio-médical, dans lequel on utilise parfois des réseaux d'électrodes (cas de l'analyse EEG) disposées sur un patient. Or il peut advenir que l'une de ces électrodes soit défaillante et la mesure soit alors manquante. En EEG, les données sont regroupées dans un tenseur après utilisation de transformées temps-fréquence [BCA12, MnMVS+04]. Dans d'autres domaines, tels que l'étude du trafic réseau, on peut perdre certains paquets échangés entre deux terminaux. Plus généralement, ce problème de la gestion des données manquantes concerne tous les domaines dans lesquels la collecte des données peut être sujette à erreurs. Si dans certains cas il suffit tout simplement d'ignorer les valeurs manquantes, dans d'autres cas tels que le domaine de la spectroscopie de fluorescence, par exemple, il n'est pas souhaitable d'ignorer ces valeurs manquantes et le problème pourrait même ne pas pouvoir être traité sans prise en compte de ces données. C'est pourquoi il s'avère alors tout à fait pertinent de développer des méthodes destinées à traiter le problème étudié tout en les considérant.
Dans ce chapitre, nous couplons la question des données manquantes à celle des tenseurs non négatifs répondant à un modèle CP. La question de la non négativité et de la décomposition CP prenant en compte cet aspect a été traitée dans le chapitre précédent (Chapitre 3).

4.2 Approches existantes

Comme mentionné précédemment, les tenseurs résultant de mesures expérimentales peuvent contenir des données manquantes. Cependant, il peut être nécessaire d'être capable de réaliser la décomposition CP en ignorant ces valeurs. Les travaux précédents sur le sujet ont consisté à introduire un tenseur de poids, que nous appellerons $\mathbf{W}$, possédant la même taille que le tenseur de données ($\mathbf{W} \in \mathbb{R}^{+I \times J \times K}$) et dont les entrées valent 0 quand une valeur est manquante ou 1 quand elle est présente [AKDM11][TB05]. Notons dès à présent que dans la nouvelle approche qui nous expliciterons dans la sous-section 4.3, les valeurs de $\mathbf{W}$ ne seront

plus supposées nécessairement binaires. En notant $\mathbf{W}_{(1)}^{I,KJ}$ le dépliement du tenseur $\mathbf{W}$ dans le premier mode, la fonction de coût exprimée dans le premier mode vaut maintenant :

$$\begin{aligned}
&\mathcal{J}(\mathbf{A},\mathbf{B},\mathbf{C}) \\
&= \|\mathbf{W}_{(1)}^{I,KJ} \boxdot \left(\mathbf{X}_{(1)}^{I,KJ} - (\mathbf{A}\boxdot\mathbf{A})((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B}))^T\right)\|_F^2 \\
&= \|\mathbf{W}_{(1)}^{I,KJ} \boxdot \boldsymbol{\delta}_{(1)}\|_F^2 = \|\boldsymbol{\beta}_{(1)}\|_F^2
\end{aligned} \tag{4.1}$$

On a le même type d'expression pour les deux autres modes :

$$\mathcal{J}(\mathbf{A},\mathbf{B},\mathbf{C}) = \|\mathbf{W}_{(2)}^{J,KI} \boxdot \boldsymbol{\delta}_{(2)}\|_F^2 = \|\boldsymbol{\beta}_{(2)}\|_F^2 \tag{4.2}$$

$$= \|\mathbf{W}_{(3)}^{K,JI} \boxdot \boldsymbol{\delta}_{(3)}\|_F^2 = \|\boldsymbol{\beta}_{(3)}\|_F^2 \tag{4.3}$$

Les quantités $\boldsymbol{\beta}_{(i)}$, avec $i = 1, ..., 3$ sont introduites de sorte à simplifier les expressions. En utilisant comme au chapitre précédent $\|\mathbf{A}\|_F^2 = \mathsf{trace}\{\mathbf{A}^T\mathbf{A}\} = \langle \mathbf{A}, \mathbf{A}\rangle$, où $\langle .\rangle$ est le produit scalaire de Frobenius, l'équation (4.1) est alors réécrite :

$$\begin{aligned}
\langle \boldsymbol{\beta}_{(1)}, \boldsymbol{\beta}_{(1)}\rangle &= \mathsf{trace}\{\boldsymbol{\beta}_{(1)}^T\boldsymbol{\beta}_{(1)}\} \\
&= \mathsf{trace}\left\{\left[\mathbf{W}_{(1)}^{I,KJ} \boxdot \left(\mathbf{X}_{(1)}^{I,KJ} - (\mathbf{A}\boxdot\mathbf{A})((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B}))^T\right)\right]^T .\right. \\
&\left.\left[\mathbf{W}_{(1)}^{I,KJ} \boxdot \left(\mathbf{X}_{(1)}^{I,KJ} - (\mathbf{A}\boxdot\mathbf{A})((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B}))^T\right)\right]\right\}
\end{aligned}$$

Nous calculons ensuite la différentielle $\mathsf{d}\mathcal{J}$ de $\mathcal{J}$ afin déterminer les trois matrices de gradient : la matrice $\nabla_{\mathbf{A}}\mathcal{J}$ de dimension $I \times F$, la matrice $\nabla_{\mathbf{B}}\mathcal{J}$ de dimension $J \times F$ et la matrice $\nabla_{\mathbf{C}}\mathcal{J}$ de dimension $K \times F$. Les calculs sont détaillés dans l'Annexe 6.7. Nous trouvons finalement que :

$$\begin{aligned}
&\mathsf{d}\mathcal{J}(\mathbf{A},\mathbf{B},\mathbf{C}) \\
&= \langle 4\left[\mathbf{A}\boxdot\left((-\boldsymbol{\beta}_{(1)}\boxdot\mathbf{W}_{(1)}^{I,KJ})[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})]\right)\right], \mathsf{d}\mathbf{A}\rangle \\
&+ \langle 4\left[\mathbf{B}\boxdot\left((-\boldsymbol{\beta}_{(2)}\boxdot\mathbf{W}_{(2)}^{J,KI})[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})]\right)\right], \mathsf{d}\mathbf{B}\rangle \\
&+ \langle 4\left[\mathbf{C}\boxdot\left((-\boldsymbol{\beta}_{(3)}\boxdot\mathbf{W}_{(3)}^{K,JI})[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})]\right)\right], \mathsf{d}\mathbf{C}\rangle
\end{aligned}$$

On en déduit alors les trois matrices de gradient, qui pourront être utilisées dans n'importe lequel des algorithmes d'optimisation dont nous avons rappelé le principe dans la section 3.4 du chapitre précédent :

$$\begin{aligned}
\nabla_{\mathbf{A}}\mathcal{J}(\mathbf{A},\mathbf{B},\mathbf{C}) &= 4\mathbf{A}\boxdot\left[(-\boldsymbol{\beta}_{(1)}\boxdot\mathbf{W}_{(1)}^{I,KJ})\,((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B}))\right] \\
\nabla_{\mathbf{B}}\mathcal{J}(\mathbf{A},\mathbf{B},\mathbf{C}) &= 4\mathbf{B}\boxdot\left[(-\boldsymbol{\beta}_{(2)}\boxdot\mathbf{W}_{(2)}^{J,KI})\,((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A}))\right] \\
\nabla_{\mathbf{C}}\mathcal{J}(\mathbf{A},\mathbf{B},\mathbf{C}) &= 4\mathbf{C}\boxdot\left[(-\boldsymbol{\beta}_{(3)}\boxdot\mathbf{W}_{(3)}^{K,JI})\,((\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A}))\right]
\end{aligned} \tag{4.4}$$

4.3 Nouvelle approche : algorithme "Varying Weights" (VW)

Dans cette variante de l'algorithme précédent, plutôt que de laisser le tenseur des poids $\mathbf{W}$ fixe au fil des itérations comme cela avait été proposé dans [AKDM11], nous suggérons de mettre à jour ce tenseur et de ré-estimer les valeurs manquantes du tenseur $\mathbf{T}$ au fil des itérations. L'algorithme d'estimation devient donc :

Algorithme. Démarrer avec un tenseur donné $\mathbf{T} \in \mathbb{R}^{+I \times J \times K}$ avec des données manquantes et un tenseur de poids $\mathbf{W}$ avec des entrées binaires au début (une faible constante ϵ au lieu de 0 peut être utilisée si la valeur est manquante, et 1 si la valeur est présente).
Ensuite, tant que le critère d'arrêt n'est pas vérifié :

1. Calculer les matrices de gradient grâce à l'équation (4.4). Ces matrices peuvent être utilisées dans n'importe quel algorithme de descente (cf. section 3.4) afin de déterminer la direction de descente. Durant les premières itérations, le tenseur de données $\mathbf{T}$ contient des valeurs fixées considérées comme manquantes. Pour le calcul du gradient, il convient de les gérer en tant qu'élément neutre (0 pour une addition, 1 pour une multiplication).

Facultatif S'l est prévu de calculer le pas d'adaptation optimal, il convient comme précédemment de gérer les valeurs manquantes du tenseur $\mathbf{T}$ comme élément neutre.

2. Mettre à jour les matrices de facteurs $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$.

3. Si la borne supérieure n'est pas atteinte (définie à l'étape 4), les valeurs manquantes de $\mathbf{T}$ sont corrigées par interpolation. D'abord, on reconstruit le tenseur généré à partir des matrices de facteurs ; cela amène au tenseur $\mathbf{U}$. Si t_{ijk} existe ou n'est pas une ancienne valeur manquante (c'est à dire $w_{ijk} < 1$), il n'y a rien à faire. Sinon, on calcule la valeur moyenne pondérée à affecter à t_{ijk} en prenant en compte les plus proches voisins (six s'ils sont tous présents, moins si certains sont non définis ou non reconstruits). Cette moyenne est pondérée par les valeurs du tenseur $\mathbf{W}$ correspondant à ces voisins.

4. Si $w_{ijk} \neq 1$, on augmente légèrement sa valeur d'une faible constante, par exemple $\alpha = 10^{-3}$. De cette manière, le poids des valeurs initialement manquantes est progressivement augmenté au fur et à mesure que le tenseur est reconstruit, et que les matrices de facteurs sont estimées. Cependant, cette reconstruction n'est pas parfaite, et pour pallier ce problème, on définit une borne supérieure $M < 1$ (par exemple $M = 0.4$, de telle sorte que pour tous les w_{ijk} liés à une valeur manquante $w_{ijk} \leq M$.

5. Vérifier si l'algorithme a convergé. Si c'est le cas, stopper, sinon passer à l'itération suivante.

4.4 Validation de la méthode sur des mélanges synthétiques

Nous générons un tenseur $\mathbf{T}$ en utilisant $F = 4$ composés. Les images de fluorescence de ces composés sont les mêmes que celles présentées dans la section 3.7. On génère une matrice $\mathbf{C}$ aléatoire et non négative de taille 15×4. Un certain pourcentage des données du tenseur a été remplacé aléatoirement par des "Not A Number" (NaN en Matlab), indiquant qu'elles ne sont pas connues. Dans ce but, les coordonnées (i, j, k) de ces points ont été choisies selon une loi uniforme. Sur la droite de la figure 4.1, les 4 matrices de fluorescence d'émission-excitation estimées par l'algorithme du gradient conjugué non négatif avec prise en compte des données manquantes ont été réprésentées. L'algorithme VW a ainsi été utilisé en considérant $M = 0.1$ et $\alpha = 10^{-5}$, avec 30% de données manquantes dans ce cas. Sur la gauche de cette même figure, on donne les résultats pour l'algorithme du gradient conjugué avec contrainte de non négativité (pas de données manquantes dans le tenseur considéré). Les résultats nous montrent que les images reconstituées avec ou sans données manquantes sont très proches.

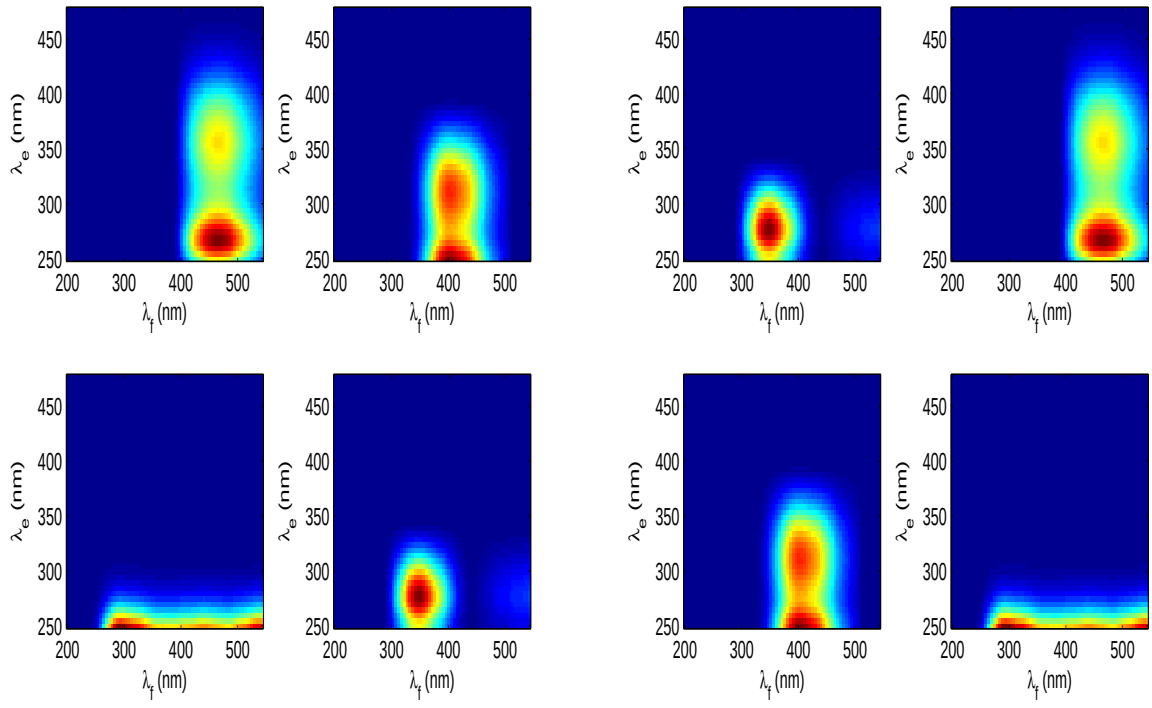


FIGURE 4.1 – Modèle exact ($F = 4$), les 4 images de fluorescence émission-excitation estiméees en utilisant : à gauche : l'algorithme du gradient conjugué avec contrainte de non négativité (pas de données manquantes). A droite : l'algorithme VW (30% de données manquantes).

Le rang est maintenant surestimé. On considère toujours $F = 4$ composés, mais étant donné la méconnaissance de cette valeur dans un cas de traitement réel des données, on réalise la décomposition CP sous l'hypothèse que le rang du tenseur est $\widehat{F} = 6$ (6 fluorophores seraient donc présents). Ici, on choisit $M = 0.15$ et l'on conserve $\alpha = 10^{-5}$. Comme dans la section 3.7,

les composés estimés de façon superflue (composés non existants) devraient être "idéalement" estimés à 0. Comme on peut l'observer sur la figure 4.2, c'est vraisemblablement plus le cas en utilisant la méthode que nous avons introduite dans la section 4.3 : seuls de faibles résidus sont visibles sur cet ensemble d'images. La même méthode avec des poids fixes et sans reconstruction tend à estimer 6 composés au lieu de 4, et le quatrième s'avère très mal estimé. Quant à la méthode CP-WOPT introduite dans [AKDM11], elle ne peut pas restaurer correctement les MEEF du fait qu'elle autorise des valeurs négatives. Effectivement, nous observons bien que de larges zones dans lesquelles devraient se situer le composé organique sont négatives. Le composé estimé sur la 5ème image est d'ailleurs le négatif de celui estimé sur la 6ème image, ce qui est un souci caractéristique des algorithmes n'intégrant pas la contrainte de non négativité sur ce type de problème.

Afin de réellement quantifier l'erreur commise au niveau de l'estimation et de la reconstruction des images de fluorescence des différents fluorophores présents dans les solutions étudiées, nous définissons deux critères d'erreur. Le premier, déjà défini dans la section 3.7, est basé sur le calcul de l'erreur entre le tenseur observé et le tenseur reconstruit :

$$E_{1\mathsf{dB}} = 10\mathsf{log}_{10}(\|T - \widehat{T})\|_F^2), \tag{4.5}$$

où $\widehat{T} = \sum_{f=1}^{F} \widehat{\mathbf{a}}_f \circ \widehat{\mathbf{b}}_f \circ \widehat{\mathbf{c}}_f$ et $\widehat{\mathbf{a}}$, $\widehat{\mathbf{b}}$ et $\widehat{\mathbf{c}}$ sont les facteurs estimés. Le second critère d'erreur (qui ne peut être utilisé que sur des données simulées et non pas sur des données expérimentales) est basé sur l'erreur entre les MEEF des estimées et les MEEF des références :

$$E_{2\mathsf{dB}} = 10\mathsf{log}_{10}(\sum_{i=1}^{F} \|\mathbf{a}_i\mathbf{b}_i^T - \widehat{\mathbf{a}}_i\widehat{\mathbf{b}}_i^T)\|_F). \tag{4.6}$$

Mais pour pouvoir mettre en oeuvre ce second indice d'erreur, il convient en premier lieu de normaliser les MEEF de référence et les MEEF estimées dans puis de les trier/classifier afin d'affecter une image de référence et une image estimée à chacun des composés organiques présents. Ce n'est qu'ensuite que les images affectées à un même composé/fluorophore pourront être comparées. Sur les figures 4.2 et 4.3, on vérifie que l'indice E_1 ne rend que très imparfaitement compte du résultat obtenu : bien que des valeurs très faibles soient atteintes dans le cas de la méthode CP-WOPT, les MEEF des composés organiques se révèlent très mal estimées ce qui conduit par conséquent à une très forte valeur de l'indice d'erreur E_2, que nous présentons sur la figure 4.3. Au contraire, avec les deux méthodes que nous avons suggérées, bien que la valeur de l'indice d'erreur E_1 soit plus importante, l'indice d'erreur E_2 tend bien, quant à lui, à être beaucoup plus faible, ce qui traduit bien une meilleure restitution des MEEF des différents fluorophores. Notons enfin que le critère glouton défini dans [CLDA09] et impliquant les 3 matrices de facteurs (et non pas seulement deux comme nous l'avons fait ici) aurait eu un coût numérique trop lourd raison pour laquelle nous ne l'utilisons pas ici. C'est pour cela que notre critère n'implique que les matrices $\mathbf{A}$ et $\mathbf{B}$.

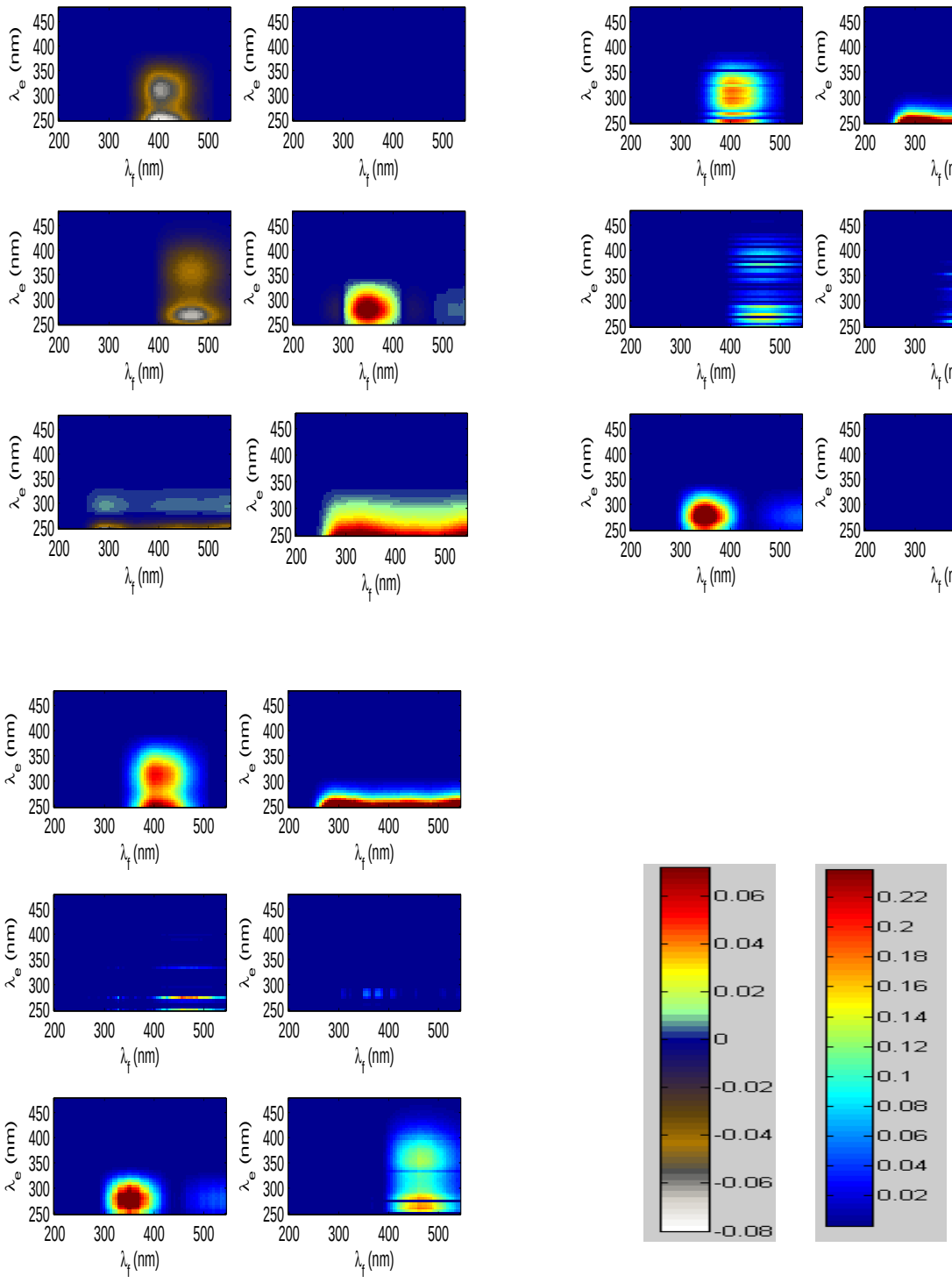


FIGURE 4.2 – 80% de données manquantes. Mélange de 4 facteurs, estimation en considérant $F = 6$. En haut à gauche : les images de fluorescence d'émission-excitation estimées en utilisant l'algorithme suggéré dans [AKDM11] (pas de contrainte de non négativité). En haut à droite : algorithme du gradient conjugué avec une contrainte de non négativité et prenant en compte de possibles données manquantes, à l'aide de poids fixes. En bas à gauche : algorithme VW. Echelle de couleurs : CP-WOPT (gauche) ; GC et VW (droite).

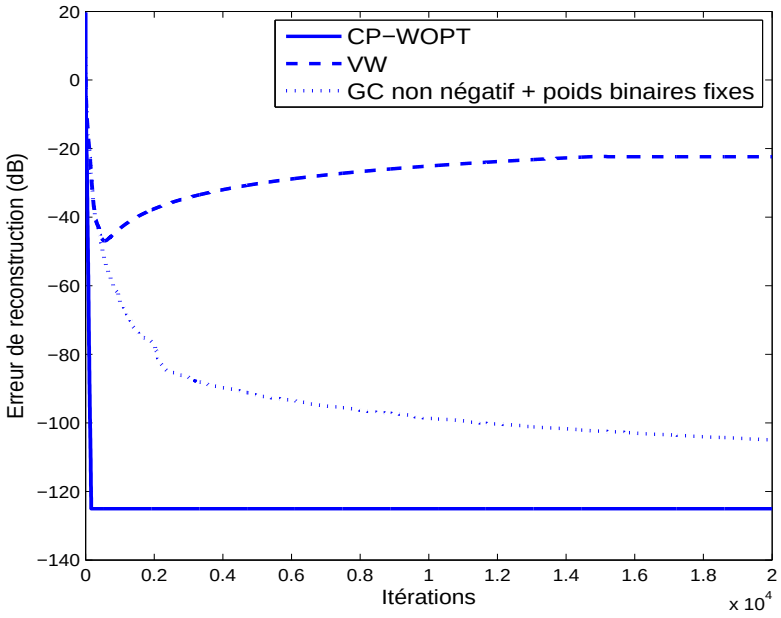


FIGURE 4.3 – 80% de données manquantes dans le cas surestimé (F est supposé égal à 6 pour l'estimation, alors que le F théorique vaut 4). La courbe montre l'évolution de l'erreur E_1 en fonction du nombre d'itérations.

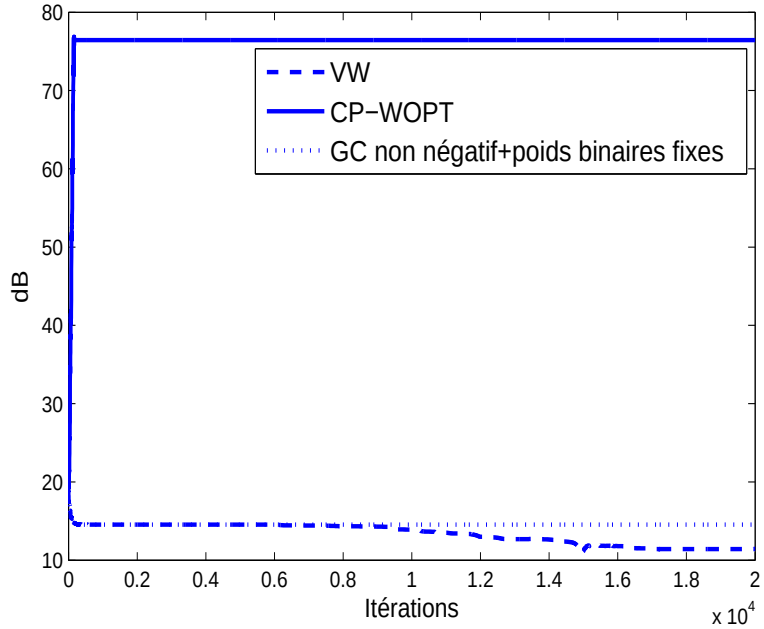


FIGURE 4.4 – 80% de données manquantes dans le cas surestimé (F est supposé égal à 6 pour l'estimation, alors que le F théorique vaut 4). La courbe montre l'évolution de l'erreur E_2 en fonction du nombre d'itérations.

4.4.1 Evolution de la reconstruction avec l'augmentation du taux de données manquantes

On compare maintenant l'évolution de la reconstruction du tenseur et de l'estimation des matrices de facteurs avec l'augmentation du pourcentage de données manquantes. On utilise un nouveau jeu d'images avec des pics de fluorescence davantage rapprochés, ce qui va encore accroître la difficulté du problème de séparation. Comme précédemment, on génère une matrice $\mathbf{C}$ de taille 15×4. Les 4 composés sont représentés par leur image de fluorescence, de taille 71×81.

Les simulations sont effectuées pour des volumes de données manquantes, insérées de façon aléatoire dans le tenseur, allant de 40% à 98%. La figure 4.5 correspond à l'image des quatre composés de référence. Les résultats sont représentés sur les figures 4.6 à 4.9. Visuellement, on constate que les images restent assez proches des images de référence jusqu'à 95% de données manquantes. A partir de 98%, on observe une dégradation notable des images reconstituées. Pour valider ce constat, on superpose sur la figure (4.10) les courbes représentant l'indice E_2 au fil des itérations, selon le pourcentage de données manquantes. Les courbes sont très proches jusqu'à 80% de données manquantes. Elles se stabilisent légèrement au dessus pour 90% et 95%. Par contre, pour 98%, l'estimation est très mauvaise et il y a un écart significatif.

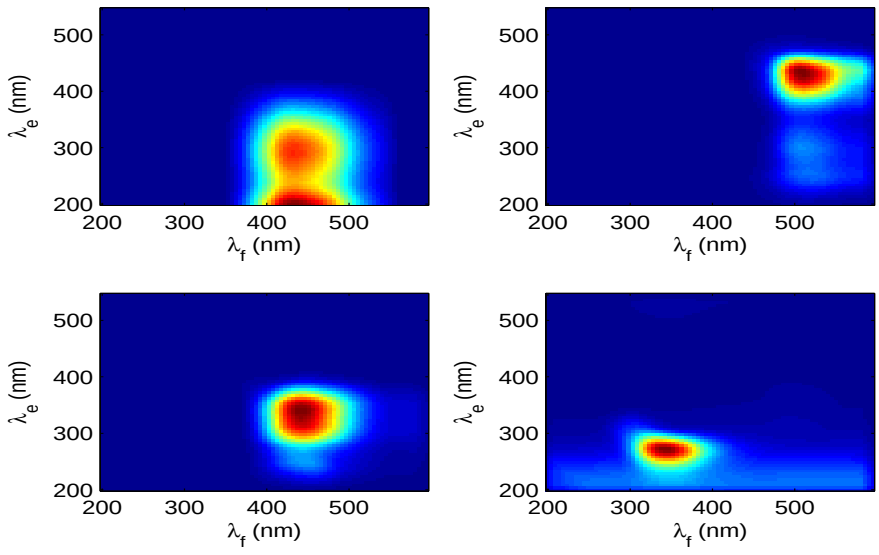


FIGURE 4.5 – Image de fluorescence de référence.

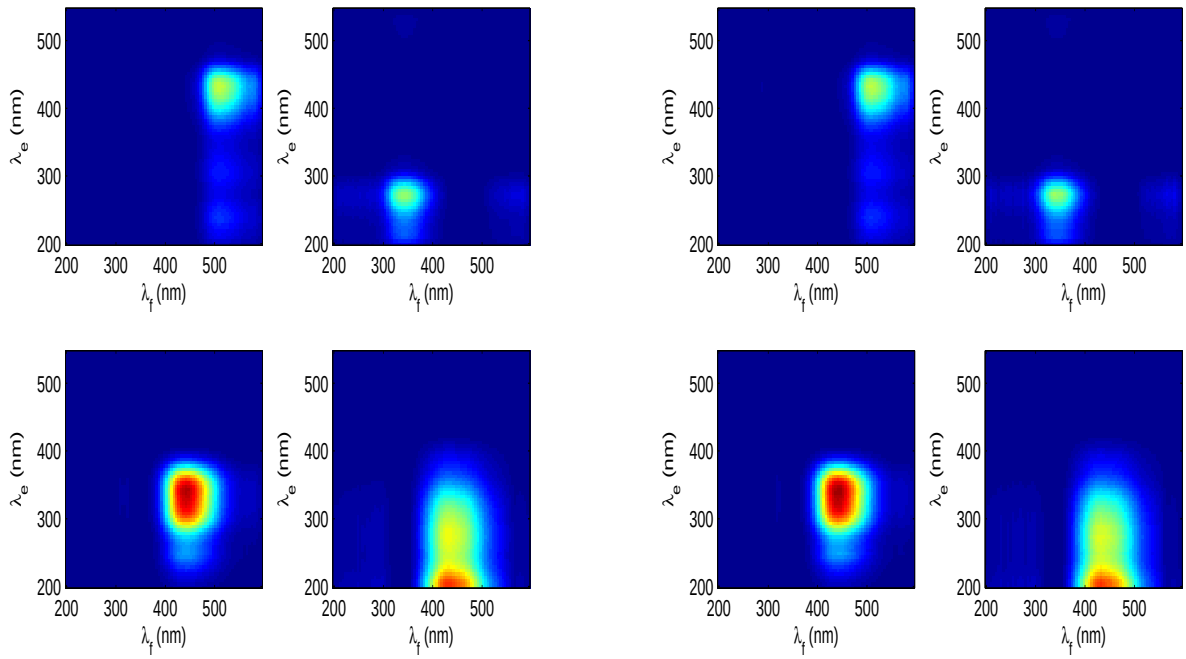


FIGURE 4.6 – MEEF reconstruites avec 40% de données manquantes (à gauche) et 60% (à droite).

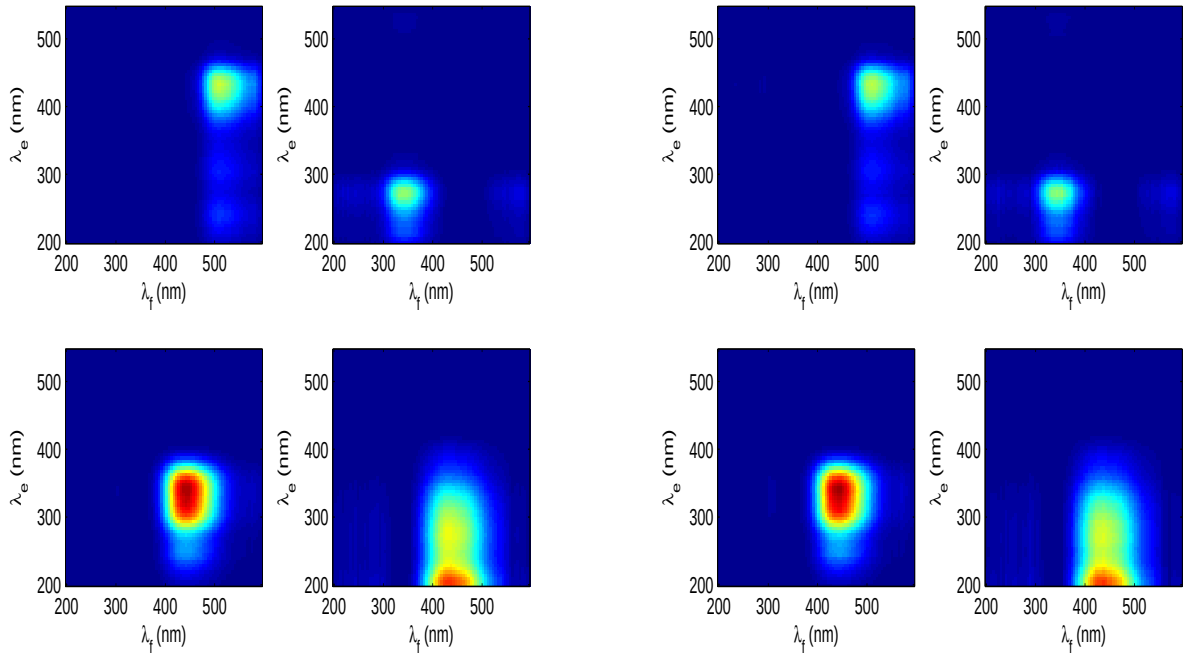


FIGURE 4.7 – MEEF reconstruites avec 70% données manquantes (à gauche) et 80 % (à droite).

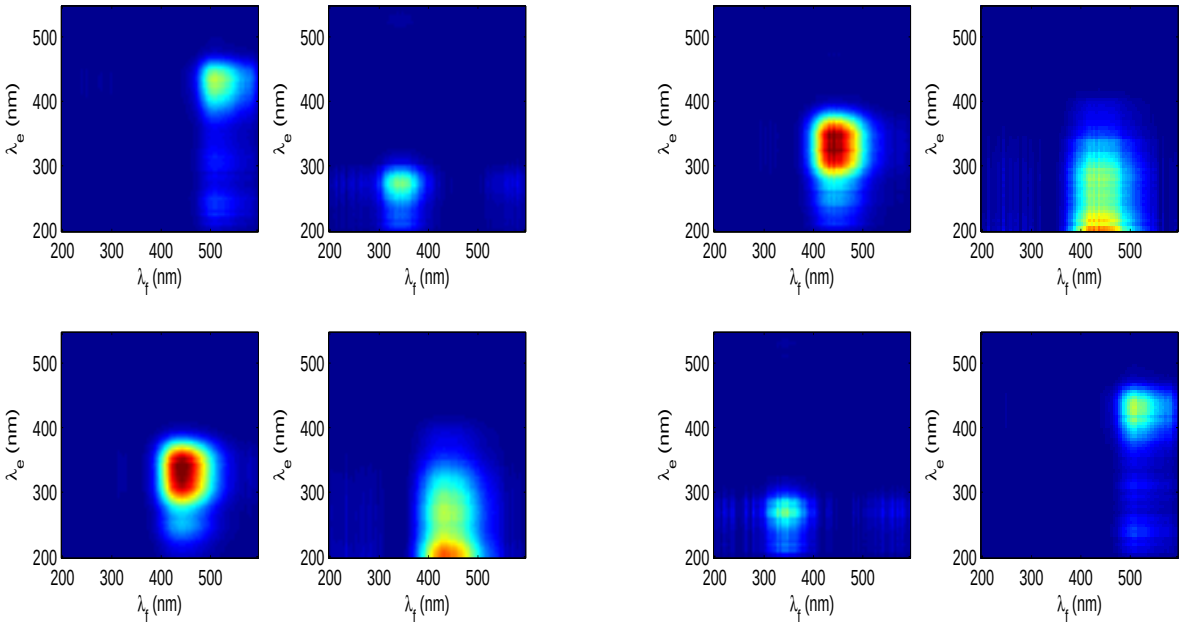


FIGURE 4.8 – Images reconstruites avec 90% de données manquantes (à gauche) et 95 % (à droite).

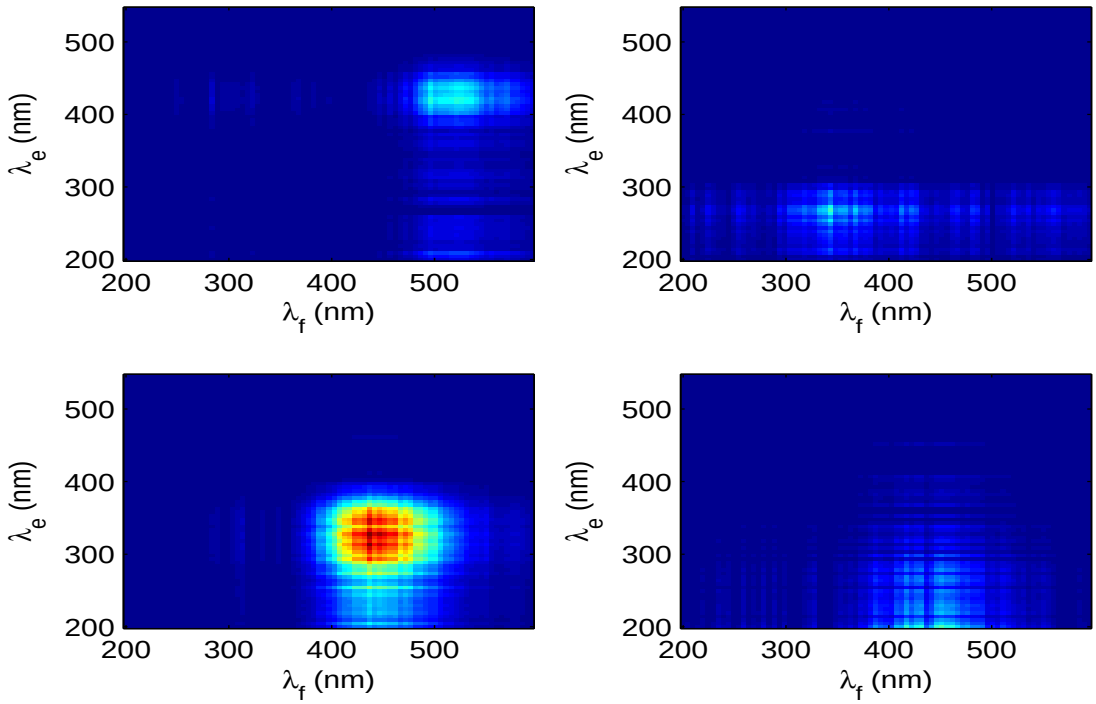


FIGURE 4.9 – Images reconstruites avec 98% de données manquantes.

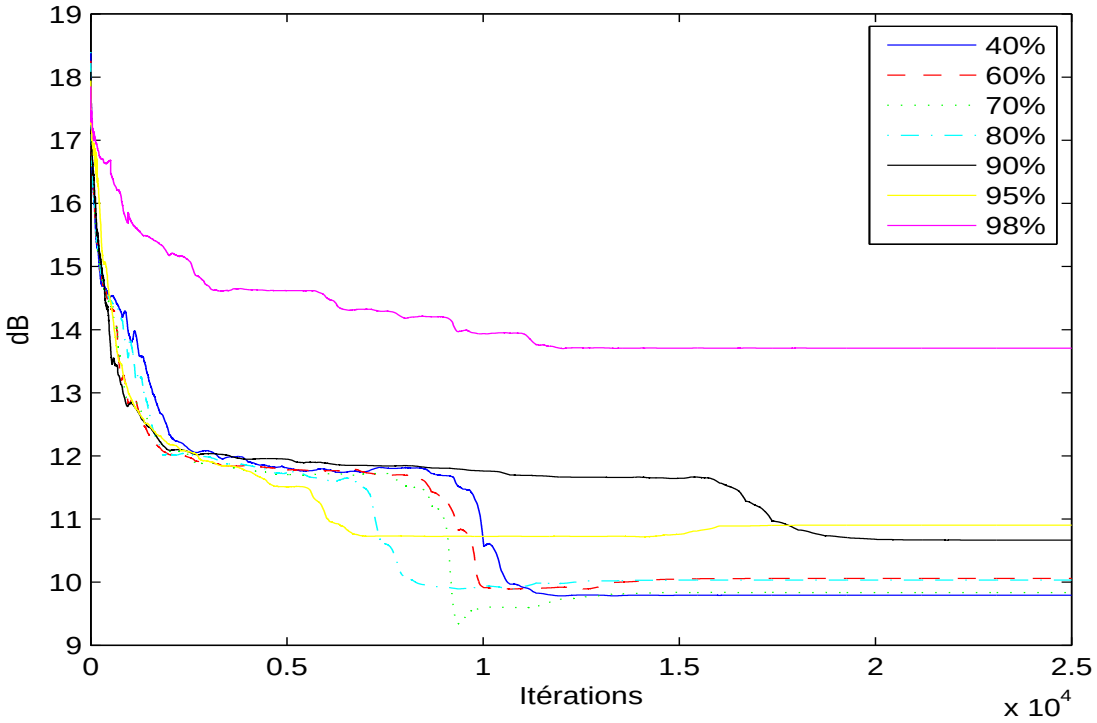


FIGURE 4.10 – Indice E_2 selon le pourcentage de données manquantes dans le cas de l'estimation à rang exact.

4.4.2 Second jeu d'images

On réutilise le jeu de données de la section 4.4.1. On applique à ce tenseur un taux de données manquantes égal à 70%. On compare les résultats des algorithmes CP-WOPT et VW à modèle exact dans un premier temps. Les résultats sont donnés sur les figures 4.15 et 4.16. Visuellement, les résultats de l'algorithme VW sont très proches des images de référence données en 4.5. Les images reconstituées à partir de l'algorithme CP-WOPT présentent un nombre non négligeable de valeurs négatives. Les courbes de performance (Figure 4.17) montrent l'évolution de la qualité de la reconstruction via l'évolution de l'erreur E_2. De façon logique, l'algorithme VW conduit à de bien meilleurs résultats (proches des 10 dB) que ceux obtenus au moyen de l'algorithme CP-WOPT, qui stagne aux alentours de 34 dB. Ceci est bien évidemment lié au fait qu'une partie des valeurs des matrices de facteurs se sont stabilisées dans des valeurs négatives après convergence.

On étudie enfin le cas surestimé. On prend $\widehat{F} = 5$. Les MEEF sont représentées sur les figures 4.11 et 4.12. Comme précédemment, l'échelle de couleurs de la figure 4.13 nous indique que le CP-WOPT converge vers des plages de valeurs négatives. Sur la figure 4.14, on peut constater que l'algorithme VW conduit à un indice E_2 proche de 15 dB après convergence, tandis que le CP-WOPT reste encore dans la zone des 40 dB.

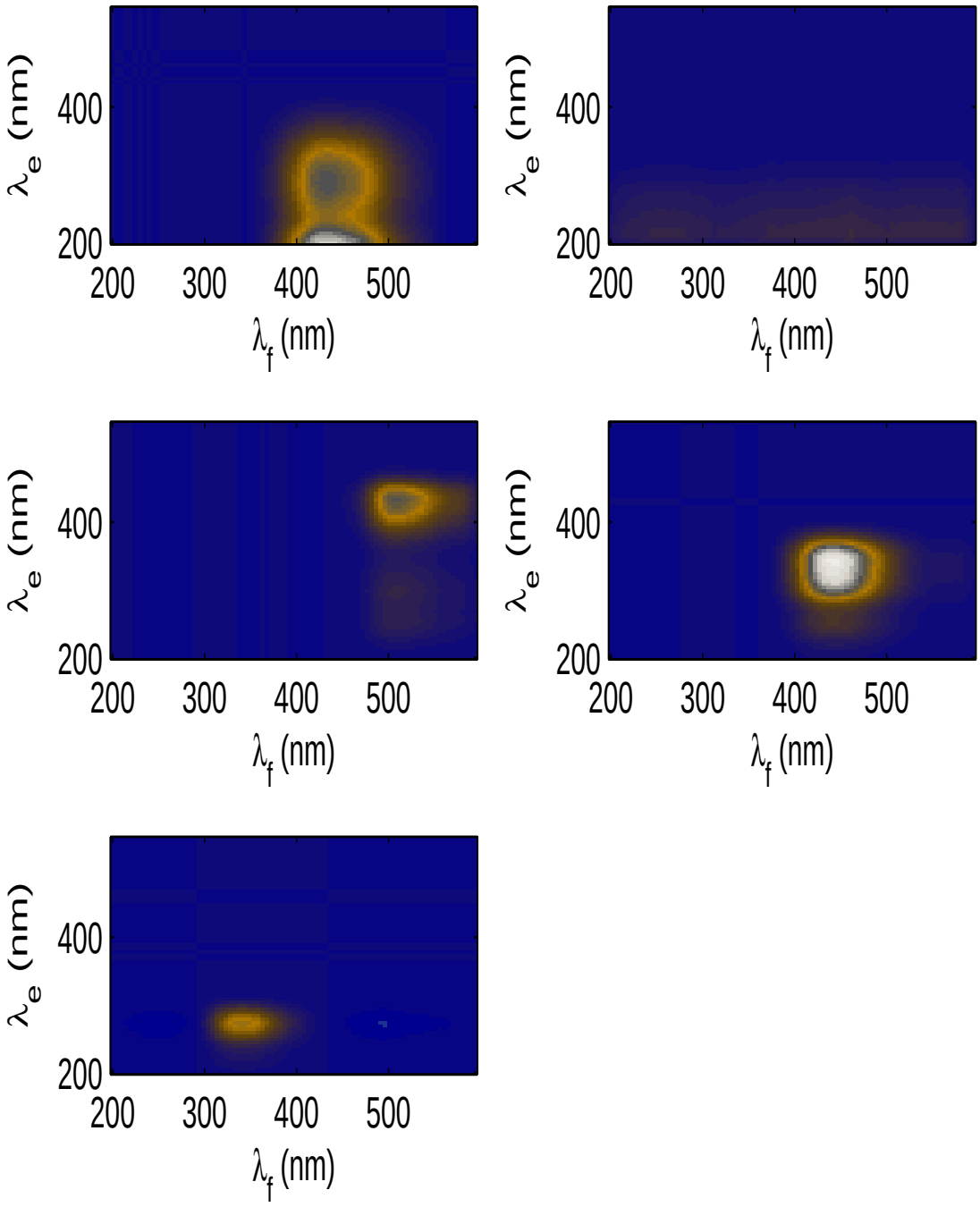


FIGURE 4.11 – Images reconstruites avec CP-WOPT pour 5 composés estimés, alors que le rang réel F vaut 4 en considérant 70% de données manquantes.

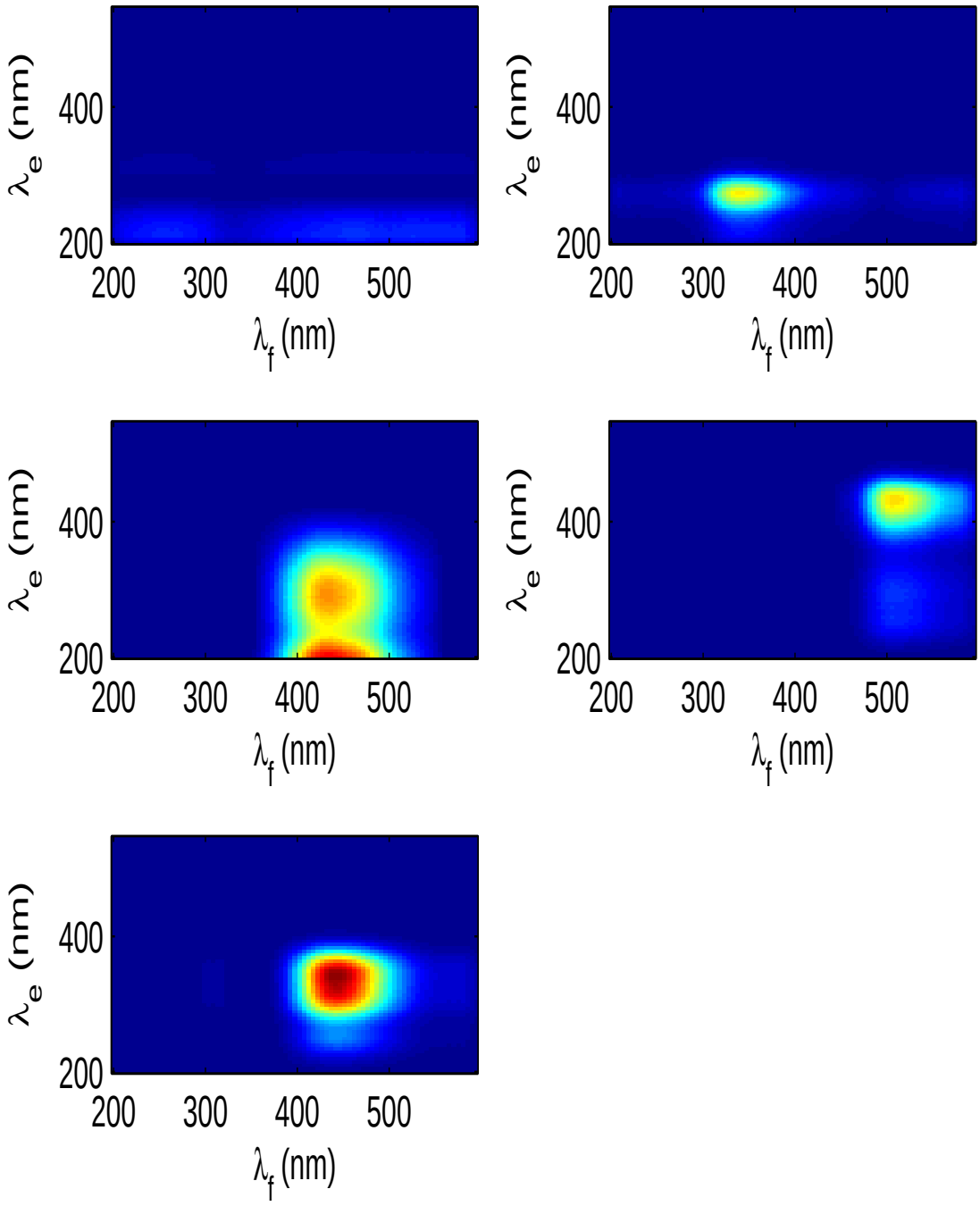


FIGURE 4.12 – Images reconstruites avec VW pour 5 composés estimés, alors que le rang réel F vaut 4 en considérant 70% de données manquantes.

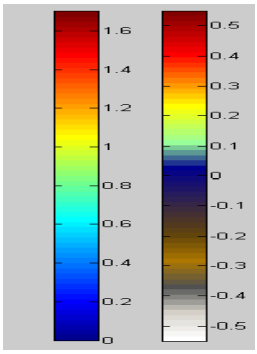


FIGURE 4.13 – Echelle de couleur des figures 4.12 et 4.11

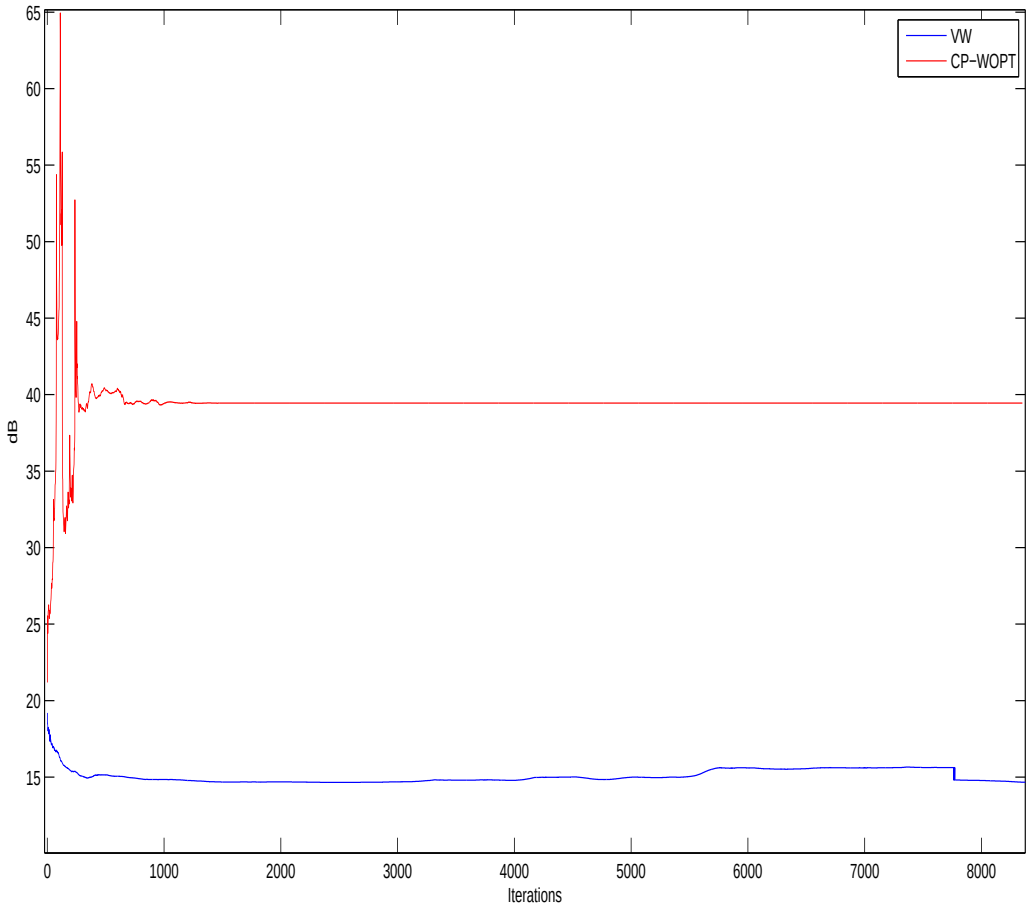


FIGURE 4.14 – 70% de données manquantes dans le cas surestimé (F est supposé égal à 5 pour l'estimation, alors que le F théorique vaut 4). La courbe montre l'évolution de E_2 en fonction du nombre d'itérations.

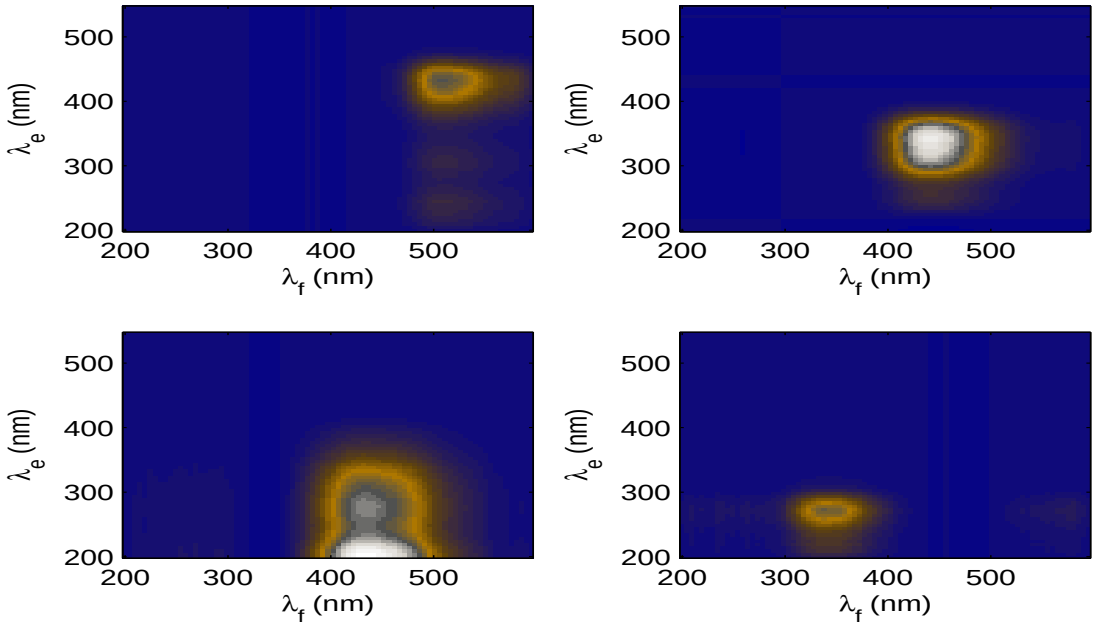


FIGURE 4.15 – 70% de données manquantes dans le cas exact avec l'algorithme CP-WOPT.

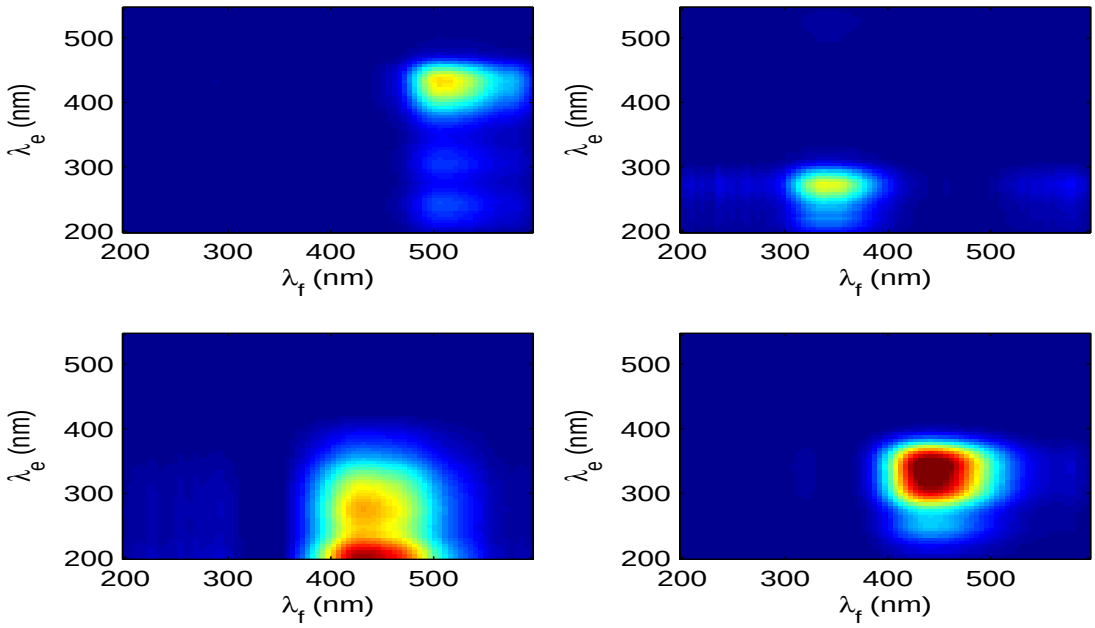


FIGURE 4.16 – 70% de données manquantes dans le cas exact avec l'algorithme VW.

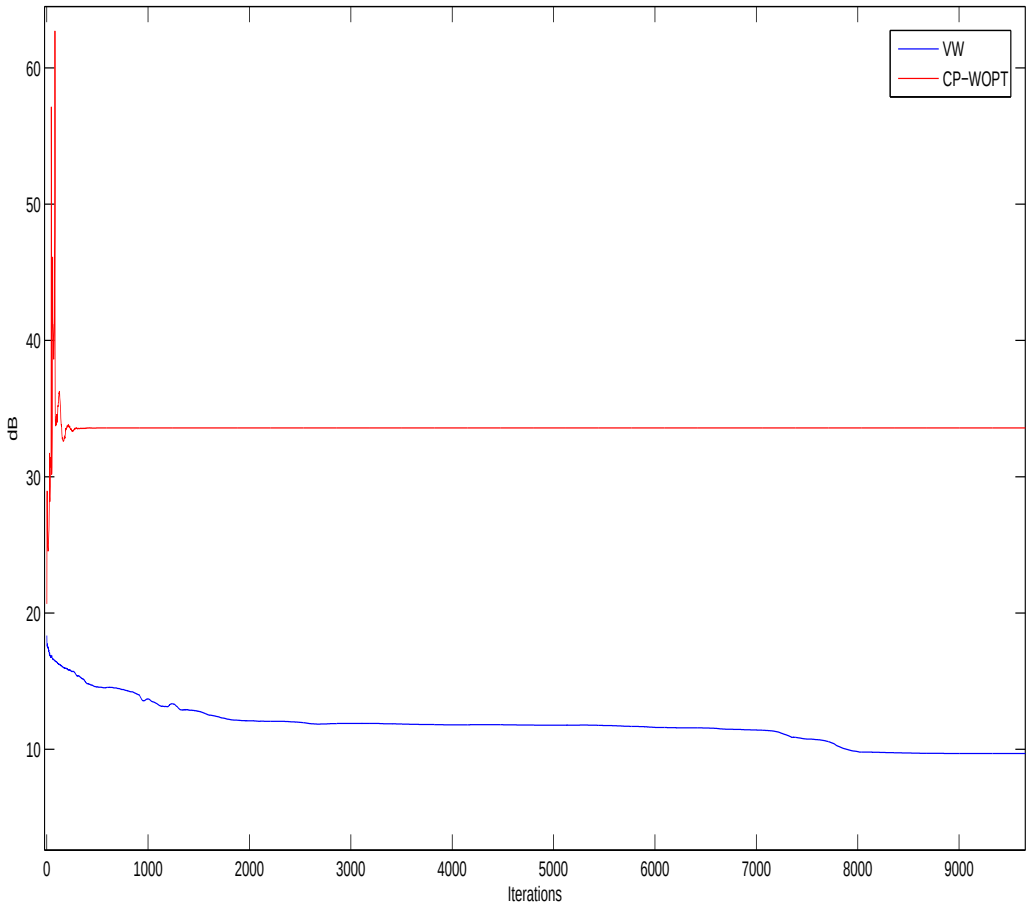


FIGURE 4.17 – 70% de données manquantes dans le cas exact. La courbe montre l'évolution de E_2 en fonction du nombre d'itérations.

4.4.3 Conclusion

Nous avons élaboré de nouveaux algorithmes permettant de réaliser la décomposition de tenseur du $3^{\text{ème}}$ ordre en prenant en compte d'éventuelles valeurs manquantes. Cela nous a amenés à généraliser les méthodes existant dans le cas sans contraintes et avec des poids fixes, avant d'en introduire une nouvelle. Nous avons alors, au travers de simulations sur des mélanges synthétiques, montré le bon comportement de nos algorithmes à rang exact et surestimé. Nous avons ainsi mis en évidence sur ces exemples, que même pour des taux particulièrement élevés de données manquantes ($> 95\%$), nous arrivons à réestimer de façon relativement correcte les matrices de facteurs.

Chapitre 5

Optimisations algorithmiques

5.1 Introduction

Les algorithmes étudiés jusqu'à présent ne fonctionnent bien évidemment pas tous de la même manière. Ainsi, certains peuvent converger en un petit nombre d'itérations, mais chacune de ces itérations peut nécessiter un temps de calcul très important, d'autres peuvent converger au bout d'un grand nombre d'itérations, mais chacune de ces itérations peut être très rapide, etc... Il est intéressant à ce stade de comparer les performances de ces algorithmes sur des critères indépendants de la machine sur laquelle ils sont exécutés.
Toujours sur ces aspects de temps de calcul, des problèmes peuvent se poser dès lors que l'on est confronté à des tenseurs de grande dimension. Certaines quantités telles que le Hessien peuvent devenir impossibles à manipuler voire à construire. Selon les méthodes de décompositions utilisées, les coûts algorithmiques explosent, il devient alors impératif d'optimiser certaines portions de code, ou certaines parties numériques.
C'est donc l'objet de ce chapitre que d'essayer de mieux cerner les coûts algorithmiques et d'essayer de les diminuer si cela s'avère possible. Nous commencerons donc par présenter un éventail de complexités algorithmiques associées aux différents algorithmes d'optimisation introduits précédemment, avant de développer des accélérations algorithmiques adaptées au cas des tenseurs d'ordre 3.

5.2 Calcul des complexités algorithmiques

Un bon moyen de comparer les temps de calcul des différents algorithmes indépendamment du processeur de calcul utilisé consiste à évaluer la complexité algorithmique.
Nous avons résumé dans le tableau donné ci-dessous les complexités liées à la décomposition CP. On se base sur un tenseur du $3^{\text{ème}}$ ordre, de taille $I \times J \times K$ de rang F. on considère que le coût pour multiplier une matrice de taille $N \times M$ par une matrice de taille $M \times P$ est de l'ordre de $O(NMP)$. Le coût de calcul d'un produit de Khatri-Rao entre une matrice $N \times M$ et une matrice $P \times M$ est en $O(NMP)$. Le coût d'une résolution d'un système linéaire $N \times N$ est de l'ordre de $O(N^3)$ (élimination de Gauss-Jordan). Il existe toutefois des algorithmes plus efficaces. Le coût d'une addition ou soustraction est considéré négligeable. Le calcul de la trace d'une matrice est également considéré de coût négligeable.

Opération	**Taille $1^{\text{ère}}$ matrice**	**Taille $2^{\text{ème}}$ matrice**	**Coût numérique**
Multiplication	$N \times M$	$M \times P$	$O(NMP)$
Produit de Khatri-Rao	$N \times M$	$P \times M$	$O(NMP)$
Résolution (Gauss-Jordan)	$N \times N$	$-$	$O(N^3)$

TABLE 5.1 – Complexité algorithmique d'opérations matricielles

En ce qui concerne l'algorithme ALS, le calcul a été fait dans [CLDA09]. Elle est relative à $O(7F^2(JK + KI + IJ) + 3FIJK)$. Pour l'algorithme du gradient, le coût numérique par itération est de $O(6IFJK)$ (puisque pour chacune des trois composantes du gradient, nous avons 4 opérations : 2 produits de matrices + 1 produit de Khatri-Rao + 1 addition. Le coût de calcul est donc régi par le coût du calcul de ces 3 matrices. Le nombre total d'opérations arithmétiques est $O(6IFJKN_{it})$ avec N_{it} représentant le nombre total d'itérations pour atteindre la convergence. Pour le gradient avec contrainte de non négativité, la complexité algorithmique est presque la même, c'est à dire $O(6IFJK)$ et le nombre total d'opérations arithmétiques est aussi $O(6IFJKN_{it})$.
Pour la méthode du gradient conjugué non linéaire (dans les deux cas avec ou sans contrainte de non négativité), la complexité algorithmique est de l'ordre de $O(6FIJK+2(I+J+K)F^2)$ (calcul du β et deux multiplications matricielles).
Pour la méthode du BFGS, dans les cas avec et sans contrainte de non négativité, le coût par itération est de l'ordre de $O(6IFJK + 4(I + J + K)^2F^2 + (I + J + K)^3F^3)$ puisque 4 multiplications de matrices et une inversion de matrice sont requises. Au final, le coût par itération équivaut à $\approx O((I + J + K)^3F^3)$, ce qui implique qu'il est principalement influencé par l'inversion de la matrice Hessienne. Si l'inversion de la matrice est évitée en utilisant par exemple (3.86) au lieu de (3.85)), le coût de calcul est ramené à $\approx O(4(I + J + K)^2F^2)$.

Pour la méthode du DFP (dans les deux cas, soit avec et sans contrainte de non négativité), la complexité algorithmique par itération vaut également $\approx O((I+J+K)^3F^3)$, puisque le coût de calcul du β est négligeable. L'ensemble de ces résultats est résumé au niveau du tableau (5.2).

Méthode	**Coût par itération**	
	cas général	cas $I = J = K$
ALS (sans contrainte de non négativité)	$7(JK + KI + IJ)F^2 + 3IJKF$	$21(IF)^2 + 3FI^3$
ALS-Cichocki	$3IJKF$	$3FI^3$
Gradient	$6IJKF$	$6FI^3$
Gradient conjugué non linéaire	$6IJKF + 2(I + J + K)F^2$	$6FI^3 + 6IF^2$
Gauss-Newton (BFGS)	$(I + J + K)^3F^3$	$27I^3F^3$
BFGS avec (3.86)	$4(I + J + K)^2F^2$	$36I^2F^2$
Gauss-Newton (DFP)	$4(I + J + K)^2F^2$	$36I^2F^2$
Levenberg-Marquardt	$(I + J + K)^3F^3$	$27I^3F^3$
Gradient conjugué non linéaire préconditionné	$(I + J + K)^3F^3$	$27I^3F^3$

TABLE 5.2 – Complexité algorithmique de différents algorithmes

5.2.1 Complexité des algorithmes en utilisant une recherche linéaire globale

Le surcoût en terme de temps lié au calcul du pas optimal est principalement régi par l'évaluation des coefficients $a_0 ... a_n$, tels qu'ils sont décrits dans la section 6.3.2 par exemple. Si on détaille ce dernier exemple, on trouve 7 coefficients, dont les complexités sont données ci-dessous :

$$
\begin{aligned}
a_6 &\sim JKI^2 + I \\
a_5 &\sim 2JKI^2 + I \\
a_4 &\sim 3JKI^2 + I \\
a_3 &\sim 4JKI^2 + I \\
a_2 &\sim 3JKI^2 + I \\
a_1 &\sim 2JKI^2 + I \\
a_0 &\sim JKI^2 + I
\end{aligned}
$$

Au final, la complexité de ces 7 coefficients est de l'ordre de $O(16JKI^2)$. Le coût ensuite inhérent à la recherche des racines du polynôme en μ (basée sur une décomposition en valeurs propres de la matrice compagnon du polynôme) est négligeable devant le coût de celui-ci.
On récapitule dans le tableau 5.3 les complexités liées au calcul du pas optimal. On peut alors présenter les complexités globales des algorithmes présentés dans la section 3.4 prenant en compte une étape de calcul du pas optimal à chaque itération (Tableau 5.4).

Méthode à pas optimal	**Coût par itération** Cas général
CP sans contrainte de non négativité	$16KJI^2 + 16I + 8\,(JKF + FKJI)$
CP avec contrainte de non négativité	$49KJI^2 + 7FKJI + 5JKF + KF + JF + 13JKF + 3IF$

TABLE 5.3 – Complexité algorithmique de l'ELS des différents algorithmes

5.3 Accélérations algorithmiques

5.3.1 Optimisation du calcul des coefficients du pas optimal

Comme cela a été montré dans le paragraphe 5.2, le calcul du pas optimal représente une part importante du coût de calcul global de l'algorithme. Toutefois, on remarque que seuls

Méthode	Coût par itération	
	Cas général	Cas $I = J = K$
ALS sans contrainte	$7(JK + KI + IJ)F^2 + 11IJKF + 9IJK$	$21I^2F^2 + 11I^3F + 9I^3$
Gradient	$49KJI^2 + 13IJKF$	$49I^4 + 13I^3F$
Gradient conjugué non linéaire	$2(I + J + K)F^2 + 49KJI^2 + 13IJKF$	$6IF^2 + 49I^4 + 13I^3F$
Gauss-Newton (BFGS)	$(I + J + K)^3F^3 + 49KJI^2 + 13IJKF$	$27I^3F^3 + 49I^4 + 13I^3F$
Gauss-Newton (BFGS avec (3.86))	$4(I + J + K)^2F^2 + 49KJI^2 + 13IJKF$	$36I^2F^2 + 49I^4 + 13I^3F$
Gauss-Newton (DFP)	$4(I + J + K)^2F^2 + 49KJI^2 + 13IJKF$	$36I^2F^2 + 49I^4 + 13I^3F$
Levenberg-Marquardt	$(I + J + K)^3F^3 + 49KJI^2 + 13IJKF$	$27I^3F^3 + 49I^4 + 13I^3F$
Gradient conjugué préconditionnné	$(I + J + K)^3F^3 + 49KJI^2 + 13IJKF$	$27I^3F^3 + 49I^4 + 13I^3F$

TABLE 5.4 – Complexité algorithmique pour les versions ELS des différents algorithmes

les termes diagonaux de la matrice résultant du produit $\mathbf{K}_i\mathbf{K}_j$ (cf. (3.105a) - (3.105m)) nous intéressent dans la mesure où il s'agit de calculer ensuite la trace de ce produit. Il n'est donc pas nécessaire de calculer toute la matrice $\mathbf{K}_i\mathbf{K}_j$. On obtient facilement la somme des termes diagonaux d'un produit de deux matrices, $\mathbf{A}^{(1)} \in M \times N$ et $\mathbf{A}^{(2)} \in N \times M$, à partir des relations suivantes :

$$d = \sum_{m=1}^{M} \left(\sum_{n=1}^{N} a_{mn}^{(1)} a_{nm}^{(2)} \right) \tag{5.1}$$

$$\tag{5.2}$$

Etant donné que chaque coefficient résulte souvent d'une somme de plusieurs traces, il faut en plus sommer sur le nombre de matrices Q impliquées dans le calcul du coefficient, sachant que les matrices vont toujours par paire :

$$d = \sum_{q=1}^{Q-1} \sum_{m}^{M} \sum_{n}^{N} a_{mn}^{(q)} a_{nm}^{(q+1)} \tag{5.3}$$

Avec le logiciel MATLAB que nous utilisons pour tous nos programmes et simulations, cette astuce n'est pas exploitable en pratique. En effet MATLAB est optimisé pour faire directement du calcul vectoriel avec ses notations propres, or l'astuce que nous suggérons fait intervenir des boucles, et la gestion boucles est interprétée en temps réel, et est donc considérablement plus lente et déconseillée sous MATLAB. Toutefois, on peut intégrer du code en langage C ou C++, et faire les appels de fonctions en utilisant l'outil MEX, acronyme de Matlab EXecutable. Dans sa forme la plus simple, MEX compile un fichier source en langage C ou C++ en une bibliothèque partagée (sous windows, des *Dynamic Link Library* (DLL)).
MATLAB dispose d'une bibliothèque de fonctions C qui permettent d'appliquer des opérations élémentaires sur les matrices et les vecteurs, déjà présentes dans les boîtes à outils MATLAB. Il est également possible d'appeler des fonctions MATLAB directement dans le code C.
La compilation se fait via l'appel de la commande *mex fichierSource.c*. La fonction présente dans le fichier C peut alors être appelée sous MATLAB via son nom sans différence avec un appel de fonction classique.

5.3.2 Découpage du tenseur

Quand on dispose de tenseurs d'ordre 3 de grande dimension, au niveau desquels on dispose de K tranches d'observation (représentatives de temps ou des lieux d'acquisition différents), on peut découper ce tenseur selon le mode des observations en un ensemble de N tenseurs plus petits, tels que $NM = K$. M est alors la taille du tenseur selon le mode tronqué. Dans le cas de tenseurs du $3^{\text{ème}}$ ordre, en admettant que le mode tronqué soit le $3^{\text{ème}}$, on peut alors lancer N décompositions, qui vont nous donner chacune de bonnes approximations des matrices de facteurs $\mathbf{A}$ et $\mathbf{B}$ (on peut aussi moyenner les résultats des N décompositions). Pour chacune des décompositions on obtiendra une partie de la matrice $\mathbf{C}$ du tenseur global, avec seulement M colonnes au lieu des K. En empilant les $\mathbf{C}_{(n)}$ matrices :

$$\mathbf{C} = \begin{pmatrix} \mathbf{C}_{(1)} \\ ... \\ \mathbf{C}_{(N)} \end{pmatrix} \tag{5.4}$$

on obtient ainsi la matrice $\mathbf{C}$ complète. La convergence de ces petits tenseurs tendant à être atteinte en moins d'itérations, le temps global de calcul est au final inférieur.
On peut représenter ainsi l'ordre des étapes :

Optimisation algorithmique par découpage du tenseur

1. Découper le tenseur en N sous-tenseurs dans le mode des observations.
2. Pour $i = 1$ à N :
   (a) Si c'est le premier sous-tenseur :
      i. initialiser les matrices de facteurs.
      ii. Appliquer un algorithme d'optimisation : on obtient $\mathbf{A}_{(1)}$, $\mathbf{B}_{(1)}$ et $\mathbf{C}_{(1)}$.
   (b) Sinon
      i. Initialiser aléatoirement $\mathbf{C}$, nommée $\mathbf{C}_{(i)}$.
      ii. Appliquer un algorithme d'optimisation, initialisé avec $\mathbf{A}_{(1)}$, $\mathbf{B}_{(1)}$ et $\mathbf{C}_{(i)}$.
3. Reconstituer $\mathbf{C}_{\text{final}}$ à partir des $\mathbf{C}_{(i)}$ et de la relation (5.4).

Optionnel : Relancer une optimisation sur le tenseur complet en utilisant $\mathbf{A}_{(1)}$, $\mathbf{B}_{(1)}$ et $\mathbf{C}_{\text{final}}$.

Application - Voici un exemple de simulation, toujours effectué dans le cadre la spectroscopie de fluorescence. On considère un tenseur $47 \times 71 \times 128$, dont les 128 concentrations ont été générées aléatoirement suivant une loi uniforme. Les 4 MEEF de référence utilisées pour générer le tenseur sont celles présentées dans la section 3.7. On dispose donc de 128 observations d'images de mélanges de fluorescence. On découpe ce cube en 16 tenseurs de taille $47 \times 71 \times 8$. On utilise l'algorithme du gradient conjugué paramétré avec des produits de Hadamard pour réaliser la décomposition.
Le temps de calcul pour les 16 étapes du tenseur tronqué est de 502 secondes, tandis que la durée de calcul sur le tenseur global est de 798 secondes. On trace sur la figure (5.1) les MEEF réestimées suite à ce découpage. Ces dernières sont identiques à celles de référence, présentées

sur la figure (3.4). Vérifions maintenant que les concentrations estiméees sont correctes. On les superpose avec celles de référence sur les figures (5.2) et (5.3). Les concentrations estimées sont également identiques à celles qui ont servi à générer le tenseur. Une remarque à ce niveau là : les matrices de facteurs, donc les concentrations sont estimées à un facteur d'échelle près. Comme nous avons réalisé 8 décompositions, il faudra recalculer autant de facteurs d'échelle le long de chaque facteur de la matrice des concentrations estimées.

En conclusion, nous venons de présenter une méthode robuste d'optimisation algorithmique, utile pour des tenseurs en grande dimension et qui permet un gain de temps notable tout en menant aux mêmes résultats dans ce type d'application. L'une des applications potentielles de ce type de technique pourrait être le monitoring de l'eau. Dans l'exemple présenté ici, il n'y a aucun recouvrement entre les tenseurs que l'on considère, il est clair que la même technique pourrait être appliquée sur des fenêtres glissantes avec un certain recouvrement.

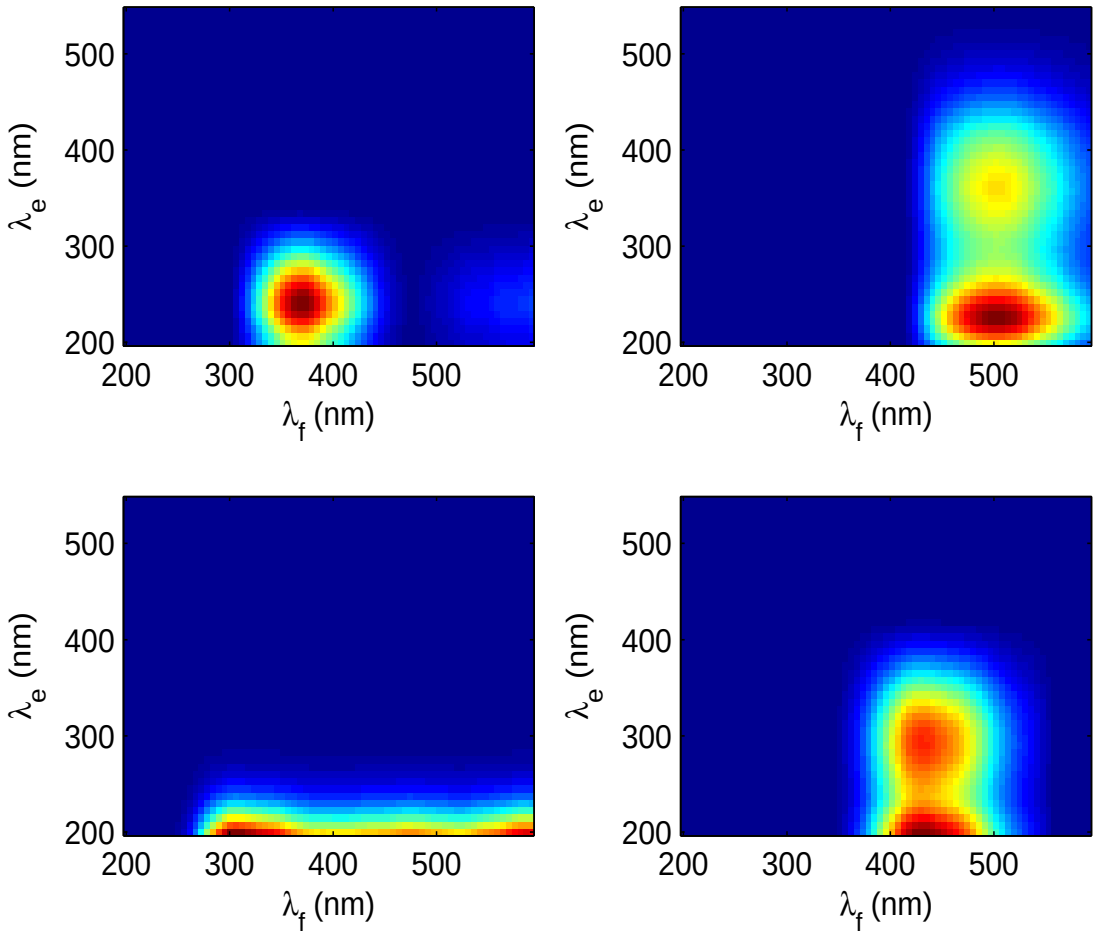


FIGURE 5.1 – MEEF estimées après optimisation par découpage d'un tenseur $47 \times 71 \times 128$ en 16 tranches.

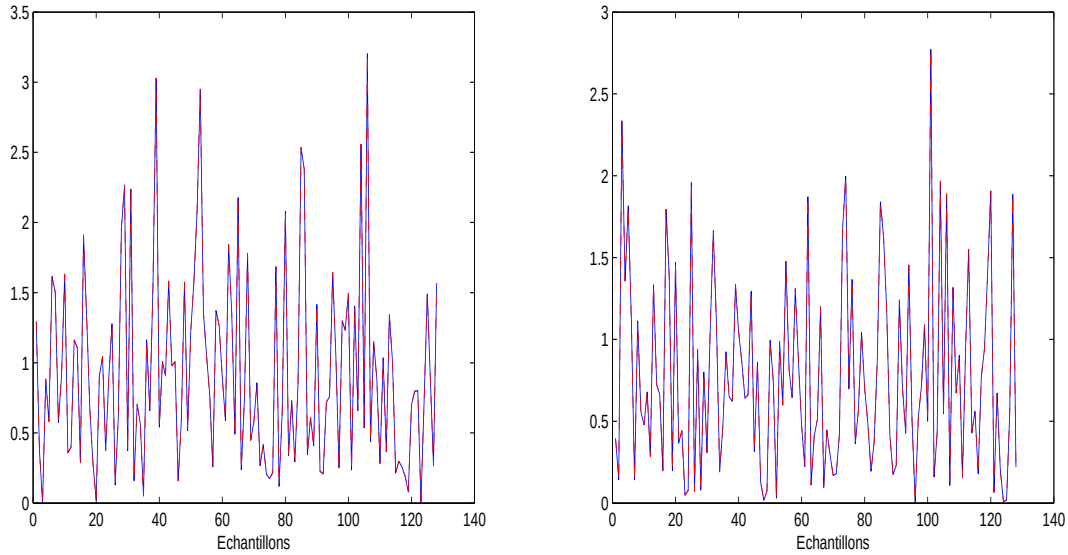


FIGURE 5.2 – Concentrations au fil des échantillons du premier (gauche) et deuxième composé (droite). En bleu, la concentration de référence, en rouge, celle réestimée.

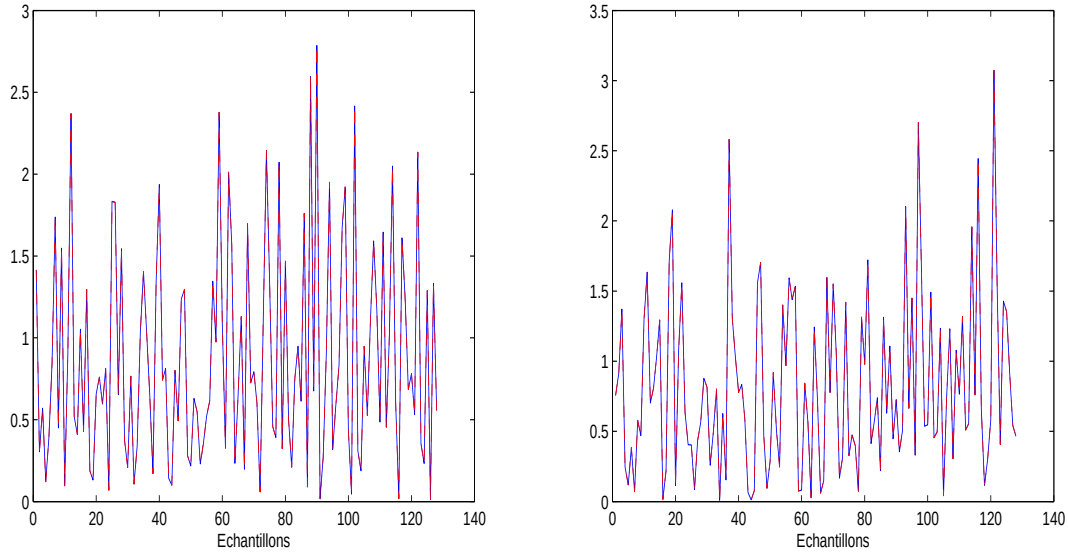


FIGURE 5.3 – Concentrations au fil des échantillons du troisième (gauche) et quatrième composé (droite). En bleu, la concentration de référence, en rouge, celle réestimée.

5.3.3 Conclusion

Nous avons présenté un certain nombre d'optimisations algorithmiques permettant d'accélerer le calcul des décompositions de tenseurs du 3$^{\text{ème}}$ ordre. Il y a d'autres choses que nous avons tentées et qui ne sont pas présentées car elles n'ont pas (encore ?) forcément abouti. Parmi elles, citons les optimisations basées sur de la parcimonie. En effet, les tenseurs que nous manipulons contiennent de larges quantités de valeurs nulles. Parmi les perspectives, des versions adaptatives de nos algorithmes pourraient être développées, avec comme application le suivi de l'environnement via la spectroscopie de fluorescence.

Chapitre 6

Application à des mélanges réels

6.1 Application à des mélanges réels pour la chimiométrie

Dans les chapitres 3 et 4, nous avons testé les algorithmes de décomposition CP que nous avons développés sur des mélanges synthétiques générés numériquement, en partant d'images, représentant la MEEF de chacun des composés présents.
L'objectif est, cette fois, d'appliquer les différents algorithmes de décomposition CP à des mélanges réels réalisés en laboratoire. Toutefois, à la différence de mesures faites in situ ou en laboratoire sur des échantillons prélevés au niveau de la mer ou de rivières dans le cadre d'applications liées à la surveillance de l'eau pour la détection de polluants (et pour lesquels on ne disposerait alors d'aucune information sur le nombre de composés de matière organique présents dans l'échantillon, la nature de ces composés et/ou leur concentration), nous connaissons dans les exemples qui vont être traités le nombre et la nature des composés de matière organique réellement présents dans le mélange, ainsi que les concentrations respectives de chaque composant dans chaque échantillon. Nous allons donc pouvoir calibrer les méthodes développées et vérifier si elles nous permettent de ré-estimer correctement les MEEF de chacun des composés présents de même que leurs concentrations respectives. Nous pourrons également tester la robustesse des algorithmes CP vis-à-vis de certaines erreurs de modèle (sur-estimation du nombre de composés présents, écart par rapport à un modèle rigoureusement trilinéaire).
Les MEEF de chaque échantillon ont été calculées au moyen d'un spectrofluorimètre. Nous reprenons, ici, une partie des jeux de données réalisés par le laboratoire PROTEE et qui avaient déjà été utilisés dans [Luc07]. Chacun des 2 jeux est divisé en 3 groupes. Un groupe contenant des fluorophores à concentration élevée et subissant l'effet d'écran, un deuxième groupe dilué d'un faible facteur servant à corriger l'effet d'écran par la méthode proposée dans [LMRB09]. Le dernier groupe est fortement dilué de sorte à ce que les MEEF ne soient plus soumises aux effets non linéaires. Connaître le facteur de dilution permettant de s'affranchir de l'effet d'écran nécessite d'utiliser des tables de dilution de référence existant pour chaque composant. Dans le cas d'un mélange réel non réalisé en laboratoire, et contenant donc des composés de nature inconnue, ce troisième groupe n'est pas réalisable.
Enfin, les concentrations relatives de chacun des composés au travers de chaque échantillon ont été choisies de sorte à ce que l'intensité de fluorescence d'un composé ne masque pas celle des autres.
Par souci de simplicité, les simulations qui suivent ne reposeront que sur l'utilisation des

TABLE 6.1 – Identification des positions des pics de fluorescence.

	Pic principal		Pic secondaire	
	Excitation	Emission	Excitation	Emission
Tyrosine	275	310		
Tryptophane	275	340-350		
Matière humique	260	380-460		
Matière marine humique	312	380-420		
Phénylalanine	257	279-282		
Sulfate de quinine	350	450	318	
Fluorescéine	480	520-540	453	

groupes fortement dilués, donc non soumis à l'effet d'écran.

Premier jeu de données - Il contient 3 échantillons, composés de 3 fluorophores : tyrosine, phénylalanine et tryptophane, de proportions respectives [1/1/2], [1/2/1] et [2/1/1]. La fente d'acquisition du spectrofluorimètre ayant été réglée sur 10 nm, les raies de fluorescence sont relativement larges. Par conséquent, ce mélange est relativement difficile à analyser. Par ailleurs, nous sommes aussi limités par la résolution de l'échantillonnage, qui est fixée à 3 nm.

Deuxième jeu de données - Il contient 5 échantillons composés de 2 fluorophores : sulfate de quinine et fluorescéine. Il est issu de la dilution d'un facteur 30 du groupe initial. Les concentrations relatives des fluorophores dans les 5 échantillons varient comme suit (sulfate de quinine / fluorescéine) : [10/90], [30/70], [50/50], [70/30], [90/10].

Nouveau jeu de données - Ce jeu a été réalisé plus récemment (il n'a pas été utilisé dans [Luc07]) et fait intervenir à travers 6 échantillons, de la fluorescéine et du sulfate de quinine, dont on connaît les proportions, soit respectivement : 66/30, 66/66, 66/100, 100/33, 100/66, 100/100. Des échantillons contenant du tryptophane font aussi partie de ce jeu, mais ils ont été écartés étant donné que sa trop faible concentration ne permet pas de ré-estimer ce composé.

Le travail de Paula G. Coble réalisé en 1996 [Cob96] permet de valider la position des pics de certains fluorophores. Les valeurs des pics pour d'autres fluorophores peuvent être également trouvées sur le site web suivant : http ://omlc.ogi.edu/spectra/PhotochemCAD/index.html. Notons que la position des pics varie également en fonction d'autres paramètres comme le solvant utilisé, ou la concentration des fluorophores ou encore le pH du solvant utilisé. Ces références ne servent donc que d'indication générale. Les positions de référence sont récapitulées dans le tableau (6.1).

6.1.1 Prétraitement

Nous avons vu dans la seconde partie du chapitre 2 que la génération des MEEF engendrait des perturbations (phénomènes de diffusion Rayleigh et Raman) pouvant masquer les signaux

d'intérêt et qui entravent la trilinéarité du modèle donné par la loi de Beer-Lambert 2.37. Il convient donc de pré-traiter les images avant d'appliquer un quelconque algorithme de décomposition CP. La méthode de correction la plus utilisée est celle qui a été suggérée par Zepp. Nous en avons détaillé le principe dans la section 2.4.3. Elle donne de bons résultats dans la majorité des cas mais elle peut parfois s'avérer trop invasive allant jusqu'à supprimer un pic de fluorescence si celui-ci se trouve dans la zone traversée par les raies de diffusion. La figure (6.1) illustre ce point.

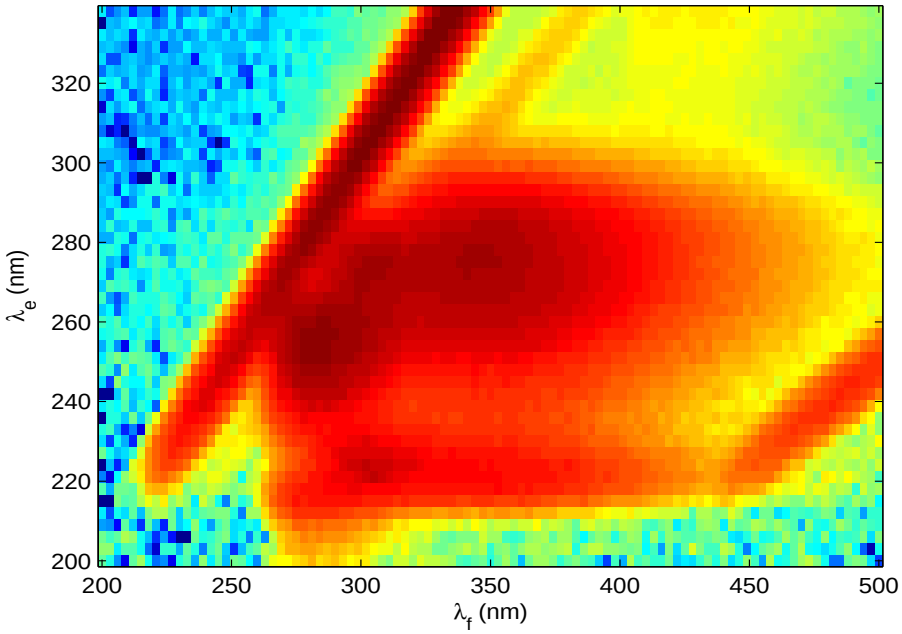


FIGURE 6.1 – Raie de diffusion traversant un pic de fluorescence.

Comme alternative à cette technique de pré-traitement, nous proposons ici une méthode fondée sur du filtrage morphologique d'image. La morphologie mathématique fut introduite en 1965 par J. Serra et G. Matheron [Ser83, MS02]. Le principe de la morphologie mathématique est d'utiliser une forme géométrique appelée élément structurant qui va servir de support pour analyser l'ensemble qu'on souhaite étudier. Cet élément structurant est défini par un masque binaire et un centre (point d'ancrage).
Afin de supprimer les raies de diffusion, nous allons réaliser une ouverture, qui combine deux opérations de morphologie mathématique : une opération de dilatation mathématique et une opération d'érosion.
Rappelons sur un exemple le mode opératoire de la dilatation quand l'image considérée est une image binaire. Pour chaque pixel de l'image, si il y a intersection entre l'élément structurant et l'ensemble, alors le pixel situé sur le point d'ancrage est annexé à l'ensemble (cf. figure (6.3)).
L'érosion est l'opération duale de la dilatation. Pour chaque pixel de l'image, si l'élément structurant n'est pas complètement inclus dans l'ensemble, alors le pixel situé sur le point d'ancrage n'appartiendra pas à l'ensemble érodé (cf. figure (6.3)).

FIGURE 6.2 – Elément structurant de type ligne. Le point d'ancrage est situé sur le carré du centre.

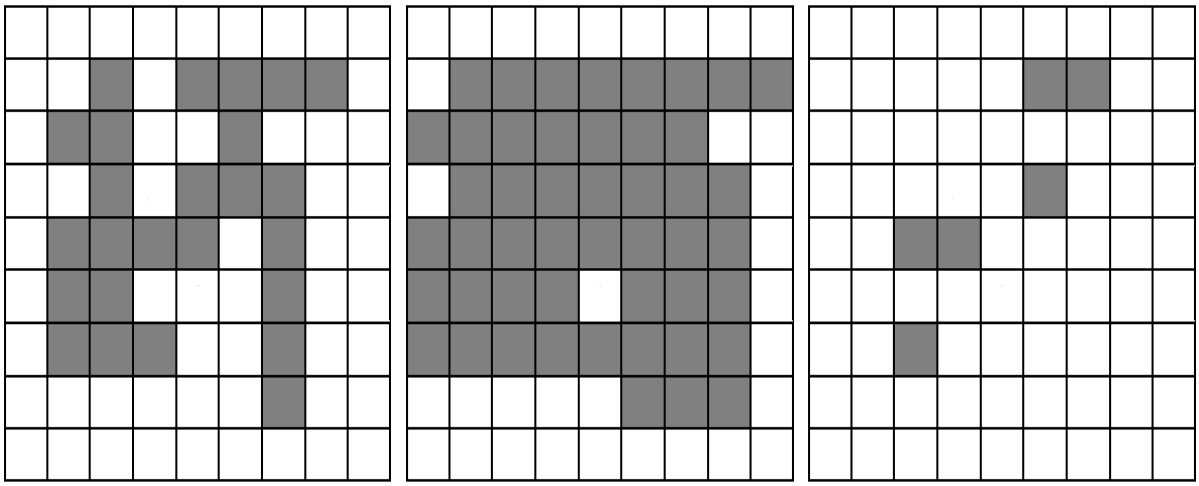


FIGURE 6.3 – On utilise l'élément structurant de la figure (6.2). A gauche, l'image d'origine. Au centre, la dilatée de l'image d'origine. A droite, l'érodée de l'image d'origine.

La morphologie mathématique définie pour des images binaires a été étendue au cas d'images en niveaux de gris [GW01], cas qui nous concerne, voire même à celui d'images en couleur. On peut alors utiliser des éléments structurants plats, comme précédemment, ou des éléments structurants en 3D (cas général, mais plus rarement utilisé). En utilisant un élément structurant plat appelé S, on définit la dilatation de l'image $\mathbf{I}$ comme

$$D_{\mathbf{I}(p)} = \mathsf{sup}\left\{\mathbf{I}\left(q\right),\, q \in \mathsf{S}_p\right\} \tag{6.1}$$

et l'érosion comme

$$E_{\mathbf{I}(p)} = \mathsf{inf}\left\{\mathbf{I}\left(q\right),\, q \in \mathsf{S}_p\right\}, \tag{6.2}$$

où q représente l'ensemble des pixels appartenant à S et S_p est le translaté de S au pixel p de l'image. Algorithmiquement, cela se traduit par le fait de parcourir les p pixels de l'image et de leur affecter la valeur maximum (dilatation) ou la valeur minimum (érosion) trouvée parmi les pixels recouverts par l'élément structurant centré en p.
Ainsi, en utilisant un élément structurant de type ligne, (approximativement) orthogonal aux raies de diffusion, ces dernières seront supprimées, alors que le reste de l'image ne sera pas "trop" altéré. L'effet positif étant surtout qu'un pic de fluorescence traversé par les raies de diffusion ne sera pas éliminé.
En partant de l'image brute donnée en figure 6.1, on obtient les résultats de la figure 6.4, après filtrage passe-bas (filtre moyenneur) des images corrigées afin de lisser les résultats. La

correction par morphologie mathématique conserve relativement bien les 2 principaux pics de fluorescence qu'on distingue dans les zones $((\lambda_e; \lambda_f))$ $(250; 280)$ et $(270; 350)$, alors que le premier de ces pics est très altéré par la méthode de Zepp. La figure 6.6 illustre sur un deuxième exemple l'apport d'une correction par morphologie mathématique plutôt que par la méthode de Zepp plus invasive là encore.

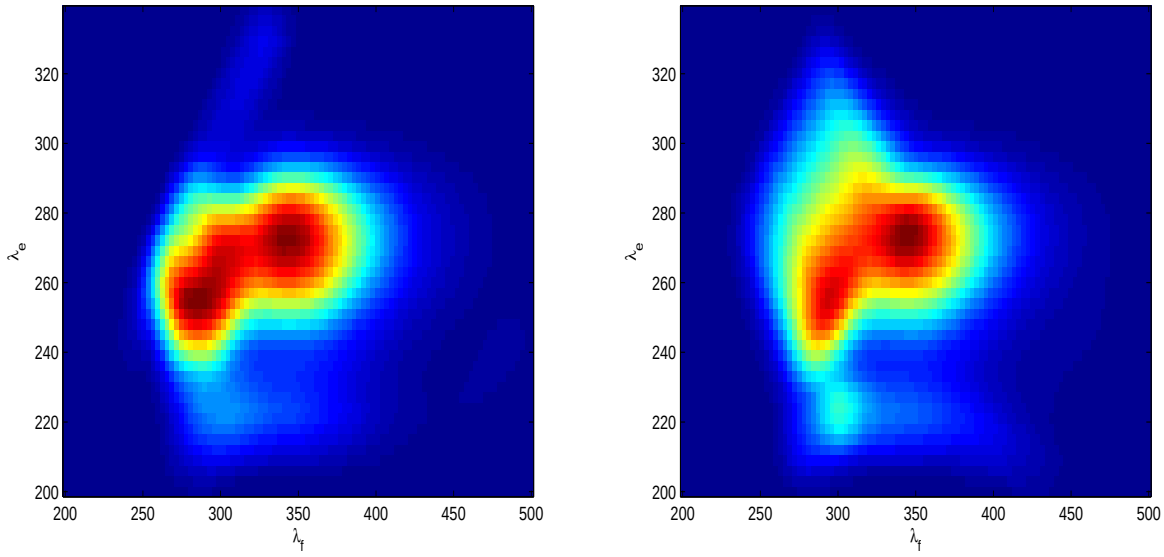


FIGURE 6.4 – Suppression des raies de diffusion de l'image (6.1). A gauche, l'image corrigée par morphologie mathématique. A droite, celle corrigée par la méthode de Zepp.

Le nettoyage des raies de diffusion par morphologie mathématique supprime toutefois un peu moins bien les raies de diffusion dans les zones où le signal est absent. Pour parer à ce problème, il est alors possible combiner les deux algorithmes, de façon pondérée. Si on appelle $\mathbf{I}_{Zepp}$ l'image corrigée par l'algorithme de Zepp, $\mathbf{I}_{Morphologie}$ l'image corrigée par morphologie mathématique, l'image résultante peut être définie par :

$$\mathbf{I}_{corrige} = \alpha \mathbf{I}_{Zepp} + (1 - \alpha)\mathbf{I}_{Morphologie} \tag{6.3}$$

où α est un coefficient compris entre 0 et 1. Un exemple d'image résultante $\mathbf{I}_{corrige}$ est donné sur la figure (6.5).

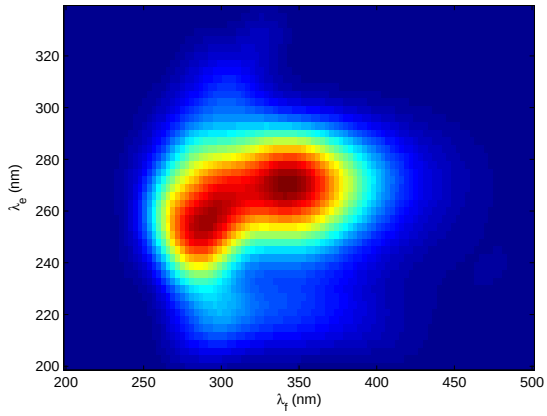


FIGURE 6.5 – Image corrigée en combinant les images données en (6.4) par la méthode de Zepp et la morphologie mathématique, dans cet exemple nous avons choisi $\alpha = \frac{1}{3}$.

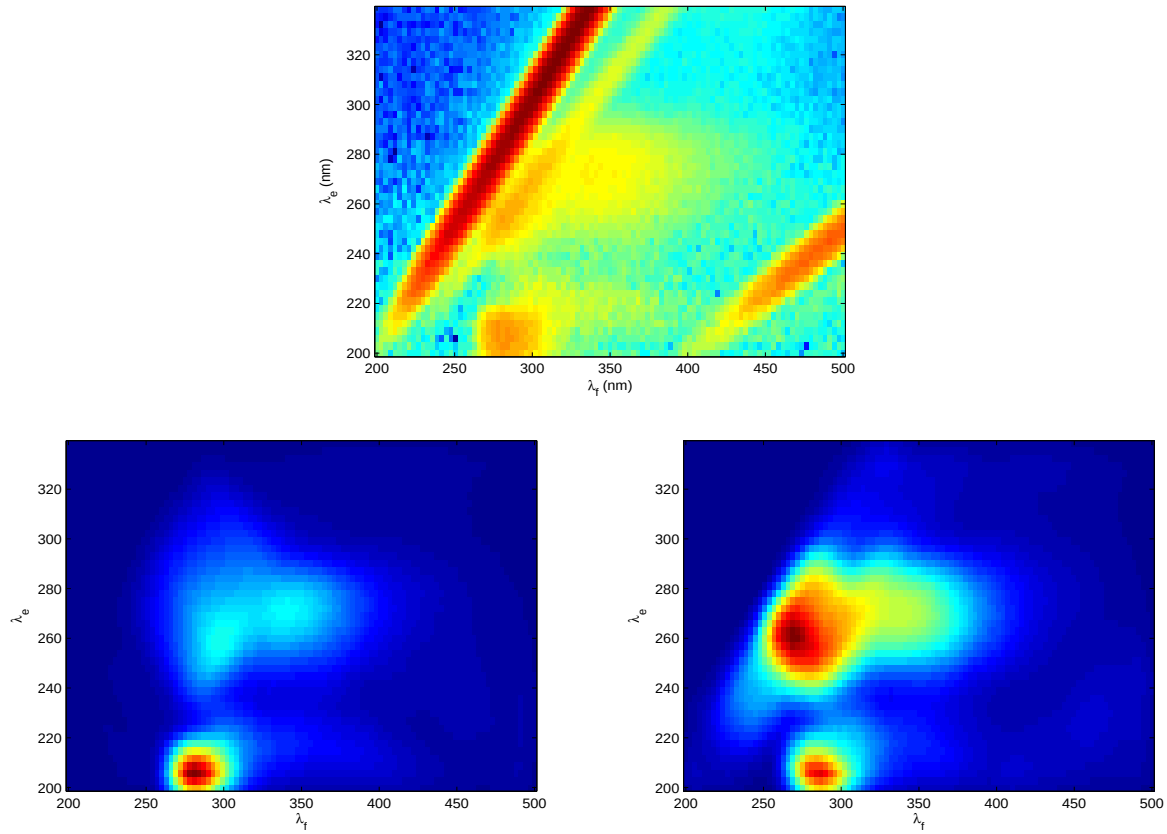


FIGURE 6.6 – Autre exemple pour lequel la correction par morphologie mathématique donne de meilleurs résultats. En haut, l'image d'origine. En bas à gauche, celle corrigée par la méthode de Zepp : la zone de fluorescence, traversée par les raies de diffusion, est pratiquement supprimée. En bas à droite, la correction par morphologie mathématique, qui conserve toutes les zones contenant initialement du signal de fluorescence.

6.1.2 Jeu de données à 3 fluorophores

Pour ce jeu à 3 constituants, on dispose d'un tenseur de taille $101 \times 47 \times 3$ sur lequel on a traité les raies de diffusion en utilisant la relation (6.3) en prenant $\alpha = \frac{1}{3}$. On réalise la décomposition CP en utilisant le gradient conjugué non négatif, initialisé avec l'algorithme DTLD [SK90].

Les spectres d'excitation et d'émission estimés après décomposition CP par gradient conjugué non négatif sont tracés sur la figure (6.7). On prétraite par filtrage morphologique combiné (cf. (6.3) avec $\alpha = \frac{1}{3}$). On superpose donc sur chacune des courbes les spectres estimés après décomposition CP , le spectre de référence, et le spectre de référence dans lequel on a supprimé (en les mettant leur valeur à zéro) les longueurs d'onde qui correspondent aux raies de diffusion (ceci afin de mieux visualiser les pics de fluorescence masqués du fait de la très forte amplitude des raies de diffusion). Chacune de ces trois courbes a été normalisée.

On observe que les spectres estimés en émission sont très bien calés par rapport aux spectres de référence pour chacun des trois fluorophores estimés. C'est un peu moins vrai en excitation, où l'on observe un décalage du pic principal de 17 nm pour la phénylalanine, de 5 nm pour la tyrosine, et de 4 nm pour le tryptophane. L'amplitude des pics secondaires a aussi été altérée.

Nous estimons que le pic du tryptophane se situe en $(272; 350)$, celui de la tyrosine en $(271; 305)$ et celui de la phénylalanine en $(255; 280)$ ce qui correspond par contre bien aux zones données par le tableau 6.1.

Sur la figure 6.8, on affiche également les spectres estimés après décomposition CP par gradient conjugué non négatif. Cependant, le prétraitement des raies de diffusion n'est plus du filtrage morphologique, mais une simple de correction par la méthode de Zepp. Bien que le pic de la phenylalanine soit mieux calé en excitation (malgré une estimation très grossière de sa largeur), ce n'est plus le cas en émission. De la même façon, la tyrosine est plus mal estimée que précédemment, que ce soit en excitation ou en émission. Le tryptophane, qui n'est pas affecté par le passage des raies de diffusion, et par conséquent, n'est pas corrigé lors du prétraitement, donne les mêmes résultats que sur la figure 6.7. On montre donc par un exemple concret les inconvénients de ce prétraitement lorsque des pics de fluorescence se trouvent dans la zone traversée par les raies de diffusion.

Concernant les concentrations relatives tracées sur la figure 6.9, on retrouve bien les proportions données dans la section précédente, et dans le bon ordre. Le tryptophane présente ainsi des proportions relatives de 0.47 puis 0.26 et 0.26, proches des rapports 2/1/1 attendus. Il y a plus de différences pour la tyrosine, où la première valeur est 0.3 (puis 0.25 et 0.45), ce qui correspond aux rapports 1/1/2. Il y a aussi des différences pour la phénylalanine, où la première valeur est basse, à 0.21 (puis 0.53 et 0.25), ce qui correspond à un rapport de 1 / 2 / 1.

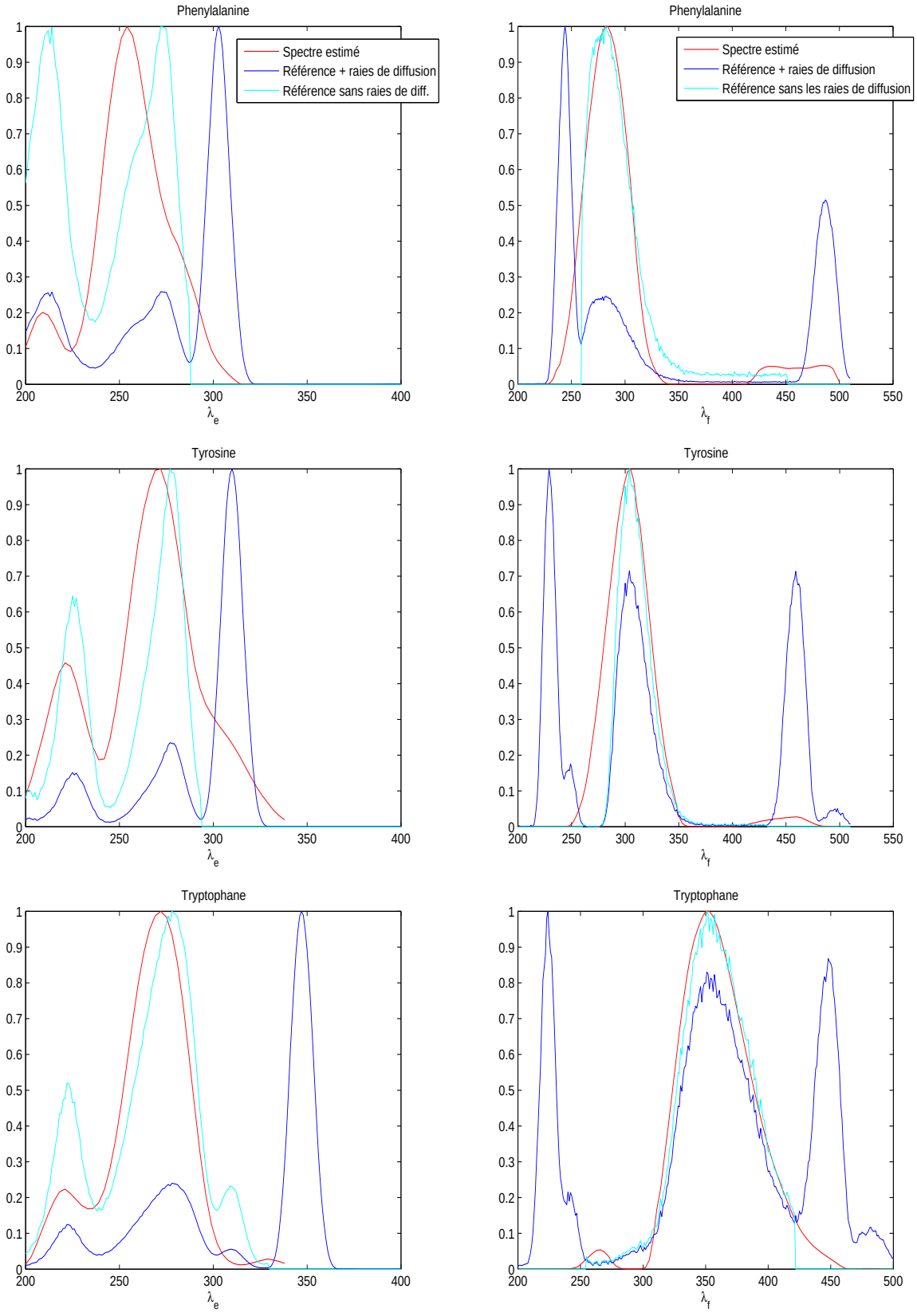


FIGURE 6.7 – Spectres estimés après prétraitement par filtrage morphologique. A gauche sont représentés tous les spectres d'excitation, et à droite les spectres d'émission. De haut en bas : la phénylalanine, la tyrosine et le tryptophane. Sur chaque figure, la courbe rouge correspond au spectre estimé par décomposition CP, la courbe bleue au spectre de référence et la courbe cyan au spectre de référence sur lequel on a tronqué la zone qui correspondait aux raies de diffusion pour plus de lisibilité.

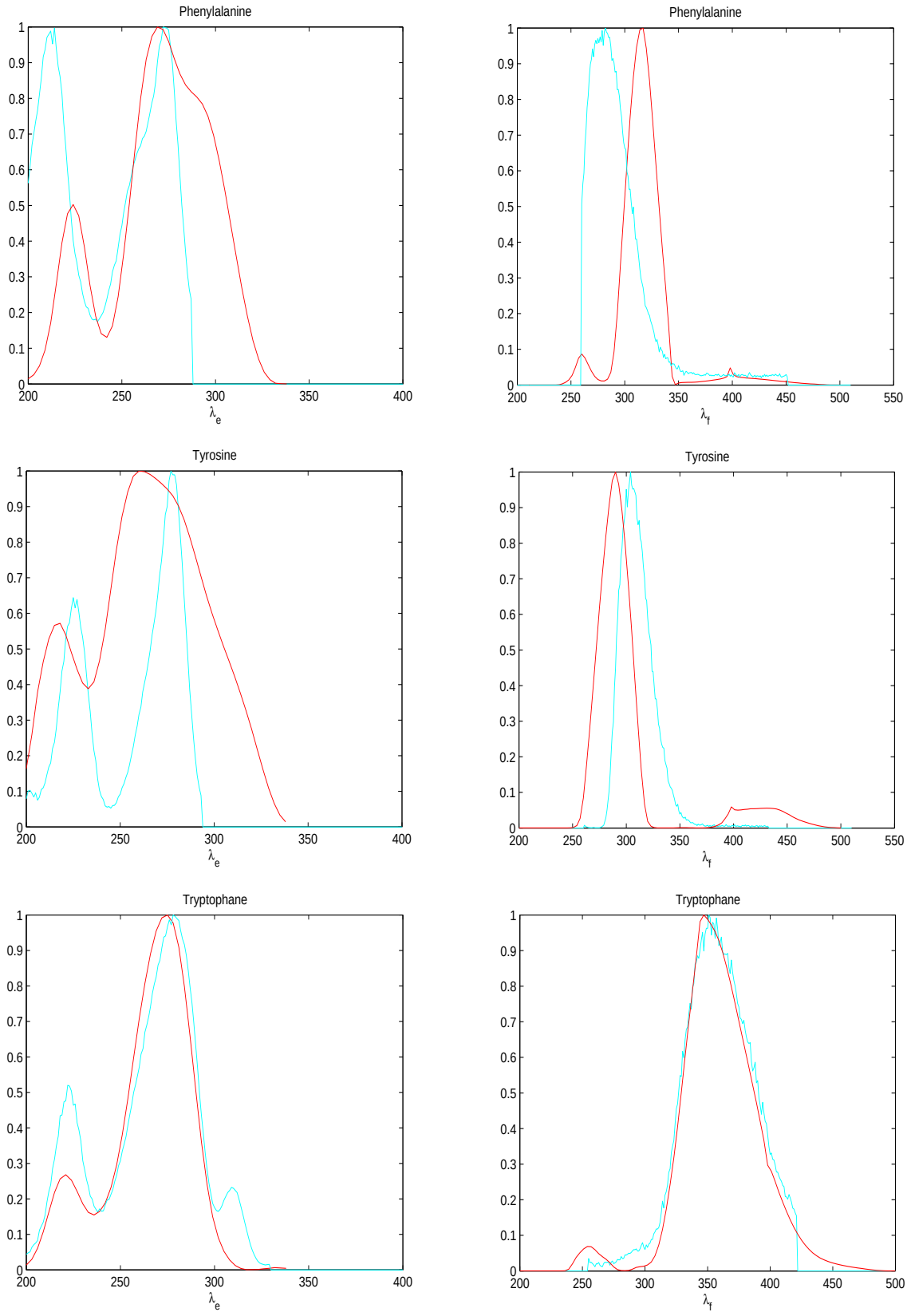


FIGURE 6.8 – Spectres estimés après correction par la méthode de Zepp. A gauche sont représentés tous les spectres d'excitation, et à droite les spectres d'émission. De haut en bas : la phénylalanine, la tyrosine et le tryptophane. Sur chaque figure, la courbe rouge correspond au spectre estimé par décomposition CP, la courbe bleue au spectre de référence et la courbe cyan au spectre de référence sur lequel on a tronqué la zone qui correspondait aux raies de diffusion pour plus de lisibilité.

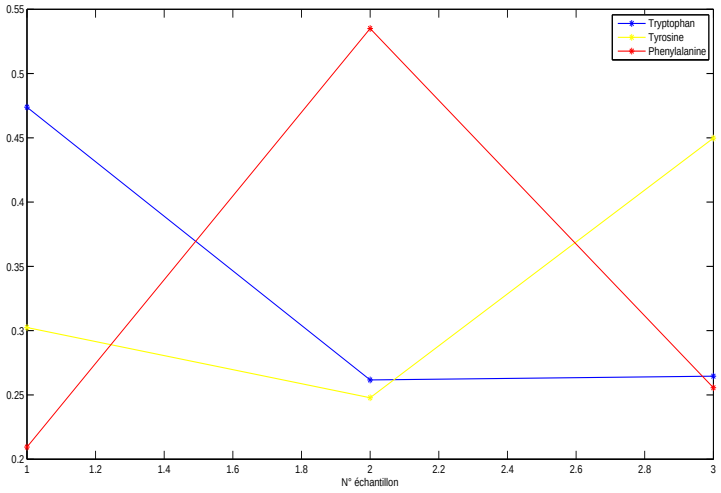


FIGURE 6.9 – Concentration relative des différents fluorophores au sein de chaque échantillon.

Les MEEF de référence sont données sur la figure 6.10. Celles estimées par décomposition CP (algorithme du gradient conjugué non négatif) sont visibles sur la figure 6.11. Conformément à ce que nous avions déjà observé au niveau des comparaisons des spectres en émission et en excitation, les images des références et des fluorophores estimés s'avèrent très proches.

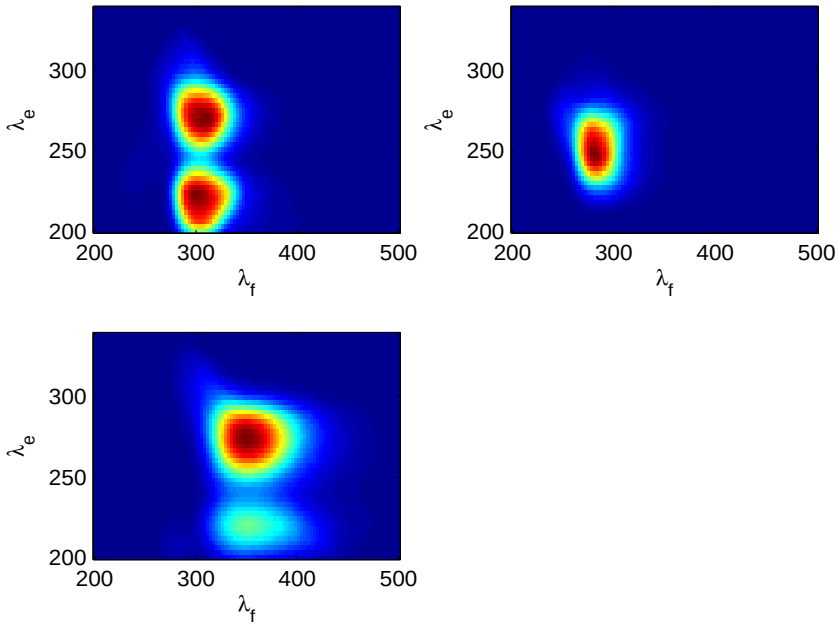


FIGURE 6.10 – Les MEEF des composés organiques de référence présents dans le mélange. En haut à gauche MEEF de la tyrosine. En haut à droite, MEEF de la phénylalanine. En bas, MEEF du tryptophane.

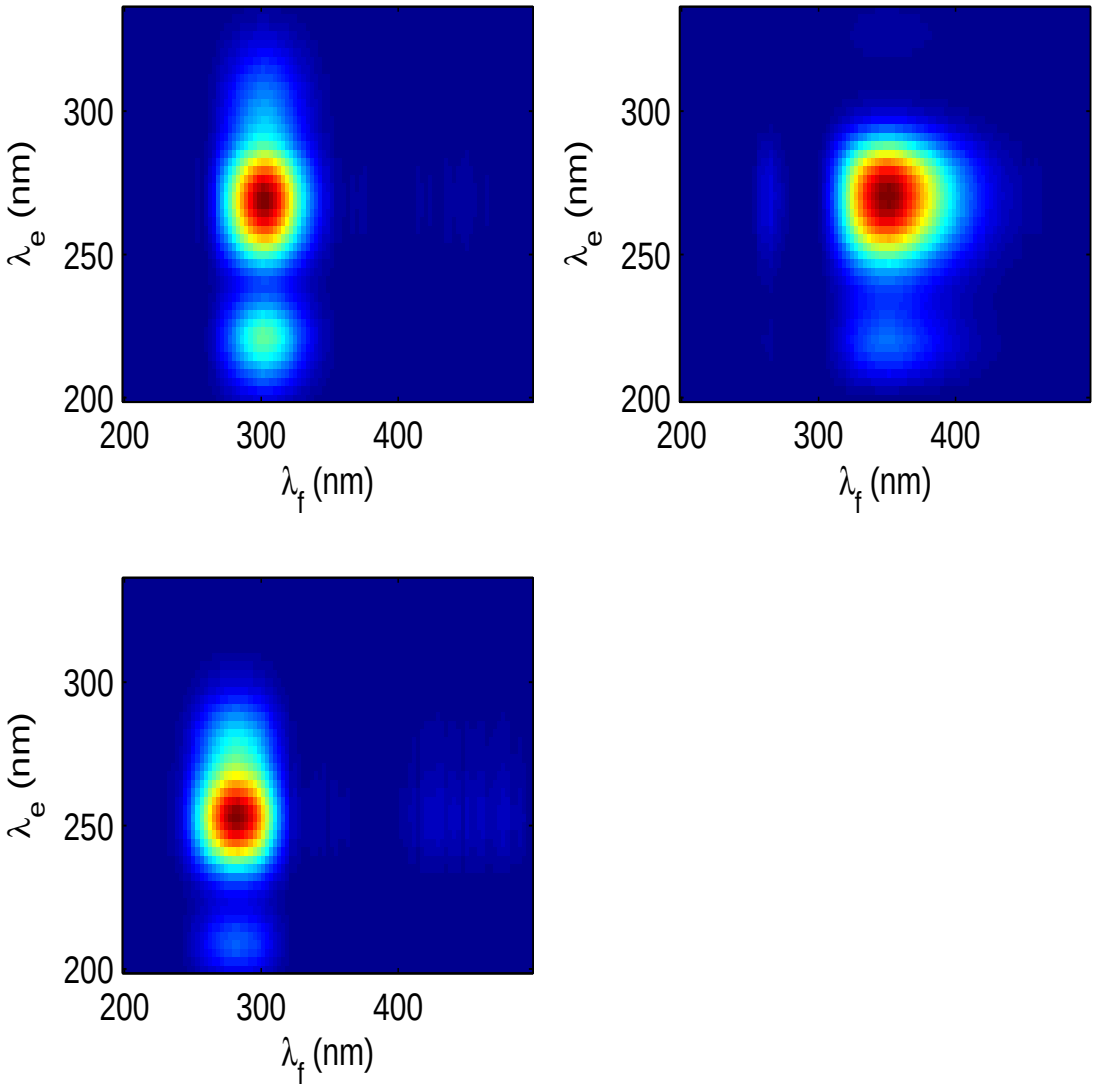


FIGURE 6.11 – Les MEEF estimées par gradient conjugué non négatif. En haut à gauche MEEF de ce que l'on estime être la tyrosine. En haut à droite, MEEF de ce que l'on estime être le tryptophane. En bas, MEEF de ce que l'on estime être la phénylalanine.

Sur la figure (6.12), on trace les MEEF des composés dans le cas où on surévalue le rang du tenseur de données. La décomposition est donc réalisée en prenant $F = 4$. On utilise pour ce faire le gradient conjugué sous contrainte de non négativité, avec une pénalisation L_1 (coefficient de pénalisation 10^{-2}.

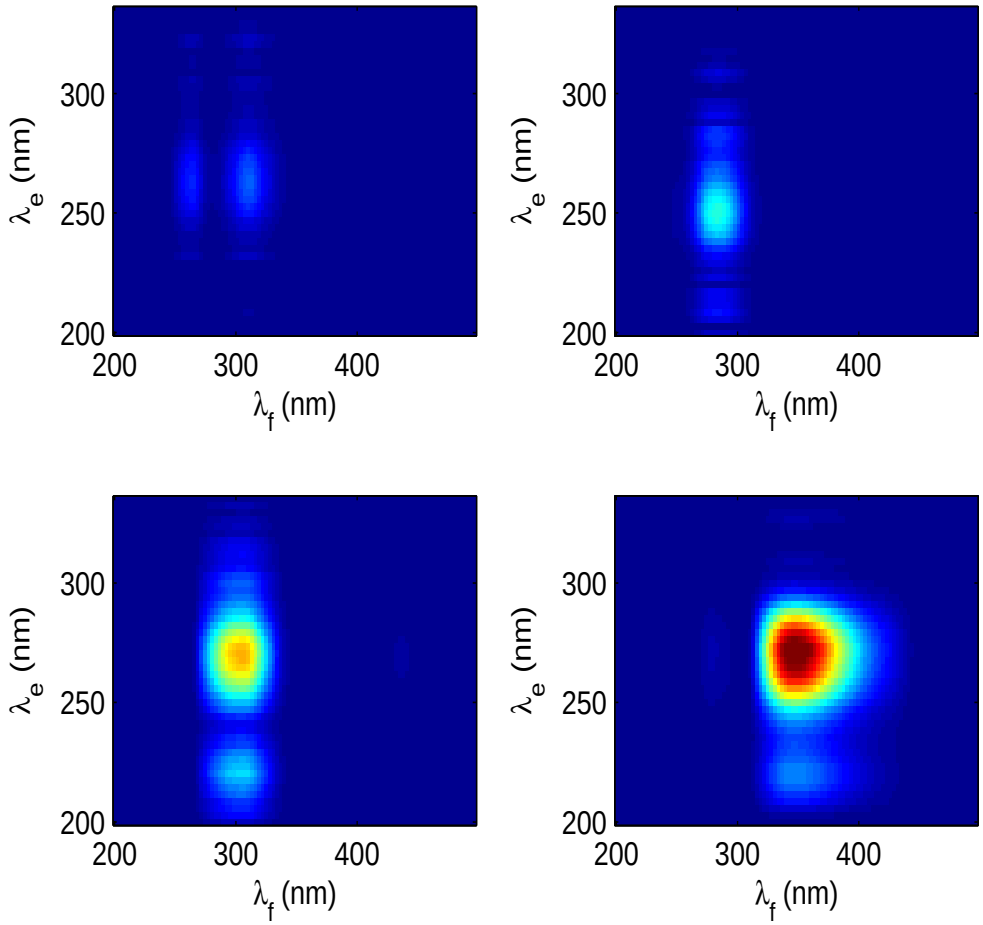


FIGURE 6.12 – Surestimation du rang du tenseur en prenant $F = 4$ au lieu de 3. En haut à droite, on reconnait la phénylalanine. En bas à gauche, la tyrosine. En bas bas à droite, le tryptophane. La MEEF en haut à gauche est logiquement d'intensité plus faible que les autres puisqu'elle ne correspond à aucun des composés du mélange.

On compare également avec la décomposition CP non contrainte via ALS + LS , laquelle est implémentée dans [BA]. Les MEEF reconstituées sur la figure 6.13 présentent des valeurs négatives. Les matrices de facteurs sont initialisées positivement et aléatoirement suivant une loi uniforme. Sur la figure 6.14, on a également tracé l'évolution des performances en fonction des itérations : pour ce faire on utilise l'indice qui a été défini au niveau de l'équation 3.109. Comme on pouvait s'y attendre par une observation visuelle des MEEF de la figure 6.13, le gradient conjugué sous contrainte de non négativité atteint de meilleures performances que l'ALS + LS sans contrainte.
Pour se rendre compte des erreurs d'estimation en utilisant un algorithme sans contrainte, on trace les spectres des matrices de facteurs sur la figure 6.15. En plus de décalages notoires du pic de fluorescence sur la phenylalanine (excitation), la tyrosine (excitation) et le tryptophane (dans une moindre mesure), on observe que ces spectres prennent des valeurs négatives pour la

phenylalanine (excitation et émission), ainsi que le tryptophane (en émission très nettement, et en excitation pour une valeur).
Pour finir, on trace les concentrations estimées par ALS + LS sur la figure 6.16. Elles sont moins bien évaluées que sur la figure 6.9. En effet, les deux premières valeurs de la tyrosine, qui devraient être les mêmes (on rappelle que le ratio est 1/1/2 pour ce composé), sont très éloignées (0.45 au lieu de 0.6). C'est la même chose pour les deux dernières valeurs du triptophane (estimées à 0.41 et 0.45). La phénylalanine est en revanche très légèrement mieux estimée.

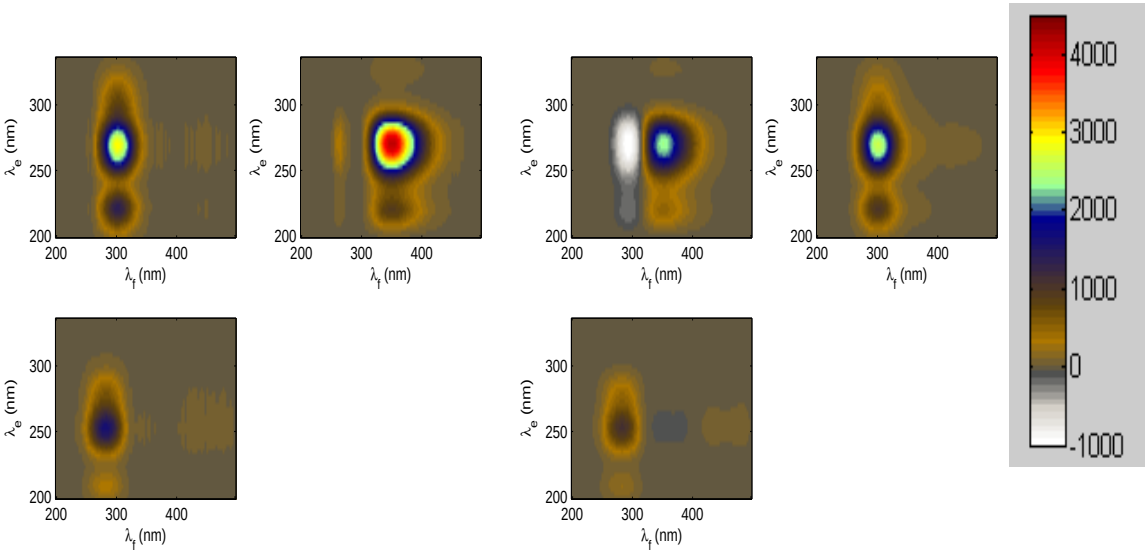


FIGURE 6.13 – Comparaison des MEEF estimées sur le jeu à 3 fluorophores au moyen des deux algorithmes suivants : gradient conjugué sous contrainte de non négativité (à gauche), ALS + LS sans contrainte de positivité (à droite).

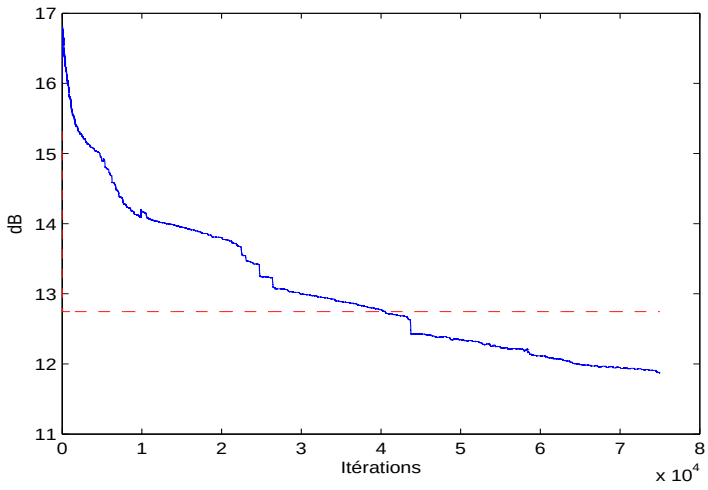


FIGURE 6.14 – Indice de performances $E_{2\text{dB}}$ sur le mélange à 3 fluorophores. Le gradient conjugué est en bleu, l'ALS + LS en rouge

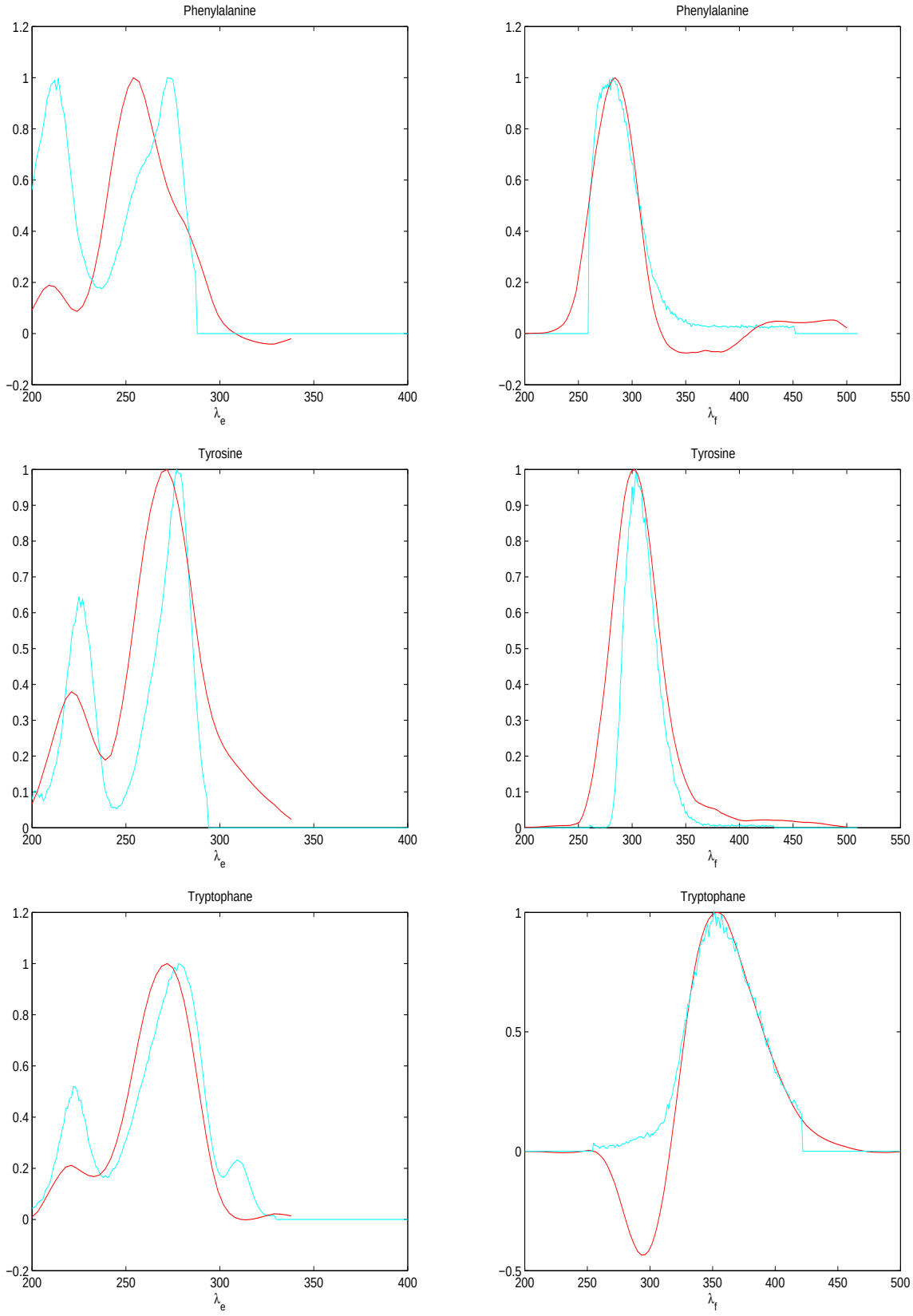


FIGURE 6.15 – Spectres estimés après prétraitement par filtrage morphologique via ALS + LS sans contrainte. A gauche sont représentés tous les spectres d'excitation, et à droite les spectres d'émission. De haut en bas : la phénylalanine, la tyrosine et le tryptophane. Sur chaque figure, la courbe rouge correspond au spectre estimé par décomposition CP, la courbe bleue au spectre de référence et la courbe cyan au spectre de référence sur lequel on a tronqué la zone qui correspondait aux raies de diffusion pour plus de lisibilité.

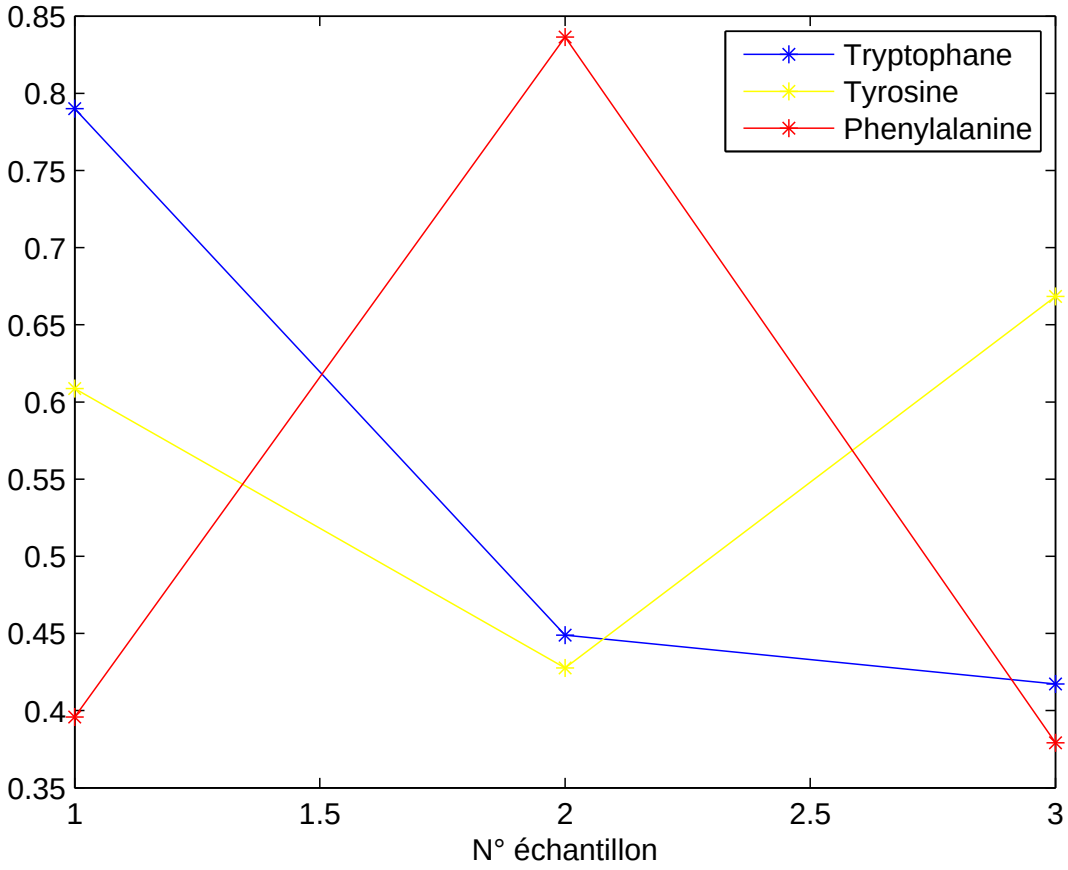


FIGURE 6.16 – Concentrations estimées par ALS + LS

6.1.3 Jeu de données à 2 fluorophores

On réalise maintenant l'analyse du premier jeu de données à 2 fluorophores. On dispose dans ce nouvel exemple d'un tenseur de taille $71 \times 46 \times 5$. On réalise la décomposition CP via l'algorithme de gradient conjugué non négatif que nous avons proposé au chapitre 3. Là encore, l'algorithme est initialisé au moyen de l'algorithme DTLD suggéré dans [SK90].
La figure (6.17) présente les spectres d'excitation et d'émission estimés sur la fluorescéine et sur le sulfate de quinine. Les amplitudes ont été normalisées par la valeur maximale afin de pouvoir comparer les courbes plus facilement. En émission, les estimations sont très proches de la référence. C'est encore plus vrai pour le sulfate de quinine, où la reconstruction est quasiment parfaite. En émission, on observe un très léger décalage du pic (5 nm) correspondant à la fluorescéine. Le sulfate de quinine présente une légère distorsion. Son pic est bien estimé, mais son estimation est plus grossière.
Les concentrations sont représentées sur la figure (6.18). Les valeurs ont été réajustées afin de se cadrer sur une échelle $[0 - 90]$ de concentrations relatives. Rappelons que les rapports de concentrations choisis sont de 10, 30, 50, 70 et 90. La courbe obtenue pour la fluorescéine est pratiquement linéaire et est donc très proche de ce qu'on attend (la $3^{\text{ème}}$ valeur est la

plus éloignée et vaut 47 au lieu de 50). La courbe du sulfate de quinine donne d'aussi bons résultats (la $4^{\text{ème}}$ valeur vaut 65 au lieu de 70).

La figure (6.19) montre les MEEF des 2 composés organiques estimés à partir de la décomposition CP (algorithme du gradient conjugué non négatif). A gauche, nous présentons la MEEF de ce que nous estimons être la fluorescéine, et à droite la MEEF de ce que nous estimons être du sulfate de quinine.

Un exemple des résultats obtenus en surestimant le nombre de fluorophores présents dans la solution est présenté au niveau de la figure (6.20). Les 2 composés sont reconnaissables, alors que la $3^{\text{ème}}$ image est pratiquement nulle. La méthode que nous avons proposée s'est avérée robuste vis-à-vis d'erreur de modèle (rang du tenseur qui correspond au nombre de composés dissous présents) dans ce cas là.

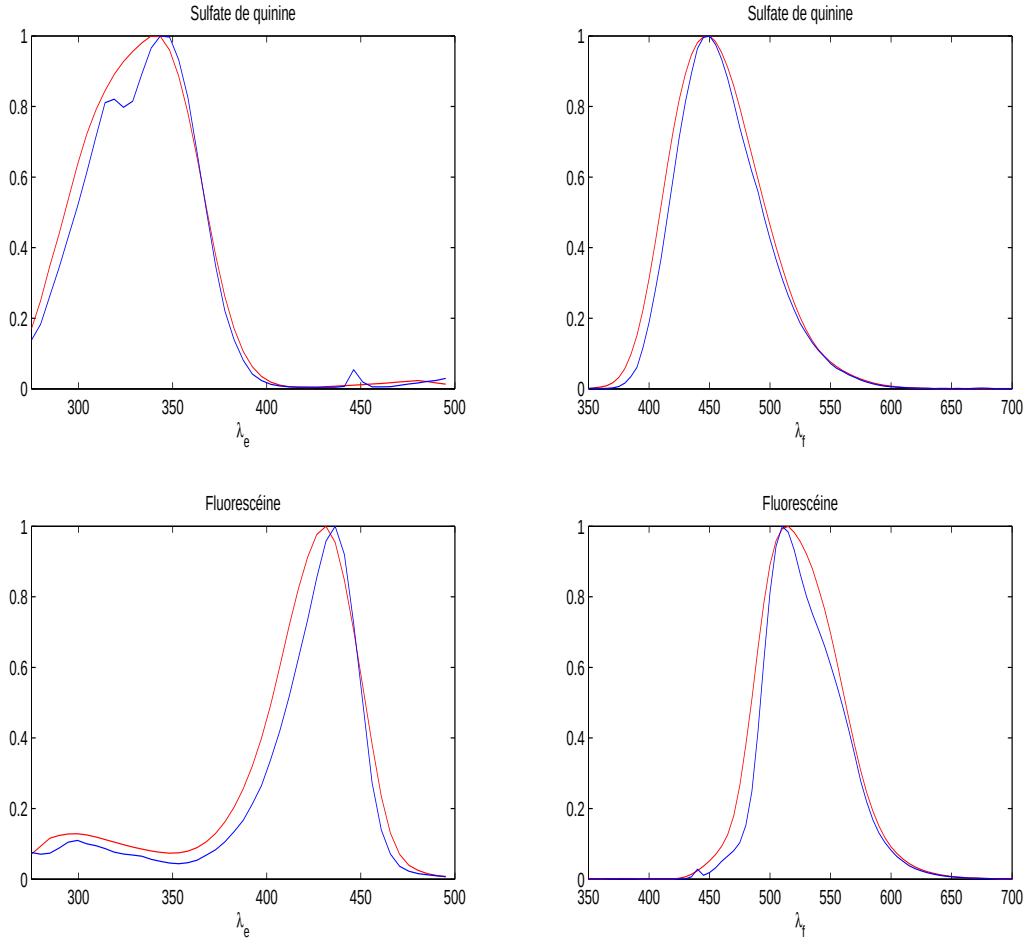


FIGURE 6.17 – En rouge, les spectres estimés par décomposition CP non négative (algorithme du gradient conjugué). En bleu, les spectres de référence. En haut, les spectres du sulfate de quinine, respectivement l'excitation à gauche et l'émission à droite. En bas, les spectres de la fluorescéine, respectivement l'excitation à gauche et l'émission à droite.

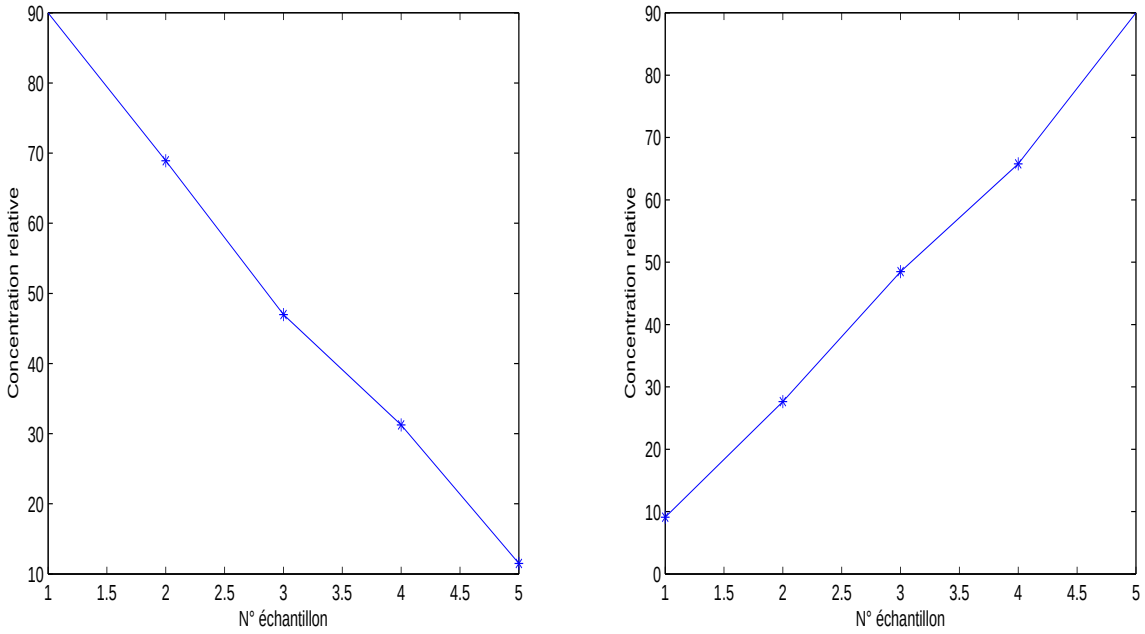


FIGURE 6.18 – Evolution des concentrations relatives au fil des échantillons. A gauche, la fluorescéine, à droite, le sulfate de quinine.

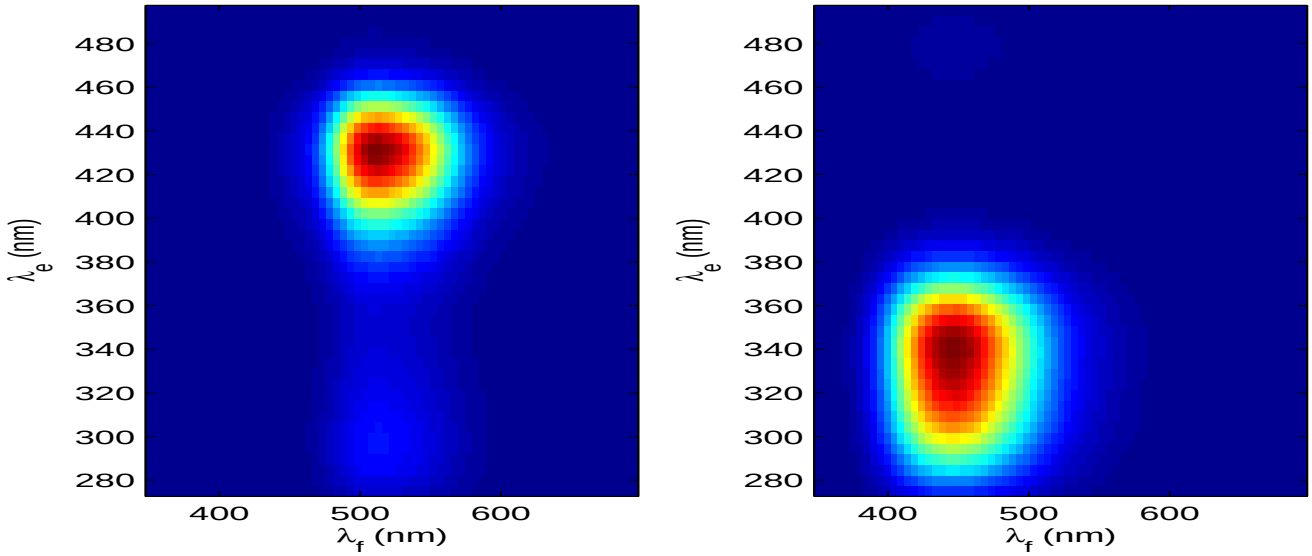


FIGURE 6.19 – MEEF des 2 composés organiques. A gauche, la fluorescéine, à droite, le sulfate de quinine.

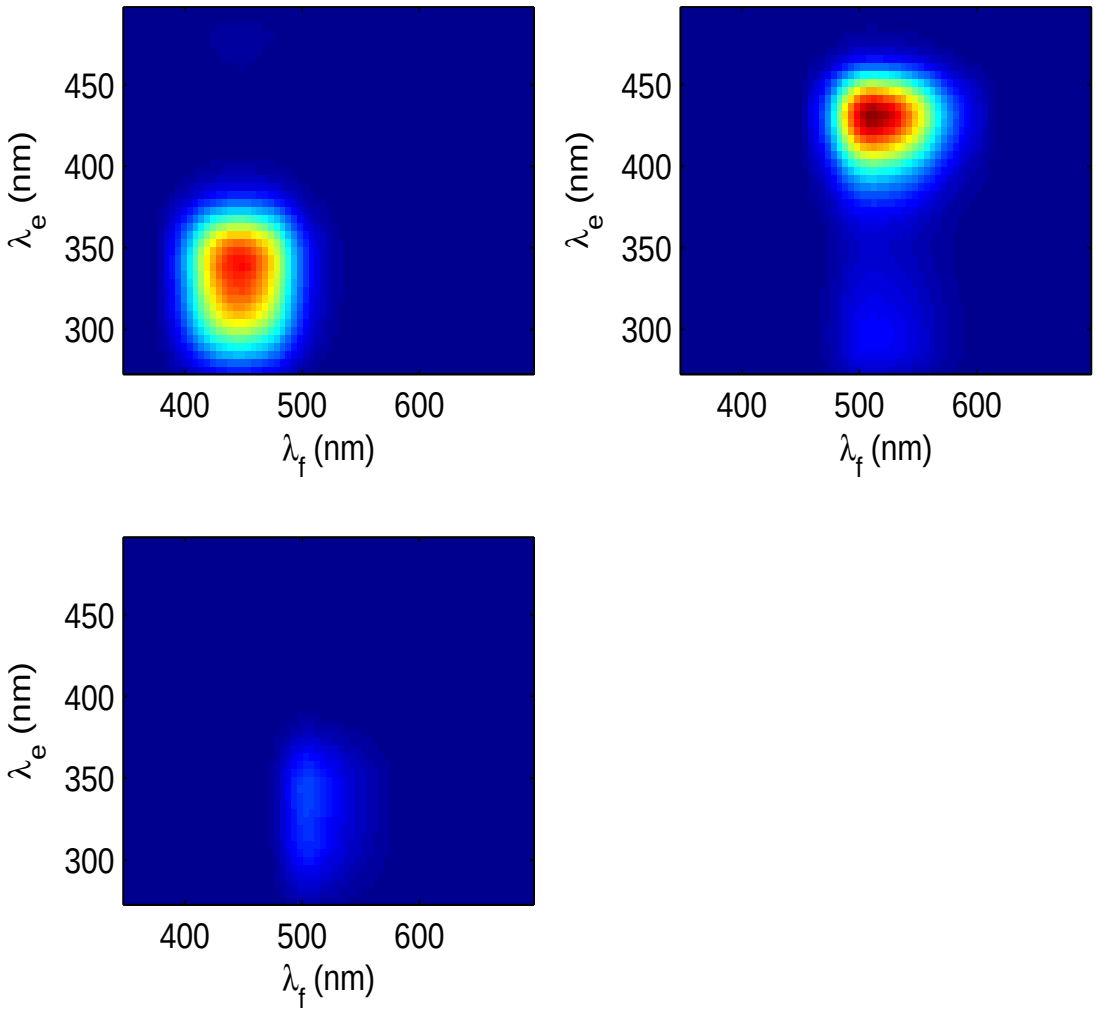


FIGURE 6.20 – 3 composés estimés pour 2 réellement présents.

6.1.4 Nouveau jeu à 2 fluorophores

On réalise maintenant l'analyse du jeu de données à 2 fluorophores. On dispose dans ce dernier cas d'un tenseur de taille $81 \times 71 \times 6$.

On présente tout d'abord les résultats sur les spectres relatifs. Les références sont obtenues en sélectionnant les spectres d'émission et d'excitation présents aux longueurs d'onde du maximum d'intensite des MEEF de chaque composé pur. La figure 6.21 présente les spectres d'excitation et d'émission estimés sur la fluorescéine et sur le sulfate de quinine. Les amplitudes ont été normalisées par la valeur maximum afin de comparer sur la même échelle de valeur. Les estimations sont quasiment confondues avec la référence, à l'exception du pic secondaire de la fluorescéine en excitation, qui n'est pas très bien reconstruit.

Les concentrations sont représentées sur la figure 6.22. Rappelons que les rapports de concen-

trations choisis sont, pour l'ensemble des 6 échantillons : 33 / 66 / 100 / 33 / 66 / 100 pour le sulfate de quinine, ainsi que 66/66/66/100/100/100 pour la fluorescéine. La courbe obtenue pour la fluorescéine présente 2 paliers comme attendu. La 5ème valeur est un peu en deçà de celle attendue, avec 0.28 au lieu de 0.31. Le ratio entre les 2 paliers n'est par contre pas respecté. Il devrait être de $\frac{100}{66} \approx 1.52$, alors qu'il est ici d'un facteur proche de 10. La courbe du sulfate de quinine présente bien 3 paliers de ratios corrects, aux alentours de 0.09, 0.18 et 0.25.
La figure (6.23) montre les MEEF des 2 composés organiques réestimés à partir de la décomposition CP. A gauche, la fluorescéine, et à droite, le sulfate de quinine. On peut comparer cette bonne estimation aux MEEF de référence des composés purs qui sont fournis au niveau de la figure (6.24).
On réalise aussi la décomposition en surestimant le rang du tenseur : rang 3 au lieu de 2. Les résultats obtenus sont présentés figure 6.25. La sulfate de quinine et la fluorescéine sont identifiables, alors que la 3ème image est d'intensité négligeable, comme attendu.

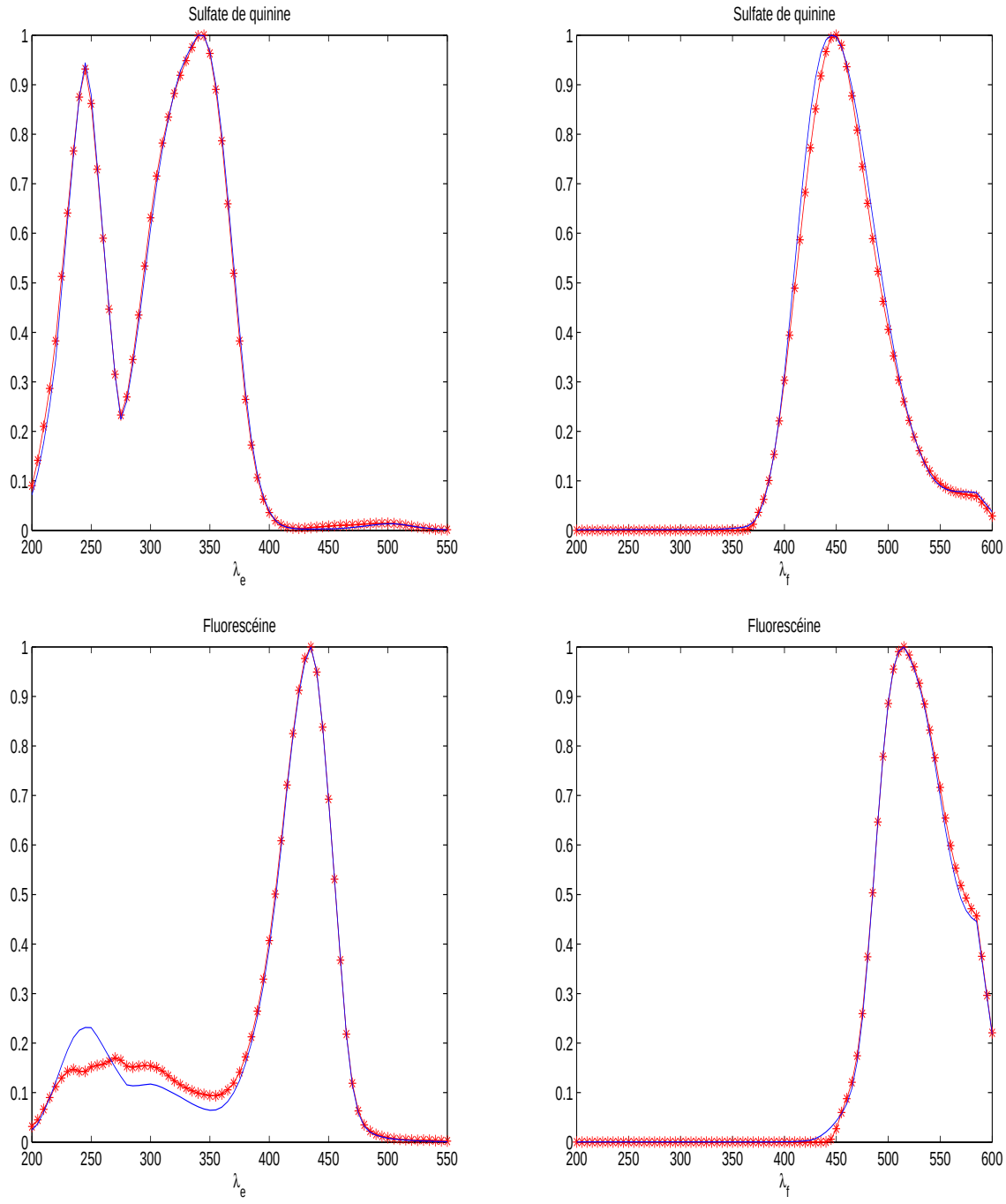


FIGURE 6.21 – En rouge, les spectres estimés par décomposition CP non négative (algorithme du gradient conjugué). En bleu, les références. En haut, les spectres du sulfate de quinine, respectivement l'excitation à gauche et l'émission à droite. En bas, les spectres de la fluorescéine, respectivement l'excitation à gauche et l'émission à droite.

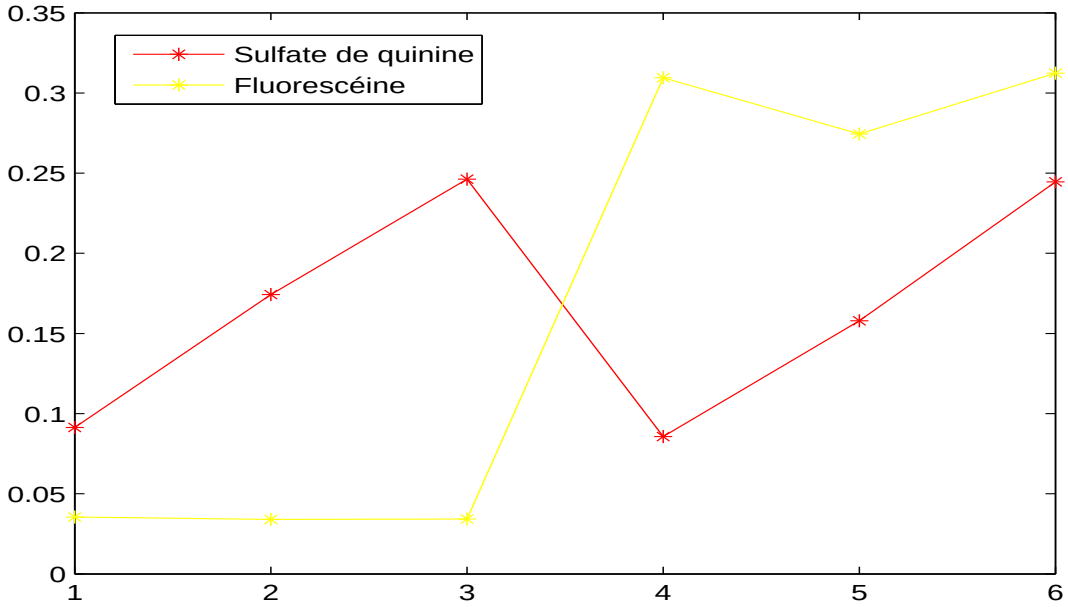


FIGURE 6.22 – Evolution des concentrations relatives de la fluorescéine et du sulfate de quinine au fil des échantillons.

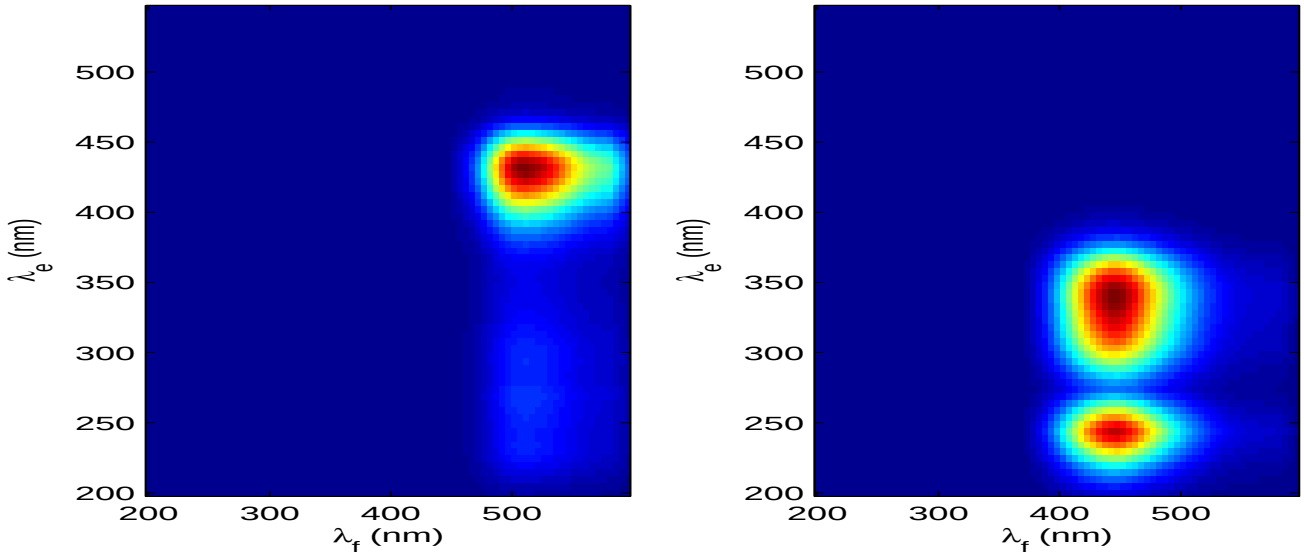


FIGURE 6.23 – MEEF des 2 composés organiques. A gauche, la fluorescéine, à droite, le sulfate de quinine.

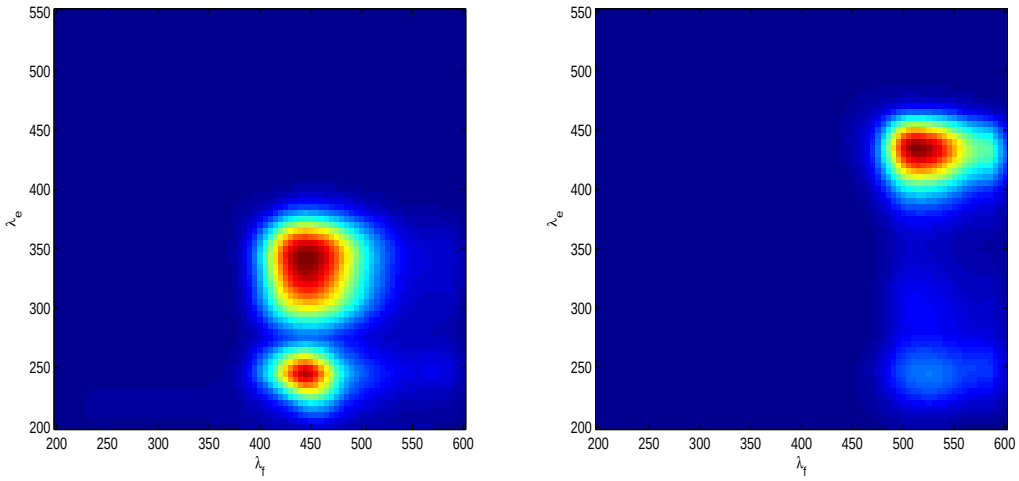


FIGURE 6.24 – MEEF de référence du sulfate de quinine (gauche) et de la fluorescéine (droite).

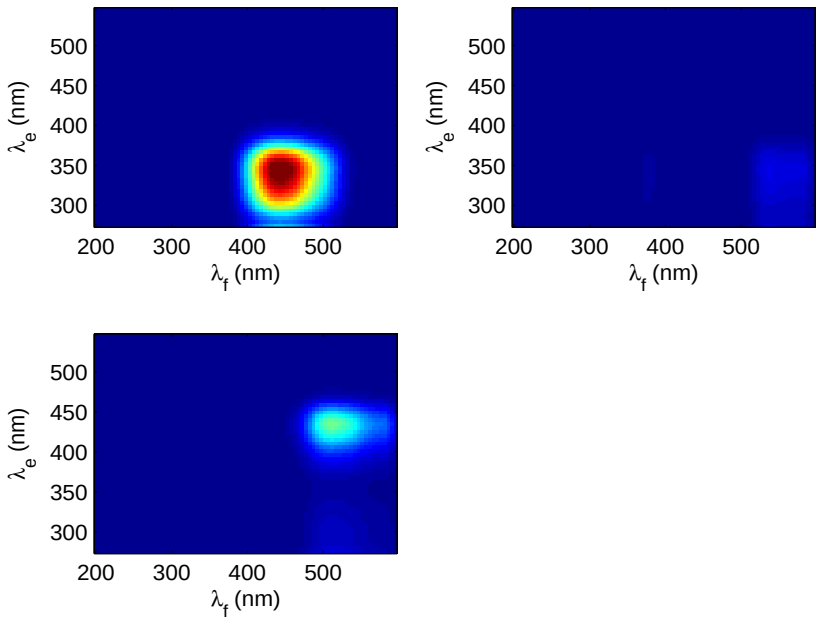


FIGURE 6.25 – 3 composés estimés pour 2 réellement présents.

6.1.5 Conclusion

A travers 3 jeux de données contenant des composés connus en concentration connue, nous avons pu valider nos algorithmes sur des mélanges réels.
La première étape nous a menés à montrer l'importance du prétraitement pour éliminer les raies de diffusion Raman et Rayleigh. A ce sujet, nous avons pointé que l'algorithme décrit par Zepp et classiquement utilisé, n'était pas toujours le meilleur choix. Pire, il peut mener

à une altération de l'information contenue dans les MEEF . Par conséquent, bien avant d'appliquer la correction des éventuels effets d'écran, ou l'optimisation des matrices de facteurs, il devient déjà impossible de retrouver ces dernières, du fait que le modèle trilinéaire a été altéré. Nous avons alors montré une méthode de correction par filtrage morphologique, qui permet davantage de conserver l'intégrité de l'information de fluorescence, quand celle-ci est traversée par les raies de diffusion.
L'application des algorithmes développés dans la section 3 à ces 3 jeux de données donne de très bons résultats. Nous avons mis en évidence l'apport de la prise en compte de la non négativité pour la réestimation des matrices de facteurs. Nos algorithmes ont été capables de reconstruire les spectres relatifs en étant très proches des références surtout sur les mélanges à 2 fluorophores, comportant moins de difficultés (pics de fluorescence plus éloignés les uns des autres, prétraitement trivial). Les concentrations ont également été très bien évaluées sur les différents sets.
Pour finir, les MEEF reconstruites lorsque l'on surestime le rang du tenseur de données donnent des résultats très satisfaisants sur ces mélanges réels contrairement aux algorithmes de la littérature. Nous avions déjà montré dans la section 3 que c'était le cas sur des cubes de données totalement synthétiques. Ici, les algorithmes proposés se montrent très robustes lorsqu'on introduit des erreurs de modèle de type surestimation du rang.

Conclusions & perspectives de recherche

Conclusions

Ce manuscrit s'est focalisé sur la décompostion CP de tenseurs du 3ème ordre non négatifs. Nous avons présenté des méthodes de paramétrisation permettant d'obtenir des matrices de facteurs positives.
Nous avons tout d'abord rappelé quelques notions d'algèbre tensorielle, et mis en évidence les problèmes de dégénérescence des colonnes des matrices de facteurs, issus des décompositions CP non contraintes. En poursuivant sur l'application des décompositions tensorielles en fluorescence, via la loi de Beer-Lambert, nous avons mis en lumière le fait que le tenseur de données est positif, et que de part leur nature (spectres et concentrations), les matrices de facteurs ne doivent pas contenir de valeurs négatives. Ceci nous a amenés à présenter diverses approches permettant de prendre en compte cet aspect.
Il existait déjà plusieurs méthodes. Après avoir brièvement rappelé celles utilisées dans le cas de la factorisation de matrices non négatives, c'est à dire NMF, nous avons présenté les méthodes non négatives de factorisation tensorielle. La plus simple consiste à mettre à 0 les valeurs négatives durant le processus itératif d'estimation, mais cette façon de faire contrarie le processus de descente vers la solution voulue. Nous avons ensuite évoqué le NNLS, un algorithme itératif basé sur la résolution d'un système vectorisé, puis des variantes de l'ALS. Nous avons ensuite décrit nos propres méthodes. Ainsi, nous avons proposé en premier lieu une paramétrisation des matrices de facteurs, de sorte à ce que chaque élément soit écrit sous la forme d'un carré. Nous avons ensuite développé une approche par régularisation exponentielle, qui incite la fonction de coût à écarter les valeurs négatives. Ces algorithmes ont été appliqués à des méthodes de descente classiques, pour lesquelles nous proposons des versions avec un pas d'adaptation dit optimal, qui est celui minimisant la fonction de coût à l'itération considérée, mais qui présente une charge de calcul élevée, et un pas d'adaptation dit approché, très rapide à calculer. Ces algorithmes ont été validés sur des exemples entièrement synthétiques de spectroscopie de fluorescence, dans lesquels concentrations et MEEF sont connus. Ces simulations ont permis de montrer que la paramétrisation par produit de Hadamard s'avère très robuste quand on surestime le rang du tenseur, davantage que celles basées sur des projections et faisant intervenir de l'ALS, et nous permet d'entrevoir des perspectives intéressantes pour déterminer le nombre de composés présents. Nous avons conclu de cette étude que le gradient conjugué utilisant notre approche offrait le meilleur compromis performances / temps. En effet, les algorithmes de Quasi-Newton peuvent s'avérer trop coûteux lorsque l'on considère de larges tenseurs et se montrent moins robustes lors de la surestimation du rang. La régula-

risation exponentielle, quant à elle, fait converger la méthode très rapidement et offre de bons résultats à rang exact. Par contre, elle ne fait pas beaucoup mieux que les décompositions basées sur de l'ALS lorsque le nombre de composants est surestimé.
Dans la suite, nous avons pris en compte le problème des données manquantes qui peuvent survenir durant le processus d'acquisition des valeurs. Après avoir présenté les méthodes existantes, basées sur des poids binaires affectés au tenseur de données, nous avons développé la nôtre, basée sur des poids variables au fil des itérations et faisant intervenir une reconstruction du tenseur, toujours dans la continuité de la non négativité et de sa prise en compte. Nous avons validé le bon comportement de la méthode par rapport à celles existantes, et par rapport à une nouvelle généralisation de ces dernières dans le cadre de la non négativité.
Plus tard, nous avons, après avoir fait l'étude de la complexité algorithmique pour les différentes méthodes proposées et pour le calcul du pas optimal, proposé des améliorations algorithmiques pour réduire le temps de calcul par itération.
Dans une dernière partie, nous avons appliqué les méthodes développées ici dans le cadre de mélanges réalisés en laboratoire, et dont on connait les composés présents et leur concentration au fil des échantillons. Nous avons également présenté une méthode de prétraitement permettant d'éliminer les raies de diffusion Raman et Rayleigh de façon plus efficace, lorsque ces dernières traversent un pic de fluorescence. Nous avons montré que nos algorithmes se montrent très robustes et donnent les meilleurs résultats pour réestimer spectres et composés, à la fois à rang exact, et aussi lorsque l'on surestime le nombre de composés à obtenir.

Perspectives de recherche

Prise en compte de la parcimonie Les tenseurs que nous avons considérés contiennent une part relativement importante de valeurs nulles ou même faibles. On pourrait se servir de cette propriété afin d'accélérer les temps de calcul en ne prenant pas en compte ces données qui contiennent pas ou moins d'informations que les autres. Des optimisations de calculs prenant en compte la parcimonie existent. Le principe est de ne conserver que les valeurs non nulles des matrices. On peut alors redéfinir les opérations de base, comme le produit matriciel, qui sont considérablement plus rapides en présence de très forts taux de données nulles. A ce sujet, on peut envisager de seuiller fortemement le tenseur de données afin de limiter le nombre de valeurs non nulles, tout en gardant les pics contenant le maximum d'information, dans le but d'obtenir une première approximation dont on se resservirait comme initialisation.

Faire du traitement adaptatif Il est parfois utile, au moins dans l'application en chimiométrie considérée, de faire suivi en temps réel des données recueillies, dans le but de détecter un changement brutal de composition des échantillons d'eau par exemple (pollution...). Cela amène à considérer le temps de façon complète, et à développer des algorithmes adaptatifs prenant en compte cet aspect, notamment pour des question de rapidité de calcul (les MEEF ne changeant que très peu de l'une à l'autre en temps normal).

Liste de publications

Revues internationales avec comité de lecture (1 + 2 en préparation) :

[A1] J.-P. Royer, N. Thirion-Moreau, P. Comon "Computing the polyadic decomposition of nonnegative third order tensors", accepted in *Signal Processing*, Elsevier, Vol. 91, Issue 9, pp 2159-2171, September 2011.

Conférences internationales avec actes et comité de lecture (4 + 2 soumises) :

[C1] J.-P. Royer, P. Comon, N. Thirion-Moreau, "Computing the nonnegative 3-way tensor factorization using Tikhonov regularization", in Proc. International Conference on Acoustic Speech and Signal Processing (*ICASSP'2011*), pp. 2732-2735, Prague, République Tchèque, May 2011.

[C1] J.-P. Royer, P. Comon, N. Thirion-Moreau, "Nonnegative 3-way tensor factorization via conjugate gradient with globally optimal stepsize", in Proc. International Conference on Acoustic Speech and Signal Processing (*ICASSP'2011*), pp. 4040-4043, Prague, République Tchèque, May 2011.

[C1] J.-P. Royer, N. Thirion-Moreau, P. Comon, "Nonnegative 3-way tensor factorization taking into account possible missing data", Proc. European Signal Processing Conference (*EUSIPCO'2012*), pp. , Bucharest, Romania, August 2012.

[C1] D. Sylla, J.-P. Royer, A. Minghelli-Roman, N. Thirion-Moreau and A. Mangin, "Images fusion based on a coupled nonnegative tensors factorization approach (CNTF), application to OLCI and ETM sensors", accepted in Proc. IEEE workshop on hyperspectral image and signal processing (*WHISPERS'2013*), pp. , Gainsville, Florida, USA, 25-28 June 2013.

[C1] J.-P. Royer, R. Redon, N. Thirion-Moreau, P. Comon "An improved preprocessing of FEEMs coupled with a nonnegative Canonical Polyadic Decomposition (PARAFAC) algorithm", International Workshop on Organic Matter Spectroscopy (*WOMS'2013*), pp. , Toulon, Var, France, 16-19 July 2013.

[C1] I. Kopriva, J.-P. Royer, N. Thirion-Moreau, P. Comon, "Error analysis of tensor factorization approach to blind source separation", submitted to International Workshop on Organic Matter Spectroscopy (*WOMS'2013*), pp. , Toulon, Var, France, 16-19 July 2013.

Conférences internationales sans actes (1) :

[M2] J.-P. Royer, N. Thirion-Moreau and P. Comon, "3-way nonnegative polyadic decomposition, computation and application", Trilateral workshop on Separation of Variables and Applications (SVA), La Colle-sur-Loup, 8-10 September, 2010.

Colloques avec actes sans comité de lecture (1)

[M3] J.-P. Royer, P. Comon, N. Thirion-Moreau, S. Mounier, R. Redon, H. Zhao, G. Féraud, C. Potot, "Water analysis with the help of tensor canonical decompositions" in Actes du Colloque International RUNSUD 2010 sur les conditions d'une coopération universitaire et scientifique franco-vietnamienne dans le contexte d'une société globale de la connaissance universitaire, pp. 407-410, Nice - Sophia Antipolis, 23-25 mars 2010.

Bibliographie

[AKDM11] E. Acar, T. G. Kolda, D. M. Dunlavy, and M. M. Mørup, *Scalable tensor factorizations for incomplete data*," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, pp. 41–56, january 2011.

[AMDDM60] A. Amir-Moez, C. Davis, D. M. Dunlavy, and M. M. Mørup, *Generalized frobenius inner products*," *Math Annalen*, vol. 141, pp. 1070–112, 1960.

[AY09] E. Acar and B. Yener, *Unsupervised multiway data analysis : a literature survey*," *IEEE Transactions on Knowledge and data engineering*, vol. 21, no. 1, pp. 6–20, january 2009.

[BA] R. Bro and C. A. Andersson. N-way toolbox for matlab. [Online]. Available : http ://www.models.life.ku.dk/nwaytoolbox

[BBL+07] M. Berry, M. Browne, A. Langville, V. Pauca, and R. Plemmons, *Algorithms and applications for approximate nonnegative matrix factorization*," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 155–173, Sep. 2007. [Online]. Available : http ://dx.doi.org/10.1016/j.csda.2006.11.006

[BCA12] H. Becker, P. Comon, and L. Albera, *Tensor-Based Preprocessing of Combined EEG/MEG Data*," in *EUSIPCO-2012*, Eurasip, Ed. Bucarest, Roumanie : Elsevier, Aug. 2012, pp. 1–5, 5 pages. [Online]. Available : http ://hal.archives-ouvertes.fr/hal-00725280

[BD97] R. Bro and S. De Jong, *A fast non-negativity-constrained least squares algorithm*," *Journal of Chemometrics*, vol. 11, pp. 393–401, 1997.

[Bre78] J. Brewer, *Kronecker products and matrix calculus in system theory*," *IEEE Trans. on Circuits and Systems*, vol. Cas-25, no. 9, pp. 772–781, September 1978.

[Bro97] R. Bro, *Parafac : tutorial and applications*," *Chemometr. Intell. Lab.*, vol. 38, pp. 149–171, 1997.

[Bro98] ——, *Multi-way analysis in the food industry : models, algorithms and applications*," Ph.D. dissertation, University of Amsterdam, Amsterdam, The Netherlands, 1998.

[BV04] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, Mar. 2004.

[CC70] J. Carroll and J.-J. Chang, *Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition*," *Psychometrika*, vol. 35, no. 3, pp. 283–319, September 1970. [Online]. Available : http ://ideas.repec.org/a/spr/psycho/v35y1970i3p283-319.html

[CLDA09] P. Comon, X. Luciani, and A. L. F. De Almeida, *Tensor decompositions, alternating least squares and other tales*," *Jour. Chemometrics*, vol. 23, pp. 393–405, Aug. 2009.

[CMI12] E. Chouzenoux, S. Moussaoui, and J. Idier, *Majorize-minimize linesearch for inversion methods involving barrier function optimization*," *Inverse Problems*, vol. 28, p. 065011 (24 pages), oct. 2012.

[Cob96] P. G. Coble, *Characterization of marine and terrestrial dom in seawater using excitation-emission matrix spectroscopy*," *Marine Chemistry*, vol. 52, pp. 325–346, 1996.

[Com09] P. Comon, *Tensors, usefulness and unexpected properties*," in *IEEE Workshop on Statistical Signal Processing (SSP'09)*, Cardiff, UK, Sep. 1-3 2009, keynote.

[CZPA09] A. Cichocki, R. Zdunek, A. H. Phan, and S. I. Amari, *Non negative matrix and tensor factorizations : Application to exploratory multi-way data analysis and blind separation.* Wiley, 2009.

[dAFM07] A. L. F. de Almeida, G. Favier, and J. a. C. M. Mota, *Parafac-based unified tensor modeling for wireless communication systems with application to blind multiuser equalization*," *Signal Process.*, vol. 87, no. 2, pp. 337–351, Feb. 2007. [Online]. Available : http ://dx.doi.org/10.1016/j.sigpro.2005.12.014

[FBD09] C. Févotte, N. Bertin, and J.-L. Durrieu, *Nonnegative matrix factorization with the itakura-saito divergence : With application to music analysis*," *Neural Comput.*, vol. 21, no. 3, pp. 793–830, Mar. 2009. [Online]. Available : http ://dx.doi.org/10.1162/neco.2008.04-08-771

[FG91] A. S. Field and D. Graupe, *Topographic component (parallel factor) analysis of multichannel evoked potentials : practical issues in trilinear spatiotemporal decomposition*," *Brain topography*, vol. 3, no. 4, pp. 407–423, 1991.

[Fra92] A. Franc, *Etude algébrique des multi-tableaux : apport de l'algèbre tensorielle*," PhD thesis, University of Montpellier II, Montpellier, France, 1992.

[GMB+11] X. Guo, S. Miron, D. Brie, S. Zhu, and X. Liao, *A candecomp/parafac perspective on uniqueness of doa estimation using a vector sensor array.*" 2011, pp. 3475–3481.

[GTMMA10] H. Ghennioui, N. Thirion-Moreau, E. Moreau, and D. Aboutajdine, *Gradient based joint block diagonalization algorithms : application to blind separation of fir convolutive sources mixtures*," *Eurasip Signal Processing*, vol. 90, no. 6, pp. 1836–1849, June 2010, doi :10.1016/ j.sigpro.2009.12.002.

[GW01] R. Gonzalez and R. Woods, *Digital image processing*, P. Hall, Ed. Pearson Educational International, 2001, vol. Second Edition.

[Har70] R. A. Harshman, *Foundation of the Parafac procedure : models and conditions for an explanatory multimodal factor analysis*," *UCLA Working papers in phonetics*, vol. 16, pp. 1–84, 1970.

[HD04] P. O. Hoyer and P. Dayan, *Non-negative matrix factorization with sparseness constraints*," *Journal of Machine Learning Research*, vol. 5, pp. 1457–1469, 2004.

[Hit27] F. L. Hitchcock, *The expression of a tensor or a polyadic as a sum of products*," *J. Math. and Phys.*, vol. 6, pp. 165–189, 1927.

[HL84] R. A. Harshman and M. E. Lundy, *The PARAFAC model for three-way factor analysis and multidimensional scaling*," *H. G. Law, C.W. Snyder Jr., J. Hattie, and R. P. McDonald*, vol. 16, 1984.

[JWLY99] J. H. Jiang, H. L. Wu, Y. Li, and R. Q. Yu, *Alternating coupled vectors resolution (acover) method for trilinear analysis of three-way data*," *Journal of Chemometrics*, vol. 13, no. 6, pp. 557–578, 1999.

[KB09] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*," *Siam Review*, vol. 51, no. 3, pp. 455–500, september 2009.

[KC09] I. Kopriva and A. Cichocki, *Blind multispectral image decomposition by 3d nonnegative tensor factorization*," *Opt. Lett.*, vol. 34, no. 14, pp. 2210–2212, Jul 2009. [Online]. Available : http ://ol.osa.org/abstract.cfm ?URI=ol-34-14-2210

[KHL89] J. B. Kruskal, R. A. Harshman, and M. E. Lundy, *Multiway data analysis*," R. Coppi and S. Bolasco, Eds. Amsterdam, The Netherlands, The Netherlands : North-Holland Publishing Co., 1989, ch. How 3-MFA data can cause degenerate parafac solutions, among other relationships, pp. 115–122. [Online]. Available : http ://dl.acm.org/citation.cfm ?id=120565.120578

[Kop09] I. Kopriva, *3d tensor factorization approach to single-frame model-free blind-image deconvolution*," *Opt. Lett.*, vol. 34, no. 18, pp. 2835–2837, Sep 2009. [Online]. Available : http ://ol.osa.org/abstract.cfm ?URI=ol-34-18-2835

[Kru77] J. B. Kruskal, *Three-way arrays : Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics*," *Linear Algebra Applicat.*, vol. 18, 1977.

[KS98] H. A. L. Kiers and A. K. Smilde, *Constrained three-mode factor analysis as a tool for parameter estimation with second-order instrumental data*," *Journal of Chemometrics*, vol. 12, pp. 125–147, 1998.

[LC09] L. H. Lim and P. Comon, *Nonnegative approximations of nonnegative tensors*," *Jour. Chemometrics*, vol. 23, pp. 432–441, Aug. 2009.

[LdAC11] X. Luciani, A. L. F. de Almeida, and P. Comon, *Blind identification of underdetermined mixtures based on the characteristic function : The complex case*," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 540–553, 2011.

[LH87] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems (Classics in Applied Mathematics)*, new edition ed. Society for Industrial Mathematics, 1987.

[LMRB09] X. Luciani, S. Mounier, R. Redon, and A. Bois, *A simple correction method of inner filter effects affecting feem and its application to the parafac decomposition*," *Chemometrics and Intelligent Laboratory Systems*, vol. 96, no. 2, pp. 227 – 238, 2009, <ce :title>Chimiometrie 2007, Lyon, France, 29-30 November 2007</ce :title>. [Online]. Available : http ://www.sciencedirect.com/science/article/pii/S0169743909000203

[LS99] D. D. Lee and H. S. Seung, *Learning the parts of objects by non-negative matrix factorization*," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999. [Online]. Available : http ://dx.doi.org/10.1038/44565

[LS00] ——, *Algorithms for non-negative matrix factorization*," in *NIPS*, 2000, pp. 556–562. [Online]. Available : citeseer.ist.psu.edu/lee01algorithms.html

[LT08] S. Liu and G. Trenkler, *Hadamard, khatri-rao, kronecker and other matrix products*." *International Journal of Information & Systems Sciences*, vol. 4, no. 1, pp. 160–177, 2008.

[Luc07] X. Luciani, *Analyse numérique des spectres de fuorescence 3d issus de mélanges non linéaires*," Ph.D. dissertation, Université du Sud Toulon Var, 2007.

[Lue69] D. G. Luenberger, *Optimization by vector space methods*. Wiley, 1969.

[LY08] D. G. Luenberger and Y. Ye, *Linear and non linear programming*, 3rd ed. Wiley, 2008.

[Mau] D. Maurel, *Diagramme de jablonski*."

[MHA08] M. Mørup, L. K. Hansen, and S. M. Arnfred, *Algorithms for sparse nonnegative tucker decompositions*," *Neural Comput.*, vol. 20, no. 8, pp. 2112–2131, Aug. 2008. [Online]. Available : http ://dx.doi.org/10.1162/neco.2008.11-06-407

[MMMVS+04] F. Miwakeichi, E. Martínez-Montes, P. A. Valdés-Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi, *Decomposing eeg data into space time frequency components using parallel factor analysis*," *NeuroImage*, vol. 22, no. 3, pp. 1035 – 1045, 2004. [Online]. Available : http ://www.sciencedirect.com/science/article/pii/S1053811904001958

[MN07] J. R. Magnus and H. Neudecker, *Matrix differential calculus with applications in statistics and econometrics*, 3rd ed. Wiley, 2007.

[MS02] G. Matheron and J. Serra, *The birth of mathematical morphology*," in *VIth International Symposium : ISMM*, apr 2002.

[MZBR10] S. Mounier, H. Zhao, C. Barnier, and R. Redon, *Copper complexing properties of dissolved organic matter : Parafac treatment of fluorescence quenching*," *Biochemistry*, August 2010.

[NL08] D. Nion and L. D. Lathauwer, *An enhanced line search scheme for complex-valued tensor decompositions. application in ds-cdma*," *Signal Processing*, vol. 88, no. 3, pp. 749 – 755, 2008. [Online]. Available : http ://www.sciencedirect.com/science/article/pii/S016516840700271X

[NMSP10] D. Nion, K. N. Mokios, N. D. Sidiropoulos, and A. Potamianos, *Batch and adaptive parafac-based blind separation of convolutive speech mixtures*," *Trans.*

Audio, Speech and Lang. Proc., vol. 18, no. 6, pp. 1193–1207, Aug. 2010. [Online]. Available : http ://dx.doi.org/10.1109/TASL.2009.2031694

[NS09] D. Nion and N. D. Sidiropoulos, *A parafac-based technique for detection and localization of multiple targets in a mimo radar system*," in *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, ser. ICASSP '09. Washington, DC, USA : IEEE Computer Society, 2009, pp. 2077–2080. [Online]. Available : http ://dx.doi.org/10.1109/ICASSP.2009.4960024

[NW00] J. Nocedal and S. J. Wright, *Numerical Optimization.* Springer, Aug. 2000.

[Paa97] P. Paatero, *A weighted non-negative least squares algorithm for three-way Parafac factor analysis*," *Chemometrics Intell. Lab. Systems*, vol. 38, no. 2, pp. 223–242, 1997.

[Paa00] ——, *Construction and analysis of degenerate Parafac models*," *J. Chemometrics*, vol. 14, no. 3, pp. 285–299, 2000.

[PBdJ+02] V. Pravdova, C. Boucon, S. de Jong, B. Walczak, and D. Massart, *Three-way principal component analysis applied to food analysis : an example*," *Analytica Chimica Acta*, vol. 462, no. 2, pp. 133–148, 2002.

[PC08] A.-H. Phan and A. Cichocki, *Multi-way nonnegative tensor factorization using fast hierarchical alternating least squares algorithm (hals)*," in *Proceedings of the 2008 International Symposium on Nonlinear Theory and its Applications*, 2008, pp. 41–44.

[PC11] H. A. Phan and A. Cichocki, *Extended HALS algorithm for nonnegative tucker decomposition and its applications for multi-way analysis and classification*," *Neurocomputing*, vol. 74, pp. 1956–1969, 2011.

[PCB10] M. Plumbley, A. Cichocki, and R. Bro, *Handbook of Blind Source Separation*, ser. Academic Press. London, UK : P. Comon & C.Jutten Ed., 2010, ch. chapter 1 : Non-negative mixtures, pp. 515–547.

[Pol97] E. Polak, *Optimization algorithms and consistent approximations.* Springer, 1997.

[PTC] A. H. Phan, P. Tichavsky, and A. Cichocki, *Damped gauss-newton algorithm for nonnegative tucker decomposition*," in *Statistical Signal Processing (SSP'2011)*, Nice, France, pp. 665–668.

[PTC11] ——, *Damped gauss-newton algorithm for sparse and nonnegative tensor factorization*," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'11)*, Prague, Czech Republic, may 2011, pp. 1988–1991.

[RCH08] M. Rajih, P. Comon, and R. A. Harshman, *Enhanced line search : a novel method to accelerate Parafac*," *SIAM Journal of Matrix analysis applications*, vol. 30, no. 3, pp. 1148–1171, Sep. 2008.

[Ren08] N. Renard, *Traitement du signal tensoriel, application à l'imagerie hyperspectrale*," Ph.D. dissertation, Université Paul Cézanne, 2008.

[SB00] N. Sidiropoulos and R. Bro, *On the uniqueness of multilinear decomposition of n-way arrays*," *Journal of Chemometrics*, vol. 14, no. 3, pp. 229–239, 2000. [Online]. Available : http ://dx.doi.org/10.1002/1099-128X(200005/06)14 :3<229 : :AID-CEM587>3.0.CO ;2-N

[SBG04] A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis with applications in the chemical sciences*. Wiley, 2004.

[Ser83] J. Serra, *Image Analysis and Mathematical Morphology*. Orlando, FL, USA : Academic Press, Inc., 1983.

[She94] J. R. Shewchuk, *introduction to the conjugate gradient method without the agonizing pain*," Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, Technical Report, Aug. 1994.

[SK90] E. Sanchez and B. R. Kowalski, *Tensorial resolution : A direct trilinear decomposition*," *Journal of Chemometrics*, vol. 4, pp. 29–45, 1990.

[SM03] C. A. Stedmon and S. Markager, *Behaviour of the optical properties of coloured dissolved organic matter under conservative mixing*," *Estuarine, Coastal and Shelf Science*, vol. 57, no. 5-6, pp. 973–979, 2003.

[SM05] ——, *Resolving the variability in dissolved organic matter fluorescence in a temperate estuary and its catchment using parafac analysis*," *Limnology and Oceanography*, vol. 50, no. 2, pp. 686–697, 2005.

[SMB03a] C. A. Stedmon, S. Markager, and R. Bro, *Tracing dissolved organic matter in aquatic environments using a new approach to fluorescence spectroscopy*," *Marine Chemistry*, vol. 82, pp. 239–254, 2003.

[SMB03b] ——, *Tracing dissolved organic matter in aquatic environments using a new approach to fluorescence spectroscopy*," *Marine Chemistry*, vol. 82, no. 3-4, pp. 239–254, August 2003.

[SMBG00] N. D. Sidiropoulos, S. Member, R. Bro, and G. B. Giannakis, *Parallel factor analysis in sensor array processing*," *IEEE Trans. Signal Processing*, vol. 48, pp. 2377–2388, 2000.

[SS07] A. Stegeman and N. D. Sidiropoulos, *On Kruskal's uniqueness condition for the Candecomp/Parafac decomposition*," *Linear Algebra and Its Applications*, vol. 420, pp. 540–552, 2007.

[TB04] G. Tomasi and R. Bro, *Parafac and missing values*," *Chemometrics Intell. Lab. Systems*, 2004.

[TB05] ——, *Parafac and missing values*," *Chemometrics and Intelligent Laboratory Systems*, vol. 75, no. 2, pp. 163 – 180, 2005. [Online]. Available : http ://www.sciencedirect.com/science/article/pii/S0169743904001741

[TB06] ——, *A comparison of algorithms for fitting the Parafac model*," *Computational Statistics & Data Analysis*, vol. 50, no. 7, pp. 1700–1734, Apr. 2006, elsevier Sciences Publishers B. V.

[Tuc66] L. Tucker, *Some mathematical notes on three-mode factor analysis*," *Psychometrika*, pp. 279–311, 1966.

[ZAX12] G. Zhou, A. C. A., and S. Xie, *Fast nonnegative matrix/tensor factorization based on low-rank approximation*," *IEEE Transaction on Signal Processing*, 2012.

[ZC12] G. Zhou and A. Cichocki, *Fast and unique tucker decompositions via multiway blind source separation*," *Bulletin of Polish Academy of Science*, 2012.

[Zha11] H. Zhao, *Analyse de la matière organique et ses propriétés dans l'environnement naturel en spectroscopie de fluorescence 3D traitée par Parafac*," Ph.D. dissertation, Université du Sud Toulon-Var, Décembre 2011.

[ZSM04] R. G. Zepp, W. M. Sheldon, and M. A. Moran, *Dissolved organic fluorophores in southeastern us coastal waters : correction method for eliminating rayleigh and raman scattering peaks in excitation-emission matrices*," *Marine Chemistry*, vol. 89, no. 1 - 4, pp. 15 – 36, 2004, <ce :title>CDOM in the Ocean : Characterization, Distribution and Transformation</ce :title>. [Online]. Available : http ://www.sciencedirect.com/science/article/pii/S0304420304000799

[ZWPP08] Q. Zhang, H. Wang, R. Plemmons, and P. Pauca, *Tensors methods for hyperspectral data processing : a space object identification study*," *Journal of Optical Society of America A*, dec 2008.

Annexe 1 : calculs liés à la décomposition CP

6.2 Rappels de propriétés utiles

On utilise des propriétés similaires sur la trace à celles déjà utilisées en [GTMMA10]. Considérons trois matrices $M \times M$ carrées $\mathbf{D}_1$, $\mathbf{D}_1$, $\mathbf{D}_2$ and $\mathbf{D}_3$ et quatre matrices rectangulaires $\mathbf{D}_4$, $\mathbf{D}_5$, $\mathbf{D}_6$ and $\mathbf{D}_7$ (resp. $M \times N$, $N \times M$, $M \times N$ and $M \times N$), on a les propriétés suivantes [MN07] :

$\mathbf{P}_0$. $(\mathbf{D}_4\mathbf{D}_5)^T = \mathbf{D}_5^T\mathbf{D}_4^T$.

$\mathbf{P}_1$. $\mathsf{trace}\{\mathbf{D}_1\} = \mathsf{trace}\{\mathbf{D}_1^T\}$.

$\mathbf{P}_2$. $\mathsf{trace}\{\mathbf{D}_1 + \mathbf{D}_2\} = \mathsf{trace}\{\mathbf{D}_1\} + \mathsf{trace}\{\mathbf{D}_2\}$.

$\mathbf{P}_3$. $\mathsf{trace}\{\mathbf{D}_1\mathbf{D}_2\mathbf{D}_3\} = \mathsf{trace}\{\mathbf{D}_3\mathbf{D}_1\mathbf{D}_2\} = \mathsf{trace}\{\mathbf{D}_2\mathbf{D}_3\mathbf{D}_1\}$
$\Rightarrow \mathsf{trace}\{\mathbf{D}_1\mathbf{D}_2\} = \mathsf{trace}\{\mathbf{D}_2\mathbf{D}_1\}$.

$\mathbf{P}_4$. $\mathsf{trace}\{\mathbf{D}_4\mathbf{D}_5\} = \mathsf{trace}\{\mathbf{D}_5\mathbf{D}_4\}$.

$\mathbf{P}_5$. $\mathsf{d}(\mathbf{D}_1^T) = (\mathsf{d}\mathbf{D}_1)^T$.

$\mathbf{P}_6$. $\mathsf{d}(\mathbf{D}_1\mathbf{D}_2) = \mathsf{d}\mathbf{D}_1\mathbf{D}_2 + \mathbf{D}_1\mathsf{d}\mathbf{D}_2$.

$\mathbf{P}_7$. $\mathsf{d}(\mathbf{D}_1 + \mathbf{D}_2) = \mathsf{d}\mathbf{D}_1 + \mathsf{d}\mathbf{D}_2$.

$\mathbf{P}_8$. $\mathsf{d}(\mathsf{trace}\{\mathbf{D}_1\}) = \mathsf{trace}\{\mathsf{d}\mathbf{D}_1\}$.

$\mathbf{P}_9$. $\mathsf{d}(\mathbf{D}_1 \boxdot \mathbf{D}_2) = \mathsf{d}\mathbf{D}_1 \boxdot \mathbf{D}_2 + \mathbf{D}_1 \boxdot \mathsf{d}\mathbf{D}_2 \Rightarrow \mathsf{d}(\mathbf{D}_1 \boxdot \mathbf{D}_1) = 2\mathbf{D}_1 \boxdot \mathsf{d}\mathbf{D}_1$.

$\mathbf{P}_{10}$. $\mathbf{D}_4 \boxdot \mathbf{D}_6 = \mathbf{D}_6 \boxdot \mathbf{D}_4$.

$\mathbf{P}_{11}$. $(\mathbf{D}_4 \boxdot \mathbf{D}_6)^T = \mathbf{D}_4^T \boxdot \mathbf{D}_6^T$.

$\mathbf{P}_{12}$. $\mathsf{trace}\{\mathbf{D}_4^T(\mathbf{D}_6 \boxdot \mathbf{D}_7)\} = \mathsf{trace}\{(\mathbf{D}_4^T \boxdot \mathbf{D}_6^T)\mathbf{D}_7\}$.

6.3 Décomposition CP sans contrainte

6.3.1 Calcul des matrices de gradient

Comme dans [Fra92], notre but est d'obtenir :

$$\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \langle\frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}},\mathsf{d}\mathbf{A}\rangle + \langle\frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}},\mathsf{d}\mathbf{B}\rangle + \langle\frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}},\mathsf{d}\mathbf{C}\rangle, \qquad (6.4)$$

où $\frac{\partial \cdot}{\partial \mathbf{A}}$ représente la dérivée partielle par rapport à la matrice $\mathbf{A}$.

En utilisant des permutations circulaires, et les propriétés mentionnées précédemment en section 6.2, $\mathbf{P}_1 - \mathbf{P}_9$, on a :

$$
\begin{aligned}
\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \mathsf{trace}\left\{\mathsf{d}\boldsymbol{\delta}_{(1)}^T)\boldsymbol{\delta}_{(1)}\right\} + \mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T\mathsf{d}\boldsymbol{\delta}_{(1)}\right\} \\
&= 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T\mathsf{d}\boldsymbol{\delta}_{(1)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(2)}^T\mathsf{d}\boldsymbol{\delta}_{(2)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(3)}^T\mathsf{d}\boldsymbol{\delta}_{(3)}\right\} \\
&= 4\mathsf{trace}\Big\{-\boldsymbol{\delta}_{(1)}^T\mathsf{d}\mathbf{A}\boldsymbol{\Lambda}\left[\mathbf{C}\odot\mathbf{B}\right]^T - \boldsymbol{\delta}_{(2)}^T(\mathsf{d}\mathbf{B})\boldsymbol{\Lambda}\left[\mathbf{C}\odot\mathbf{A}\right]^T \\
&\quad -\boldsymbol{\delta}_{(3)}^T(\mathsf{d}\mathbf{C})\boldsymbol{\Lambda}\left[\mathbf{B}\odot\mathbf{A}\right]^T\Big\} \\
&= \mathsf{trace}\Big\{-4\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{B}\right)^T\boldsymbol{\delta}_{(1)})^T\mathsf{d}\mathbf{A}\Big\} \\
&\quad + \mathsf{trace}\Big\{-4\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{A}\right)^T\boldsymbol{\delta}_{(2)})^T\mathsf{d}\mathbf{B}\Big\} \\
&\quad + \mathsf{trace}\Big\{-4\boldsymbol{\Lambda}\left(\mathbf{B}\odot\mathbf{A}\right)^T\boldsymbol{\delta}_{(3)}^T\mathsf{d}\mathbf{C}\Big\} \\
&= \mathsf{trace}\Big\{\left(-4\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\odot\mathbf{B}\right)\boldsymbol{\Lambda}\right)^T\mathsf{d}\mathbf{A}\Big\} \\
&\quad + \mathsf{trace}\Big\{\left(-4\boldsymbol{\delta}_{(2)}\left(\mathbf{C}\odot\mathbf{A}\right)\boldsymbol{\Lambda}\right)^T\mathsf{d}\mathbf{B}\Big\} \\
&\quad + \mathsf{trace}\Big\{\left(-4\boldsymbol{\delta}_{(3)}\left(\mathbf{B}\odot\mathbf{A}\right)\boldsymbol{\Lambda}\right)^T\mathsf{d}\mathbf{C}\Big\} \\
&= \langle -4\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\odot\mathbf{B}\right)\boldsymbol{\Lambda}, \mathsf{d}\mathbf{A}\rangle \\
&\quad + \langle -4\boldsymbol{\delta}_{(2)}\left(\mathbf{C}\odot\mathbf{A}\right)\boldsymbol{\Lambda}, \mathsf{d}\mathbf{B}\rangle \\
&\quad + \langle -4\boldsymbol{\delta}_{(3)}\left(\mathbf{B}\odot\mathbf{A}\right)\boldsymbol{\Lambda}, \mathsf{d}\mathbf{C}\rangle
\end{aligned}
$$

Par identification avec (6.22), on trouve finalement :

$$
\begin{aligned}
\nabla_{\mathbf{A}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}} && (6.5)\\
&= -4\boldsymbol{\delta}_{(1)}\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{B}\right) && (6.6)\\
&= -4\left(\mathbf{T}_{(1)}^{I,KJ} - \mathbf{A}\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{B}\right)^T\right)\left(\mathbf{C}\odot\mathbf{B}\right)\boldsymbol{\Lambda}, && (6.7)\\
\nabla_{\mathbf{B}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}} && (6.8)\\
&= \langle -4\boldsymbol{\delta}_{(2)}\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{A}\right) && (6.9)\\
&= -4\left(\mathbf{T}_{(2)}^{J,KI} - \mathbf{A}\boldsymbol{\Lambda}\left(\mathbf{C}\odot\mathbf{A}\right)^T\right)\left(\mathbf{C}\odot\mathbf{A}\right)\boldsymbol{\Lambda}, && (6.10)\\
\nabla_{\mathbf{C}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}} && (6.11)\\
&= -4\boldsymbol{\delta}_{(3)}\boldsymbol{\Lambda}\left(\mathbf{B}\odot\mathbf{A}\right) && (6.12)\\
&= -4\left(\mathbf{T}_{(3)}^{K,JI} - \mathbf{A}\boldsymbol{\Lambda}\left(\mathbf{B}\odot\mathbf{A}\right)^T\right)\left(\mathbf{B}\odot\mathbf{A}\right)\boldsymbol{\Lambda}. && (6.13)
\end{aligned}
$$

6.3.2 Calcul du pas optimal

On souhaite minimiser l'expression suivante par rapport à μ :

$$\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \|\mathbf{T}_{(1)}^{I,KJ} - [(\mathbf{A}+\mu\mathbf{D_A})]\mathbf{\Lambda}[((\mathbf{C}+\mu\mathbf{D_C})\odot((\mathbf{B}+\mu\mathbf{D_B})]^T\|^2$$

$$\begin{aligned}\mathcal{F}(.) &= \|\mathbf{T}_{(1)}^{I,KJ} - (\mathbf{A}+\mu\mathbf{D_A})\,\mathbf{\Lambda}\,[\mathbf{C}\odot\mathbf{B}+\mathbf{C}\odot\mu\mathbf{D_B}+\mu\mathbf{D_C}\odot\mathbf{B}+\mu\mathbf{D_C}\odot\mu\mathbf{D_B}]^T\,\|^2\\ &= \|\mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mathbf{D_B})^T\,\mu^3 + \left(\mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{D_B})^T + \mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mathbf{B})^T + \mathbf{A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mu\mathbf{D_B})^T\right)\mu^2\\ &+ \left(\mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{B})^T + \mathbf{A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{D_B})^T + \mathbf{A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mathbf{B})^T\right)\mu + (\mathbf{T}_{(1)}^{I,KJ} - \mathbf{A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{B})^T\,\|^2\end{aligned}$$

Définissons des variables intermédiaires :

$$\begin{aligned}\mathbf{K}_3 &= \mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mathbf{D_B})^T\\ \mathbf{K}_2 &= \mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{D_B})^T + \mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mathbf{B})^T + \mathbf{A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mu\mathbf{D_B})^T\\ \mathbf{K}_1 &= \mathbf{D_A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{B})^T + \mathbf{A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{D_B})^T + \mathbf{A}\mathbf{\Lambda}\,(\mathbf{D_C}\odot\mathbf{B})^T\\ \mathbf{K}_0 &= \mathbf{T}_{(1)}^{I,KJ} - \mathbf{A}\mathbf{\Lambda}\,(\mathbf{C}\odot\mathbf{B})\end{aligned}$$

On obtient donc :

$$\begin{aligned}\mathcal{F}(.) &= \mathsf{trace}\left\{(\mathbf{K_3}\mu^3+\mathbf{K_2}\mu^2+\mathbf{K_1}\mu+\mathbf{K_0})\left(\mathbf{K_3}\mu^3+\mathbf{K_2}\mu^2+\mathbf{K_1}\mu+\mathbf{K_0}\right)^T\right\} && (6.14)\\ &= \mathsf{trace}\left\{\mathbf{K_3}\mathbf{K_3}^T\mu^6\right. && (6.15)\\ &+ \left(2\mathbf{K_3}\mathbf{K_2}^T\right)\mu^5 && (6.16)\\ &+ \left(2\mathbf{K_3}\mathbf{K_1}+\mathbf{K_2}\mathbf{K_2}^T\right)\mu^4 && (6.17)\\ &+ \left(2\left(\mathbf{K_3}\mathbf{K_0}^T+\mathbf{K_2}\mathbf{K_1}^T\right)\right)\mu^3 && (6.18)\\ &+ \left(2\mathbf{K_2}\mathbf{K_0}^T+\mathbf{K_1}\mathbf{K_1}^T\right)\mu^2 && (6.19)\\ &+ \left(2\mathbf{K_1}\mathbf{K_0}^T\right)\mu && (6.20)\\ &+ \left.\mathbf{K_0}\mathbf{K_0}^T\right\} && (6.21)\end{aligned}$$

Les sept coefficients $a_0,\ldots,a_6$ sont alors obtenus par identification.

6.4 Décomposition CP sous contrainte de non négativité

6.4.1 Paramétrisation au moyen d'un produit de Hadamard

6.4.1.1 Calcul des matrices de gradient

Comme dans [Fra92], notre but est d'obtenir :

$$\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \langle\frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}},\mathsf{d}\mathbf{A}\rangle + \langle\frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}},\mathsf{d}\mathbf{B}\rangle + \langle\frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}},\mathsf{d}\mathbf{C}\rangle, \qquad (6.22)$$

où $\frac{\partial \cdot}{\partial \mathbf{A}}$ représente la dérivée partielle par rapport à la matrice $\mathbf{A}$.

En utilisant des permutations circulaires, et les propriétés mentionnées précédemment en section 6.2, $\mathbf{P}_1 - \mathbf{P}_9$, on a :

$$\begin{aligned}
\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \mathsf{trace}\left\{\mathsf{d}(\boldsymbol{\delta}_{(1)}^T)\boldsymbol{\delta}_{(1)}\right\} + \mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T \mathsf{d}\boldsymbol{\delta}_{(1)}\right\} \\
&= 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T \mathsf{d}\boldsymbol{\delta}_{(1)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(2)}^T \mathsf{d}\boldsymbol{\delta}_{(2)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(3)}^T \mathsf{d}\boldsymbol{\delta}_{(3)}\right\} \\
&= 4\mathsf{trace}\Big\{-\boldsymbol{\delta}_{(1)}^T(\mathbf{A}\boxdot \mathsf{d}\mathbf{A})\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]^T - \boldsymbol{\delta}_{(2)}^T(\mathbf{B}\boxdot \mathsf{d}\mathbf{B})\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T \\
&\quad -\boldsymbol{\delta}_{(3)}^T(\mathbf{C}\boxdot \mathsf{d}\mathbf{C})\boldsymbol{\Lambda}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T\Big\} \\
&= \mathsf{trace}\left\{4\left(\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]^T(-\boldsymbol{\delta}_{(1)})^T\right)(\mathbf{A}\boxdot \mathsf{d}\mathbf{A})\right\} \\
&\quad + \mathsf{trace}\left\{4\left(\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T(-\boldsymbol{\delta}_{(2)})^T\right)(\mathbf{B}\boxdot \mathsf{d}\mathbf{B})\right\} \\
&\quad + \mathsf{trace}\left\{4\left(\boldsymbol{\Lambda}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T(-\boldsymbol{\delta}_{(3)})^T\right)(\mathbf{C}\boxdot \mathsf{d}\mathbf{C})\right\}
\end{aligned}$$

En utilisant la propriété $\mathbf{P}_{10} - \mathbf{P}_{12}$ ([MN07], p. 53) et le fait que $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^T$ puisque $\boldsymbol{\Lambda}$ est diagonale, on a :

$$\begin{aligned}
\mathsf{d}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) &= \mathsf{trace}\left\{4\left[\left(\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]^T(-\boldsymbol{\delta}_{(1)})^T\right)\boxdot\mathbf{A}^T\right]\mathsf{d}\mathbf{A}\right\} \\
&\quad + \mathsf{trace}\left\{4\left[\left(\boldsymbol{\Lambda}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T(-\boldsymbol{\delta}_{(2)})^T\right)\boxdot\mathbf{B}^T\right]\mathsf{d}\mathbf{B}\right\} \\
&\quad + \mathsf{trace}\left\{4\left[\left(\boldsymbol{\Lambda}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]^T(-\boldsymbol{\delta}_{(3)})^T\right)\boxdot\mathbf{C}^T\right]\mathsf{d}\mathbf{C}\right\} \\
&= \mathsf{trace}\left\{4\left[\mathbf{A}\boxdot\left(-\boldsymbol{\delta}_{(1)}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]\boldsymbol{\Lambda}\right)\right]^T\mathsf{d}\mathbf{A}\right\} \\
&\quad + \mathsf{trace}\left\{4\left[\mathbf{B}\boxdot\left(-\boldsymbol{\delta}_{(2)}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]\boldsymbol{\Lambda}\right)\right]^T\mathsf{d}\mathbf{B}\right\} \\
&\quad + \mathsf{trace}\left\{4\left[\mathbf{C}\boxdot\left(-\boldsymbol{\delta}_{(3)}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]\boldsymbol{\Lambda}\right)\right]^T\mathsf{d}\mathbf{C}\right\} \\
&= \left\langle 4\left[\mathbf{A}\boxdot\left(-\boldsymbol{\delta}_{(1)}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]\boldsymbol{\Lambda}\right)\right], \mathsf{d}\mathbf{A}\right\rangle \\
&\quad + \left\langle 4\left[\mathbf{B}\boxdot\left(-\boldsymbol{\delta}_{(2)}\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right]\boldsymbol{\Lambda}\right)\right], \mathsf{d}\mathbf{B}\right\rangle \\
&\quad + \left\langle 4\left[\mathbf{C}\boxdot\left(-\boldsymbol{\delta}_{(3)}\left[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right]\boldsymbol{\Lambda}\right)\right], \mathsf{d}\mathbf{C}\right\rangle
\end{aligned}$$

Par identification avec (6.22), on trouve finalement :

$$\nabla_{\mathbf{A}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}} = 4\mathbf{A}\boxdot\left((-\boldsymbol{\delta}_{(1)})\left[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right]\boldsymbol{\Lambda}\right), \tag{6.23}$$

$$\nabla_{\mathbf{B}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}} = 4\mathbf{B}\boxdot\left((-\boldsymbol{\delta}_{(2)})[(\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})]\boldsymbol{\Lambda}\right), \tag{6.24}$$

$$\nabla_{\mathbf{C}}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{\partial\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}} = 4\mathbf{C}\boxdot\left((-\boldsymbol{\delta}_{(3)})[(\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})]\boldsymbol{\Lambda}\right). \tag{6.25}$$

6.4.1.2 Calcul des gradients des termes de pénalité

On cherche à calculer l'expression des gradients des termes de pénalité sous contrainte de non négativité par produits de Hadamard.

Norme L1

$$\mathcal{H}_{L_1}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + \alpha_{\mathbf{A}}\|\mathbf{A}\boxdot\mathbf{A}\|_1 + \alpha_{\mathbf{B}}\|\mathbf{B}\boxdot\mathbf{B}\|_1 + \alpha_{\mathbf{C}}\|\mathbf{C}\boxdot\mathbf{C}\|_1.$$

Rappelons qu'on considère que $\|\mathbf{M}\|_1 = \sum_i \sum_j |m_{ij}|$.
Dans notre cas, $a_{ij} \geq 0 \quad \forall \quad i,j$, donc $\|\mathbf{M}\|_1 = \sum_i \sum_j m_{ij}$.
Dérivons pour l'exemple le terme relatif à la matrice de facteurs $\mathbf{A}$:

$$\begin{aligned}\mathsf{d}\left[\alpha_{\mathbf{A}}\|\mathbf{A}\boxdot\mathbf{A}\|_1\right] &= \alpha_{\mathbf{A}}\left(\|\mathsf{d}\mathbf{A}\boxdot\mathbf{A}\|_1 + \|\mathbf{A}\boxdot\mathsf{d}\mathbf{A}\|_1\right)\\ &= 2\alpha_{\mathbf{A}}\mathbf{A}\boxdot\mathsf{d}\mathbf{A}\end{aligned}$$

Donc la matrice de gradient de $\mathbf{A}$ résultant de la différentielle précédente est :

$$\nabla_{\mathbf{A}}\left(\alpha_{\mathbf{A}}\|\mathbf{A}\boxdot\mathbf{A}\|_1\right) = \frac{\mathsf{d}\left[\alpha_{\mathbf{A}}\|\mathbf{A}\boxdot\mathbf{A}\|_1\right]}{\mathsf{d}\mathbf{A}} = \alpha_{\mathbf{A}}2\mathbf{A}.$$

On trouve par analogie les matrices de gradients des termes de pénalité pour les matrices de facteurs $\mathbf{B}$ et $\mathbf{C}$.

Norme L2

$$\mathcal{H}_{L_2}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) + \alpha_{\mathbf{A}}\|\mathbf{A}\boxdot\mathbf{A}\|_F^2 + \alpha_{\mathbf{B}}\|\mathbf{B}\boxdot\mathbf{B}\|_F^2 + \alpha_{\mathbf{C}}\|\mathbf{C}\boxdot\mathbf{C}\|_F^2.$$

Dérivons pour l'exemple le terme relatif à la matrice de facteurs $\mathbf{A}$:

$$\begin{aligned}\mathsf{d}\left[\alpha_A\|\mathbf{A}\boxdot\mathbf{A}\|_F^2\right] &= \alpha_{\mathbf{A}}\left[\mathsf{trace}\left(\mathsf{d}\left(\mathbf{A}\boxdot\mathbf{A}\right)^T\left(\mathbf{A}\boxdot\mathbf{A}\right)\right) + \mathsf{trace}\left(\left(\mathbf{A}\boxdot\mathbf{A}\right)^T\mathsf{d}\left(\mathbf{A}\boxdot\mathbf{A}\right)\right)\right]\\ &= 2\alpha_{\mathbf{A}}\mathsf{trace}\left(\left(\mathbf{A}\boxdot\mathbf{A}\right)^T\mathsf{d}\left(\mathbf{A}\boxdot\mathbf{A}\right)\right)\\ &= 2\alpha_{\mathbf{A}}\mathsf{trace}\left(\left(\mathbf{A}\boxdot\mathbf{A}\right)^T\left(2\mathbf{A}\boxdot\mathsf{d}\mathbf{A}\right)\right).\end{aligned}$$

En utilisant la propriété $\mathbf{P}_{12}$ de l'annexe 6.2, on a :

$$\begin{aligned}\mathsf{d}\left[\alpha_A\|\mathbf{A}\boxdot\mathbf{A}\|_F^2\right] &= 4\alpha_{\mathbf{A}}\mathsf{trace}\left(\mathbf{A}^T\boxdot\mathbf{A}^T\boxdot\mathbf{A}^T\mathsf{d}\mathbf{A}\right)\\ &= 4\alpha_{\mathbf{A}}\mathsf{trace}\left(\left(\mathbf{A}\boxdot\mathbf{A}\boxdot\mathbf{A}\right)^T\mathsf{d}\mathbf{A}\right)\\ &= 4\alpha_{\mathbf{A}}\langle\mathbf{A}\boxdot\mathbf{A}\boxdot\mathbf{A},\mathsf{d}\mathbf{A}\rangle.\end{aligned}$$

Donc la matrice de gradient de $\mathbf{A}$ résultant de la différentielle précédente est :

$$\nabla_{\mathbf{A}}\left(\alpha_A\|\mathbf{A}\boxdot\mathbf{A}\|_F^2\right) = \frac{\partial\left[\alpha_A\|\mathbf{A}\boxdot\mathbf{A}\|_F^2\right]}{\partial\mathbf{A}} = 4\alpha_{\mathbf{A}}\mathbf{A}\boxdot\mathbf{A}\boxdot\mathbf{A}.$$

On procède de manière analogue pour déterminer les gradients des termes de pénalité provenant des matrices de facteurs $\mathbf{B}$ et $\mathbf{C}$.

6.4.2 Ajout d'un terme de régularisation exponentielle

6.4.2.1 Calcul des gradients des termes de pénalité

On cherche à calculer les dérivées des termes de pénalité dans le but d'obtenir les trois gradients des matrices de facteurs. Le détail du calcul des gradients sans termes de pénalité a été donné dans l'annexe 6.3. On donne l'exemple dans le cas de la matrice de facteurs $\mathbf{A}$ de taille $I\times F$. Par analogie, on peut faire de même pour les deux autres matrices de facteurs :

$$\mathsf{d}f(\mathbf{A}) = \mathsf{d}\|e^{-\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2. \tag{6.26}$$

$\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}$ est une matrice carrée de taille $IF\times IF$. On a la propriété suivante pour une matrice $\mathbf{M}$, carrée, de taille $N\times N$ par exemple :

$$\mathbf{D}_0.e^{\mathbf{M}} = \sum_{k=0}^{\infty}\frac{\mathbf{M}^k}{k!} \tag{6.27}$$

Alors :

$$\|e^{-\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 = \mathsf{trace}\{(e^{-\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}})^T.$$
$$e^{-\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\} = \mathsf{trace}\{e^{-2\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\}.$$

En utilisant la propriété $\mathbf{D}_0$, on a :

$$\|e^{-2\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 = \sum_{k=0}^{\infty}\frac{(-2\gamma)^k}{k!}\mathsf{trace}\{[\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}]^k\}$$

En considérant seulement les trois premiers termes de la série, on trouve :

$$\begin{aligned}\mathsf{d}\|e^{-\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 &= \frac{\partial\left[\mathbf{I}_{IF}-2\gamma\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}+2\gamma^2[\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}]^2\right]}{\partial\mathsf{vec}\{\mathbf{A}\}}\mathsf{dvec}\{\mathbf{A}\}\\ &\simeq \left[-2\gamma\mathbf{I}_{IF}+4\gamma^2\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}\right]\mathsf{dvec}\{\mathbf{A}\}\\ &\simeq 4\gamma^2\left[\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}-\frac{\mathbf{I}_{IF}}{2\gamma}\right]\mathsf{dvec}\{\mathbf{A}\}.\end{aligned} \tag{6.28}$$

Si on transforme ce résultat sous forme matricielle, on obtient :

$$\mathsf{d}\|e^{-\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 = 4\gamma^2 \left[\mathbf{A} - \frac{\mathbf{1}_{I,F}}{2\gamma}\right] \mathsf{d}\mathbf{A}. \tag{6.29}$$

6.4.3 Calcul du pas optimal sous contrainte de non négativité

On souhaite minimiser l'expression suivante par rapport à μ :

$$\begin{aligned}\mathcal{H}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \|\mathbf{T}^{I,JK} - &[(\mathbf{A}+\mu\mathbf{D_A}) \boxdot (\mathbf{A}+\mu\mathbf{D_A})]\mathbf{\Lambda} \\ &[((\mathbf{C}+\mu\mathbf{D_C}) \boxdot (\mathbf{C}+\mu\mathbf{D_C})) \odot ((\mathbf{B}+\mu\mathbf{D_B}) \boxdot (\mathbf{B}+\mu\mathbf{D_B}))]^T\|^2\end{aligned}$$

Premièrement, pour clarifier les expressions, on définit des quantités intermédiaires :

$$\begin{aligned}
\mathbf{E_0} &= \mathbf{A} \boxdot \mathbf{A} \\
\mathbf{E_1} &= \mathbf{A} \boxdot \mathbf{D_A} + \mathbf{D_A} \boxdot \mathbf{A} = 2\mathbf{A} \boxdot \mathbf{D_A} \\
\mathbf{E_2} &= \mathbf{D_A} \boxdot \mathbf{D_A} \\
\mathbf{F_0} &= (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{B}) \\
\mathbf{F_1} &= (\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{B}) + (\mathbf{D_C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{B}) \\
&\quad + (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{D_B}) + (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{D_B} \boxdot \mathbf{B}) \\
&= 2\left[(\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{B}) + (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{D_B})\right] \\
\mathbf{F_2} &= (\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{D_B}) + (\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{D_B} \boxdot \mathbf{B}) + (\mathbf{D_C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{D_B}) \\
&\quad + (\mathbf{D_C} \boxdot \mathbf{C}) \odot (\mathbf{D_B} \boxdot \mathbf{B}) + (\mathbf{D_C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{B}) + (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{D_B} \boxdot \mathbf{D_B}) \\
&= 4\left[(\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{D_B})\right] + (\mathbf{D_C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{B}) + (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{D_B} \boxdot \mathbf{D_B}) \\
\mathbf{F_3} &= (\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{D_B} \boxdot \mathbf{D_B}) + (\mathbf{D_C} \boxdot \mathbf{C}) \odot (\mathbf{D_B} \boxdot \mathbf{D_B}) \\
&\quad + (\mathbf{D_C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{D_B}) + (\mathbf{D_C} \boxdot \mathbf{D_C}) \odot (\mathbf{D_B} \boxdot \mathbf{B}) \\
&= 2[(\mathbf{C} \boxdot \mathbf{D_C}) \odot (\mathbf{D_B} \boxdot \mathbf{D_B}) + (\mathbf{D_C} \boxdot \mathbf{D_C}) \odot (\mathbf{B} \boxdot \mathbf{D_B})] \\
\mathbf{F_4} &= (\mathbf{D_C} \boxdot \mathbf{D_C}) \odot (\mathbf{D_B} \boxdot \mathbf{D_B})
\end{aligned}$$

En developpant, on obtient :

$$\begin{aligned}
\mathcal{H}(.) &= \|\mathbf{T}^{I,JK} - [\mathbf{E_0} + \mathbf{E_1}\mu + \mathbf{E_2}\mu^2]\mathbf{\Lambda}[\mathbf{F_4}\mu^4 + \mathbf{F_3}\mu^3 + \mathbf{F_2}\mu^2 + \mathbf{F_1}\mu + \mathbf{F_0}]^T\|^2 \\
&= \|(-\mathbf{E_2}\mathbf{\Lambda}{\mathbf{F_4}}^T)\mu^6 + (-\mathbf{E_1}\mathbf{\Lambda}{\mathbf{F_4}}^T - \mathbf{E_2}\mathbf{\Lambda}{\mathbf{F_3}}^T)\mu^5 \\
&\quad + (-\mathbf{E_0}\mathbf{\Lambda}{\mathbf{F_4}}^T - \mathbf{E_1}\mathbf{\Lambda}{\mathbf{F_3}}^T - \mathbf{E_2}\mathbf{\Lambda}{\mathbf{F_2}}^T)\mu^4 + (-\mathbf{E_0}\mathbf{\Lambda}{\mathbf{F_3}}^T - \mathbf{E_1}\mathbf{\Lambda}{\mathbf{F_2}}^T - \mathbf{E_2}\mathbf{\Lambda}{\mathbf{F_1}}^T)\mu^3 \\
&\quad + (-\mathbf{E_0}\mathbf{\Lambda}{\mathbf{F_2}}^T - \mathbf{E_1}\mathbf{\Lambda}{\mathbf{F_1}}^T - \mathbf{E_2}\mathbf{\Lambda}{\mathbf{F_0}}^T)\mu^2 + (-\mathbf{E_0}\mathbf{\Lambda}{\mathbf{F_1}}^T - \mathbf{E_1}\mathbf{\Lambda}{\mathbf{F_0}}^T)\mu \\
&\quad + \mathbf{T}^{I,JK} - \mathbf{E_0}\mathbf{\Lambda}{\mathbf{F_0}}^T\|^2
\end{aligned}$$

Une nouvelle fois, on définit des variables intermédiaires :

$\mathbf{K_0}=\mathbf{T}^{I,JK} - \mathbf{E_0\Lambda F_0}^T$ $\quad$ $\mathbf{K_4}=-\mathbf{E_0\Lambda F_4}^T - \mathbf{E_1\Lambda F_3}^T - \mathbf{E_2\Lambda F_2}^T$

$\mathbf{K_1}=-\mathbf{E_0\Lambda F_1}^T - \mathbf{E_1\Lambda F_0}^T$ $\quad$ $\mathbf{K_5}=-\mathbf{E_1\Lambda F_4}^T - \mathbf{E_2\Lambda F_3}^T$

$\mathbf{K_2}=-\mathbf{E_0\Lambda F_2}^T - \mathbf{E_1\Lambda F_1}^T - \mathbf{E_2\Lambda F_0}^T$ $\quad$ $\mathbf{K_6}=-\mathbf{E_2\Lambda F_4}^T$

$\mathbf{K_3}=-\mathbf{E_0\Lambda F_3}^T - \mathbf{E_1\Lambda F_2}^T - \mathbf{E_2\Lambda F_1}^T$

$$\begin{aligned}\mathcal{H}(.) = \mathsf{trace}\big\{&(\mathbf{K_6}\mu^6 + \mathbf{K_5}\mu^5 + \mathbf{K_4}\mu^4 + \mathbf{K_3}\mu^3 + \mathbf{K_2}\mu^2 + \mathbf{K_1}\mu + \mathbf{K_0}) \\ &\big(\mathbf{K_6}\mu^6 + \mathbf{K_5}\mu^5 + \mathbf{K_4}\mu^4 + \mathbf{K_3}\mu^3 + \mathbf{K_2}\mu^2 + \mathbf{K_1}\mu + \mathbf{K_0}\big)^T\big\} \qquad (6.30)\end{aligned}$$

$$\begin{aligned}&= \mathsf{trace}\big\{(\mathbf{K_6K_6}^T)\mu^{12} \\ &+ (\mathbf{K_6K_5}^T + \mathbf{K_5K_6}^T)\mu^{11} \\ &+ (\mathbf{K_6K_4}^T + \mathbf{K_5K_5}^T + \mathbf{K_4K_6}^T)\mu^{10} \\ &+ (\mathbf{K_6K_3}^T + \mathbf{K_5K_4}^T + \mathbf{K_4K_5}^T + \mathbf{K_3K_6}^T)\mu^9 \\ &+ (\mathbf{K_6K_2}^T + \mathbf{K_5K_3}^T + \mathbf{K_4K_4}^T + \mathbf{K_3K_5}^T + \mathbf{K_2K_6}^T)\mu^8 \\ &+ (\mathbf{K_6K_1}^T + \mathbf{K_5K_2}^T + \mathbf{K_4K_3}^T + \mathbf{K_3K_4}^T + \mathbf{K_2K_5}^T + \mathbf{K_1K_6}^T)\mu^7 \\ &+ (\mathbf{K_6K_0}^T + \mathbf{K_5K_1}^T + \mathbf{K_4K_2}^T + \mathbf{K_3K_3}^T + \mathbf{K_2K_4}^T + \mathbf{K_1K_5}^T + \mathbf{K_0K_6}^T)\mu^6 \\ &+ (\mathbf{K_5K_0}^T + \mathbf{K_4K_1}^T + \mathbf{K_3K_2}^T + \mathbf{K_2K_3}^T + \mathbf{K_1K_4}^T + \mathbf{K_0K_5}^T)\mu^5 \\ &+ (\mathbf{K_4K_0}^T + \mathbf{K_3K_1}^T + \mathbf{K_2K_2}^T + \mathbf{K_1K_3}^T + \mathbf{K_0K_4}^T)\mu^4 \\ &+ \mathbf{K_3K_0}^T + \mathbf{K_2K_1}^T + \mathbf{K_1K_2}^T + \mathbf{K_0K_3}^T)\mu^3 \\ &+ (\mathbf{K_2K_0}^T + \mathbf{K_1K_1}^T + \mathbf{K_0K_2}^T)\mu^2 \\ &+ (\mathbf{K_1K_0}^T + \mathbf{K_0K_1}^T)\mu \\ &+\mathbf{K_0K_0}^T\big\} \qquad (6.31)\end{aligned}$$

Les treize coefficients $a_0, \ldots, a_{12}$ sont finalement obtenus par identification.

Annexe 2 : calculs liés à la décomposition de Tucker3

6.5 Décomposition de Tucker3 sans contrainte

6.5.1 Calcul des gradients matriciels

Considérons la fonction de coût suivante :

$$\mathcal{T} = \|\mathbf{T}_{(1)}^{I,KJ} - \mathbf{A}\mathbf{G}_{(1)}^{F_1,F_3F_2}(\mathbf{C}\otimes\mathbf{B})^T\|_F^2 = \|\boldsymbol{\delta}_{(1)}\|_F^2 \tag{6.32}$$

$$= \|\mathbf{T}_{(2)}^{J,KI} - \mathbf{B}\mathbf{G}_{(2)}^{F_2,F_3F_1}(\mathbf{C}\otimes\mathbf{A})^T\|_F^2 = \|\boldsymbol{\delta}_{(2)}\|_F^2 \tag{6.33}$$

$$= \|\mathbf{T}_{(2)}^{J,KI} - \mathbf{C}\mathbf{G}_{(3)}^{F_3,F_2F_1}(\mathbf{B}\otimes\mathbf{A})^T\|_F^2 = \|\boldsymbol{\delta}_{(3)}\|_F^2, \tag{6.34}$$

où le tenseur $\mathbf{T}$ déplié ci-dessus dans les trois modes est de taille $I \times J \times K$, $\mathbf{A}$ est de taille $I \times F_1$, $\mathbf{B}$ est de taille $J \times F_2$, $\mathbf{C}$ est de taille $K \times F_3$. Le tenseur $\mathbf{G}$ est quant à lui de taille $F_1 \times F_2 \times F_3$. Notons les définitions implicites de $\boldsymbol{\delta}_{(1)}$, $\boldsymbol{\delta}_{(2)}$ et $\boldsymbol{\delta}_{(3)}$.

Il s'agit de calculer les gradients des matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, et du tenseur coeur $\mathbf{G}$. On se base sur les propriétés données en 6.2.

$$
\begin{aligned}
\mathsf{d}\mathcal{T}(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{G}) &= \mathsf{trace}\left\{\mathsf{d}\boldsymbol{\delta}_{(1)}^T)\boldsymbol{\delta}_{(1)}\right\} + \mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T\mathsf{d}\boldsymbol{\delta}_{(1)}\right\} \\
&= 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T\mathsf{d}\boldsymbol{\delta}_{(1)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(2)}^T\mathsf{d}\boldsymbol{\delta}_{(2)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(3)}^T\mathsf{d}\boldsymbol{\delta}_{(3)}\right\} \\
&= 2\mathsf{trace}(-\boldsymbol{\delta}_{(1)}^T\mathsf{d}\mathbf{A}\mathbf{G}_{(1)}^{F_1,F_3F_2}(\mathbf{C}\otimes\mathbf{B})^T) \\
&+ 2\mathsf{trace}(-\boldsymbol{\delta}_{(1)}^T\mathbf{A}\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}(\mathbf{C}\otimes\mathbf{B})^T) \\
&+ 2\mathsf{trace}(-\boldsymbol{\delta}_{(2)}^T\mathsf{d}\mathbf{B}\mathbf{G}_{(2)}^{F_2,F_3F_1}(\mathbf{A}\otimes\mathbf{C})^T) \\
&+ 2\mathsf{trace}(-\boldsymbol{\delta}_{(3)}^T\mathsf{d}\mathbf{C}\mathbf{G}_{(3)}^{F_3,F_2F_1}(\mathbf{B}\otimes\mathbf{A})^T) \\
&= 2\mathsf{trace}(-\mathbf{G}_{(1)}^{F_1,F_3F_2}(\mathbf{C}\otimes\mathbf{B})^T\boldsymbol{\delta}_{(1)}^T\mathsf{d}\mathbf{A}) \\
&+ 2\mathsf{trace}(-(\mathbf{C}\otimes\mathbf{B})^T\boldsymbol{\delta}_{(1)}^T\mathbf{A}\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}) \\
&+ 2\mathsf{trace}(-\mathbf{G}_{(2)}^{F_2,F_3F_1}(\mathbf{A}\otimes\mathbf{C})^T\boldsymbol{\delta}_{(2)}^T\mathsf{d}\mathbf{B}) \\
&+ 2\mathsf{trace}(-\mathbf{G}_{(3)}^{F_3,F_2F_1}(\mathbf{B}\otimes\mathbf{A})^T\boldsymbol{\delta}_{(3)}^T\mathsf{d}\mathbf{C}) \\
&= 2\mathsf{trace}(-\left(\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)^T\right)^T\mathsf{d}\mathbf{A} \\
&+ 2\mathsf{trace}(-\left(\mathbf{A}^T\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right)\right)^T\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2} \\
&+ 2\mathsf{trace}(-\left(\boldsymbol{\delta}_{(2)}\left(\mathbf{C}\otimes\mathbf{A}\right)\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)^T\right)^T\mathsf{d}\mathbf{B} \\
&+ 2\mathsf{trace}(-\left(\boldsymbol{\delta}_{(3)}\left(\mathbf{B}\otimes\mathbf{A}\right)\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)^T\right)^T\mathsf{d}\mathbf{C} \\
&= 2\langle-\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right)\mathbf{G}_{(1)}^{F_1,F_3F_2},\mathsf{d}\mathbf{A}\rangle \\
&+ 2\langle-\mathbf{A}^T\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right)\rangle \\
&+ 2\langle-\boldsymbol{\delta}_{(2)}\left(\mathbf{A}\otimes\mathbf{C}\right)\mathbf{G}_{(2)}^{F_2,F_3F_1},\mathsf{d}\mathbf{B}\rangle \\
&+ 2\langle-\boldsymbol{\delta}_{(3)}\left(\mathbf{B}\otimes\mathbf{A}\right)\mathbf{G}_{(3)}^{F_3,F_2F_1},\mathsf{d}\mathbf{C}\rangle
\end{aligned}
$$

Par identification, on trouve :

$$\frac{\partial\mathcal{T}}{\partial\mathbf{A}} = -2\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)^T \tag{6.35}$$

$$\frac{\partial\mathcal{T}}{\partial\mathbf{B}} = -2\boldsymbol{\delta}_{(2)}\left(\mathbf{A}\otimes\mathbf{C}\right)\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)^T \tag{6.36}$$

$$\frac{\partial\mathcal{T}}{\partial\mathbf{C}} = -2\boldsymbol{\delta}_{(3)}\left(\mathbf{B}\otimes\mathbf{A}\right)\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)^T \tag{6.37}$$

$$\frac{\partial\mathcal{T}}{\partial\mathbf{G}} = -2\mathbf{A}^T\boldsymbol{\delta}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right) \tag{6.38}$$

$$\tag{6.39}$$

6.5.2 Calcul du pas optimal

On souhaite minimiser l'expression suivante par rapport à μ :

$$\mathcal{T}(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{G}) = \|\mathbf{T}^{I,JK} - (\mathbf{A}+\mu\mathbf{D_A})\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}+\mu\mathbf{D_G}\right)[(\mathbf{C}+\mu\mathbf{D_C})\otimes(\mathbf{B}+\mu\mathbf{D_B})]^T\|_F^2$$

Développons l'expression :

$$\begin{aligned}\mathcal{T}(.) &= \|\mathbf{T}_{(1)}^{I,KJ} - (\mathbf{A}+\mu\mathbf{D_A})\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}+\mu\mathbf{D_G}\right)[(\mathbf{C}+\mu\mathbf{D_C})\otimes(\mathbf{B}+\mu\mathbf{D_B})]^T\|_F^2\\ \mathcal{T}(.) &= \|\mathbf{T}_{(1)}^{I,KJ} - \left[\mathbf{A}\mathbf{G}_{(1)}^{F_1,F_3F_2}+\mu\left(\mathbf{A}\mathbf{D_G}+\mathbf{D_A}\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)+\mu^2\mathbf{D_A}\mathbf{D_G}\right]\\ &\quad\left[\mathbf{C}\otimes\mathbf{B}+\mu\left(\mathbf{C}\otimes\mathbf{D_B}+\mathbf{D_C}\otimes\mathbf{B}\right)+\mu^2\mathbf{D_C}\odot\mathbf{D_B}\right]\|_F^2\end{aligned}$$

Définissons des valeurs intermédiaires :

$$\begin{aligned}\mathbf{E_0} &= \mathbf{A}\mathbf{G}_{(1)}^{F_1,F_3F_2}\\ \mathbf{E_1} &= \mathbf{A}\mathbf{D_G}+\mathbf{D_A}\mathbf{G}_{(1)}^{F_1,F_3F_2}\\ \mathbf{E_2} &= \mathbf{D_A}\mathbf{D_G}\\ \mathbf{F_0} &= \mathbf{C}\otimes\mathbf{B}\\ \mathbf{F_1} &= \mathbf{C}\otimes\mathbf{D_B}+\mathbf{D_C}\otimes\mathbf{B}\\ \mathbf{F_2} &= \mathbf{D_C}\otimes\mathbf{D_B}\end{aligned}$$

D'où :

$$\begin{aligned}\mathcal{T}(.) &= \|\mathbf{T}_{(1)}^{I,KJ} - \left[\mathbf{E_0}+\mu\mathbf{E_1}+\mu^2\mathbf{E_2}\right]\left[\mathbf{F_0}+\mu\mathbf{F_1}+\mu^2\mathbf{F_2}\right]^T\|_F^2\\ \mathcal{T}(.) &= \|\mathbf{T}_{(1)}^{I,KJ} - \mathbf{E_0}\mathbf{F_0}^T+\mu\left(-\mathbf{E_0}\mathbf{F_1}^T-\mathbf{E_1}\mathbf{F_0}^T\right)+\mu^2\left(-\mathbf{E_0}\mathbf{F_2}^T-\mathbf{E_1}\mathbf{F}_1^T-\mathbf{E_2}\mathbf{F_0}^T\right)+\\ &\quad\mu^3\left(-\mathbf{E_1}\mathbf{F_2}^T-\mathbf{E_2}\mathbf{F_1}^T\right)+\mu^4\left(-\mathbf{E_2}\mathbf{F_2}^T\right)\|_F^2\end{aligned}$$

En posant :

$$\begin{aligned}\mathbf{K_0} &= \mathbf{T}_{(1)}^{I,KJ}-\mathbf{E_0}\mathbf{F_0}^T\\ \mathbf{K_1} &= -\mathbf{E_0}\mathbf{F_1}^T-\mathbf{E_1}\mathbf{F_0}^T\\ \mathbf{K_2} &= -\mathbf{E_0}\mathbf{F_2}^T-\mathbf{E_1}\mathbf{F}_1^T-\mathbf{E_2}\mathbf{F_0}^T\\ \mathbf{K_3} &= -\mathbf{E_1}\mathbf{F_2}^T-\mathbf{E_2}\mathbf{F_1}^T\\ \mathbf{K_4} &= -\mathbf{E_2}\mathbf{F_2}^T\end{aligned}$$

on obtient :

$$\mathcal{T}(.) = \mathsf{trace}\left\{\left(\mathbf{K_0} + \mu\mathbf{K_1} + \mu^2\mathbf{K_2} + \mu^3\mathbf{K_3} + \mu^4\mathbf{K_4}\right)\left(\mathbf{K_0} + \mu\mathbf{K_1} + \mu^2\mathbf{K_2} + \mu^3\mathbf{K_3} + \mu^4\mathbf{K_4}\right)^T\right\}$$

$$\begin{aligned}\mathcal{T}(.) = \mathsf{trace}\{&\left(\mathbf{K_4}\mathbf{K_4}^T\right)\mu^8\\ &\left(\mathbf{2K_3}\mathbf{K_4}^T\right)\mu^7+\\ &\left(2\mathbf{K_2}\mathbf{K_4}^T + \mathbf{K_3}\mathbf{K_3}^T\right)\mu^6+\\ &\left(2\left(\mathbf{K_1}\mathbf{K_4}^T + \mathbf{K_2}\mathbf{K_3}^T\right)\right)\mu^5+\\ &\left(2\left(\mathbf{K_4}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_3}^T\right) + \mathbf{K_2}\mathbf{K_2}^T\right)\mu^4+\\ &\left(2\left(\mathbf{K_3}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_2}^T\right)\right)\mu^3+\\ &\left(\mathbf{2K_2}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_1}^T\right)\mu^2+\\ &\left(2\mathbf{K_1}\mathbf{K_0}^T\right)\mu+\\ &\mathbf{K_0}\mathbf{K_0^T}\}\end{aligned}$$

On obtient ensuite les neuf coefficients $a_0 ... a_8$ par identification

6.6 Décomposition de Tucker3 avec contrainte de non négativité

6.6.1 Calcul des gradients matriciels

Considérons la fonction de coût suivante :

$$\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{G}) = \|\mathbf{T}_{(1)}^{I,KJ} - (\mathbf{A}\boxdot\mathbf{A})\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B}))^T\|_F^2 = \|\boldsymbol{\delta}_{(1)}\|_F^2 \tag{6.40}$$

$$= \|\mathbf{T}_{(2)}^{J,KI} - (\mathbf{B}\boxdot\mathbf{B})\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A}))^T\|_F^2 = \|\boldsymbol{\delta}_{(2)}\|_F^2 \tag{6.41}$$

$$= \|\mathbf{T}_{(3)}^{K,JI} - (\mathbf{C}\boxdot\mathbf{C})\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A}))^T\|_F^2 = \|\boldsymbol{\delta}_{(3)}\|_F^2, \tag{6.42}$$

où le tenseur $\mathbf{T}$ déplié ci-dessus dans les trois modes est de taille $I \times J \times K$, $\mathbf{A}$ est de taille $I \times F_1$, $\mathbf{B}$ est de taille $J \times F_2$, $\mathbf{C}$ est de taille $K \times F_3$. Le tenseur $\mathbf{G}$ est quant à lui de taille $F_1 \times F_2 \times F_3$. Notons les définitions implicites de $\boldsymbol{\delta}_{(1)}$, $\boldsymbol{\delta}_{(2)}$ et $\boldsymbol{\delta}_{(3)}$.
Il s'agit toujours de calculer les gradients des matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, et du tenseur coeur $\mathbf{G}$. On utilise pour cela les propriétés données en 6.2.

$$
\begin{aligned}
\mathsf{d}\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{G}) &= \mathsf{trace}\left\{\mathsf{d}\boldsymbol{\delta}_{(1)}^T)\boldsymbol{\delta}_{(1)}\right\} + \mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T\mathsf{d}\boldsymbol{\delta}_{(1)}\right\} \\
&= 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(1)}^T\mathsf{d}\boldsymbol{\delta}_{(1)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(2)}^T\mathsf{d}\boldsymbol{\delta}_{(2)}\right\} = 2\mathsf{trace}\left\{\boldsymbol{\delta}_{(3)}^T\mathsf{d}\boldsymbol{\delta}_{(3)}\right\} \\
&= 2\mathsf{trace}\left(-\boldsymbol{\delta}_{(1)}^T\left(2\mathbf{A}\boxdot\mathsf{d}\mathbf{A}\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\right) \\
&+ 2\mathsf{trace}\left(-\boldsymbol{\delta}_{(2)}^T\left(2\mathbf{B}\boxdot\mathsf{d}\mathbf{B}\right)\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\right) \\
&+ 2\mathsf{trace}\left(-\boldsymbol{\delta}_{(3)}^T\left(2\mathbf{C}\boxdot\mathsf{d}\mathbf{C}\right)\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)\left((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\right) \\
&+ 2\mathsf{trace}\left(\boldsymbol{\delta}_{(1)}^T\left(-\left(\mathbf{A}\boxdot\mathbf{A}\right)\left(2\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\right)\right) \\
&= 4\mathsf{trace}\left(-\boldsymbol{\delta}_{(1)}^T\left(\mathbf{A}\boxdot\mathsf{d}\mathbf{A}\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\right) \\
&+ 4\mathsf{trace}\left(-\boldsymbol{\delta}_{(2)}^T\left(\mathbf{B}\boxdot\mathsf{d}\mathbf{B}\right)\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\right) \\
&+ 4\mathsf{trace}\left(-\boldsymbol{\delta}_{(3)}^T\left(\mathbf{C}\boxdot\mathsf{d}\mathbf{C}\right)\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)\left((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\right) \\
&+ 4\mathsf{trace}\left(-\boldsymbol{\delta}_{(1)}^T\left(\left(\mathbf{A}\boxdot\mathbf{A}\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\right)\right) \\
&= 4\mathsf{trace}\left(-\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\boldsymbol{\delta}_{(1)}^T\left(\mathbf{A}\boxdot\mathsf{d}\mathbf{A}\right)\right) \\
&+ 4\mathsf{trace}\left(-\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\boldsymbol{\delta}_{(2)}^T\left(\mathbf{B}\boxdot\mathsf{d}\mathbf{B}\right)\right) \\
&+ 4\mathsf{trace}\left(-\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)\left((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\boldsymbol{\delta}_{(3)}^T\left(\mathbf{C}\boxdot\mathsf{d}\mathbf{C}\right)\right) \\
&+ 4\mathsf{trace}\left(-\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\boldsymbol{\delta}_{(1)}^T\left(\left(\mathbf{A}\boxdot\mathbf{A}\right)\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\right)\right) \\
&= 4\mathsf{trace}\left(-\left[\left(\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\boldsymbol{\delta}_{(1)}^T\right)\boxdot\mathbf{A}^T\right]\mathsf{d}\mathbf{A}\right) \\
&+ 4\mathsf{trace}\left(-\left[\left(\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\boldsymbol{\delta}_{(2)}^T\right)\boxdot\mathbf{B}^T\right]\mathsf{d}\mathbf{B}\right) \\
&+ 4\mathsf{trace}\left(-\left[\left(\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)\left((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\boldsymbol{\delta}_{(3)}^T\right)\boxdot\mathbf{C}^T\right]\mathsf{d}\mathbf{C}\right) \\
&+ 4\mathsf{trace}\left(-\left(\left[\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\boldsymbol{\delta}_{(1)}^T\left(\mathbf{A}\boxdot\mathbf{A}\right)\right]\boxdot\mathbf{G}^T\right)\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}\right) \\
&= 4\mathsf{trace}\left(-\left[\mathbf{A}\boxdot\left(\boldsymbol{\delta}_{(1)}\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)^T\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)^T\right)\right]^T\mathsf{d}\mathbf{A}\right) \\
&+ 4\mathsf{trace}\left(-\left[\mathbf{B}\boxdot\left(\boldsymbol{\delta}_{(2)}\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\left(\mathbf{G}_{(2)}^{F_2,F_3F_1}\boxdot\mathbf{G}_{(2)}^{F_2,F_3F_1}\right)^T\right)\right]^T\mathsf{d}\mathbf{B}\right) \\
&+ 4\mathsf{trace}\left(-\left[\mathbf{C}\boxdot\left(\boldsymbol{\delta}_{(3)}\left((\mathbf{B}\boxdot\mathbf{B})\otimes(\mathbf{A}\boxdot\mathbf{A})\right)^T\left(\mathbf{G}_{(3)}^{F_3,F_2F_1}\boxdot\mathbf{G}_{(3)}^{F_3,F_2F_1}\right)^T\right)\right]^T\mathsf{d}\mathbf{C}\right) \\
&+ 4\mathsf{trace}\left(-\left(\mathbf{G}_{(1)}^{F_1,F_3F_2}\boxdot\left[\left(\mathbf{A}\boxdot\mathbf{A}\right)^T\boldsymbol{\delta}_{(1)}\left((\mathbf{C}\boxdot\mathbf{C})\otimes(\mathbf{B}\boxdot\mathbf{B})\right)\right]\right)^T\mathsf{d}\mathbf{G}_{(1)}^{F_1,F_3F_2}\right)
\end{aligned}
\tag{6.43}
$$

On en déduit les gradients suivants :

$$\frac{\partial \mathcal{T}_c}{\partial \mathbf{A}} = -4\mathbf{A} \boxdot \left[\boldsymbol{\delta}_{(1)} \left((\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{B} \boxdot \mathbf{B}) \right) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right)^T \right] \tag{6.44}$$

$$\frac{\partial \mathcal{T}_c}{\partial \mathbf{B}} = -4\mathbf{B} \boxdot \left[\boldsymbol{\delta}_{(2)} \left((\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{A} \boxdot \mathbf{A}) \right) \left(\mathbf{G}_{(2)}^{F_2,F_3F_1} \boxdot \mathbf{G}_{(2)}^{F_2,F_3F_1} \right)^T \right] \tag{6.45}$$

$$\frac{\partial \mathcal{T}_c}{\partial \mathbf{C}} = -4\mathbf{C} \boxdot \left[\boldsymbol{\delta}_{(3)} \left((\mathbf{B} \boxdot \mathbf{B}) \otimes (\mathbf{A} \boxdot \mathbf{A}) \right) \left(\mathbf{G}_{(3)}^{F_3,F_2F_1} \boxdot \mathbf{G}_{(3)}^{F_3,F_2F_1} \right)^T \right] \tag{6.46}$$

$$\frac{\partial \mathcal{T}_c}{\partial \mathbf{G}_{(1)}^{F_1,F_3F_2}} = -4\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \left[(\mathbf{A} \boxdot \mathbf{A})^T \boldsymbol{\delta}_{(1)} \left((\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{B} \boxdot \mathbf{B}) \right) \right] \tag{6.47}$$

6.6.2 Calcul du pas optimal

On considère la fonction de coût suivante :

$$\mathcal{T}_c(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{G}) = \| \mathbf{T}_{(1)}^{I,KJ} - [(\mathbf{A} + \mu\mathbf{D_A}) \boxdot (\mathbf{A} + \mu\mathbf{D_A})] \left[\left(\mathbf{G}_{(1)}^{F_1,F_3F_2} + \mu\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \boxdot \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} + \mu\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \right.$$
$$\left. [((\mathbf{C} + \mu\mathbf{D_C}) \boxdot (\mathbf{C} + \mu\mathbf{D_C})) \odot ((\mathbf{B} + \mu\mathbf{D_B}) \boxdot (\mathbf{B} + \mu\mathbf{D_B}))]^T \right] \|_F^2$$

Développons :

$$\begin{aligned}
\mathcal{T}_c(.) = \| \mathbf{T}_{(1)}^{I,KJ} - \Big[& (\mathbf{A} \boxdot \mathbf{A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right) \\
& - \mu \left(2 (\mathbf{A} \boxdot \mathbf{A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) + 2 (\mathbf{A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right) \right) \\
& - \mu^2 \left((\mathbf{A} \boxdot \mathbf{A}) \left(\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) + 4 (\mathbf{A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \right. \\
& \left. + (\mathbf{D_A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right) \right) \\
& - \mu^3 \left(2 (\mathbf{A} \boxdot \mathbf{D_A}) \left(\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) + (\mathbf{D_A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \right) \\
& - \mu^4 (\mathbf{D_A} \boxdot \mathbf{D_A}) \left(\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \Big] \\
& \Big[((\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{B} \boxdot \mathbf{B}))^T \\
& + \mu \left(2 (\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{B} \boxdot \mathbf{D_B}) + ((\mathbf{C} \boxdot \mathbf{D_C}) \otimes (\mathbf{B} \boxdot \mathbf{B})) \right)^T \\
& + \mu^2 4 (\mathbf{C} \boxdot \mathbf{D_C}) \otimes (\mathbf{B} \boxdot \mathbf{D_B}) + (\mathbf{D_C} \boxdot \mathbf{D_C}) \otimes (\mathbf{B} \boxdot \mathbf{B}) + (\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{D_B} \boxdot \mathbf{D_B}) \Big] \|_F^2
\end{aligned}$$

En posant :

$$
\begin{aligned}
\mathbf{E_0} &= (\mathbf{A} \boxdot \mathbf{A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right) \\
\mathbf{E_1} &= 2\,(\mathbf{A} \boxdot \mathbf{A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) + 2\,(\mathbf{A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right) \\
\mathbf{E_2} &= (\mathbf{A} \boxdot \mathbf{A}) \left(\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) + 4\,(\mathbf{A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \\
&\quad + (\mathbf{D_A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{G}_{(1)}^{F_1,F_3F_2} \right) \\
\mathbf{E_3} &= 2\,(\mathbf{A} \boxdot \mathbf{D_A}) \left(\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) + (\mathbf{D_A} \boxdot \mathbf{D_A}) \left(\mathbf{G}_{(1)}^{F_1,F_3F_2} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \\
\mathbf{E_4} &= (\mathbf{D_A} \boxdot \mathbf{D_A}) \left(\mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \boxdot \mathbf{D}_{\mathbf{G}_{(1)}^{F_1,F_3F_2}} \right) \\
\mathbf{F_0} &= (\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{B} \boxdot \mathbf{B}) \\
\mathbf{F_1} &= 2\,(\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{B} \boxdot \mathbf{D_B}) + ((\mathbf{C} \boxdot \mathbf{D_C}) \otimes (\mathbf{B} \boxdot \mathbf{B})) \\
\mathbf{F_2} &= 4\,(\mathbf{C} \boxdot \mathbf{D_C}) \otimes (\mathbf{B} \boxdot \mathbf{D_B}) + (\mathbf{D_C} \boxdot \mathbf{D_C}) \otimes (\mathbf{B} \boxdot \mathbf{B}) + (\mathbf{C} \boxdot \mathbf{C}) \otimes (\mathbf{D_B} \boxdot \mathbf{D_B})
\end{aligned}
$$

On obtient :

$$
\begin{aligned}
\mathcal{T}_c(.) = \| & \left(-\mathbf{E}_4\mathbf{F}_4^T\right) \mu^8 \\
& + \left(-\mathbf{E}_4\mathbf{F}_3^T - \mathbf{E}_3\mathbf{F}_4^T\right) \mu^7 \\
& + \left(-\mathbf{E}_4\mathbf{F}_2^T - \mathbf{E}_3\mathbf{F}_3^T - \mathbf{E}_2\mathbf{F}_4^T\right) \mu^6 \\
& + \left(-\mathbf{E}_4\mathbf{F}_1^T - \mathbf{E}_3\mathbf{F}_2^T - \mathbf{E}_2\mathbf{F}_3^T - \mathbf{E}_1\mathbf{F}_4^T\right) \mu^5 \\
& + \left(-\mathbf{E}_4\mathbf{F}_0^T - \mathbf{E}_3\mathbf{F}_1 - \mathbf{E}_2\mathbf{F}_2^T - \mathbf{E}_1\mathbf{F}_3^T - \mathbf{E}_0\mathbf{F}_4^T\right) \mu^4 \\
& + \left(-\mathbf{E}_3\mathbf{F}_0^T - \mathbf{E}_2\mathbf{F}_1^T - \mathbf{E}_1\mathbf{F}_2^T - \mathbf{E}_0\mathbf{F}_3^T\right) \mu^3 \\
& + \left(-\mathbf{E}_2\mathbf{F}_0 - \mathbf{E}_1\mathbf{F}_1^T - \mathbf{E}_0\mathbf{F}_2^T\right) \mu^2 \\
& + \left(-\mathbf{E}_1\mathbf{F}_0 - \mathbf{E}_0\mathbf{F}_1^T\right) \mu^1 \\
& + \mathbf{T}_{(1)}^{I,KJ} - \mathbf{E}_0\mathbf{F}_0\|_F^2
\end{aligned}
$$

Si on pose :

$$
\begin{aligned}
\mathbf{K_0} &= \mathbf{T}_{(1)}^{I,KJ} - \mathbf{E_0}\mathbf{F_0}^T \\
\mathbf{K_1} &= -\mathbf{E_0}\mathbf{F_1}^T - \mathbf{E_1}\mathbf{F_0}^T \\
\mathbf{K_2} &= -\mathbf{E_0}\mathbf{F_2}^T - \mathbf{E_1}\mathbf{F}_1^T - \mathbf{E_2}\mathbf{F_0}^T \\
\mathbf{K_3} &= -\mathbf{E_1}\mathbf{F_2}^T - \mathbf{E_2}\mathbf{F_1}^T \\
\mathbf{K_4} &= -\mathbf{E_2}\mathbf{F_2}^T \\
\mathbf{K_5} &= -\mathbf{E}_4\mathbf{F}_1^T - \mathbf{E}_3\mathbf{F}_2^T - \mathbf{E}_2\mathbf{F}_3^T - \mathbf{E}_1\mathbf{F}_4^T \\
\mathbf{K_6} &= -\mathbf{E}_4\mathbf{F}_2^T - \mathbf{E}_3\mathbf{F}_3^T - \mathbf{E}_2\mathbf{F}_4^T \\
\mathbf{K_7} &= -\mathbf{E}_4\mathbf{F}_3^T - \mathbf{E}_3\mathbf{F}_4^T \\
\mathbf{K_8} &= -\mathbf{E}_4\mathbf{F}_4^T
\end{aligned}
$$

on arrive à :

$$
\begin{aligned}
\mathcal{T}_c(.) = \mathsf{trace}\Big\{ & \left(\mathbf{K_0} + \mu\mathbf{K_1} + \mu^2\mathbf{K_2} + \mu^3\mathbf{K_3} + \mu^4\mathbf{K_4} + \mu^5\mathbf{K_5} + \mu^6\mathbf{K_6} + \mu^7\mathbf{K_7} + \mu^8\mathbf{K_8}\right) \\
& \left(\mathbf{K_0} + \mu\mathbf{K_1} + \mu^2\mathbf{K_2} + \mu^3\mathbf{K_3} + \mu^4\mathbf{K_4} + \mu^5\mathbf{K_5} + \mu^6\mathbf{K_6} + \mu^7\mathbf{K_7} + \mu^8\mathbf{K_8}\right)^T \Big\} \\
\mathcal{T}_c(.) = \mathsf{trace}\Big\{ & \left(\mathbf{K_8}\mathbf{K_8}^T\right)\mu^{16} + \\
& \left(2\mathbf{K_7}\mathbf{K_8}^T\right)\mu^{15} + \\
& \left(2\mathbf{K_6}\mathbf{K_8}^T + \mathbf{K_7}\mathbf{K_7}^T\right)\mu^{14} + \\
& \left(2\left(\mathbf{K_5}\mathbf{K_8}^T + \mathbf{K_6}\mathbf{K_7}^T\right)\right)\mu^{13} + \\
& \left(2\left(\mathbf{K_4}\mathbf{K_8}^T + \mathbf{K_5}\mathbf{K_7}^T\right) + \mathbf{K_6}\mathbf{K_6}^T\right)\mu^{12} + \\
& \left(2\left(\mathbf{K_3}\mathbf{K_8}^T + \mathbf{K_4}\mathbf{K_7}^T + \mathbf{K_5}\mathbf{K_6}^T +\right)\right)\mu^{11} + \\
& \left(2\left(\mathbf{K_2}\mathbf{K_8}^T + \mathbf{K_3}\mathbf{K_7}^T + \mathbf{K_4}\mathbf{K_6}^T\right) + \mathbf{K_5}\mathbf{K_5}^T\right)\mu^{10} + \\
& \left(2\left(\mathbf{K_1}\mathbf{K_8}^T + \mathbf{K_2}\mathbf{K_7}^T + \mathbf{K_3}\mathbf{K_6}^T + \mathbf{K_4}\mathbf{K_5}^T\right)\right)\mu^9 + \\
& \left(2\left(\mathbf{K_0}\mathbf{K_8}^T + \mathbf{K_1}\mathbf{K_7}^T + \mathbf{K_2}\mathbf{K_6}^T + \mathbf{K_3}\mathbf{K_5}^T\right) + \mathbf{K_4}\mathbf{K_4}^T\right)\mu^8 \\
& \left(2\left(\mathbf{K_0}\mathbf{K_7}^T + \mathbf{K_1}\mathbf{K_6}^T + \mathbf{K_2}\mathbf{K_5}^T + \mathbf{K_3}\mathbf{K_4}^T\right)\right)\mu^7 + \\
& \left(2\left(\mathbf{K_2}\mathbf{K_4}^T + \mathbf{K_1}\mathbf{K_5}^T + \mathbf{K_0}\mathbf{K_6}^T\right) + \mathbf{K_3}\mathbf{K_3}^T\right)\mu^6 + \\
& \left(2\left(\mathbf{K_1}\mathbf{K_4}^T + \mathbf{K_2}\mathbf{K_3}^T\right)\right)\mu^5 + \\
& \left(2\left(\mathbf{K_4}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_3}^T\right) + \mathbf{K_2}\mathbf{K_2}^T\right)\mu^4 + \\
& \left(2\left(\mathbf{K_3}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_2}^T\right)\right)\mu^3 + \\
& \left(\mathbf{2K_2}\mathbf{K_0}^T + \mathbf{K_1}\mathbf{K_1}^T\right)\mu^2 + \\
& \left(2\mathbf{K_1}\mathbf{K_0}^T\right)\mu + \\
& \mathbf{K_0}\mathbf{K_0^T}\Big\}
\end{aligned}
$$

On identifie ainsi les 17 coefficients $a_0 ... a_{16}$.

Annexe 3 : calculs liés aux données manquantes

6.7 Calcul des gradients matriciels avec contrainte de non négativité et en présence de données manquantes

Calculons les dérivées partielles $\frac{\partial \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda})}{\partial \mathbf{A}}$, $\frac{\partial \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda})}{\partial \mathbf{B}}$ et $\frac{\partial \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda})}{\partial \mathbf{C}}$ données par le calcul de la différentielle suivante :

$$\mathsf{d}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = \langle \frac{\partial \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda})}{\partial \mathbf{A}}, \mathsf{d}\mathbf{A} \rangle + \langle \frac{\partial \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda})}{\partial \mathbf{B}}, \mathsf{d}\mathbf{B} \rangle + \langle \frac{\partial \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda})}{\partial \mathbf{C}}, \mathsf{d}\mathbf{C} \rangle.$$

On part de la fonction de coût prenant en compte la présence de données manquantes :

$$\begin{aligned} \mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= \|\mathbf{W}_{(1)}^{I,KJ} \boxdot \left(\mathbf{T}_{(1)}^{I,KJ} - (\mathbf{A} \boxdot \mathbf{A})((\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{B}))^T\right) \|_F^2 \\ &= \|\mathbf{W}_{(1)}^{I,KJ} \boxdot \delta_{(1)}\|_F^2 \\ &= \|\boldsymbol{\beta}_{(1)}\|_F^2 \end{aligned} \tag{6.48}$$

En posant :

$$\begin{aligned} \mathbf{M}_1 &= (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{B} \boxdot \mathbf{B}), \\ \mathbf{M}_2 &= (\mathbf{C} \boxdot \mathbf{C}) \odot (\mathbf{A} \boxdot \mathbf{A}), \\ \mathbf{M}_3 &= (\mathbf{B} \boxdot \mathbf{B}) \odot (\mathbf{A} \boxdot \mathbf{A}), \end{aligned}$$

on peut réécrire l'équation (6.48) et ensuite calculer sa différentielle :

$$
\begin{aligned}
&\mathsf{d}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) = 2\mathsf{trace}\{\boldsymbol{\beta}_{(1)}^T \mathsf{d}\boldsymbol{\beta}_{(1)}\} + 2\mathsf{trace}\{\boldsymbol{\beta}_{(2)}^T \mathsf{d}\boldsymbol{\beta}_{(2)}\} \\
&+ 2\mathsf{trace}\{\boldsymbol{\beta}_{(3)}^T \mathsf{d}\boldsymbol{\beta}_{(3)}\} \\
&= 2\mathsf{trace}\{\boldsymbol{\beta}_{(1)}^T \mathsf{d}[\mathbf{W}_{(1)}^{I,KJ} \boxdot (\mathbf{T}_{(1)}^{I,KJ} - (\mathbf{A}\boxdot\mathbf{A})\mathbf{M}_1^T)]\} \\
&+ 2\mathsf{trace}\{\boldsymbol{\beta}_{(2)}^T \mathsf{d}[\mathbf{W}_{(2)}^{J,KI} \boxdot (\mathbf{T}_{(2)}^{J,KI} - (\mathbf{B}\boxdot\mathbf{B})\mathbf{M}_2^T)]\} \\
&+ 2\mathsf{trace}\{\boldsymbol{\beta}_{(3)}^T \mathsf{d}[\mathbf{W}_{(3)}^{K,JI} \boxdot (\mathbf{T}_{(3)}^{K,JI} - (\mathbf{C}\boxdot\mathbf{C})\mathbf{M}_3^T)]\} \\
&= 4\mathsf{trace}\{-\boldsymbol{\beta}_{(1)}^T \left[\mathbf{W}_{(1)}^{I,KJ} \boxdot \left((\mathbf{A}\boxdot \mathsf{d}\mathbf{A})\mathbf{M}_1^T\right)\right]\} \\
&+ 4\mathsf{trace}\{-\boldsymbol{\beta}_{(2)}^T \left[\mathbf{W}_{(2)}^{J,KI} \boxdot \left((\mathbf{B}\boxdot \mathsf{d}\mathbf{B})\mathbf{M}_2^T\right)\right]\} \\
&+ 4\mathsf{trace}\{-\boldsymbol{\beta}_{(3)}^T \left[\mathbf{W}_{(3)}^{K,JI} \boxdot \left((\mathbf{C}\boxdot \mathsf{d}\mathbf{C})\mathbf{M}_3^T\right)\right]\} \\
&= 4\mathsf{trace}\{-\boldsymbol{\beta}_{(1)}^T \boxdot \mathbf{W}_{(1)}^{I,KJ^T} (\mathbf{A}\boxdot \mathsf{d}\mathbf{A})\mathbf{M}_1^T\} \\
&+ 4\mathsf{trace}\{-\boldsymbol{\beta}_{(2)}^T \boxdot \mathbf{W}_{(2)}^{J,KI^T} (\mathbf{B}\boxdot \mathsf{d}\mathbf{B})\mathbf{M}_2^T\} \\
&+ 4\mathsf{trace}\{-\boldsymbol{\beta}_{(3)}^T \boxdot \mathbf{W}_{(3)}^{K,JI^T} (\mathbf{C}\boxdot \mathsf{d}\mathbf{C})\mathbf{M}_3^T\} \\
&= 4\mathsf{trace}\{\mathbf{M}_1^T(-\boldsymbol{\beta}_{(1)} \boxdot \mathbf{W}_{(1)}^{I,KJ})^T (\mathbf{A}\boxdot \mathsf{d}\mathbf{A})\} \\
&+ 4\mathsf{trace}\{\mathbf{M}_2^T(-\boldsymbol{\beta}_{(2)} \boxdot \mathbf{W}_{(2)}^{J,KI})^T (\mathbf{B}\boxdot \mathsf{d}\mathbf{B})\} \\
&+ 4\mathsf{trace}\{\mathbf{M}_3^T(-\boldsymbol{\beta}_{(3)} \boxdot \mathbf{W}_{(3)}^{K,JI})^T (\mathbf{C}\boxdot \mathsf{d}\mathbf{C})\} \\
&= 4\mathsf{trace}\{\mathbf{M}_1^T(-\boldsymbol{\beta}_{(1)} \boxdot \mathbf{W}_{(1)}^{I,KJ})^T \boxdot \mathbf{A}^T \mathsf{d}\mathbf{A})\} \\
&+ 4\mathsf{trace}\{\mathbf{M}_2^T(-\boldsymbol{\beta}_{(2)} \boxdot \mathbf{W}_{(2)}^{J,KI})^T \boxdot \mathbf{B}^T \mathsf{d}\mathbf{B})\} \\
&+ 4\mathsf{trace}\{\mathbf{M}_3^T(-\boldsymbol{\beta}_{(3)} \boxdot \mathbf{W}_{(3)}^{K,JI})^T \boxdot \mathbf{C}^T \mathsf{d}\mathbf{C})\} \\
&= \langle 4\left[\mathbf{A}\boxdot\left((-\boldsymbol{\beta}_{(1)} \boxdot \mathbf{W}_{(1)}^{I,KJ})\mathbf{M}_1\right)\right], \mathsf{d}\mathbf{A}\rangle \\
&+ \langle 4\left[\mathbf{B}\boxdot\left((-\boldsymbol{\beta}_{(2)} \boxdot \mathbf{W}_{(2)}^{J,KI})\mathbf{M}_2\right)\right], \mathsf{d}\mathbf{B}\rangle \\
&+ \langle 4\left[\mathbf{C}\boxdot\left((-\boldsymbol{\beta}_{(3)} \boxdot \mathbf{W}_{(3)}^{K,JI})\mathbf{M}_3\right)\right], \mathsf{d}\mathbf{C}\rangle
\end{aligned}
$$

On en déduit donc les trois matrices de gradients suivants :

$$
\begin{aligned}
\nabla_{\mathbf{A}}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= \frac{\partial\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{A}} = 4\mathbf{A}\boxdot\left[(-\boldsymbol{\beta}_{(1)} \boxdot \mathbf{W}_{(1)}^{I,KJ})\left((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{B}\boxdot\mathbf{B})\right)\right] \\
\nabla_{\mathbf{B}}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= \frac{\partial\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{B}} = 4\mathbf{B}\boxdot\left[(-\boldsymbol{\beta}_{(2)} \boxdot \mathbf{W}_{(2)}^{J,KI})\left((\mathbf{C}\boxdot\mathbf{C})\odot(\mathbf{A}\boxdot\mathbf{A})\right)\right] \\
\nabla_{\mathbf{C}}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C};\mathbf{\Lambda}) &= \frac{\partial\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial\mathbf{C}} = 4\mathbf{C}\boxdot\left[(-\boldsymbol{\beta}_{(3)} \boxdot \mathbf{W}_{(3)}^{K,JI})\left((\mathbf{B}\boxdot\mathbf{B})\odot(\mathbf{A}\boxdot\mathbf{A})\right)\right]
\end{aligned}
$$