Adéquation algorithme architecture pour l'accélération de méthodes d'inversion de données en grande dimension

Nicolas Gac

université
PARIS-SACLAY

Adéquation algorithme architecture pour l'accélération de méthodes d'inversion de données en grande dimension

Mémoire pour l'obtention de

l'habilitation à diriger des recherches de l'Université Paris Saclay

École doctorale n° 580 Sciences et technologies de l'information et de la communication (STIC)

Spécialité : Traitement du Signal

Laboratoire des Signaux et Systèmes (L2S)

soutenue le 23 novembre 2020, par

Nicolas GAC

Composition du jury :

Jérôme Idier Directeur de Recherche, CNRS (L2SN)	Rapporteur
Olivier Romain Professeur, Université Cergy-Pontoise (ETIS)	Rapporteur
Michaël Krajecki Professeur, Université de Reims Champagne-Ardenne (CReSTIC)	Rapporteur
André Ferrari Professeur, Université Côte d'Azur (Lagrange)	Examinateur
Stéphane Vialle Professeur, CentraleSupelec (LRI)	Examinateur
Alain Mérigot Professeur, Université Paris Saclay (SATIE)	Examinateur
Françoise Peyrin Directrice de recherche, INSERM (CREATIS)	Présidente

Remerciements

Je remercie tout d'abord les rapporteurs d'avoir accepté d'examiner ce manuscrit. Leurs remarques contenues dans leurs rapports m'ont été très utiles ; je remercie également les examinateurs d'avoir accepté de participer au jury. Je remercie tous les membres du jury pour la discussion suite à la soutenance que j'ai particulièrement apprécié malgré les conditions particulières dans lesquelles elles se sont déroulées, malheureusement entièrement à distance.

Je tiens à remercier le laboratoire L2S pour m'avoir accueilli depuis 2009. Mes remerciements vont tout particulièrement au Groupe Problèmes Inverses où mon activité de recherche a pu s'épanouir. Merci à Ali Djafari pour m'avoir accompagné tout au long de mes premières années au GPI ; merci à Thomas Rodet pour ses bons conseils ; merci à François Orieux et Charles Soussen pour leur soutien lors ma préparation pour cette HDR. Cette habilitation à diriger des travaux est en grande partie le fruit des travaux des valeureux doctorants et étudiants en stage que j'ai eu le plaisir d'encadrer ; Merci à Ning Chu, Thomas Boulay, Long Chen, Li Wang, Camille Chapdeplaine, Maxime Martelli, Mickael Seznec, Daouda Diakite et Nicolas Monnier. Je tiens à les remercier sincèrement pour leurs implications dans leurs travaux de thèse ; les accompagner jusqu'à l'accouchement de leur manuscrit de thèse devient alors une vraie source de satisfaction pour leurs encadrants.

Table des matières

Table des figures

Liste des tableaux

Organisation du manuscrit

Ce manuscrit est composé de deux parties : la première présente mon parcours de chercheur depuis mon obtention du doctorat en 2008 ; la deuxième synthétise mes travaux de recherche effectués au sein du Groupe Problèmes Inverses (GPI).

Dans la première partie, après une synthèse de ma carrière, mes activités scientifique et pédagogique ainsi que les responsabilités collectives auxquelles j'ai pu prendre part sont exposées. Mes participations à des encadrements de thèse et la liste de mes publications* sont respectivement détaillées dans les sections 2 et 3 des activités scientifiques.

Dans la deuxième partie, mes travaux sont exposés en cinq chapitres :

Le chapitre 1 présente la problématique de l'accélération sur cartes GPU et FPGA, puis mon positionnement par rapport au domaine de l'Adéquation Algorithme Architecture.

Le chapitre 2 pose le contexte algorithmique des problèmes inverses en grande dimension avec une description des méthodes bayésiennes employées. Mes contributions à l'application de ces méthodes en reconstruction tomographique y sont exposées. La présentation des enjeux algorithmiques pour la radioastronomie clôt ce chapitre.

Le chapitre 3 présente mes contributions pour l'accélération sur GPU et FPGA des opérateurs utilisés par les algorithmes itératifs bayésiens : paire de projection et rétroprojection $\mathcal{P}/\mathcal{R}$ en reconstruction tomographique et convolution $\mathcal{C}$ en radioastronomie.

Le chapitre 4 présente les stratégies de parallélisation sur serveurs multi-GPU mises en oeuvre pour y faire résider l'ensemble des boucles itératives faisant appel aux opérateurs $\mathcal{P}/\mathcal{R}$ en tomographie et $\mathcal{C}$ en déconvolution.

Le chapitre 5 dresse des perspectives à court et moyen terme à ces travaux.

Enfin, troix annexes et une bibliographie viennent compléter cette deuxième partie : l'annexe A expose les principaux modèles de projection et rétroprojection utilisés en reconstruction tomographique ; l'annexe B présente la liste des serveurs et cartes accélératrices utilisées ; l'annexe C présente les logiciels de reconstruction tomographique développés depuis mon arrivée au GPI.

* les citations à mes publications [R$_{\text{année}}$] et [C$_{\text{année}}$] possèdent un lien cliquable redirigeant vers leur référence HAL avec accès au pdf (en haut à droite dans le cadre fichier).

Première partie

Parcours depuis le doctorat

Synthèse de carrière

Etat civil

Né le 04 Décembre 1979 à Rennes (35)
Nationalité Française
Marié, 2 enfants
Résident à Paris 13ième

Situation actuelle

Depuis le 01/09/2009, je suis maître de conférences en section 61 à l'université Paris-Saclay*(7ième échelon au 15/08/2020). J'enseigne à l'IUT de Cachan situé à 25 km de mon laboratoire de recherche, le L2S (UMR 8506 CentraleSupélec-CNRS-UPSaclay).

Enseignement *systèmes numériques embarqués* (architectures microcontroleurs et FPGA ; langages C et VHDL) au département de Génie Electrique Informatique Industrielles (GEII) ; *calcul parallèle sur GPU* au M2 SETI de l'UFR des sciences.

Recherche Groupe Problèmes Inverses (GPI) du Pôle Signaux et Statistiques au L2S : *adéquation algorithme architecture, calcul parallèle, processeurs many cores, GPU, FPGA, synthèse de haut niveau, problèmes inverses, reconstruction tomographique, déconvolution pour la radioastronomie.*

Coordonnées

Page web : **https://www.l2s.centralesupelec.fr/u/gac-nicolas/**

Coordonnées L2S

Adr.	L2S (UMR 8506 UPSaclay/CNRS/CS) CentraleSupélec 3, rue Joliot-Curie 91192 Gif-sur-Yvette cedex
Tel.	01 69 85 17 38
E-mail	nicolas.gac@l2s.centralesupelec.fr

* l'université Paris Sud est devenue l'université Paris Saclay au 1er janvier 2020

Coordonnées IUT Cachan

Adr.	IUT de Cachan
	9 av. de la division Leclerc
	94230 Cachan
Tel.	01 41 24 11 48
E-mail	nicolas.gac@universite-paris-saclay.fr

Cursus Universitaire et Diplômes

2020 Inscription à l'HDR à l'université Paris Saclay

2018 Obtention de la PEDR

2008 Doctorat en Traitement du Signal et des Images, Grenoble-INP

Titre :*Adéquation Algorithme Architecture pour la reconstruction 3D en imagerie médicale TEP.*

Mots-clés : reconstruction tomographique, calcul parallèle, FPGA, SoPC, GPU, cache mémoire.

Soutenue le 17 juillet 2008 devant le jury composé de :

D. Demigny, Pr. Univ. Rennes, ENSSAT (CNU 61)	rapporteur
Y. Mathieu, Pr. Telecom Paristech, LTCI	rapporteur
L. Desbat, Pr. Univ. J. Fourier, TIMC (CNU 26)	président
B. Granado, Pr. Unv. Cergy, ETIS (CNU 63)	examinateur
M. Desvignes, Pr. INP Grenoble, Gipsa-lab (CNU 27)	directeur
S. Mancini, Mcf INP Grenoble, Gipsa-lab (CNU 61)	co-enc.

2004 Master Signal Image Parole et Télécommunications, Grenoble-INP.

2003 DEA en Microélectronique, UJF, Grenoble

2003 Ingénieur en Télécommunication, ENSIMAG, Grenoble-INP

1996 Classes Préparatoires MP, lycée Henri Bergson, Angers (49)

Parcours Professionnel

2017-19 Délégation CNRS (50%) à l'Observatoire de la Côte d'Azur

2009-... **Maître de conférences à l'université Paris Saclay**

2008-09 Post-Doctorat, Laboratoire L2S et CEA-List, Saclay (91)
Parallélisation GPU pour la reconstruction tomographique

2007-08 ATER à l'université de Cergy, IUT Neuville/laboratoire ETIS

2003-07 Vacataire/moniteur à Grenoble-INP, ENSERG/ENSIMAG

Activité scientifique chiffrée

Revues avec comité de lecture	Congrès avec actes	Brevets	Logiciels
10 Int. / **2** Nat.	**41** Int. / **10** Nat.	**1**	**3**

TABLE 1 – Production scientifique

Thèses		Post-doc	Stages M2 en labo	Parcours Recherche	Stages L2/M1
soutenues	en cours				
6 *>50% (3)* *[30%, 50%[(3)*	**3** *> 40% (3)* *1 en dir.*	**3** *>25% (3)*	**13** *100% (6)* *50% (6)*	**2** 80% (1) 50% (1)	**4** M1 **4** L2

TABLE 2 – Encadrement doctoral et scientifique

Projet		**Période**	**Budget**	**Rôle**
PRC DARK-ERA	ANR	2021-25	500 kE	Coordinateur Principal
PRC MAGELLAN	ANR	2015-18		Participant
PEPS SKALLAS	CNRS	2018-19	20 kE	Porteur
Mastodon Hyperstars	CNRS	2017-18		Participant
ExaSKA	Région IdF	2020-22	100 kE	Resp. scientifique
SAFRAN, Thalès et Atos-Bull		2016-22	320 kE	Resp. scientifique
Trophy et Thalès		2010-14		Participant

TABLE 3 – Contrats de recherche

Activité pédagogique (faits marquants)

2010-17 Création de projets originaux pour le parcours S4-TI

2011-16 Responsable pédagogique du S4-TI (24 étud.) - *Prime 12h*

2015-... Création de la maquette d'Etude et Réalisation PARSEC

2017-... Responsable pédagogique du S2 (100 étud.) - *Prime 48h*

2018 Apprentissage Par Problèmes expérimenté pour l'UE SNE

2018-... Responsable de l'UE *GPU/FPGA* (5 ECTS) du M2 SETI

Principales responsabilités collectives

2013-15 Membre élu du conseil de département GEII - IUT Cachan

2014-21 Membre élu de la CCSU* 60/61/62 de l'université Paris Sud

2018-... Responsable du pôle calcul scientifique du L2S

2020-24 Membre élu du comité de direction du L2S (6 membres)

* mêmes missions que celles du CNU mais au niveau local

Activité scientifique

Sommaire

Mes travaux de recherche en Adéquation Algorithme Architecture au sein du Groupe Problèmes Inverses (GPI) au L2S ont débuté en 2008 en tant que post-doc puis maître de conférences. Ils sont focalisés sur l'accélération sur cartes GPU ou FPGA des problèmes inverses en grande dimension. Ils sont principalement appliqués à la reconstruction tomographique et à la radioastronomie. Ce dernier champs applicatif est exploré depuis une délégation CNRS en 2017-2019 au laboratoire Lagrange de l'Observatoire de la Côte d'Azur (OCA). Durant cette période, j'y ai effectué régulièrement des séjours courts afin de développer cette nouvelle collaboration.

1 Responsabilités, animation et rayonnement

Responsabilités scientifiques pour des contrats de recherche

Gestion cumulée de ∼550 kE en tant que responsable scientifique

2021-25 **ANR DARK-ERA** : *Dataflow Algorithm aRchitecture co-design of SKA pipeline for Exascale Radio Astronomy*. Porteur de cette collaboration (500 kE) entre les laboratoires IETR, IRISA, Labgrange, Obs. Nançay et L2S.

2020-24 **Plateforme expérimentale SACHEMS** : SAClay High-end Equipment for the Monitoring of Structures, financement SESAME de l'IdF piloté par le CEA-List, *Membre du comité de pilotage*

2019-22 **Région Ile de France + Atos Bull** : 100kE (région) + complément d'Atos Bull, porteur du projet ExaSKA

2019-22 **Université Paris Sud** : Bourse de l'ED STIC de l'université Paris Saclay (100 kE), directeur de la thèse de Daouda Diakite (en attente de passage de l'HDR)

2018-19 **CNRS - PEPS Astro Info** : 20kE sur 2 ans porteur du projet SKALLAS (SKA paraLLel Architecture & Software) impliquant 4 laboratoires (L2S, Lagrange, IETR, Obs Paris) - Adéquation Algorithme Architecture pour relever le défi de traitement des données de taille radioastronomique SKA

2018-21 **Thales TRT Palaiseau** (Thèse CIFRE) : responsable scientifique de cette collaboration industrielle portant sur l'élaboration d'un flot d'optimisations pour le calcul haute performance temps réel sur systèmes embarqués parallèles hétérogènes

2016-19 **Thales TSA Elancourt** (Thèse CIFRE) : responsable scientifique de cette collaboration industrielle portant sur l'accélération d'algorithmes de simulation radar sur FPGA en collaboration avec le laboratoire SATIE

2014-19 **SAFRAN** : responsable scientifique de ce contrat industriel portant sur la reconstruction tomographique de volume de grande taille sur serveur multi-GPU (post-doc 15 mois + thèse CIFRE)

Participation à des projets de recherche nationaux ou locaux (6)

2019-... **Plateforme TOMX**, membre du comité scientifique et d'animation de ce consortium non institutionnel entre laboratoires et industriels sur le plateau de Saclay autour de la Tomographie par Rayons X

2017-18 **Projet HyperFusion**, département STIC de l'université Paris Saclay impliquant 3 EC du L2S (15 kE)

2017-18 **Projet HyperStars**, Programme Mastodon CNRS, impliquant 6 C et EC de 3 laboratoires : CEA (M.A. Miville-Deschenes, porteur), le L2S et le LIP6

2015-18 **Projet ANR Magellan** participant en tant que partenaire ENS Paris Saclay, en collaboration avec Observatoire de la Côte d'Azur (porteur) et Télécom Paristech

2011-14 **Carestream Dental** (Thèse CIFRE) : membre de cette collaboration en tant que co-encadrant de thèse

2010-13 **Thales Limours** (Thèse CIFRE) : coordinateur de cette collaboration sous la responsabilité d'Ali Djafari (dir. de thèse)

Invitations à des jurys scientifiques

Examinateur à des jurys de thèses (4)

2019 Thèse de Samir Bouindour soutenue le 17 décembre 2019 à l'université technonologique de Troyes : *Apprentissage profond appliqué à la détection d'évènements anormaux dans les flux vidéos*

2017 Thèse de Thilbault Notargiacomo soutenue le 14 février 2017 à l'université Grenoble Alpes : *Optimisation d'implémentation d'algorithmes de reconstruction 3D interactif traditionnel et innovant, reposant sur le compressed sensing pour des architectures parallèles*

2016 Thèse de Mahmoud Soua soutenue le 8 novembre 2016 à l'université Paris-Est : *Extraction hybride et description structurelle de caractères pour une reconnaissance efficace de texte dans les documents hétérogènes scannés : Méthodes et Algorithmes parallèles*

2015 Thèse de Maxime Legendre soutenue le 22 avril 2015 à l'école centrale de Nantes : *Estimation des cartes d'abondances en imagerie hyperspectrale et l'utilisation des algorithmes d'optimisation sous contraintes et leur implémentation sur des GPU*

Autres jurys (3)

2020 Jury pour le grade d'ingénieur du CNAM pour F. Tounier : *Accélérateurs matériels FPGA/GPU pour le traitement d'image*

2018 Evaluation à mi-parcours de la thèse de C. Kervazo, 11/04/18 au CEA-Irfu à Paris Saclay : *Learning and image restauration*

2017 Evaluation à mi-parcours de la thèse de PA Rodesch, 16/10/17 2017 au CEA-Léti à Grenoble : *Méthodes statistiques de reconstruction tomographique multi-énergie pour des systèmes de détection spectrométriques*

Invitations pour donner des séminaires à audience locale (7), nationale (7) ou internationale (1)

2019 Atelier Hyperspectral, GEOPS (Orsay)

2019 Journée "Avancées et perspectives pour l'industrie du futur", Centralesupélec / *reconstruction tomographique pour le CND*

2018 GDR ISIS - Journée Radioastronomie SKA

2017 Séminaire Lagrange (Nice)

2017 Colloque Inversion de données spatiales, IRS Spaceobs, Saclay

2017 Séminaire LRDE de l'Epita, Le Kremlin-Bicêtre (94)

2016 **Meeting on Tomography (TAIR)** organisé par un COST (European Cooperation in Science and Technology), Milan

2016 Workshop Imagin'in MULTIPLANETO, GEOPS (Orsay)

2010-17 Ecole CNRS sur les GPUs, éditions 2010-2011-2015-2017

2012 Introduction (1h) d'une journée thématique du GDR ISIS : *Résolution de problèmes inverses : optimisation et parallélisation*

2012 Séminaire Gipsa-lab (INP Grenoble)

2009 Séminaire invité pour le colloque MI2B/CERIMED, Obernai

Organisation de conférences ou journées scientifiques

2019 Organisateur principal d'une journée au L2S en l'honneur d'Ali Djafari lors de son départ à la retraite (4 séminaires, 20/30 participants dont 50% d'extérieurs)

2014 Comité d'organisation de la conférence internationale MaxEnt

Travail d'expertise

Rapporteurs d'articles pour revues (8)

Eurasip Embedded System (2), Elsevier Parallel Computing (2), Springler Real Time processing (3), Elsevier Signal Processing (1)

Rapporteurs d'articles pour conférences

Gretsi (2017-2015) et MaxEnt (2014)

2 Encadrement de travaux de recherche

- Rapports et soutenances disponibles sur :
  https://www.l2s.centralesupelec.fr/u/gac-nicolas/phd/
- Le taux d'encadrement est indiqué sous la date de début [X%]

Thèses (9)

Direction de thèse en cours (1)

09/2018 [50%]
Mickael Seznec - Thèse CIFRE avec Thalès TRT à Palaiseau (91)
Directeur de thèse : N. Gac (Dérogation HDR)
Co-encadrants : F. Orieux (L2S) [40%] et A. Sashala (Thalès) [10%]
De l'algorithme à l'implémentation, élaboration d'un flot d'optimisations pour le calcul haute performance temps réel sur systèmes embarqués parallèles hétérogènes
Publications : [C_{2020b}], [C_{2020d}], [C_{2018b}]

Co-encadrement de thèse en cours (2)

12/2019 [40%]
Nicolas Monnier - Thèse Région Ile de France avec Atos-Bull
Directeur de thèse : C. Soussen (et N. Gac sous réserve HDR)
Co-encadrants : C. Tasse (Obs Paris) [30%] et F. Orieux (L2S) [30%]
ExaSKA : Parallelization on a High Performance Computing server for the exascale radiotelescope SKA

09/2019 [60%]
Daouda Diakite - Thèse Université Paris-sud
Directeur de thèse : T. Rodet [10%] (et N. Gac sous réserve HDR)
Co-encadrant : F. Orieux (L2S) [30%]
High level synthesis (HLS) on FPGA for image processing. Applications to tomography and radioastronomy.
Publications : [C_{2020a}]

Co-encadrement de thèse soutenues (4)

12/2016 [60%] *(3 ans)*
Maxime Martelli - Thèse CIFRE avec Thalès DMS (78)
Directeur de thèse : A. Mérigot (SATIE) [30%]
Co-encadrant : C. Enderli (Thalès) [10%]
Approche haut niveau pour l'accélération d'algorithmes sur des architectures hétérogènes CPU/GPU/FPGA. Application à la qualification des radars et des systèmes d'écoute électromagnétique.
Publications : [R_{2018a}], [C_{2020a}], [C_{2019b}], [C_{2018d}], [C_{2017g}]
Devenir : Ingénieur conseil

05/2016 [50%] *(3 ans)*
Camille Chapdelaine - Thèse CIFRE Safran Tech à Magny (78)
Directeur de thèse : A. Djafari (L2S) [40%]
Co-encadrante : E. Parra (Safran) [10%]
Reconstruction 3D par rayons X pour le Contrôle Non Destructif de pièces aéronautiques
Publications : [L_{2019}], [R_{2019b}], [R_{2017b}], [C_{2019a}], [C_{2019c}], [C_{2018c}], [C_{2018a}], [C_{2017b}], [C_{2017e}]
Devenir : Ingénieur R&D chez Safran Tech

10/2014 [40%] *(3 ans + 2 mois)*
Li Wang - Thèse CSC (China Scholarship Council)
Directeur de thèse : A. Djafari (L2S) [60%]
Fast and Accrate 3D X Ray image reconstruction for non destructive testing industrial applications
Publications : [R_{2018b}], [R_{2017a}], [C_{2017a}], [C_{2017c}], [C_{2017f}], [C_{2017j}], [C_{2016a}], [C_{2016b}], [C_{2015b}]
Devenir : Post-doc au L2S-Pôle Telecoms *(Deep Neural Network)*

11/2011 [30%] *(3 ans + 3 mois)*
Long Chen - Thèse CIFRE avec Carestream Dental à Lognes (77)
Directeur de thèse : T. Rodet (SATIE) [55%]
Co-encadrante : C. Maury (Carestream) [15%]
Méthodes itératives de reconstruction tomographique pour la réduction des artefacts métalliques et de la dose en imagerie dentaire
Publications : [C_{2014d}], [C_{2013d}]
Devenir : Ingénieur R&D en Chine (Drones)

11/2010 [50%] *(3 ans)*
Thomas Boulay - Thèse CIFRE avec Thales à Limours (91)
Directeur de thèse : A. Djafari (L2S) [50%]
Développement d'algorithme pour la fonction NCTR / Mise en oeuvre du calcul parallèle sur carte GPUs
Publications : [R_{2013}], [C_{2013a}], [C_{2012}]
Devenir : Ingénieur R&D chez Valéo (Véhicules autonomes)

10/2010 [30%] *(3 ans + 2 mois)*
Ning Chu - Thèse CSC (China Scholarship Council)
Directeur de thèse : A. Djafari (L2S) [40%]
Co-encadrant : J. Picheral (Supélec) [30%]
Méthodes de Super-résolution par deconvolution appliquées à la localisation de source
Publications : [R_{2014a}], [C_{2015c}], [C_{2014a}], [C_{2014b}], [C_{2014c}], [C_{2013c}]
Devenir : Enseignant/chercheur à Zhejiang University

Stages de recherche (15)

Stages de niveau M2 (13)

2020 Jérémy Besson
[70%] Co-encadrement avec F. Orieux
(4 mois) *Accélération sur GPU de l'algorithme ROHSA pour la séparation de sources de données hyperspectrales en radioastronomie*

2019 Mohammed Chghaf (M2R SETI - UPSaclay)
[100%] *Multi-GPU Parallelization of Iterative Reconstruction Methods in 3D X-ray Computed Tomography*
(6 mois) Publications : [C_{2020c}]

2019 Daouda Diakite (M2R SETI - UPSaclay)
[50%] Co-encadrement avec M. Martelli
(6 mois) *Accélération de la reconstruction tomographique : implémentation sur FPGA avec outils HLS du projecteur de Siddon*
Publications : [C_{2020a}]

2018 Nicolas Georgin (CentraleSupélec- stage de césure)
[50%] Co-encadrement avec C. Chapdelaine
(6 mois) *Parallélisation sur GPU d'une paire projecteur-rétroprojecteur pour le contrôle non-destructif en tomographie 3D par rayons X*
Publications : [C_{2019a}]

2018 Olivier Pérard (M2R Informatique - UPSaclay)
[100%] *Parallélisation sur machine de calcul multi-GPU d'algorithmes de*
(6 mois) *déconvolution et de séparation de source pour la radioastronomie (projet SKA)*
Remarque : Etudiant en situation d'handicap (autisme) encadré en 2013 lors de son S4-TI à l'IUT de Cachan. Le projet a été adapté à sa situation.

2017 Youssouf Samuet Bouhaïk (M2R SETI - UPSaclay)
[100%] *Parallélisation sur serveur de calcul intensif de méthodes d'appren-*
(6 mois) *tissage pour le très grand réseau d'antennes en radioastronomie SKA*

2014 Li Wang (M2R ATSI - UPSud)
[50%] *Fast and Accrate 3D X Ray image reconstruction for non destructive*
(6 mois) *testing industrial applications*
Devenir : poursuite en thèse au L2S

2011 Long Chen (M2R ATSI - UPSud)

[50%] *Développement d'une méthode itérative de reconstruction d'objet 3D*
(6 mois) *en imagerie dentaire à rayons X*
Devenir : poursuite en thèse au L2S

2010 Ning Chu (M2R ATSI - UPSud)
[50%] *Création d'une séquence d'images hautes résolutions à partir d'une*
(6 mois) *séquence d'images basses résolutions*
Devenir : poursuite en thèse au L2S

2011 Krayni Anis (Ecole d'ingénieur SUP'COM - Tunisie)
[100%] *Parallélisation sur un serveur multi-GPUs d'algorithmes de recons-*
(4 mois) *truction en tomographie 3*D
Devenir : poursuite en thèse

2010 Thomas Boulay (Master Spécialisé SIRF - Télécom ParisTech)
[50%] *Algorithmes de reconnaissance cibles non coopératifs et mise en*
(4 mois) *oeuvre en calcul parallèle sur cartes graphiques*
Publications : [C_{2011c}]
Devenir : poursuite en thèse au L2S

2009 Fei Wang (M2P SESIS - UPSud)
(4 mois) *Accélération d'un algorithme bayésien de restauration d'image*

2009 Asier Rabanal (Erasumus - UPV(Univ. Pays Basque)/UPSud)
[100%] *Accélération sur cartes graphiques de l'opérateur de projection*
(6 mois) Devenir : poursuite en CDD au L2S (9 mois) puis ingénieur

Parcours recherche Centralesupélec* **(2)**

2019 Jérémy Besson (CentraleSupelec)
[80%] Co-encadrement avec F.Orieux
(3 ans) *Adéquation Algorithme Architecture pour le radiotélescope Square Kilometer Array (SKA)*
Remarque : stage de 6 mois à Toronto dans le cadre du projet de recherche

2020 Jean Cohen (CentraleSupelec)
(1 an) Encadrement à 50% pour la deuxième année
Reconstruction tomographique pour l'imagerie médicale
Remarque : adaptation de notre logiciel TomoBayes [$L_{2015-2019}$] aux données de GE HealthCare (Buc)

∗ parcours pour étudiants de CentraleSupélec basé sur la conduite d'un projet de recherche sur trois ans au sein d'un des laboratoires de CentraleSupélec.

Stages courts (8)

Co-encadrement de stage M1 (4)

2019 Fabio Eid Morooka (2A INSA Rennes)
[50%] Encadrement à Co-encadrement F. Orieux
(4 mois) *Modélisation DASK de la deconvolution dans DDFacet (projet SKA)*

2018 Vincent Samy (2A CentraleSupélec)
(3 mois) *Comparaison de toolkit de reconstuction pour la Tomographie X*

2011 Aditya Gautam (IIIT Hyderabad, Inde)
[100%] *Optimisations des transferts CPU-GPU sur serveur 8 GPUs*
(4 mois) Devenir : ingénieur logiciel chez Google pour le machine learning

2010 Samsophath Nhean (M1 Physique Appliquée et Mécanique - UPSud)
[50%] Collaboration avec le laboratoire GEOPS
(4 mois) *Traitement des données spectrales de l'instrument Planetary Fourier Spectrometer de la sonde Mars Express*

Encadrement de stages L2 (4)

2013 XiangYang Gan (IUT Cachan - Dept GEII)
(2 mois) *Accélération sur GPU des méthodes itératives de déconvolution pour la localisation de sources acoustiques*
Publications : [C_{2014a}]
Devenir : Ecole d'ingénieur Télécom ParisTech

2012 Rémi Navarre (IUT Cachan - Dept GEII)
(2 mois) *Parallélisation multi-GPU de la deconvolution de données spectrales de Mars*
Devenir : ENS Cachan

2011 Alexandre Frizac (IUT Cachan - Dept GEII)
(2 mois) *Traitement de flux vidéo sur GPUs*
Devenir : Ecole d'ingénieur EPITECH

2010 Benoit Pencrec'h (IUT Cachan - Dept GEII)
(2 mois) *Accélération sur processeurs graphiques de la convolution 2D et 3D*
Devenir : Ecole d'ingenieur ISEP

Post-docs et ingénieurs d'étude (6)

Encadrement de post-doc (3)

2016 Mircea Dumitru - Post-doc (financement interne L2S)
[50%] Co-encadrement avec A. Djafari
(18 mois) *Méthodes bayésiennes de reconstruction tomographique*
Publications : [L_{2018}], [C_{2017d}], [C_{2017h}], [C_{2017i}], [C_{2017k}]

2014 Thomas Boulay - Post-doc sur un contrat industriel avec SAFRAN
[70%] Co-encadrement avec A. Djafari
(15 mois) *Reconstruction 3D par rayons X pour le Contrôle Non Destructif de pièces aéronautiques*
Devenir : Ingénieur en R&D chez Valeo (vehicules autonomes)

2013 Olivier Schwander - Post-doc DIGITEO
[25%] Co-encadrement avec J.Picheral et A. Djafari
(12 mois) *Développement de méthodes de déconvolution pour l'imagerie acoustique en milieu réverbérant et bruité*
Publications : [C_{2015a}]
Devenir : MCF à l'UPMC

Encadrement d'ingénieurs d'étude (3)

2018 Olivier Pérard
[100%] *Projet SKA*
(8 mois) Etudiant en situation d'handicap (autisme) encadré en 2013 lors de son S4-TI à l'IUT de Cachan et en 2018 lors de son M2 informatique. Le projet a été adapté à sa situation.

2018 Mickael Seznec
[60%] Co-encadrement avec F. Orieux
(7 mois) *Projet SKA*
Publications : [C_{2018b}]

2009 Asier Rabanal
[100%] *Reconstruction tomographique CT sur serveur multi-GPU*
(8 mois) Publications : [C_{2011a}], [C_{2011b}]

3 Production scientifique

- La liste de publications (avec pdf) est disponible sur ma page de chercheur et sur HAL https://cv.archives-ouvertes.fr/nicolas-gac.
- Les co-auteurs sous ma supervision apparaissent en rouge (doctorants), bleu (stagiaires) et magenta (post-doc).
- Les indicateurs Impact Factor (JCR) et SJR avec *Quartiles* (www.scimagojr.com) des revues sont données pour la période de publication des articles.

Revues avec comité de sélection (12)

[R$_{2019a}$] A. Marchal, M.A. Miville-Deschênes, F. Orieux, N. Gac, C. Soussen, M.J. Lesot, A. Revault d'Allonnes, Q. Salomé, **ROHSA : Regularized Optimization for Hyper-Spectral Analysis**, *Astronomy and Astrophysics - AA (2018 IF 6.209/SJR 2.527-Q1 Astronomy and Astrophysics), EDP Sciences, 2019, 626, pp.A101.*

[R$_{2019b}$] C. Chapdelaine, A. Mohammad-Djafari, N. Gac, E. Parra, **Error-Splitting Forward Model for Iterative Reconstruction in X-ray Computed Tomography and application with Gauss-Markov-Potts prior**, *IEEE Transactions on Computational Imaging (2018 IF 4.546/SJR 0.837-Q1 Computer science applications)*

[R$_{2018a}$] M. Martelli, N. Gac, A. Merigot, C. Enderli, **3D Tomography backprojection parallelization on Intel FPGAs using OpenCL**, *Journal of Signal Processing Systems (2018 IF 1.035/SJR 0.203-Q3 Hardware and Architecture), Springer, 2018*

[R$_{2018b}$] L. Wang, A. Mohammad-Djafari, N. Gac, M. Dumitru, **3D X-ray Computed Tomography with a Hierarchical Prior model for Sparsity in Haar Transform domain**, *Entropy (2018 IF 2.419/SJR 0.524-Q2 Physics and astronomy), Special Issue Probabilistic Methods for Inverse Problems, MDPI, 2018*

[R$_{2017a}$] L. Wang, A. Mohammad-Djafari, N. Gac, **X-ray Computed Tomography using a sparsity enforcing prior model based on Haar transformation in a Bayesian framework**, *Special Issue of Fundamenta Informaticae (2018 IF 1.204/SJR 0.355-Q2 information Systems), IOS Press, 2017*

[R$_{2017b}$] C. Chapdelaine, A. Mohammad-Djafari, N. Gac, E. Parra, **A 3D Bayesian Computed Tomography Reconstruction Algorithm with Gauss-Markov-Potts Prior Model and its Application on Real Data**, *Special Issue of Fundamenta Informaticae (2018 IF 1.204/SJR 0.355-Q2 information Systems), IOS Press, 2017*

[R$_{2014a}$] N. Chu, J. Picheral, A. Mohammad-Djafari, N. Gac, **A robust super-resolution approach with sparsity constraint in acoustic imaging**, *Applied Acoustics (2014 IF 1.024/SJR 0.701-Q1 Acoustics and Ultrasonics), Elsevier, 2014, 76, pp.197-208.*

[R$_{2014b}$] F. Schmidt, I. Shatalina, M. Kowalski, N. Gac, B. Saggin, et al., **Toward a numerical deshaker for PFS**, *Planetary and Space Science (2015 IF 1.942/SJR 1.01-Q2 Space and Planetary science), Elsevier, 2014, 91, pp.45 - 51.*

[R$_{2013}$] T. Boulay, N. Gac, A. Mohammad-Djafari, J. Lagoutte, **Algorithmes de reconnaissance NCTR et parallélisation sur GPU**, *Traitement du Signal (2013 IF 0.023/SJR 0.114), Lavoisier, 2013, 6, pp.309-342.*

[R$_{2012}$] M.L. Gallin-Martel., Y. Grondin, N. Gac, Y. Carcagno, L.Gallin-Martel, D. Grondin, M. Marton, J.-F.Muraz, O. Rossetto, F. Vezzu, **Experimental results and first 22Na source image reconstruction by two prototype modules in coincidence of a liquid Xenon Positron Emission Tomograph for small animal imaging**, *Nuclear Instruments and Methods in Physics Research Section A : Accelerators, Spectrometers, Detectors and Associated Equipment (2013 IF 1.316/SJR 0.946-Q1 Instrumentation), Elsevier, 2012, 682, pp.66-74*

[R$_{2009}$] N. Gac, S.Mancini, M. Desvignes et D. Houzet, **High Speed 3D Tomography on CPU, GPU and FPGA**, *EURASIP Journal on Embedded systems (2013 IF 1.5/2010 SJR 0.39-Q2 Computer Science), SpringerOpen, 2009 (2019 citation googlescholar : 63*

[R$_{2006}$] S. Mancini, N. Gac, M. Desvignes, O. Bourrion et O. Rosseto, **Application d'un cache 2D prédictif à l'accélération de la rétroprojection TEP 2D**, *Traitement du Signal (2009 IF 0.121/2011 SJR 0.101), 2006, vol. 23, n 5-6-NS, p. 391-404.*

Conférences internationales avec comité de sélection (41)

[C$_{2020a}$] D. Diakite, M. Martelli, N. Gac, **An OpenCL pipeline implementation on Intel FPGA for 3D backprojection**, *6th International Conference on Image Formation in X-Ray Computed Tomography, Aug 2020, Regensburg, Germany*

[C$_{2020b}$] M. Seznec, N. Gac, F. Orieux, A. Sashala Naik, **An efficiency-driven approach for real-time optical flow processing on parallel hardware**, *IEEE International Conference on Image Processing (ICIP), Oct 2020, Abu Dhabi, United Arab Emirates*

[C$_{2020c}$] M. Chghaf, N. Gac, **Data distribution on a multi-GPU node for TomoBayes CT reconstruction**, *the 26th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTSCA), Aug 2020, South Korea*

[C$_{2020d}$] M. Seznec, N. Gac, F. Orieux, A. Sashala Naik, **A new convolutions algorithm to leverage tensor cores**, *GPU Technology Conference (GTC), May 2020, Silicon Valley, United States*

[C$_{2019a}$] N. Georgin, C. Chapdelaine, N. Gac, A. Mohammad-Djafari, E. Parra,**Multi-streaming and multi-GPU optimization for a matched pair of Projector and Backprojector** , *2019 International Conference on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, Jun 2019, Philadelphia, United States*

[C$_{2019b}$] M. Martelli, C. Enderli, N. Gac, A. Merigot, E. Parra, **GPU Acceleration : OpenACC for Radar Processing Simulation**, *2019 International Radar Conference, Sept 2019, Toulon*

[C$_{2019c}$] C. Chapdelaine, A. Mohammad-Djafari, N. Gac, E. Parra, **Variational Bayesian approach in model-based iterative reconstruction for 3D X-ray computed tomography with Gauss-Markov-Potts prior**, *MaxEnt 2019 : Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, Jun 2019, Garching/Munich, Germany*

[C$_{2018a}$] C. Chapdelaine, N. Gac, A. Mohammad-Djafari, E. Parra,**New GPU implementation of Separable Footprint (SF) Projector and Backprojector : first results** , *International Meeting on Image Formation in X-Ray Computed Tomography, Salt Lake City, Utah, US, May 2018, Proceedings of CT meeting*

[C$_{2018b}$] M. Seznec, N. Gac, A. Ferrari, F. Orieux, **A Study on Convolution Using Half–Precision Floating–Point Numbers on GPU for Radioastronomy Deconvolution** , *IEEE SIPS, Cape Town, South Africa, October 2018*

[C$_{2018c}$] A. Mohammad-Djafari, M. Dumitru, C. Chapdelaine, N. Gac, **Bayesian Inference with Error Variable Splitting and Sparsity Enforcing Priors for Linear Inverse Problems** , *26th European Signal Processing Conference (EUSIPCO), Rome, September 2018*

[C$_{2017a}$] L. Wang, A. Mohammad-Djafari, N. Gac, M. Dumitru, **3D X-ray computed tomography reconstruction using sparsity enforcing hierarchical model based on haar tranformation**, *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, Mar 2017,New Orleans , Etats-Unis. IEEE Proceedings ICASSP, 2017*

[C$_{2017b}$] C. Chapdelaine, A. Mohammad-Djafari, N. Gac, E. Parra, **A joint segmentation and reconstruction algorithm for 3D bayesian computed tomography using Gaus-Markov-Potts Prior Mode**, *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, Mar 2017,New Orleans , Etats-Unis. IEEE Proceedings ICASSP, 2017 (accepted for Ph.D./M.Sc. Forum Proposal)*

[C$_{2017c}$] L. Wang, A. Mohammad-Djafari, N. Gac, M. Dumitru, **3D X-ray Computed Tomography reconstruction using sparsity enforcing Hierarchical Model based on Haar Transformation** , *Fully3D, 2017*

[C$_{2017d}$] M. Dumitru, N. Gac ,L. Wang, A. Mohammad-Djafari, **Unsupervised sparsity enforcing iterative algorithms for 3D image reconstruction in X-ray Computed Tomography**, *Fully3D, 2017*

[C$_{2017e}$] C. Chapdelaine, A. Mohammad-Djafari, N. Gac, E. Parra, **Joint Reconstruction and Segmentation of Real 3D Data in Computed Tomography thanks to a Gauss-Markov-Potts Prior Model**, *Fully3D, 2017*

[C$_{2017f}$] L. Wang, A. Mohammad-Djafari, N. Gac, **Bayesian method with sparsity enforcing prior of dual-tree complex wavelet transform coefficients for X-ray CT image reconstruction**, *The 25th European Signal Processing Conference (EUSIPCO), Aug 2017, Kos island, Greece*

[C$_{2017g}$] M. Martelli, N. Gac, A. Mérigot, C. Enderli. **3D Tomography parallelization on FPGAs via HLS tools**, *DASIP, Sep 2017, dresden, Germany*

[C$_{2017h}$] M. Dumitru, L. Wang, N. Gac, A. Mohammad-Djafari, **Performance comparison of Bayesian iterative algorithms for three classes of sparsity enforcing priors with application in computed tomography**, *2017 IEEE International Conference on Image Processing, Sep 2017, Beijing, China*

[C$_{2017i}$] M. Dumitru, L. Wang, A. Mohammad-Djafari, N. Gac, **Model selection in the sparsity context for inverse problems in Bayesian framework**, *37th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, Jul 2017, Jarinu, Brazil*

[C$_{2016a}$] L. Wang, A. Mohammad-Djafari, N. Gac, M. Dumitru, **Computed tomography reconstruction based on a hierarchical model and variational Bayesian method**, *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, Mar 2016, Shangai, Chine. IEEE Proceedings ICASSP, pp. 883-887, 2016*

[C$_{2016b}$] L. Wang, A. Mohammad-Djafari, N. Gac, **Bayesian X-ray Computed Tomography using a Three-level Hierarchical Prior Model**, *Maxent 2016, Jul 2016, Gent, Belgium. AIP Conference, published in december 2016*

[C$_{2016c}$] A. Mohammad-Djafari, L. Wang, N. Gac, F. Bleichrodt, **A Student-t based sparsity enforcing hierarchical prior for linear inverse problems and its ecient Bayesian computation for 2D and 3D Computed Tomography**, *itwist 2016, Aug 2016, Aalborg, Danemark.*

[C$_{2015a}$] O. Schwander, J. Picheral, N. Gac, A. Mohammad Djafari, D. Blacodon, **Aero-acoustics source separation with sparsity inducing priors in the frequency domain**, *Sep 2014, Amboise, France, Proceedings of the 34th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, 1641, pp.422 - 431, 2015, AIP Conf. Proc.*

[C$_{2015b}$] L. Wang, N. Gac, A. Mohammad-Djafari, **Bayesian 3D X-ray computed tomography image reconstruction with a scaled Gaussian mixture prior model**, *Sep 2014, Amboise, France, Proceedings of the 34th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, 1641, pp.556-563, 2015, AIP Conf. Proc.*

[C$_{2015c}$] N. Chu, A. Mohammad-Djafari, N. Gac, J. Picheral, **A hierarchical variational Bayesian approximation approach in acoustic imaging**, *Sep 2014, Amboise, France, Proceedings of the 34th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, 1641, pp.572 - 579, 2015, AIP Conf. Proc.*

[C$_{2014a}$] N. Chu, N. Gac, J. Picheral, A. Mohammad-Djafari, **Convolution Models with Shift-invariant kernel based on Matlab-GPU platform for Fast Acoustic Imaging**, *ISAV 2014, Dec 2014, Tehran, Iran. Proceedings of the 4th International Conference on Acoustics and Vibration, 2014*

[C$_{2014b}$] N. Chu, N. Gac, J. Picheral, A. Mohammad-Djafari, **2D Convolution model using (in)variant kernels for fast acoustic imaging**, *BEBEC 2014, Feb 2014, Berlin, Germany. Proceedings of the Berlin Beamforming Conference, 15 p., 2014*

[C$_{2014c}$] N. Chu, A. Mohammad-Djafari, N. Gac, J. Picheral, **A variational Bayesian approximation approach via a sparsity enforcing prior in acoustic imaging**, *WIO 2014, Jul 2014, Neuchatel, Switzerland. Proceedings of the 2014 13th Workshop on Information Optics, pp.1 - 4, 2014*

[C$_{2014d}$] L. Chen, T. Rodet, N. Gac, **A simple and efficient super-short-scan algorithm of fan-beam reconstruction for multiple circular trajectories : solution towards the truncated data**, *International Meeting on Image Formation in X-Ray Computed Tomography, Salt Lake City, Utah, US, June 2014, Proceedings of CT meeting, pp. 212-215, 2014*

[C$_{2013a}$] T. Boulay, A. Mohammad-Djafari, N. Gac, Julien Lagoutte, **High-Dimensional Range Profile Geometrical Visualization and Performance Estimation of Radar Target Classification via a Gaussian Mixture Model**, *GSI2013-Geometric Science of Information, Aug 2013, Paris, France. Part XXIII, pp.829-836, 2013*

[C$_{2013b}$] I. Shatalina, F. Schmidt, B. Saggin, N. Gac, M. Kowalski, et al., **Analytical model and spectral correction of vibration effects on Fourier transform spectrometer**, *SPIE Remote sensing 2013, Sep 2013, Dresden, Germany. 8890, pp.88900S-88900S-9, 2013*

[C$_{2013c}$] N. Chu, A. Mohammad-Djafari, J. Picheral, N. Gac, **An efficient variational Bayesian inference approach via Studient's-t priors for acoustic imaging in colored noises**, *POMA - ICA 2013, Jun 2013, Montreal, Canada. 19, 055031 (9p.), 2013*

[C$_{2013d}$] L. Chen, T. Rodet, N. Gac, **A penalized weighted least-squares image reconstruction based on scatter correction methods for X-ray CT**, *2013 IEEE Nuclear Science Symposium and Medical Imaging Conference, Oct 2013, Seoul, South Korea. pp.2, 2013*

[C$_{2012}$] T. Boulay, J. Lagoutte, A. Mohammad-Djafari, N. Gac, **A Fuzzy-Logic based non cooperative target recognition**, *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems, Naples, 2012*

[C$_{2011a}$] N. Gac, A.Vabre, A. Mohammad-Djafari, **Multi GPU parallelization of 3D bayesian CT algorithm and its application on real foam reconstruction with incomplete data set**, *Forum on recent developments in Volume Reconstruction techniques applied to 3D fluid and solid mechanics, Poitiers, décembre 2011*

[C$_{2010}$] N. Gac, A.Vabre, A. Mohammad-Djafari, A.Rabanal, F.Buyens, **GPU Implementation of 3D Bayesian CT Algorithm and its application on real foam reconstruction**, *International Meeting on Image Formation in X-Ray Computed Tomography, Salt Lake City, US, Proceedings of CT meeting, pp. 151-155, 2010*

[C$_{2008a}$] N. Gac, S. Mancini, M. Desvignes, F. Deboissieu et A. Reilhac, **A Hardware Projector/Backprojector Pair for 3D PET Reconstruction**, *SPIE Physics of Medical Imaging 2008 conf. proc., volume 6913*

[C$_{2007}$] N. Gac, S. Mancini, M. Desvignes et D.Houzet, **Algorithm Architecture Adequacy for High Speed 3D Tomography**, *2007 workshop on Design and Architectures for Signal and Image Processing*

[C$_{2006a}$] N. Gac, S. Mancini et M. Desvignes, **Hardware/software 2D-3D backprojection on a SoPC platform**, *Proc. of the 2006 ACM symposium on Applied computing, vol.4, pp. 222-228*

[C$_{2006b}$] Y. Grondin, L. Desbat, M. Defrise, T. Rodet, N. Gac, M. Desvignes and S. Mancini, **Data Sampling in Multislice Mode PET for Multi-Ring Scanner**, *2006 IEEE Nuclear Science Symposium conf. record, pp. 2180-2184*

[C$_{2005a}$] M. Vacher, D. Istrate, J-F. Serignat et N. Gac, **Detection and Speech/Sound Segmentation in a smart room environment**, *Proc. of SpeD'05, pp. 37- 48*

Conférences francophones avec comité de sélection (10)

[C$_{2018d}$] M. Martelli, N. Gac, A. Mérigot, C. Enderli, **Harnessing FPGAs potential with OpenCL**, *Colloque nationale SOC-SIP, June 2018, Paris*

[C$_{2017j}$] L. Wang, N. Gac, A. Mohammad-Djafari, **Reconstruction 3D en tomographie à rayons X à l'aide d'un modèle *a priori* hiérarchique utilisant la transformation de Haar**, *Colloque GRETSI 2017, Sep 2017, Juan-Les-Pins, France*

[C$_{2017k}$] M. Dumitru, L. Wang, N. Gac, A. Mohammad-Djafari, **Comparaison des performances d'algorithmes itératifs bayésiens basés sur trois classes de modèles *a priori* parcimonieux appliqués à la reconstruction tomographique**, *GRETSI, Sep 2017, Juans-Les-Pins, France*

[C$_{2011b}$] N. Gac, A. Vabre, A.Mohammad-Djafari, F. Buyens, **Accélération sur serveur multi-GPUs de la reconstruction 3D d'une mousse de nickel par méthodes itératives algébriques régularisées**, *GRETSI, Bordeaux, Septembre, 2011*

[C$_{2011c}$] T. Boulay, N. Gac, A.Mohammad-Djafari, J. Lagoutte, **Algorithmes de Reconnaissance Non Coopérative de Cibles et implémentation sur GPU**, *GRETSI, Bordeaux, Septembre, 2011*

[C$_{2009}$] N. Gac, A. Vabre, A.Mohammad-Djafari, F. Buyens, S. Legoupil, **Parallélisation sur GPU d'un algorithme de reconstruction 3D Bayésien en tomographie X**, *Développement d'applications de calcul intensif sur carte graphique pour l'imagerie moléculaire, Obernai, Mai, 2009*

[C$_{2008b}$] N. Gac, S. Mancini et M. Desvignes, **Une paire matérielle projecteur / rétroprojecteur pour la reconstruction TEP 3D**, *2ème colloque du GDR System On Chip - System In Package (SOC-SIP), 2008*

[C$_{2005b}$] S. Mancini, N. Gac et M. Desvignes, **Rétroprojection 2D sur plateforme SOPC**, *GRETSI, 2005, pp. 1060-1063*

[C$_{2005c}$] S. Mancini, N. Gac et M. Desvignes, **Rétroprojection 2D sur plateforme SOPC, premiers résultats**, *Journées Sciences, Technologies et Imagerie pour la Médecine, 2005, pp. 132-136*

[C$_{2005d}$] S. Mancini, N. Gac et M. Desvignes, **Etude d'un cache 2D adaptatif et prédictif pour le traitement d'image**, *Journées Francophones sur l'Adéquation Algorithme Architecture, 2005, pp. 260-265*

Logiciels (3) et Brevet (1)

[L$_{2015-2019}$] C. Chapdelaine, N. Gac, T. Boulay, A. Djafari, E. Parra, Y. Le Guilloux, N. Georgin, **TomoBayes - logiciel de reconstruction en tomographie CT**, *v1.0 2015 v2.0 2019*

[B$_{2018}$] C. Chapdelaine, A. Mohammad-Djafari, E. Parra, N. Gac, **Procédé et dispositif de contrôle non-destructif d'une pièce**, *France, Num de brevet : 18/53648. 2018*

[L$_{2018}$] M. Dumitru, L. Wang, N. Gac et A. Mohammad-Djafari, **iterTomoGPI-GPL - Plugin Matlab pour la toolbox opensource Astra mettant à disposition les méthodes bayésiennes du GPI pour la reconstruction tomographique**, *Ref interne CNRS du pré-dépôt APP 1562-02, 2018*

[L$_{2014}$] N. Gac et A.Mohammad-Djafari, **opgpuTomoGPI - librairie C/CUDA d'accélération sur multi-GPU d'opérateur de la reconstructon tomographique**, *Ref interne CNRS du pré-dépôt APP 1562-01, 2014*

Activité pédagogique

Détails disponibles sur **www.l2s.centralesupelec.fr/u/gac-nicolas/teaching/**

Activité d'enseignement à l'IUT de Cachan

Mon poste est rattaché au département GEII (Génie Electrique Informatique Industrielle) de l'IUT de Cachan à l'université Paris Saclay. J'y enseigne en formation initiale à des groupes de 24 étudiants les systèmes numériques embarqués (*computer science for embedded systems*).

Enseignements en SNE (Systèmes Numériques Embarqués)

- Module de cours/TD/TP (50h) sur la *programmation des microcontroleurs* en S2 et S3 (2009-..) : ports, interruptions, timers, convertisseurs analogique/numérique, liaison série... En 2018, j'ai enseigné ce module à l'aide d'une pédagogie d'**apprentissage par problèmes** suite à une formation donnée par l'université de Louvain (2+3 jours) et à la dynamique insufflée par des collègues à l'IUT pour cette pédagogie ; le but de ces classes "inversées" est de rendre l'étudiant plus actif.
- Module de cours/TD/TP (40h) sur la *synthèse logique* en S2 (2009-..) : langage VHDL, outils de synthèse sur FPGA, logiques combinatoire et séquentielles, testbenchs, automates...
- Création de sujets (2) pour la semaine de *projet génie logiciel* finalisant l'apprentissage du langage C (2010-2013).
- Co-création d'un module complémentaire (20h) sur le *co-design C/VHDL sur FPGA* (2013-2016)
- Co-création d'un module complémentaire (20h) sur la *synthèse haut niveau (HLS) en OpenCL sur FPGA* (2020-...)
- Création d'une Etude et Réalisation (E&R) en S3 depuis 2015 : PARSEC *Parallélisation d'Algorithmes de tRaitement d'image sur Systèmes Embarqués avec Caméra* (cibles : Raspberry Pi et Jetson GPU) (60h).

Responsabilité et création de projets pour le Parcours S4-TI

Lors du semestre 4, l'accent est mis sur la **pédagogie par projet** avec 20h par semaine dédiées aux séances projet avec trois thématiques proposées : ROBOtique, SYstèmes COmmunicants et Traitement de l'Information (TI). L'objectif final est de réaliser un démonstrateur et de présenter un projet avec un rapport et une soutenance préparant ainsi au stage de fin d'année. Je me suis fortement impliqué

sur le thème TI avec la création de nombreux projets et la prise de responsabilité de ce thème pendant six années de 2011 à 2016. En tant que responsable S4-TI, mon rôle était : d'animer l'équipe pédagogique sur la partie projet (6 enseignants) afin de proposer chaque année une douzaine de projets originaux pour les 24 étudiants, de gérer l'emploi du temps et la collecte des notes. Le thème TI est axé sur l'implémentation d'algorithmes de traitement du signal sur cibles matérielles ; j'ai ainsi pu créer plusieurs **projets originaux proches de mes thématiques de recherche** : *la déconvolution accélérée sur GPU* (2010-2012), *la reconnaissance de la parole* (2011-2015), *la super-résolution accélérée sur GPU* (2013-2016), *la stéréovision sur Raspberry Pi* (2016) et *le comptage de personnes par caméra infrarouge* (2017)... Cinq étudiants ont poursuivis ces projets au sein de mon laboratoire en stage. Pour ce parcours S4-TI, j'ai créé également un complément de cours/TP sur l'initiation à la *programmation GPU* (2h cours + 8h TP).

Responsabilité pédagogique du semestre 2

Depuis 2017, j'assure la responsabilité pédagogique du semestre 2. Pour cette **promotion d'une centaine d'étudiants**, j'ai en charge : la gestion de l'emploi du temps (4 groupes), le bilan à mi-semestre pour chaque groupe, le pré-jury de fin de semestre, la gestion des notes (+ de 160 notes à saisir dans le logiciel scodoc...) et la comptabilité du service des 40 enseignants intervenant sur ce semestre. J'ai pu notamment remettre en place les **projets tuteurés** S2 en m'appuyant sur la vitalité de l'IUT de Cachan (ateliers d'écriture, de robotique ou d'entrepreneuriat...) et proposer des créneaux d'**Etudes Encadrées ou en Autonomie** afin de d'inciter les étudiants à travailler en groupe au sein de l'IUT. Par ailleurs, afin de réaliser l'**Evaluation des Enseignements par les Etudiants**, j'anime dans chacun des groupes des séances sous forme de débriefing permettant de recueillir leurs remarques et propositions ce qui aboutit à la rédaction annuelle d'un Recueil de Conclusions et Perspectives (RCP).

Activité d'enseignement en M2

Depuis 2011, j'interviens à différentes formations de niveau M2 en tant que vacataire[1] pour des cours et TP sur la *parallélisation des calculs sur processeurs many cores* : ESIEE (2011-2016), M2R ESA à l'ENSEA (2015-2018) et M2 SETI de l'université Paris Sud (2016-2017). Depuis 2018, je suis **responsable d'un module** (5 ECTS) sur la *parallélisation GPU/FPGA* du master SETI où je donne un cours/TP de 18h. Par ailleurs, je donne un cours+TP sur la *parallélisation d'algorithmes de reconstruction tomographique* depuis 2016 en master ATSI à Paris Saclay (6h) et occasionnellement en formation continue à Supélec. J'interviens également pour le suivi d'étudiants de CentraleSupélec en parcours recherche développant sur leurs

1. Depuis 2019, mes interventions aux masters SETI et ATSI (rattachés à l'UFR des sciences) sont intégrés à mon service à l'IUT de Cachan.

trois années d'études des projets directement rattachés à mes activités de recherche (15h/semestre/étudiant).

Présentation synthétique des enseignements

Mes enseignements sont très majoritairement (près de 90%) à un niveau L1/L2 à l'IUT de Cachan, à hauteur de 10% en M2 et à moins de 1% en Formation Continue (FC). A l'IUT de Cachan, mes enseignements se font devant des groupes de 24 étudiants avec une frontière assez mince entre le cours et le TD (27%) qui sont rapidement suivis de TP (43%). La pédagogie par projet (S4-TI et E&R S3) explique la part importante des heures projets (26%). Les responsabilités ont pris une part plus importante dans mon service d'enseignement avec la gestion du semestre 2 depuis 2017 (48h). Mon volume horaire qui était en moyenne de 2009 à 2016 de 291h avec une sollicitation importante comme c'est souvent le cas dans les IUT a diminué suite à mes deux années de demi-délégation CNRS (96h de service) [2] et l'obtention de la PEDR qui me permet d'afficher un plafond d'heures supplémentaires (50h à l'université Paris Sud).

Année 20xx	CM/ TD	TP	projet	stage	resp.	**Total** (eq.TD)	L1/ L2	M2	FC
09-16	78	124,1	76,6	3	8,6	**290,8**	94,9%	4,5%	0,6%
16-17	70,5	136,3	69	3	48	**331,5**	83,1%	15,6%	1,3%
17-18	27,5	34,5	4		48	**115**	83,5%	13,9%	2,6%
18-19	21	52	32		48	**153**	62,7%	37,3%	
19-20	27,3	73	93		48	**241,3**	70,6%	29,4%	
09-20	63 26,9%	105,8 42,7%	66,7 26,4%	2,2 1%	22,9 3%	**261,5**	87,6%	11,6%	0,7%

Diffusion, rayonnement, activités internationales

Ecole thématique du CNRS sur les GPUs

Depuis 2010, je suis intervenu pour un séminaire invité (1h) à 4 éditions de l'école *Signal Images : Architecture et programmation des GPUs* à Grenoble, 2010-2011-2013-2015-2017 (2021 en projet). Je suis membre du conseil scientifique depuis 2013 et j'étais *chairman* d'une session en 2017.

Prix Intel FPGA

J'ai encadré l'équipe d'étudiants (2 étudiants M2 aidés par un doctorant) lauréate du concours *Intel Innovate Europe Design Contest*, catégorie *High Perfomance Compunting* avec le projet « An openCL design for tomographic reconstruction on FPGA ». Ce travail au L2S s'intégrait dans l'UE Projet du master SETI qui se déroule pendant la période académique.

2. Interventions extérieures en CM/TP à titre gracieux durant ma délégation CNRS

Activités internationales

De 2013 à 2017, je me suis impliqué à la formation FLE (Français Langue Etrangère) de l'IUT de Cachan destinée à un groupe d'une vingtaine d'étudiants chinois accueillis de février à juin avant leur rentrée en DUT. J'ai donné des cours de vocabulaire en informatique et électronique (10h chaque année) et participé au jury de fin d'année.

Lien avec le monde de l'entreprise

En 2017, dans le cadre des projets S4-TI à l'IUT de Cachan, je me suis associé avec la société locale Eurecam (https://eurecam.net/) afin de réaliser une pré-étude sur l'utilisation d'une *caméra Infra rouge pour le comptage de personne*. Ce travail initié dans le parcours S4-TI a été poursuivi par un étudiant lors de son stage de DUT (3 mois) au sein de la société afin d'y développer un produit.

Responsabilités collectives

Conseils au sein de l'université Paris Saclay

2020-...	**Membre élu au comité de direction du L2S** (6 membres) en tant que représentant du personnel de l'université Paris Saclay. Participation aux réunions hebdomadaires mettant à l'ordre du jour les affaires courantes du laboratoire.
2020-...	Membre nommé au conseil de laboratoire du L2S (invité de 2018-19).
2020	Membre du jury CNRS pour le recrutement sur concours interne d'un Ingénieur d'Etudes (admin. systeme/calcul scientifique) au L2S.
2019-20	Forte implication pour la refonte du site web du L2S : cahier des charges, sélection du prestataire, suivi du projet...
2016-...	Suivis de thèses pour l'ED STIC Paris-Saclay en tant que référent*.
2015-18	Membre nommé à la CCS IOGS (Institut d'optique).
2014-20	**Membre élu de la CCSU⋆ 60-61-62** à l'université Paris-sud.
2013-15	**Membre élu du conseil de département GEII 1** et de la Commission Consultatives du Choix des Enseignants à l'IUT de Cachan. Depuis 2017, membre invité en tant que responsable S2.
2011-19	**Membre de six commissions de recrutements à l'université Paris-sud** : 2011+2012+2019 (3 MCF et 1 PRAG Anglais à l'IUT de Cachan), 2015 (MCF IUT d'Orsay), 2016 (MCF Polytech).

Calcul scientifique au laboratoire L2S

Depuis 2018, je suis **responsable du pôle calcul scientifique du L2S**, pôle technico-administratif créé en 2013 après le constat que les activités croissantes du laboratoire dans le développement d'algorithmes pour une très grande diversité de problématiques en recherche fondamentale ou applicative gagneraient à être **capitalisés et valorisés au niveau logiciel** avec une mutualisation des efforts au sein de l'unité. Mon rôle est avant tout de **recruter un ingénieur CNRS (IE)** pouvant remplir cette mission avec la rédaction de la fiche de poste et le suivi de la campagne de recrutement. En 2020, le poste sera en effet ouvert en promotion interne. Par ailleurs, je suis référent du laboratoire pour les **mésocentres de calcul** Saclay-IA et moulon (CentraleSupélec/ENS) diffusant l'information pour l'accès

∗ le rôle des référents est d'effectuer des entretiens brefs avec les doctorants en fin de première année afin de détecter d'éventuels problèmes.

⋆ la CCSU a les mêmes missions que celles du CNU pour les promotions mais au niveau local. En outre, les membres de cette CCSU gèrent les dossiers d'ATER, les profils de postes, la mise en place de comités de sélections ainsi que les demandes de professeurs invités.

et l'usage de ces ressources de calcul parallèle à tous les membres du L2S. Avant 2018, je m'étais déjà impliqué pour la politique de calcul scientifique du laboratoire :

2017-...	Membre du comité scientifique du calculateur Saclay-IA de l'université Paris Saclay (350kE 2018) hébergé à l'IDRIS.
2016	Achat d'un serveur de calcul mutualisé avec GEOPS (serveur 10 GPUs pour 25kE).
2015	Porteur initial de l'appel à projet de l'INS2I pour l'achat du serveur de calcul multi-GPU (70kE obtenu en 2016) de l'IRS CDS 2.0 (Center for Data Science) de l'université Paris Saclay.
2014-18	Définition et co-administration du système commun pour la dizaine de postes de travail du GPI (Groupe Problèmes Inverses).

Deuxième partie

Synthèse des travaux de recherche

Notations

MESURES DE L'INSTRUMENT ET DE L'OBJET A RECONSTRUIRE

$\boldsymbol{f}$	Objet d'intérêt (vectorisé), *volume en recontruction tomographique ou hypercube en radioastronomie*
$\boldsymbol{g}$	Mesures de l'instrument (vectorisé)
$\boldsymbol{\varepsilon}$	Incertitudes sur les mesures et le modèle d'instrument (vectorisé)
$\boldsymbol{H}$	Matrice d'acquisition appelée aussi matrice système
H_{ij}	Elément de $\boldsymbol{H}$ pris sur la ligne i et la colonne j
$\boldsymbol{H}^t$	Transposée de $\boldsymbol{H}$
$\boldsymbol{f}^{(n)}$	Estimée nième du volume dans un algorithme itératif
$\hat{\boldsymbol{f}}$	Solution au problème inverse, *définie par un modèle puis recherchée par un algorithme d'optimisation*
N_f	Dimension de $\boldsymbol{f}$
$\boldsymbol{\delta}_g$	Ecart entre les mesures $\boldsymbol{g}$ et les mesures estimées $\boldsymbol{Hf}$ (vectorisé)
$\mathbb{M}_D$	Modèle Direct de l'instrument
$\mathbb{M}_f$	Modèle *a priori* de l'objet $\boldsymbol{f}$

ALGORITHMES D'OPTIMISATION

DG	Algorithme de Descente de Gradient
VBA	Approche Bayésienne Variationnelle
$J(\boldsymbol{f})$	Critère J à optimiser pour trouver la solution $\hat{\boldsymbol{f}}$
$\nabla J(\boldsymbol{f})$	Gradient de $J(\boldsymbol{f})$
λ	Paramètre de régularisation
$\|\boldsymbol{g}\|^2_{\boldsymbol{V}}$	Norme $\boldsymbol{g}^t\boldsymbol{V}^{-1}\boldsymbol{g}$, avec $\boldsymbol{V}$ une matrice diagonale aux éléments positifs non nuls*
R	Fonctionelle convexe utilisée pour le terme de régularisation
$\boldsymbol{D}$	Opérateur linéaire utilisé pour le terme de régularisation
$\mathcal{L}$	Laplacien ; correspondant en 1D au noyau [-1 2 -1]
$\mathcal{O}$	Transformée en ondelettes
α	Pas de descente du gradient

* définition valable plus généralement si $\boldsymbol{V}$ est une matrice symétrique définie positive

MODELES BAYESIENS HIERARCHIQUES

$p(\boldsymbol{f})$	Probabilité de $\boldsymbol{f}$
$\mathcal{N}(m, v)$	Loi normale de moyenne m et de variance v
$\mathcal{IG}$	Loi Inverse Gamma
$\mathcal{S}t_g(.\|\alpha, \beta)$	Loi de Student-t généralisée
$\boldsymbol{\theta}_f$	Paramètres de la loi *a priori* $p(\boldsymbol{f})$ définie par $\mathcal{M}_f$
$\boldsymbol{\theta}_D$	Paramètres de la vraisemblance $p(\boldsymbol{g}\|\boldsymbol{f})$ définie par $\mathcal{M}_D$
$\boldsymbol{v}_\varepsilon$	Variance de la variable ε (vectorisé) : *utilisée pour modéliser un bruit non-stationnaire par une loi normale*
v_ε	Variance de la variable ε (scalaire) : *utilisée pour modéliser un bruit stationnaire par une loi normale*
$\alpha_{\varepsilon_0}, \beta_{\varepsilon_0}$	Paramètres α et β de la loi de Student-t généralisée de la variable $\boldsymbol{\varepsilon}$
$\boldsymbol{V}_\varepsilon$	Matrice de co-variance diagonale $\boldsymbol{V}_\varepsilon = \text{diag}(\boldsymbol{v}_\varepsilon)$

OPERATEURS POUR LE CALCUL DE $\boldsymbol{H}$ et $\boldsymbol{H}^t$

$\mathcal{C}$	Convolution
$\mathcal{P}$	Projecteur en tomographie à faisceau conique
$\mathcal{P}_{RD}$	Projecteur *Ray-Driven*
$\mathcal{P}_{Siddon}$	Projecteur *ray-driven* de Siddon
$\mathcal{P}_{Regular}$	Projecteur *ray-driven* à échantillonnage régulier
$\mathcal{P}_{Joseph}$	Projecteur *ray-driven* de Joseph
$\mathcal{P}_{VD}$	Projecteur *Voxel-Driven*
$\mathcal{R}$	Rétroprojecteur en tomographie à faisceau conique
$\mathcal{R}_{RD}$	Rétroprojecteur *Ray-Driven*
$\mathcal{R}_{VD}$	Rétroprojecteur *Voxel-Driven*
$\mathcal{R}_{VDL}$	Rétroprojecteur *Voxel-Driven* avec Interpolation bilinéaire
$\mathcal{P}_{SF}/\mathcal{R}_{SF}$	Paire de projection/rétroprojection *Separable Footprint*
$\mathcal{P}_{DD}/\mathcal{R}_{DD}$	Paire de projection/rétroprojection *Distance-Driven*

RECONSTRUCTION TOMOGRAPHIQUE

x, y, z	Coordonnées physiques dans le repère du volume
u, v	Coordonnées physiques dans le repère du plan de détecteurs
ϕ	Angle d'acqusition
un	Coordonnée discrète de u définie par $un = \dfrac{u}{\delta_u} + un_0$, avec δ_u le pas de discrétisation et un_0 un offset
un_e, ε_u	Partie entière et décimale de la coordonnée discrète $un = un_e + \varepsilon_u$
$N_{\texttt{un}}$	Dimension de la coordonnée discrète un
un	Indice de boucle sur la coordonnée discrète un

Jeu de données $[N_{\texttt{phi}} \times N_{\texttt{un}} * N_{\texttt{vn}}; N_{\texttt{xn}} * N_{\texttt{yn}} * N_{\texttt{zn}}]$: volume de $N_{\texttt{xn}} * N_{\texttt{yn}} * N_{\texttt{zn}}$ voxels reconstruit à patir de $N_{\texttt{phi}}$ projections sur des plans de $N_{\texttt{un}} * N_{\texttt{vn}}$ pixels détecteur.

RADIOASTRONOMIE

x, y	Coordonnées spatiales dans l'hypercube $\boldsymbol{f}$ du ciel
i, j	Coordonnées spatiales discrètes associées à x, y
λ	Longueur d'onde dans l'hypercube du ciel
l	Coordonnée spectrale discrète associée à λ
h	Noyau de convolution
R	Rayon du noyau de convolution h
r	Indice de parcours du noyau de convolution
$\boldsymbol{I}_l$	Image 2D de l'hypercube $\boldsymbol{f}$ à la longueur d'onde l
$\tilde{\boldsymbol{I}}_l$	Image *dirty* à la longueur d'onde l à déconvoluer
$\boldsymbol{S}_{i,j}$	Spectre 1D de l'hypercube $\boldsymbol{f}$ pour la ligne de visée i, j
μ_I, μ_S	Paramètres scalaires de régularisation spatiale et spectrale
$\boldsymbol{D}_I, \boldsymbol{D}_S$	Opérateurs linéaires appliqués à une image ou à un spectre

ACCELERATION SUR GPU ET FPGA

$\texttt{CPU}_A$	Processeur CPU d'architecture A
$\texttt{GPU}_{A-X}$	Processeur GPU d'architecture A sur la carte X
$\texttt{FPGA}_A$	Puce FPGA d'architecture A
$\texttt{S}_M$	Serveur multi-GPU de la machine M
A_n	Accélération sur un serveur multi-GPU obtenue grâce à l'utilisation de n cartes par rapport à l'emploi d'une seule carte
η	Efficacité de parallélisation sur serveur multi-GPU : $\eta = \dfrac{A_n}{n}$
$\texttt{Mem}_{CPU/GPU}$	Stockage des données sur le CPU hôte ou sur les cartes GPU.

Chapitre 1

Accélération : problématique et positionnement

Sommaire

L'amélioration constante de la résolution des instruments parallèlement à la complexité grandissante des méthodes de reconstruction basées sur des modèles de plus en plus précis, s'accompagne d'un besoin croissant en puissance de calcul. Les cartes accélératrices comme les GPUs ou les FPGAs sont une opportunité pour réduire ce fossé technologique existant entre les systèmes d'acquisition et de reconstruction.

Au sein du GPI, mes travaux visent à lever le verrou lié à la masse imposante de données traitées par les algorithmes itératifs de résolution de problèmes inverses en grande dimension. Afin de démontrer leur intérêt pratique, ces méthodes doivent conserver un temps de traitement « raisonnable » variable selon le contexte applicatif. En reconstruction tomographique, ce temps peut correspondre à quelques secondes ou minutes afin de pouvoir détecter, par exemple, dans le domaine médical, un problème éventuel d'acquisition requérant le repositionnement du patient, ou bien dans le domaine industriel, le défaut d'une pièce sur une ligne de production. En radioastronomie, dans le cas du futur radiotélescope SKA, cette contrainte de temps peut être encore plus forte et s'apparenter au temps réel requis sur les systèmes embarqués.

Si mes travaux n'ont pas vocation à atteindre une mise en production des méthodes avec une accélération optimale, une accélération effective doit être atteinte. D'une part, elle est requise comme démonstration de concepts architecturaux et algorithmiques permettant d'atteindre un temps de traitement adapté au contexte applicatif. D'autre part, elle est nécessaire lors de l'exploration algorithmique et paramétrique des méthodes développées au sein de l'équipe de recherche, et in fine lors de leur

validation sur données réelles. En amont de cette mise en oeuvre d'une accélération sur cible technologique, la recherche d'une adéquation entre l'algorithme et l'architecture de calcul est au coeur de mes travaux de recherche.

Ce chapitre d'introduction présente tout d'abord les cartes accélératrices utilisées dans mes travaux comme moteurs de calcul : les GPUs, processeurs *many cores* et les FPGAs, architectures à base de logique programmable. Mon positionnement par rapport au domaine de l'Adéquation Algorithme Architecture est enfin exposé.

1.1 Démocratisation des supercalculateurs

Apparu au début des années 2000, le *personal supercomputer* est un modèle de plateforme d'accélération des calculs alternative au modèle des grilles de calcul apparu quelques années auparavant. Ce modèle à petite échelle avec des prix nettement plus abordables que les *supercomputers* hébergés dans les mésocentres de calcul ont permis une diffusion rapide de leur utilisation dans les laboratoires de recherche. Son architecture est généralement mono-noeud avec un processeur CPU généraliste, hôte d'une ou plusieurs cartes accélératrices comme illustré Fig. 1.1. Celles-ci sont basées sur des co-processeurs *many cores* comme les Graphic Processing Unit (GPU) ou les *Massively Parallel Processor Array* (MPPA) de Kalray, sur des *Digital Signal Processing* (DSP), sur des puces FPGA ou plus récemment sur des *Neural Processing Unit* (NPU).

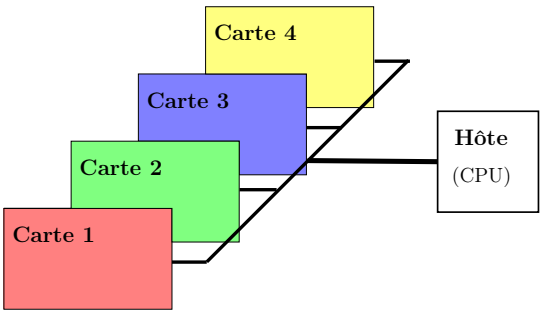


FIGURE 1.1 – Serveur hôte avec cartes accélératrices (*device*)

Dans le cadre de mes travaux de recherche, je me suis principalement intéressé aux plateformes *personal supercomputer* faisant appel aux GPUs (de Nvidia), et aux FPGA (du fabricant Xilinx durant ma thèse puis ceux d'Intel). Plus récemment, j'étends le cadre d'étude de ces cibles GPU et FPGA aux systèmes embarqués (thèse de Mickael Seznec) et au calcul haute performance dans un contexte collaboratif avec l'IETR, l'IRISA, les observatoires de Paris et de la Côte d'Azur et Atos Bull (projet SKA).

1.1.1 GPU : un co-processeur massivement parallèle

L'adoption du modèle de plateforme *personal supercomuter* a été amplifiée avec la mutation des GPUs en *General Purpose* GPU ; l'architecture des GPUs était jusqu'alors conçu quasi exclusivement pour le rendu 3D, et pouvait à ce titre être qualifié de *Domain Specific Architecture* (DSA). Apparus en 2006, le modèle d'architecture et l'API associée, CUDA (*Common Unified Device Architecure*) de Nvidia ont été particulièrement plébiscités. Ces GPUs peuvent alors être utilisés comme des co-processeurs *many cores* avec un modèle de programmation proche du C. A noter que les « coeurs » de calcul (près de 7000 sur le dernier modèle Ampère) ne sont pas comparables à ceux des CPUs mais correspondent plus exactement à toutes les unités de multiplications et additions flottantes disponibles. Le modèle CUDA a constitué un véritable saut technologique pour de nombreux champs applicatifs ; en effet, grâce à sa puissance de calcul en constante augmentation ($\times$30 sur les douze dernières années), les algorithmes réguliers et massivement parallélisables sont éligibles à une accélération d'un voire deux ordres de magnitude par rapport aux CPUs.

Le domaine de la reconstruction tomographique l'a par exemple rapidement adopté comme l'illustre la Tab. 1.1 dénombrant les cartes d'accélération utilisées dans les publications à la conférence Fully 3D. En 2007, l'année suivant l'apparition de CUDA, un workshop *High-Performance Image Reconstruction* (HPIR) y a été organisé comparant les performances obtenues sur différentes architectures d'accélération. Au fil des éditions de cette conférence, les GPUs de Nvidia ont monopolisé ce domaine de recherche. Les architectures alternatives comme le Cell d'IBM, les GPU d'AMD ou d'Intel (Larabee, Xeon phi) n'ont pas su convaincre et motiver l'utilisation d'un autre modèle de programmation que CUDA.

	2003	2005	2007	2009	2011	2013	2015	2017	2019
GPU			**10**	**17**	**18**	**9**	**13**	**20**	**30**
Nvidia			10	14	17	7	10	20	
AMD				1	1		1		
Intel(Larabee,Xeon phi)				2		2	2		
multi-GPU					3	3	3		1
CNN								2	13
Autres processeurs	**2**	**3**	**10**	**7**	**3**	**2**	**1**	**2**	
CPU (MPI/Open MP)	2	3	5	6	3	2	1	2	
Cell (IBM)			3						
DSP			2	1					
FPGA			**4**		**1**		**1**		

TABLE 1.1 – Cartes accélératrices utilisées dans les travaux publiés à la conférence Fully 3D.

Outre ses nombreux « coeurs » de calcul, l'architecture des GPUs se révèle être une riche boite à outils d'accélération : les accès mémoire sont accélérés grâce aux tex-

tures possédant un cache 2D adjoint à un interpolateur bilinéaire matériel, à de la mémoire locale (*shared memory*) et à des caches L1/L2 ; les précisions demi, simple ou double précision flottante sont disponibles pour un compromis entre précision et rapidité de calcul à décider selon le contexte algorithmique ; des unités de calcul plus spécialisés que les « coeurs » sont également disponibles comme par exemple les *tensors cores* dédiés à l'accélération de la multiplication de matrices employée par les réseaux de neurones convolutifs.

Une critique communément formulée à l'encontre de cet engouement pour les GPUs de Nvidia est que le langage CUDA est destiné à un usage exclusif des architectures du fabricant Nvidia ; cela ne concourt pas à une plus grande diversité d'architectures comme nous avons pu le constater en Tab. 1.1. Pour palier à cela, le standard de programmation ouvert OpenCL, basé également sur la syntaxe du C, a été proposé par le consortium Khronos [Khronos 2011] afin de cibler tous types de cartes accélératrices : CPU *multi cores*, GPU *many cores*, DSP ou FPGA ; en implémentant les drivers adaptés à ce standard, chaque constructeur rend ainsi son architecture programmable en OpenCL. En complément à ces langages bas niveau, des directives comme OpenACC [Wienke 2012] et OpenMP [Chapman 2007], ou bien des DSL (*Domain Specific Langage*) comme Matlab ou TensorFlow faisant appel à des librairies bas niveau, peuvent être employés. Lors des travaux de thèse de Maxime Martelli, la directive OpenACC a ainsi été utilisée pour un algorithme en simulation radar ; elle s'est révélée être un bon compromis entre temps de développement et performances d'accélération [C$_{2019b}$].

1.1.2 FPGA : une architecture dédiée

Les FPGAs (*Field Programmable Gate Array*) sont des circuits logiques programmables par opposition aux ASICs (*Application Specific Integrated Circuit*) pour lesquels l'architecture est « figée » après fabrication chez le fondeur. Pour offrir cette souplesse de reconfiguration, ils sont constitués d'une matrice reprogrammable d'interconnexion de blocs logiques dénommés ALM [1] (*Adaptative Logic Modules*) ; ces blocs constitués de LUTs (*Look-up Tables*), de *full adders* et de registres, permettent d'être configurés via leur mémoire SRAM ; ils génèrent ainsi plusieurs types d'opérations arithmétiques et logiques, qui interconnectés entre eux et avec des blocs plus spécialisés comme les IOB (*Input Output Bloc*), les DSP (*Digital Signal Processing*) et les BRAM (*Blocks RAM*), permettent de concevoir des fonctions logiques complexes correspondant à une architecture numérique sur mesure d'un algorithme donné.

Leur maîtrise de la consommation énergétique et leur déterminisme au coup d'horloge près en font des architectures traditionnellement utilisées pour les systèmes

1. dénomination du fabricant Intel

embarqués produits en faible quantité (<10 000 unités). Leur capacité de reconfiguration leur permettent également d'être un bon outil de prototypage rapide d'ASICs. Par ailleurs, cette technologie s'est ouverte depuis quelques années au marché des *data centers* grâce à ses bonnes performances par Watt et sa capacité à gérer des flux importants de données. En 2015, Intel a ainsi racheté Altera, l'un des principaux fabriquants de FPGA. Cette alliance permet de faire naître de nouveaux produits où le FPGA apparaît explicitement comme un accélérateur au coté d'un CPU hôte ; la carte FLIK en est un récent exemple.

La conception sur FPGA se fait typiquement à l'aide de langages HDL (*Hardware Decsription Language*) comme le VHDL ou le Verilog. Un outil de synthèse transcrit cette description à niveau plus bas d'abstraction, le RTL (*Register Transfer Level*) ; celui-ci permettra dans la suite du flot de conception de générer le *bitstream* reconfigurant le FPGA. Ce flot de conception numérique de circuit requiert une forte expertise. Ainsi, afin de le rendre plus rapide et de réduire sa forte disparité avec le flot de développement logiciel, des outils HLS (*High Level Synthesis*) académiques ou industriels ont émergé depuis les années 90, parmi lesquels nous pouvons citer : AUGH [TIMA 2012], GAUT [Coussy 2008], CHiMPS [Putnam 2008], ROCC [ROCCC 2013], LegUp [Canis 2011] ou Catapult-C par Mentor Graphics [Bollaert 2008]. Les constructeurs de FPGA ont bénéficié de ces travaux de recherche et ont ainsi pu proposer leur propres outils : *Xilinx Vivado HLS* apparu en 2012 et *Intel HLS Compiler* apparu en 2017[2]. Ces outils permettent de décrire des blocs matériels en langage C et non plus en HDL afin de les intégrer ensuite dans le *design* matériel complet ; le temps et la complexité de conception sont ainsi réduits. Toutefois, ils ne permettent pas de s'abstraire totalement du flot classique de conception. C'est pourquoi une étape supplémentaire a été franchie avec les outils *SDx* (*Software Defined*) de Xilinx et *FPGA SDK for OpenCL* d'Intel. Ces outils permettent à partir d'une description purement logicielle en C, C++ ou OpenCL de générer le *bitstream* qui va configurer le FPGA. En ce sens, nous pouvons les appeler outils « VHLS » (*Very High Level Synthesis*[3]) afin de les différencier des outils HLS décrits précedemment. Si les outils HLS ou VHLS permettent une portabilité fonctionnelle du code, ils ne garantissent pas la portabilité des performances ; il s'agit cependant d'un bon compromis entre temps de conception et performances du circuit.

1.2 Adéquation Algorithme Architecture (A^3)

L'Adéquation Algorithme Architecture (thème C du GDR ISIS) vise à chercher la meilleure combinaison d'architectures et d'algorithmes pour respecter les contraintes de temps, de consommation énergétique, de coût et bien entendu de précision des résultats. Nous présenterons ici notre positionnement dans ce domaine de recherche

2. Altera avait sorti auparavant son outil *OpenCL compiler* en 2013
3. dénomination non standard

après avoir exposé les parallélismes offerts par les architectures CPU, GPU et FPGA ainsi que le modèle *roofline* pouvant guider la recherche de performances optimales.

1.2.1 Parallélismes

Avec la fréquence de fonctionnement de l'architecture, le parallélisme est le principal levier d'accélération. Nous utiliserons pour illustrer les différentes stratégies de parallélisation que permettent les architectures CPU, GPU et FPGA, l'algorithme de la Fig. 1.2 (a) ; il est un exemple simple d'algorithme massivement parallèle qui doit appliquer sur une masse de données N (sans dépendance entre elles), une même séquence d'opérations avec notamment une boucle d'indice i d'accumulation.

Pipeline CPU

Sur un CPU mono-coeur, l'algorithme 1.2 (a) s'exécutera séquentiellement ; les instructions associées aux différentes opérations sur les données se succéderont dans le pipeline du processeur comme illustré Fig. 1.2 (b). Le parallélisme d'instructions est alors la seule stratégie exploitée dans ce cas : les étages du pipeline permettent d'exécuter simultanément des instructions à différent niveau d'accomplissement. A noter que les parallélisations multi coeur ou vectorielle sont également possibles ; ce manuscrit se concentrant sur les GPUs et FPGAs, elles ne seront pas explorées.

Pipelines GPU

Sur un processeur *many cores* comme les GPUs, les nombreux coeurs de calcul apportent un facteur puissant de parallélisation. Les GPUs de Nvidia sont basés sur un modèle d'exécution SIMT (*Single Instruction Multiple Threads*) où plusieurs tâches logicielles élémentaires, les *threads*, exécutent les mêmes instructions, i.e le même programme. En associant à chaque *thread* le traitement d'une donnée, il correspond alors au modèle SIMD (*Single Instruction Multiple Data*). Au niveau le plus élémentaire, l'exécution se fera par *warps* qui regroupe 32 *threads* ; s'ils possèdent la même séquence d'instructions, ils seront exécutés simultanément sinon séquentiellemement ; les divergences de branche sont ainsi sources d'une plus faible efficacité pour des algorithmes massivement parallèles mais non réguliers.

Le parallélisme de *threads* à l'intérieur d'un *warp* est démultiplié par le nombre de *warps* pouvant être exécute simultanément grâce aux nombreux *Streams Multiprocessors* (SM) existant sur les GPUs ; par exemple, sur l'architecture Volta, chacun des 80 SM est constitué de 64 coeurs, ce qui leur permet d'exécuter simultanément deux *warps*. Par ailleurs, afin de remplir les pipelines d'exécution des SMs, plusieurs *warps* sont exécutés séquentiellement. Par exemple, sur la Fig. 1.2 (c), après un premier *warp* traitant les données [1,W], un deuxième *warp* traitant les données [W+1,2W] est lancé un cycle suivant à l'instant t_2. Ce *Thread Level Parallelism* (TLP) a pour stratégie de maximiser l'*occupancy* du GPU (nombre de *warps* actifs

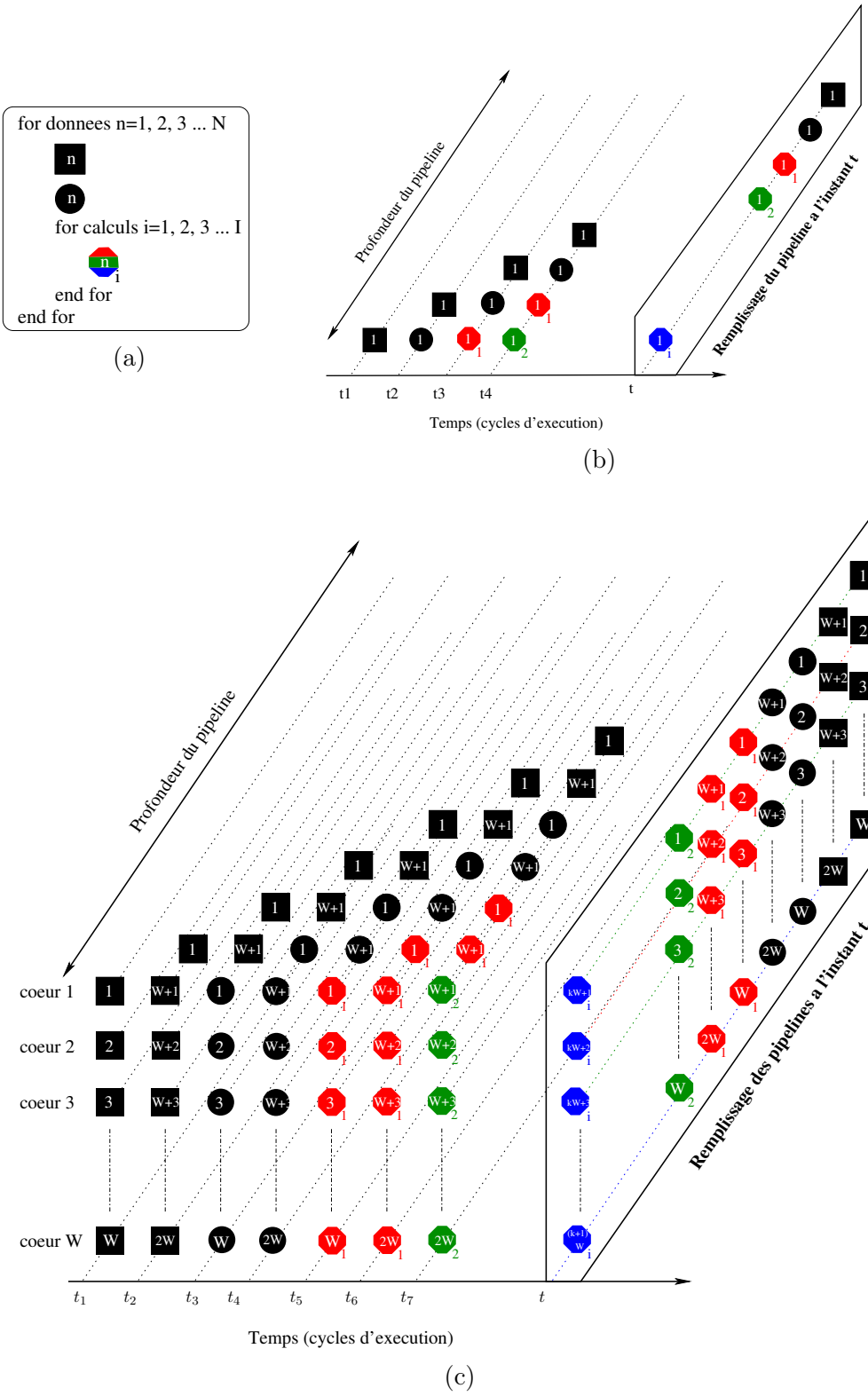


FIGURE 1.2 – Algorithme de traitement de N données avec I calculs pour chaque donnée n (a) executé sur le pipeline du CPU (b) et du GPU (c).

sur le nombre de *warps* résidents maximal des SMs) afin garantir le « remplissage » des pipelines. Pour mettre en oeuvre cette stratégie TLP, le développeur définira des blocs de *threads* de taille correspondante à plusieurs *warps*. Une autre stratégie est l'*Instruction Level Parallelism* (ILP) qui permet d'atteindre cet objectif sans augmenter l'*occupancy* [Volkov 2016] ; un même *thread* traitera plusieurs données, par exemple de cette manière les données [1,W] et [W+1,2W] de la Fig. 1.2 (c) seront traitées par le même *warp*. En pratique, un compromis est à trouver entre les deux stratégies TLP et ILP ; le nombre de *threads* lancés sera plus grand que le nombre de coeurs (TLP) et chaque *thread* pourra être chargé de plusieurs données (ILP). Le réglage fin de ces paramètres a été exploré dans le cadre des travaux de thèse de Thomas Boulay lors du calcul de distances entre signatures radar [R_{2013}].

En conclusion, l'architecture des GPUs permet de mettre en oeuvre un parallélisme de données à grande échelle : les opérations sur une ou quelques données (cf. ILP) sont exécutées séquentiellement et les données traitées en parallèle.

Pipeline FPGA

Sur un FPGA, une architecture sur-mesure sera conçue. Avec les outils HLS, une architecture en pipeline sera vraisemblablement préférée lors de la synthèse de l'algorithme de la Fig. 1.2 (a) : chaque étage correspondra à une opération ou micro-opération de l'algorithme. Il s'agit là d'une différence fondamentale avec les processeurs ; à chaque opération correspondra un opérateur matériel et non une instruction logicielle. Lors de ce parallélisme de pipeline, cas particulier du parallélisme de tâches, toutes les opérations seront exécutées en parallèle ; une nouvelle donnée sera mise dans le « tuyau » à chaque cycle d'horloge comme illustré Fig. 1.2 (c) où à l'instant t_{3P}, 3P données sont dans le pipeline mais à différents niveaux d'accomplissement.

Le pipeline sur FPGA atteindra idéalement un débit de calcul d'une donnée traitée par cycle d'horloge. Il sera ainsi supérieur à celui des processeurs mono-coeur dont le débit sera fonction du nombre de cycles pour exécuter toute la séquence d'instructions nécessaire au traitement d'une donnée. Ce débit n'est que théorique et pourra notamment être affecté par le temps d'établissement du pipeline (Intervalle d'Initialisation dans la terminologie d'Intel). Par ailleurs, le FPGA pourra totalement ou partiellement dupliquer son pipeline. Comme illustré sur la Fig. 1.2 (c), un déroulage de la boucle sur les calculs i se transforme ainsi en une réplication d'opérateurs matériels. Comparés aux processeurs GPU, les principales limitations des architectures FPGA sont leur plus faible fréquence de fonctionnement de l'ordre de 100 à 200 MHz ($\div \sim 8$ par rapport au 1,4 GHz des GPUs) et leur plus faible densité en ressources matérielles.

Les outils VHLS d'Intel utilisent le standard de programmation OpenCL pour exprimer le parallélisme ; le découpage de l'algorithme se fait en *work-items* (équivalents

des *threads* CUDA) accédant à de la mémoire privée, i.e. les registres ; ces *work-items* sont regroupés en *work-groups* (équivalents des blocs de *threads* CUDA) accédant à de la mémoire locale, i.e la mémoire RAM sur la puce (*shared memory* pour CUDA). Deux types de noyau de calcul (*kernels*) sont alors disponibles : le *Single Work Item Kernel* (SWIK) et le *NDRange Kernel* (NDRK). Le dernier correspond à une transcription du modèle SIMT des GPUs sur les FPGAs ; il n'est pas conseillé par les fabricants de FPGA privilégiant le noyau SWIK synthétisant des architectures se singularisant des GPUs. Avec ce modèle, un seul *work-item* est défini correspondant de facto à la mise en pipeline de l'algorithme entier.

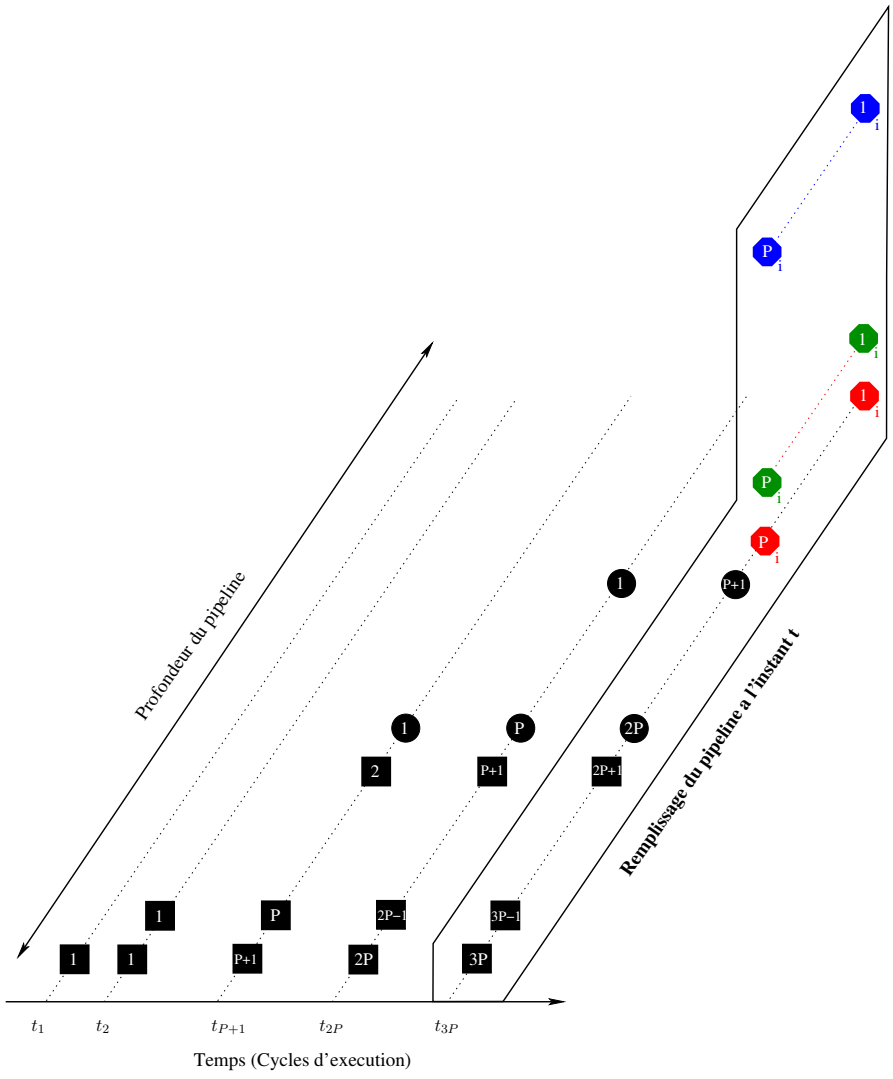


FIGURE 1.3 – Algorithme 1.2 (a) exécuté sur le pipeline d'une architecture FPGA.

En conclusion, les FPGAs permettent de concevoir une architecture avec un paral-

lélisme de pipeline à grande échelle : les données seront traitées en parallèles mais à différents niveaux d'accomplissement correspondant aux différents étages de ce pipeline ; le résultat d'une donnée pourra alors sortir de ce « tuyau » à chaque cycle d'horloge. Un parallélisme de données à petite échelle pourra par la suite être mis en oeuvre par réplication du pipeline en fonction des ressources disponibles sur le FPGA.

1.2.2 Optimisations guidées par le *roofline model*

L'obtention d'un parallélisme « plein » sur une architecture donnée requiert une analyse en amont des propriétés de l'algorithme à implémenter ainsi qu'une connaissance fine du potentiel d'accélération de l'architecture. Afin de prendre le recul nécessaire lors de cette recherche de performances optimales, le *roofline model* [Williams 2009] constitue un outil synthétique utile en première approche. Sur le Fig. 1.4, nous l'avons représenté comme le débit arithmétique D_a exprimé en FLOPs (*Floating Point OPeration per second*) fonction de l'intensité arithmétique I_a exprimée en FLOP/octet, i.e. le nombre d'opérations flottantes rapporté au nombre d'accès à une mémoire ; dans ce modèle « naïf », cette mémoire est la DRAM de la carte de calcul (*global memory* dans la terminologie CUDA).

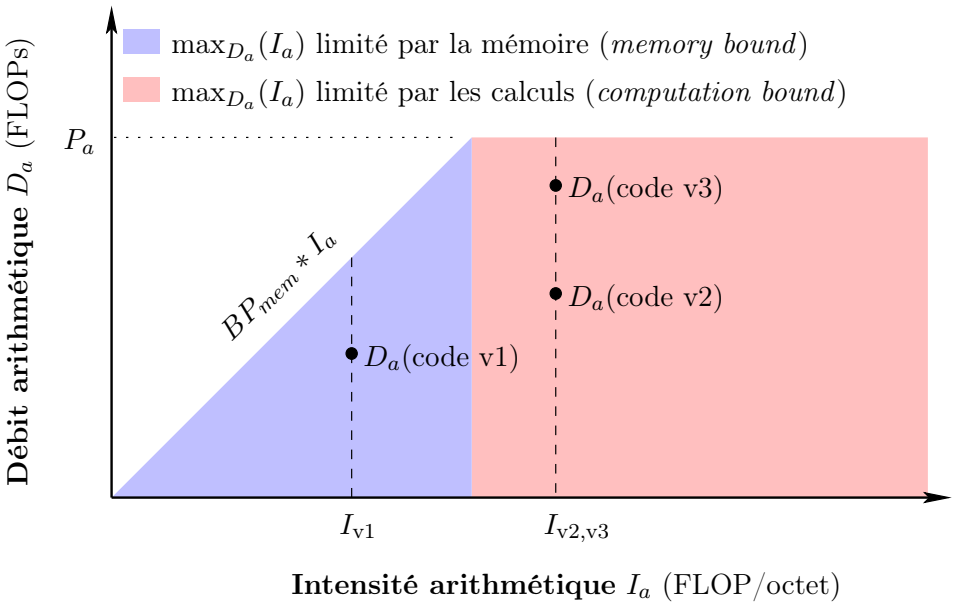


FIGURE 1.4 – *Roofline model* naïf

Sur le graphe du modèle *roofline*, l'implémentation d'un algorithme sera représenté par un point ; par exemple, la version v1 d'un code permettra d'atteindre le débit D_a(v1). L'intérêt de ce modèle est de situer une version du code par rapport à deux facteurs limitant d'une architecture : le débit arithmétique maximal P_a, i.e.

la performance crête (*Peak*), et le débit mémoire maximal BP_{mem}, i.e. la bande passante du bus mémoire reliant les unités de calcul à la mémoire. A une intensité arithmétique I_a donné, caractérisant une version du code, correspondra ainsi une performance maximale $\max_{D_a}(I_a)$; cette dernière sera limitée soit par BP_{mem} (*memory bound*), soit par P_a (*computation bound*). Les optimisations de code permettront ainsi de proche en proche de tendre vers cette performance maximale. Le problème pourra aussi être déporté en jouant sur l'intensité arithmétique. Ainsi, l'expression dans le code d'une plus grande localité spatiale et temporelle lors des accès mémoire, permettra par exemple d'augmenter cette intensité arithmétique : des données déjà accédées ou voisines spatialement de celles-ci seront ainsi respectivement réutilisées ou chargées en avance. Pour ce faire, le code définira une nouvelle séquence d'instructions correspondant à une séquence d'accès mémoire plus adaptée à l'architecture. Cette stratégie mémoire fera appel notamment aux registres, aux mémoires locales ou aux caches mémoire ; sur GPU, elle veillera notamment à la « coalescence » des accès en mémoire des *threads* appartenant à un même *warp*. Ces techniques d'optimisations des performances sur GPU sont détaillées dans de nombreux guides de programmation CUDA [Sanders 2010, Kirk 2016, Khairy 2019].

Les grandes absentes de ce modèle *roofline* naïf sont les latences : arithmétique L_a, i.e. le temps d'exécution d'une opération arithmétique, et mémoire L_{mem}, i.e le temps d'acheminement d'une donnée stockée en mémoire. Pour obtenir un parallélisme « plein », elles sont pourtant des facteurs essentiels ; avec les débits D_a et D_{mem}, elles permettent en effet d'exprimer le nombre maximal de calcul et d'accès mémoire effectuées en parallèle à différents niveaux d'accomplissement respectivement par $L_a * D_a$ et $L_{mem} * D_{mem}$. Par exemple, pour un GPU aux caractéristiques représentatives en première approche de l'architecture Volta (freq=1 GHz, P_a=20 TFLOPs, BP_{mem}=1 To/s, L_{mem}=200 cycles, L_a=20 cycles), une implémentation tirera au maximum bénéfice du potentiel de cette architecture si ses « tuyaux » sont chargés de 400 kFLOP de calculs et de 200Ko d'accès mémoire.

En pratique, le facteur empêchant d'atteindre la borne $\max_{D_a}(I_a)$ est bien souvent la latence mémoire qui est de l'ordre de quelques centaines de cycles d'horloge pour les mémoires globales présentes sur les cartes GPUs. Les efforts d'optimisation se focaliseront de manière prioritaire sur des mécanismes permettant de masquer cette latence mémoire [Volkov 2016]. Par exemple, le code v2 de la Fig. 1.4 se situe sur la zone dite *computation bound* mais c'est grâce à une optimisation mémoire avec une version v3 que ces performances pourraient potentiellement être améliorées en masquant cette latence mémoire. La stratégie couramment employée est de tirer parti des parallélismes TLP ou ILP décrits en 1.2.1. Nous utiliserons la figure 1.2 (c) afin d'illustrer ce principe ; les accès mémoires correspondant à une instruction des *threads* traitant les données [1,W] peuvent bloquer le pipeline s'ils n'arrivent pas à temps avant l'exécution de l'instruction suivante requérant ces données. C'est pourquoi entre ces deux instructions, le pipeline est « rempli », d'instructions traitant d'autres données comme par exemple les données [W+1, 2W] (par les mêmes

threads en ILP ou par d'autres *threads* en TLP).

Sur les FPGAs, l'architecture n'étant pas figée comme sur les processeurs, le modèle *roofline* ne peut pas s'appliquer directement. Des adaptations de ce modèle au FPGA inspirées des premiers travaux sur ce sujet par [Silva 2013, Yali 2015], ont été ainsi proposées dans le cadre de la thèse de Maxime Martelli [Martelli 2019].

1.2.3 Approche globale A^3

Outre les progrès de l'industrie des semi-conducteurs couplés à ceux des outils de compilation et de synthèse, la recherche d'une meilleure adéquation entre algorithmes et architectures se révèle un levier primordial pour accroître l'accélération des calculs. Plusieurs approches d'Adéquation Algorithme Architecture (A^3) sont proposées dans cette optique. Le développement d'outils académiques ou industriels tels que Syndex [Sorel 2004], PREESM [Pelcat 2014] ou Spear [Petrisor 2011] vise par exemple à rendre automatique l'exploration architecturale. A partir d'un modèle en flot de données de l'algorithme et d'un modèle d'architecture, ils permettent un prototypage rapide d'applications. Une autre approche est de s'appuyer sur certaines caractéristiques communes de classes d'algorithmes afin de développer des modules matériels dédiés, à l'instar d'une bibliothèque logicielle. Le cache 3D-AP [Mancini 2004] que j'ai utilisé lors de mes travaux de thèse est un exemple de cette approche.

Je développe mes travaux au sein d'une équipe de chercheurs en traitement du signal développant plus spécifiquement des méthodes pour la résolution des problèmes inverses en grande dimension. Cela a orienté tout naturellement l'approche A^3 que je développe en son sein. Cette approche s'appuie sur les expertises propres de chaque membre de l'équipe afin de faire les choix de méthodes, de modèles d'instrument et d'objet, d'algorithmes d'optimisation mais également de cibles technologiques et d'implémentations, les plus adéquats. Cette approche se focalise sur des cas spécifiques, les plus emblématiques possibles des problèmes inverses comme la reconstruction tomographique ou la radioastronomie. Des principes généraux sont tirés de ces études de cas afin de mieux guider la recherche d'adéquation dans de futurs cas d'applications. Cela a abouti par exemple à un manuel pratique de présentation des leviers d'accélération offerts par les outils VHLS pour l'accélération sur FPGA [Martelli 2019].

Dans cette approche globale, les limitations et le potentiel des architectures d'accélération sont pris en compte en amont de la conception de la méthode de résolution du problème inverse. Les accélérations algorithmiques, logicielles et matérielles y sont conjointement recherchées. Par exemple, une méthode à la complexité théorique importante mais massivement parallèle sera vraisemblablement à privilégier par rapport à une méthode aux propriétés théoriques plus intéressantes mais intrinsèquement séquentielle. La problématique de l'accélération des problèmes inverses

adressée à la fois de manière algorithmique et architecturale, peut être ainsi qualifiée d'« experte » par opposition aux approches A^3 donnant un poids plus important aux outils. Dans le chapitre 2, le cadre algorithmique des problèmes inverses sera posé, puis mes contributions à leur application en reconstruction tomographique seront exposées. Dans les chapitres 3 et 4, leurs accélérations seront étudiées.

Chapitre 2

Problèmes inverses en grande dimension

Sommaire

L'inversion d'un problème linéaire (ou linéarisé au besoin) consiste à obtenir l'objet d'intérêt $\boldsymbol{f}$ à partir des données $\boldsymbol{g}$ de l'instrument. Le modèle direct $\mathbb{M}_D$ d'acquisition s'exprime ainsi sous la forme :

$$\mathbb{M}_D : \boldsymbol{g} = \boldsymbol{H}\boldsymbol{f} + \boldsymbol{\varepsilon}, \tag{2.1}$$

avec $\boldsymbol{H}$ le modèle linéaire de l'instrument, et $\boldsymbol{\varepsilon}$ les incertitudes sur ce modèle et les mesures (e.g. le bruit). Outre l'existence et l'unicité de « la » solution $\hat{\boldsymbol{f}}$, le bon conditionnement de la matrice $\boldsymbol{H}$ est requis pour qu'un problème inverse soit qualifié de « bien posé » au sens de J. Hadamard ; cette condition non satisfaite est synonyme d'instabilités potentielles de la solution $\hat{\boldsymbol{f}}$ à de petites variations des données $\boldsymbol{g}$ comme celles apportées par exemple par les incertitudes $\boldsymbol{\varepsilon}$. Dans bon nombre d'applications, à cause du manque de données ou de la nature mal conditionnée de $\boldsymbol{H}$, le problème est « mal posé ». A cette première difficulté vient s'ajouter celle de la taille du problème inverse à résoudre ; l'accroissement constant du nombre de détecteurs sur les instruments fait exploser la taille de la matrice $\boldsymbol{H}$ qui devient alors difficilement inversible. La résolution des problèmes inverses mal posés en grande dimension ouvre ainsi un large champs de recherche [Idier 2001] visant à définir des

méthodes itératives convergeant rapidement vers la solution $\hat{\boldsymbol{f}}$ la plus précise possible. En reconstruction tomographique par exemple, si des méthodes analytiques d'inversion comme la rétroprojection filtrée offrent des solutions satisfaisantes dans de nombreux cas de figure, les méthodes itératives dénommées MBIR (*Model Based Iterative Reconstruction*) sont devenues prépondérantes dans la littérature et deviennent de plus en plus utilisées en pratique médicale ou industrielle. Ces méthodes permettent en effet d'aller au delà des limitations théoriques des méthodes analytiques trop attachées au modèle mathématique de la transformée de Radon [Radon 1917] ; elles intègrent plus de degrés de liberté pour définir des modèles de la physique d'acquisition (noté $\mathbb{M}_D$) et de l'objet à reconstruire (noté $\mathbb{M}_f$). Elles offrent par exemple une qualité de reconstruction bien supérieure lorsqu'un faible nombre d'angles d'acquisition est utilisé afin de réduire la dose de rayon X au patient ou le temps d'acquisition sur des chaînes de production en CND ; de cette manière, leurs modèles $\mathbb{M}_D$ et $\mathbb{M}_f$ pallient au manque de données.

Afin de définir une méthode d'inversion de données, les approches variationnelles et bayésiennes sont étudiées par le GPI, équipe de recherche à laquelle j'appartiens. La première cherche à optimiser une fonction de coût régularisée ; suivant cette approche, de nombreux travaux dans la communauté du traitement du signal proposent des algorithmes rapides et efficaces en grande dimension, comme par exemple les approches Majoration-Minoration pour l'optimisation [Chouzenoux 2010]. L'approche bayésienne se distincte par une interprétation alternative de toutes les informations liées aux modèles $\mathbb{M}_D$ et $\mathbb{M}_f$ qui prennent la forme de probabilités d'hypothèses. Elles aboutissent à des méthodes semi-supervisées permettant de régler de manière automatique bon nombre de paramètres. Cet avantage nous a amené à la préférer par rapport à l'approche variationnelle, notamment pour l'inversion en tomographie à rayons X.

Dans ce chapitre, nous exposerons tout d'abord les principes généraux des méthodes bayésiennes en mettant en avant les choix devant être faits pour définir une méthode car ceux-ci impacteront son potentiel d'accélération. Nous présenterons ensuite mes contributions au sein du GPI pour l'application de ces méthodes à la reconstruction tomographique. Enfin, nous exposerons les algorithmes de traitement des données des radiotélescopes auxquels je me suis recemment intéressé ; ils constituent d'autres exemples de méthodes d'inversion de données en très grande dimension.

2.1 Méthodes bayésiennes

A partir d'un modèle direct $\mathbb{M}_D$, les méthodes bayésiennes se définissent notamment par les choix d'un modèle *a priori* et d'un estimateur de loi a posteriori. Dans le cas de l'estimateur par le Maximum A Posteriori (MAP), que nous avons tout particulièrement utilisé lors de nos travaux, un algorithme d'optimisation devra également être choisi afin d'atteindre la solution $\hat{\boldsymbol{f}}_{MAP}$. Enfin, pour démontrer leur

intérêt, ces méthodes doivent se confronter au réel dans divers domaines applicatifs.

2.1.1 Loi a posteriori et choix de l'estimateur

L'inférence bayésienne [Idier 2001] est une méthode statistique d'inversion qui cherche à estimer l'objet $\boldsymbol{f}$ à partir de la loi a posteriori $p(\boldsymbol{f}|\boldsymbol{g})$. Celle-ci est dérivée de la vraisemblance et de la loi *a priori* à travers la loi de Bayes :

$$p(\boldsymbol{f}|\boldsymbol{g}) = \frac{p(\boldsymbol{g}|\boldsymbol{f};\mathbb{M}_D)p(\boldsymbol{f};\mathbb{M}_f)}{p(\boldsymbol{g})}. \tag{2.2}$$

La vraisemblance dépend des choix des modèles définissant $\mathbb{M}_D$: modèle linéaire $\boldsymbol{H}$ de l'instrument, et modèle des incertitudes $\boldsymbol{\varepsilon}$ pour lesquels un bruit gaussien de co-variance diagonale $\boldsymbol{V}_\varepsilon$ est généralement considéré :

$$p(\boldsymbol{g}|\boldsymbol{f};\mathbb{M}_D) = \mathcal{N}(\boldsymbol{H}\boldsymbol{f}, \boldsymbol{V}_\varepsilon). \tag{2.3}$$

La loi *a priori* dépend du modèle de l'objet $\mathbb{M}_f$ et peut être exprimée de manière générale sous la forme :

$$p(\boldsymbol{f}|\mathbb{M}_f) \propto \exp(-\lambda R(\boldsymbol{D}\boldsymbol{f})), \tag{2.4}$$

où R est une fonctionnelle, et $\boldsymbol{D}$ un opérateur linéaire.

L'expression de la loi a posteriori ouvre alors la voie à l'obtention d'estimateurs ponctuels comme le Maximum A Posteriori (MAP) :

$$\hat{\boldsymbol{f}}_{MAP} = \arg\max_{\boldsymbol{f}} p(\boldsymbol{f}|\boldsymbol{g}). \tag{2.5}$$

A partir des expressions des lois $p(\boldsymbol{f})$ en 2.4 et de $p(\boldsymbol{g}|\boldsymbol{f})$ en 2.3, la solution $\hat{\boldsymbol{f}}_{MAP}$ de 2.5 correspond alors au minimum de la fonction de coût $J(\boldsymbol{f})$:

$$\hat{\boldsymbol{f}}_{MAP} = \arg\min_{\boldsymbol{f}} J(\boldsymbol{f}) = \arg\min_{\boldsymbol{f}} \frac{1}{2}\|\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}\|^2_{\boldsymbol{V}_\varepsilon} + \lambda R(\boldsymbol{D}\boldsymbol{f}). \tag{2.6}$$

De manière similaire aux méthodes variationnelles, deux termes apparaissent : l'attache aux données $\frac{1}{2}\|\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}\|^2_{V_\varepsilon}$ et la régularisation $\lambda R(\boldsymbol{D}\boldsymbol{f})$. Le premier vise à se rapprocher le plus possible des données de l'instrument alors que le deuxième cherche à pénaliser certaines solutions aux caractéristiques non adaptées à la problématique. Ce deuxième terme compense ainsi les potentielles données manquantes et atténue les effets du bruit qui entache les mesures. Le paramètre de régularisation λ est à fixer afin de donner un poids plus ou moins important à cette régularisation. Dans la méthodologie bayésienne, il peut être intégré dans l'expression des paramètres des lois $p(\boldsymbol{f})$ et $p(\boldsymbol{g}|\boldsymbol{f})$.

L'atout des méthodes bayésiennes par rapport aux méthodes variationnelles est de pouvoir associer des lois de probabilité pour les paramètres $\boldsymbol{\theta}_\mathcal{D}$ et $\boldsymbol{\theta}_f$ des modèles

$\mathbb{M}_D$ et $\mathbb{M}_f$ qui sont alors estimés conjointement avec l'objet. Moins sensibles aux réglages de ces paramètres, ces méthodes hiérarchiques peuvent ainsi être qualifiées de semi-supervisées. Elles définissent alors comme une solution au problème inverse, l'estimateur JMAP (Joint Maximum A Posteriori) :

$$\hat{\boldsymbol{f}}_{JMAP}, \hat{\boldsymbol{\theta}_D}, \hat{\boldsymbol{\theta}_f} = \arg \max_{\boldsymbol{f}, \boldsymbol{\theta}_\mathcal{D}, \boldsymbol{\theta}_f} p(\boldsymbol{f}, \boldsymbol{\theta}_\mathcal{D}, \boldsymbol{\theta}_f | \boldsymbol{g}). \tag{2.7}$$

D'autres estimateurs ponctuels peuvent être obtenus à partir de l'expression de loi a posteriori 2.2 comme l'Espérance A Posteriori (EAP) :

$$\hat{\boldsymbol{f}}_{EAP} = \int \boldsymbol{f}\, p(\boldsymbol{f}|\boldsymbol{g})\, d\boldsymbol{f}. \tag{2.8}$$

Cette EAP est un estimateur pouvant être considéré comme plus naturel que le MAP et peut lui être adjoint en prime des intervalles de confiance. Mais en grande dimension dans le cas de variables non séparables, il requiert des méthodes stochastiques de type `MCMC` [Robert 1997, Robert 2005] aux coûts calculatoires souvent trop lourds pour la génération d'échantillons. Même si des méthodes alternatives basées sur des approximations comme l'approche bayésienne variationnelle `VBA` [Demoment 1989, Zheng 2014] permettent une accélération de ces calculs, l'estimateur de la loi a posteriori le plus pratique demeure $\boldsymbol{f}_{MAP}$ (et par extension $\boldsymbol{f}_{JMAP}$) ; il sera principalement considéré dans le cadre de ce manuscrit.

2.1.2 Choix du modèle *a priori*

Si la physique d'acquisition de l'instrument oriente grandement la définition du modèle direct $\mathbb{M}_D$, un plus grand degré de liberté est offert quant au choix du modèle de l'objet $\mathbb{M}_f$. Trois modèles parmi les plus classiques sont exposés Tab. 2.1.2 lorsqu'une pénalité sur une variation trop importante de l'objet $\boldsymbol{f}$ est recherchée comme c'est généralement le cas pour des images ou volumes possédant des zones homogènes. Cet *a priori* de douceur pénalise malheureusement également les contours que l'on souhaite conserver. La norme L1 est ainsi souvent préférée à la norme L2 car préserve mieux ces fortes variations. Toutefois, la non différentiabilité de cette norme implique des algorithmes d'optimisation plus délicats à mettre en oeuvre que la simple régularisation quadratique.

Dans la méthodologie bayésienne, ces régularisations se justifient par une loi *a priori* $p(\boldsymbol{f})$ associée à l'objet. Ainsi à la régularisation quadratique $\|\mathcal{L}(\boldsymbol{f})\|_2^2$ peut correspondre une loi *a priori* normale centrée $p(\boldsymbol{f}) = \mathcal{N}(0, \boldsymbol{V}_f)$ avec $\boldsymbol{V}_f = diag[\boldsymbol{v}_f]$ et $\boldsymbol{v}_f$ la variance du volume considérée comme un des paramètres fixé à l'initialisation ou estimé conjointement avec l'objet $\boldsymbol{f}$. Pour tout problème inverse, le choix du modèle *a priori* de l'objet $\mathbb{M}_f$ le plus étroitement adapté au contexte applicatif et potentiellement plus complexe que ceux exposés en Tab. 2.1.2, est un levier essentiel afin de définir la solution la plus pertinente que devra par la suite rechercher un algorithme d'optimisation.

	R	$\boldsymbol{D}$	$R(\boldsymbol{D}\boldsymbol{f})$
Régularisation Quadratique (RQ)	L2	Laplacien	$\|\mathcal{L}(\boldsymbol{f})\|_2^2$
Total Variation (TV)	L1	Gradient (anisotropic)	$\|\nabla \boldsymbol{f}\|_1 = \sum_j \|(\nabla_x \boldsymbol{f})_j\| + \|(\nabla_y \boldsymbol{f})_j\| + \|(\nabla_z \boldsymbol{f})_j\|$
		(isotropic)	$\sum_j \sqrt{(\nabla_x \boldsymbol{f})_j^2 + (\nabla_y \boldsymbol{f})_j^2 + (\nabla_z \boldsymbol{f})_j^2}$
Huber	L2/L1	Laplacien	$R_H\big(\mathcal{L}(\boldsymbol{f})\big) = \begin{cases} \big(\mathcal{L}(\boldsymbol{f})\big)^2 & si\ \|\mathcal{L}(\boldsymbol{f})\| \leq \varepsilon_h \\ 2\varepsilon_h \|\mathcal{L}(\boldsymbol{f})\| - \varepsilon_h^2 & si\ \|\mathcal{L}(\boldsymbol{f})\| > \varepsilon_h \end{cases}$ avec ε_h à régler

TABLE 2.1 – Termes de régularisation pour différentes fonctionnelles R et opérateurs $\boldsymbol{D}$.

2.1.3 Choix de l'algorithme d'optimisation

Une fois les modèles $\mathbb{M}_D$ et $\mathbb{M}_f$ définis, un algorithme d'optimisation doit être choisi afin d'obtenir le minimum de la fonction de coût $J(\boldsymbol{f})$ défini en 2.6 comme la solution $\hat{\boldsymbol{f}}_{MAP}$. Pour les fonctions de coût convexe, des algorithmes de Descente de Gradient `DG` peuvent être employés afin de converger vers le minimum global, solution au problème. Si le gradient conjugué est une méthode de choix pour faire de l'optimisation locale et permet de converger plus rapidement que la descente de gradient simple pour un coût calculatoire équivalent, dans la suite du manuscrit, nous préférerons le gradient simple afin d'alléger les équations ; ainsi à chaque itération n, l'estimée $\boldsymbol{f}^{(n+1)}$ est obtenue à partir de l'estimée courante $\boldsymbol{f}^{(n)}$ en descendant dans la direction opposée au gradient $\nabla J(\boldsymbol{f}^{(n)})$:

$$\boldsymbol{f}^{(n+1)} = \boldsymbol{f}^{(n)} - \alpha \nabla J(\boldsymbol{f}^{(n)}), \tag{2.9}$$

avec α, le pas fixe ou estimé à chaque itération. Sans régularisation, `DG` correspond aux mises jour suivantes de l'estimée de l'objet :

$$\boldsymbol{f}^{(n+1)} = \boldsymbol{f}^{(n)} + \alpha 2 \boldsymbol{H}^t(\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}). \tag{2.10}$$

Lorsque le terme $R(\boldsymbol{D}\boldsymbol{f})$ n'est pas différentiable comme c'est le cas par exemple avec la régularisation TV, d'autres algorithmes telles que ADMM [Boyd 2011, Esser 2009], FISTA [Beck 2009, Chambolle 2015], Split Bregman [Goldstein 2009, Esser 2009] ou Champolle-Pock [Chambolle 2011] doivent être employées. Si ces algorithmes peuvent faire appel à plus de variables que les algorithmes `DG` avec une empreinte mémoire plus grande, le caneva algorithmique reste sensiblement le même : une boucle itérative qui met à jour l'estimée de l'objet $\boldsymbol{f}^{(n+1)}$ à partir de l'estimée

courante $\boldsymbol{f}^{(n)}$ et des mesures $\boldsymbol{g}$ en faisant appel à $\boldsymbol{H}$ et $\boldsymbol{H}^t$ au coût calculatoire important.

Dans le cas des méthodes bayésiennes hiérarchiques, la recherche du minimum $\hat{\boldsymbol{f}}_{JMAP}$ sur la loi jointe décrite en 2.7 n'est pas aussi directe. Une des techniques employées est l'optimisation alternée où à chaque itération n, $\boldsymbol{f}^{(n+1)}$ est estimée à partir des données $\boldsymbol{g}$ et des paramètres $\boldsymbol{\theta}_{\mathcal{D}}^{(n)}$ et $\boldsymbol{\theta}_{f}^{(n)}$, puis les paramètres $\boldsymbol{\theta}_{\mathcal{D}}^{(n+1)}$ et $\boldsymbol{\theta}_{f}^{(n+1)}$ le sont à partir de $\boldsymbol{f}^{(n+1)}$. Si cette technique d'optimisation alternée se révèle sensible à l'initialisation, elle a l'avantage de ne pas constituer un coût calculatoire supplémentaire important lorsqu'une expression analytique de l'expression des paramètres $\boldsymbol{\theta}$ existent.

2.1.4 Applications

Mes travaux développés au sein du GPI sont principalement appliqués à la reconstruction tomographique depuis mon arrivée en tant que post-doc en 2008 puis maître de conférences en 2009. Si mes travaux de thèse [R_{2009}] ou leur prolongement [R_{2012}] ont porté sur la Tomographie à Emission de Positons (TEP), ceux effectués au sein du GPI se sont focalisés sur la tomographie à rayons X. Par ailleurs, la radioastronomie est un nouveau champs applicatif exploré depuis ma délégation CNRS à l'Observatoire de la Côte d'Azur (2017-2019) initiée par la participation au projet ANR MAGELLAN.

D'autres champs applicatifs non traités dans ce manuscrit ont été ou sont actuellement investigués :

Localisation de sources acoustiques L'inversion a été vue sous l'angle d'un problème de déconvolution pour lesquelles des méthodes bayésiennes ont été développées dans le cadre de la thèse de Ning Chu [Chu 2013] sous la direction d'Ali Djafari [R$_{2014a}$, C$_{2015c}$, C$_{2014c}$,C$_{2013c}$]. Une modélisation par des noyaux de convolution séparable a été notamment proposée afin de d'accroître le bénéfice d'une accélération sur GPU [C$_{2014a}$, C$_{2014b}$].

Radar Dans le cadre de la thèse de Thomas Boulay [Boulay 2013] sous la direction d'Ali Djafari en collaboration avec Thalès, la reconnaissance de signaux radar en tant que problème de classification supervisée a été accélérée sur GPU afin de traiter la quantité massive de données [R$_{2013}$, C$_{2013a}$, C$_{2012}$]. Si la thèse de Maxime Martelli [Martelli 2019] en collaboration avec Thalès avait comme champs applicatif final la simulation de signaux radar, mes contributions ont principalement portées sur la reconstruction tomographique qui a été le cas d'étude principal pour le développement d'une méthodologie d'accélération sur FPGA via les outils HLS.

Flot optique dense Dans le cadre de la thèse de Mickael Seznec en collaboration avec Thalès TRT sous ma direction, une méthodologie générale d'Adéquation Algorithme Architecture (A^3) permettant de croiser en amont du flot

de conception, les choix algorithmiques, de cibles et d'implémentations, est recherchée. L'application prise comme cas d'études est le flot optique dense qui vise à obtenir les vecteurs de déplacement spatial en chaque pixel d'une séquence temporelle d'images [C$_{2020b}$]. Ce manuscrit se concentrera sur la démarche A^3 pour la résolution de ce problème inverse sans approfondir ces aspects applicatifs.

Planétologie A travers la collaboration que j'ai initiée avec le laboratoire GEOPS, le débruitage de l'instrument PFS (*Planetary Fourier Spectrometer*) a été traité sous l'angle d'une déconvolution 1D massive des spectres de Mars [R$_{2014b}$, C$_{2013b}$].

2.2 Reconstruction tomographique

Au sein du GPI, j'ai contribué à l'utilisation de méthodes bayésiennes pour la reconstruction tomographique. Ces méthodes ont été construites en définissant des modèles d'instrument $\boldsymbol{H}$, d'incertitudes $\boldsymbol{\varepsilon}$ et d'objet $\boldsymbol{f}$ les plus adaptées possibles à l'application visée. Durant ces travaux, nous nous sommes plus particulièrement intéressés au Contrôle Non Destructif (CND).

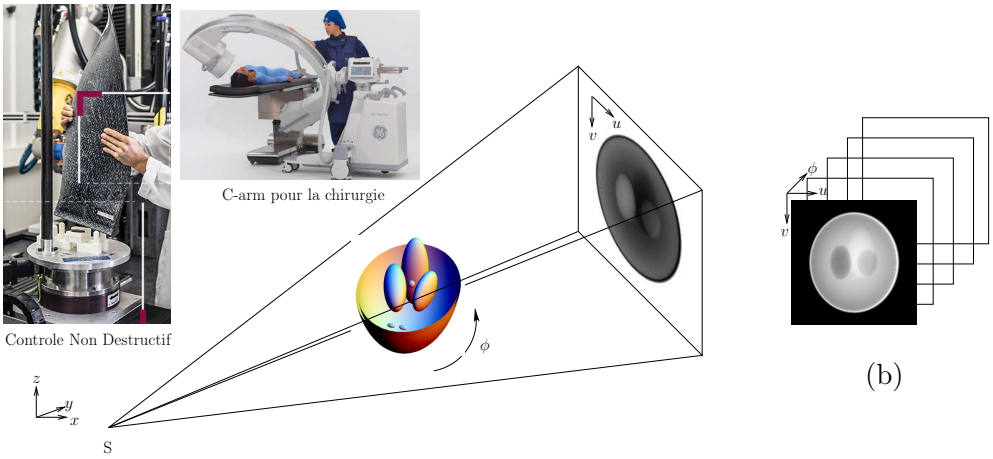


FIGURE 2.1 – Tomographie X : (a) système d'acquisition à géométrie conique utilisé en imagerie médicale ou en CND ; (b) collection de projections 2D correspondant à l'atténuation des rayons traversant le volume à imager.

2.2.1 $\mathbb{M}_D$: modèle de l'instrument H

La tomographie à rayons X est une technique d'imagerie ayant pour but de reconstruire la structure interne d'un volume. Ces trois principaux champs d'application sont : l'imagerie médicale, le Contrôle Non Destructif (CND) et la sécurité pour le

contrôle des bagages. Le volume à imager se situe entre une source de rayons X (ponctuelle dans le cadre de nos travaux) et un plan de détecteurs comme illustré sur la Fig. 2.1 (a). Les détecteurs enregistrent l'énergie des rayons X provenant du faisceau conique de la source après leur diminution lors de la traversée du volume. Les mesures d'atténuation de l'intensité source I_0 forment une projection 2D du volume 3D à chaque angle ϕ d'acquisition. La collection de projections 2D est obtenue en faisant tourner le système source-détecteur autour du patient ou en faisant tourner l'objet sur lui-même en CND.

La décroissance de l'intensité des rayons X dépend de la densité de matière rencontrée dans le volume ; elle est modélisée par la loi de Beer-Lambert qui dans le cas de rayons monochromatiques [Sauer 1993] s'exprime sous la forme :

$$I_i = I_0 \exp\left[-\int_{\mathcal{L}_i} f(\boldsymbol{r})dl\right], \tag{2.11}$$

où I_i et I_0 sont le nombre de photons respectivement enregistrés par le capteur i et envoyés par la source (« image de blanc »), $\mathcal{L}_i$ est la ligne suivie par le rayon à travers le volume, et la quantité $f(\boldsymbol{r})$ exprime le coefficient d'atténuation du volume au point $\boldsymbol{r}$. Afin d'imager l'intérieur du volume, la reconstruction 3D consiste à estimer $f(\boldsymbol{r})$ pour tous les $\boldsymbol{r}$ dans le champs de vue. Le problème est alors linéarisé en estimant $f(\boldsymbol{r})$ à partir de

$$g_i = -\ln\left(\frac{I_i}{I_0}\right) = \int_{\mathcal{L}_i} f(\boldsymbol{r})dl \tag{2.12}$$

plutôt que I_i. Il s'exprime ainsi sous la forme générique $\boldsymbol{g} = \boldsymbol{H}\boldsymbol{f}$ avec $\boldsymbol{H}$ la matrice dite système modélisant la géométrie d'acquisition, et $\boldsymbol{f}$ le volume discrétisé. Collectés sur tout le plan de détecteurs et pour tous les angles d'acquistion, ces g_i forment les données de projections $\boldsymbol{g}$ comme illustré Fig. 2.1 (b).

2.2.2 $\mathbb{M}_D$: modèle des incertitudes ε

Dans les travaux de thèse de L. Wang [Wang 2017] et C. Chapdelaine [Chapdelaine 2019] sous la direction d'Ali Djafari, le modèle de bruit gaussien exprimé en 2.3 a été généralisé dans un cadre bayésien afin de rendre les méthodes semi-supervisées. Une loi inverse gamma $\mathcal{IG}$ est ainsi associée à la variance $\boldsymbol{v}_\varepsilon$, paramètre estimée conjointement avec $\boldsymbol{f}$. La loi sur le bruit devient alors un mélange de lois avec deux hyper-paramètres à fixer $\alpha_{\varepsilon_0}, \beta_{\varepsilon_0}$:

$$\begin{cases} \forall i, & p(\boldsymbol{\varepsilon}_i|v_{\varepsilon_i}) = \mathcal{N}(\varepsilon_i|0, v_{\varepsilon_i}) \\ & p(v_{\varepsilon_i}|\alpha_{\varepsilon_0}, \beta_{\varepsilon_0}) = \mathcal{IG}(v_{\varepsilon_i}|\alpha_{\varepsilon_0}, \beta_{\varepsilon_0}) \end{cases} \tag{2.13}$$

Ce mélange correspond plus directement à une loi de student-t généralisée [Dumitru 2016] :

$$p(\varepsilon_i|\alpha_{\varepsilon_0}, \beta_{\varepsilon_0}) = \mathcal{S}t_g(\varepsilon_i|\alpha_{\varepsilon_0}, \beta_{\varepsilon_0}), \forall i. \tag{2.14}$$

Cette construction hiérarchique permet de n'avoir à fixer réglage que les hyper-paramètres α_{ε_0} et β_{ε_0} ; leur réglage s'avère moins sensible que celui des paramètres $\boldsymbol{v}_\varepsilon$ comme nous avons pu l'étudier sur des données simulées [C_{2017h}] pour différents mélanges de lois. Selon le paramétrage choisi, la loi de student $\mathcal{S}t_g$ se caractérise par une distribution se rapprochant soit d'une loi normale soit d'une loi dite à queue lourde qui favorise les évènements rares.

Dans l'article [R_{2019b}] et le brevet [B$_{2018}$] liés aux travaux de thèse de Camille Chapdelaine, le modèle des incertitudes a été raffiné en le scindant en deux parties :

$$\begin{cases} \boldsymbol{g} = \boldsymbol{g}_0 + \boldsymbol{\xi} \\ \boldsymbol{g}_0 = \boldsymbol{H}\boldsymbol{f} + \boldsymbol{\zeta}, \end{cases} \tag{2.15}$$

avec $\boldsymbol{\xi}$ les incertitudes des mesures modélisées par une loi normale, et $\boldsymbol{\zeta}$ les incertitudes du modèle linéaire de l'instrument modélisées par une loi de student généralisée $\mathcal{S}t_g$. Cette approche vise à trouver un compromis entre complexité du modèle physique d'acquisition [Nuyts 2013] et simplicité de la méthode à mettre en oeuvre. Elle propose ainsi une alternative pragmatique aux modèles polychromatiques plus complets ; si ces derniers permettent de mieux prendre en compte les effets de durcissement de faisceau ou de diffusé, génèrent des calculs trop lourds pour être appliqués en 3D [Martinez 2018].

Dans le cadre de la thèse de Long Chen [Chen 2015] en collaboration avec Carestream Dental sous la direction de Thomas Rodet, les incertitudes $\boldsymbol{\varepsilon}$ ont également joué un rôle plus large que celui de simple bruit gaussien sur les mesures $\boldsymbol{g}$. La méthode de réduction des artefacts métalliques proposée [C_{2013d}] se décompose en deux étapes : (i) corrections $\boldsymbol{g}_c$ des mesures afin de soustraire le bruit entachant $\boldsymbol{g}$ dû au durcissement de faisceau et/ou du diffusé , puis (ii) reconstruction statistique par régularisation pondérée (PWLS) ayant pour fonction de coût :

$$J(\boldsymbol{f}) = (\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f})^t \boldsymbol{W} (\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}) + \lambda R(\boldsymbol{D}\boldsymbol{f}), \tag{2.16}$$

avec $\boldsymbol{W} = \boldsymbol{V}_\varepsilon^{-1}\boldsymbol{W}_c$ et $\boldsymbol{W}_c$ une matrice de poids sur les données $\boldsymbol{g}$. Un poids plus faible est attribué aux données impactées par la correction de données, la confiance associée à ces données étant plus faible.

2.2.3 $\mathbb{M}_f$: modèles par fonctions homogènes par morceaux

En CND, l'objet à imager est dans de nombreux cas, composé de quelques matériaux homogènes. Un modèle *a priori* basé sur des fonctions homogènes par morceaux se révèle ainsi adéquat. Un modèle de Gauss Markov Potts a été ainsi utilisé dans le cadre de la thèse de Camille Chapdelaine [Chapdelaine 2019] pour l'étude de la santé matière de pièces aéronautiques produites par Safran. Ce modèle emploie une variable cachée $\boldsymbol{z}$ suivant une loi de Potts-Markov qui associe à chaque voxel un matériau. A chaque segment du volume est associé une loi gausienne renforçant ainsi l'homogénéité des matériaux. Le modèle bayésien hiérarchique mis en oeuvre pour

la première fois sur des données réels 3D de grande dimension est décrit dans l'article [R_{2017b}]. La principale originalité de cette méthode est son estimation itérative d'une segmentation du volume comme illustré en Fig. 2.2 et 2.3.

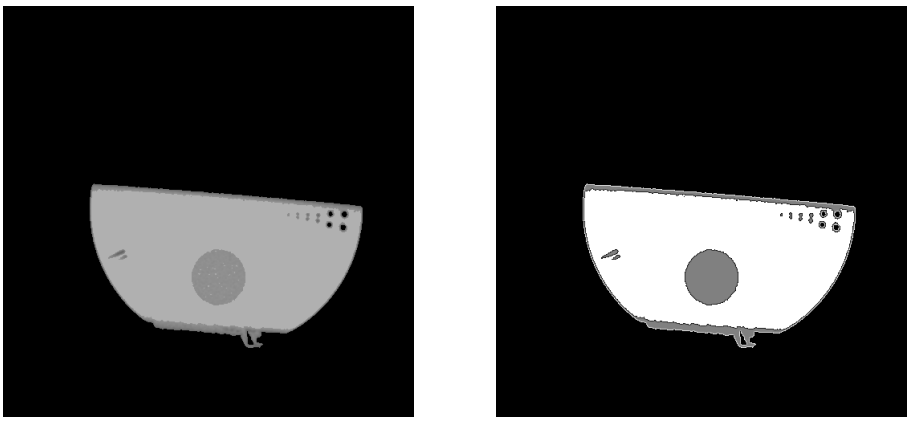


FIGURE 2.2 – Image Quality Indicator (IQ) reconstruit à partir de 300 projections par la méthode de Gauss-Markov-Potts avec sa segmentation conjointement estimée ; *source : [Chapdelaine 2019].*

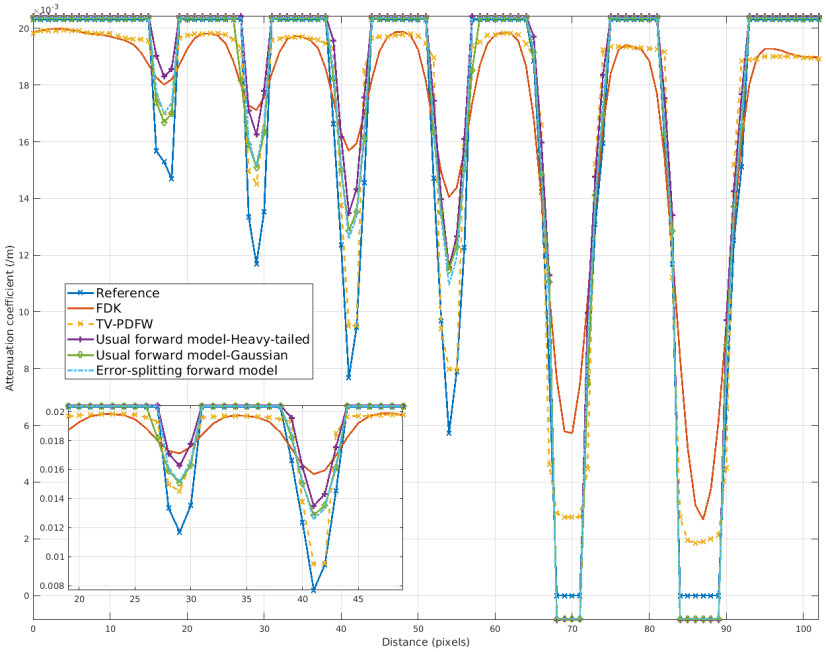


FIGURE 2.3 – Profil sur la ligne de trous de l'IQI pour différentes méthodes de reconstruction à partir de 300 projections (la reconstruction avec 2400 projections est prise comme référence) : FDK, TV-PDFW [Ongie 2018] et méthodes de Gauss-Markov-Potts avec *a priori* sur les incertitudes gaussien, student ou mixte avec le modèle *error splitting* ; *source : [Chapdelaine 2019].*

La recherche d'une représentation parcimonieuse de l'objet $\boldsymbol{f}$, i.e. avec un nombre de valeurs non nuls beaucoup plus important que dans le volume lui même, est une approche qui s'est largement répandue ces dernières années. Le problème inverse mal posé lié à la reconstruction de l'objet revient alors à l'estimation des coefficients parcimonieux. Ainsi les régularisations ou *a priori* sont focalisés sur cette représentation des données et favorise ainsi efficacement les solutions présentant cette caractéristique singulière de parcimonie. Les représentations plus élaborées que le simple gradient sont basées sur les coefficients de transformation d'ondelettes ou de dictionnaires. Dans ce courant de méthodes, les travaux de thèse de Li Wang ont ainsi utilisé comme variable cachée $\boldsymbol{z}$ les coefficients de transformées par ondelettes ; ces derniers sont particulièrement bien adaptés au contexte du CND où les volumes sont composés de matériaux homogènes par morceaux. Un modèle par synthèse a d'abord été proposé [C_{2016a}] où la transformée par ondelettes $\boldsymbol{z} = \mathcal{O}(\boldsymbol{f})$ est d'abord estimée puis $\boldsymbol{f}$ obtenue en post-traitement à partir de $\boldsymbol{z}$. Ce modèle a été ensuite amélioré en prenant en compte $\boldsymbol{\xi}$ les incertitudes liées aux calculs des coefficients de la transformée en ondelettes :

$$\boldsymbol{z} = \mathcal{O}(\boldsymbol{f}) + \boldsymbol{\xi}. \tag{2.17}$$

Toujours dans un cadre bayésien hiérarchique, les paramètres des lois (les variances $\boldsymbol{v}_\varepsilon$, $\boldsymbol{v}_\xi$ et $\boldsymbol{v}_z$) et la variable cachée $\boldsymbol{z}$ ont été estimées conjointement avec le volume $\boldsymbol{f}$ sur des données 2D [R_{2017a}]. Cette méthode a été ensuite étendue à des données 3D réelles avec des comparaisons et analyses plus approfondies [R_{2018b}].

2.3 Radioastronomie

Mes travaux en radioastronomie portent sur : (i) la **déconvolution** au coeur des pipelines de formation des images du ciel et (ii) le **démélange de gaz** dans la matière interstellaire, correspondant à un post-traitement sur ces images. Le volume astronomique des données traitées par les radiotélescopes comme le futur SKA constitue un défi calculatoire pour ces deux applications.

2.3.1 Déconvolution

En mode imageur, les radiotélescopes génèrent des collections d'images du ciel observé à différentes longueurs d'onde appelées hypercubes. Dans les réseaux d'interféromètres, chaque paire d'antennes génère à un instant et à une fréquence donné, une mesure appelée visibilité correspondant à la corrélation des signaux sur ces deux antennes. Si on néglige les nombreux et importants effets dépendants de la direction (DDE), ces visibilités correspondent à une représentation dans le domaine de Fourier du ciel. Mais l'échantillonnage en fréquence spatiale par le système d'acquisition ne correspondant pas à celle du volume, un ré-échantillonnage sur une grille cartésienne appelé *gridding* est utilisée afin de bénéficier de la rapidité de l'algorithme FFT. L'image ainsi produite est appelée *dirty image* car correspond à une image floutée comme illustré sur la Fig. 2.3.1 ; elle peut être modélisée par une convolution de

l'image du ciel. La déconvolution cherche ainsi à retirer les effets parasites de l'instrument dans le domaine image. A noter qu'une particularité de cette déconvolution est que le noyau est de la taille de l'image. La méthode CLEAN [Högbom 1974] reste une méthode de référence avec de nombreuses extensions proposées dans la littérature comme l'algorithme MS-MFS (Multi-Scale Multi-Frequency Synthesis) [Rau 2011]. Toutefois, les méthodes basées sur des représentations parcimonieuses ont prouvé depuis leur supériorité par rapport à cette méthode classique : sur données simulées [Wiaux 2009] avec des techniques de *compressed sensing* ou sur données réelles provenant des télescopes LOFAR [Garsden 2015] ou VLA [Dabbech 2018].

Dans ce contexte, j'ai effectué une délégation CNRS de 2017 à 2019 au laboratoire Lagrange. Ce laboratoire a développé l'algorithme **MUFFIN** (MUlti-Frequency sparse radio Interferometric imagiNg) [Ammanouil 2019] pour lequel une distribution des calculs sur les différents noeuds de calcul a été proposée et mise en oeuvre. Ma collaboration avec André Ferrari se focalisent sur l'accélération GPU ; un algorithme simple de déconvolution par descente de gradient avec régularisation quadratique est actuellement le premier support à cette étude. Le gain en performances obtenu avec le calcul et le stockage en demi-précision flottant [C$_{2018b}$] est présenté au chapitre 3, et la parallélisation multi-GPU rendue délicate avec la double régularisation spatiale et spectrale au chapitre 4.

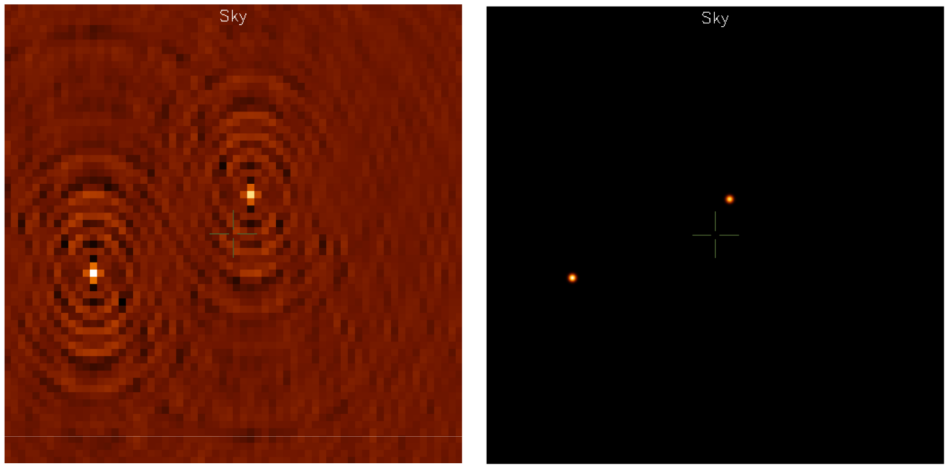


FIGURE 2.4 – Déconvolution en radioastronomie : (gauche) *dirty image*, (droite) image du ciel avec des sources ponctuelles.
Source : I.M. Steward, Bielefeld University

2.3.2 Démélange de gaz

Dans le cadre du projet Projet HyperStars (Programme Mastodon du CNRS), le GPI collabore avec le département d'astrophysique du CEA Saclay pour apporter son expertise en problème inverse en grande dimension aux travaux menés par Marc Antoine Miville-Deschênes sur le démélange de gaz interstellaires.

Les hypercubes générés par les radiotélescopes correspondent à une collection de cartes 2D de la température de brillance $\boldsymbol{T}_B(\lambda)$ du ciel à différentes longueurs d'onde λ. Comme illustré sur la Fig. 2.5, chaque pixel sur ces cartes correspond à une ligne de visée (x, y). Les longueurs d'onde étant proportionnelles à la vitesse de déplacement des sources par effet de Doppler, l'axe de profondeur des hypercubes est donc indiqué de manière équivalente en longueur d'onde λ ou en vitesse v_z (km/s). La méthode ROHSA (*Regularized Optimization for Hyper-Spectral Analysis*) développée par Antoine Marchal a pour but de séparer les différentes sources de rayonnement thermique. Dans ce but, un modèle de température de brillance $\widetilde{\boldsymbol{T}}_B(v_z)$ composé de N_G gaussiennes est utilisé :

$$\widetilde{\boldsymbol{T}}_B(v_z) = \sum_{n=1}^{N_G} G\big(v_z, \boldsymbol{\theta}_n\big), \tag{2.18}$$

avec $\boldsymbol{\theta}_n \in I\!R^{3*N_x*N_y}$ correspondant aux cartes 2D de paramètres de chaque gaussienne G, i.e. ses amplitudes a_n, ses moyennes μ_n et ses variances σ_n. Afin d'obtenir une variation douce de ces paramètres, la décomposition $\widetilde{\boldsymbol{T}}_B(v_z)$ correspondant le mieux aux données $\boldsymbol{T}_B(v_z)$ est recherchée en minimisant un critère régularisé :

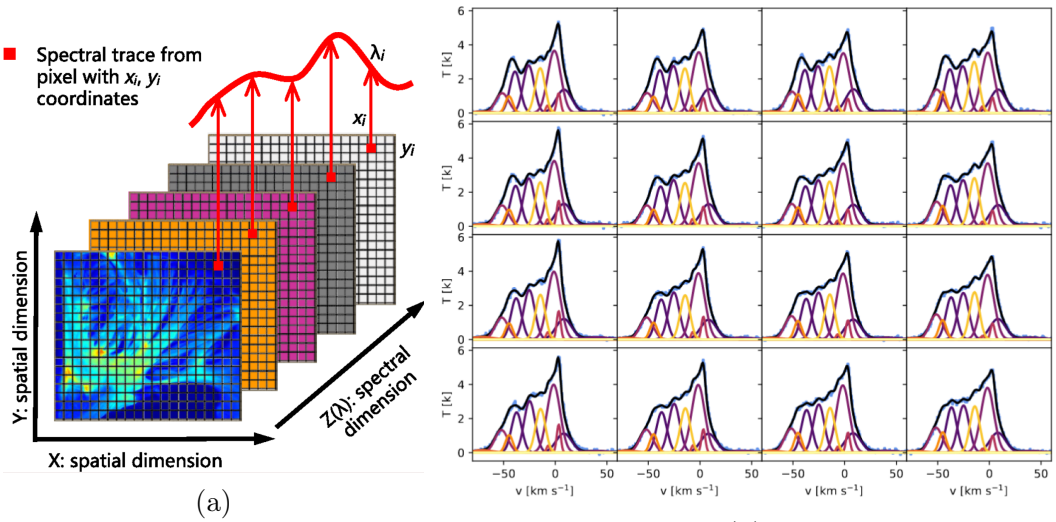
$$\begin{aligned} J(\boldsymbol{\theta}) \quad = \quad & \frac{1}{2}\|\widetilde{\boldsymbol{T}}_B(v_z) - \boldsymbol{T}_B(v_z)\|^2_{\boldsymbol{V}_\varepsilon} + \\ & \frac{1}{2}\sum_{n=1}^{N_G} \lambda_a||\mathcal{L}(\boldsymbol{a}_n)||_2^2 + \lambda_\mu||\mathcal{L}(\boldsymbol{\mu}_n)||_2^2 + \lambda_\sigma||(\boldsymbol{\sigma}_n)||_2^2, \end{aligned} \tag{2.19}$$

avec λ_a, λ_μ et λ_σ des paramètres de régularisation à fixer, et $\boldsymbol{V}_\varepsilon$ la variance spatiale du bruit de mesure. Un terme de régularisation a été ajouté afin que chaque gaussienne définisse un gaz aux propriétés spatialement homogènes :

$$\lambda'_\sigma||\boldsymbol{\sigma}_n - m_n||_2^2, \tag{2.20}$$

avec m_n une valeur scalaire, et λ'_σ un paramètre à fixer. Les variances « moyenne » m_n de chaque gaussienne sont estimées conjointement avec les cartes de paramètres $\boldsymbol{\theta}$.

Antoine Marchal a associé l'ensemble des participants du projet Hyperstar à la publication de ses résultats de thèse dans le journal [R$_{2019a}$] et a rendu cette méthode accessible à la communauté en astrophysique via la diffusion du logiciel ROHSA. Une version accélérée sur GPU est en cours de développement et fait l'objet des travaux en parcours recherche à CentraleSupélec de J. Besson que j'encadre.



(a) (b)

FIGURE 2.5 – Démélange de gaz par la méthode ROHSA : (a) hypercube formé des températures de brillance $T_B(x, y, v_z) = \boldsymbol{T}_B(v_z)$ de gaz mélangés, et (b) décomposition de $\boldsymbol{T}_B(v_z)$ en gaussiennes dans l'axe v_z (km/s) pour différentes lignes de visée (x, y).

2.4 Conclusion

Dans ce chapitre, le cadre algorithmique général pour la résolution des problèmes inverses en grande dimension a été présenté. En reconstruction tomographique, les méthodes bayésiennes hiérarchiques développées par le GPI permettent une amélioration de la qualité de reconstruction grâce à des modèles plus proches du système d'acquisition et de l'objet intérêt. Ces méthodes semi-supervisées facilitent par ailleurs le difficile réglable des paramètres des algorithmes avec régularisation. Toutefois, elles s'accompagnent d'une augmentation du coût calculatoire avec des variables cachées ou des paramètres de la taille de $\boldsymbol{f}$ ou $\boldsymbol{g}$. En radioastronomie, les méthodes pour la déconvolution ou le démélange de gaz présentent un caneva algorithmique très proche avec une boucle itératif faisant appel à des opérateurs coûteux en temps de calcul. Dans le chapitre suivant, une parallélisation massive sur GPU ou FPGA pour l'accélération de l'application de la matrice $\boldsymbol{H}$ et $\boldsymbol{H}^t$ sur les données, sera exposée. Dans le chapitre 4, le calcul distribué sur serveur multi-GPU faisant apparaître des dépendances de données lors de la boucle itérative sera ensuite présenté.

Chapitre 3

Calcul intensif par opérateurs de $\boldsymbol{H}\boldsymbol{f}$ et $\boldsymbol{H}^t\boldsymbol{g}$

Sommaire

Les méthodes itératives d'inversion de données comme la descente de gradient (cf. équation 2.10) utilisent à de nombreuses reprises le modèle d'instrument $\boldsymbol{H}$ lors du calcul de données estimées $\hat{\boldsymbol{g}} = \boldsymbol{H}\boldsymbol{f}^{(n)}$ et de corrections sur l'objet $\boldsymbol{H}^t\boldsymbol{\delta}_g = \boldsymbol{H}^t(\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}^{(n)})$. Ces deux étapes constituent le principal coût calculatoire des méthodes itératives en grande dimension et le principal levier d'accélération.

La matrice $\boldsymbol{H}$ contient les contributions H_{ij} de chaque élément j de l'objet $\boldsymbol{f}$ dans le détecteur i des données $\boldsymbol{g}$. Sa dimension $N_f \times N_g$ est gigantesque mais s'avère dans de nombreuses applications creuse. Ainsi en pratique, les calculs de $\boldsymbol{H}\boldsymbol{f}$ et $\boldsymbol{H}^t\boldsymbol{\delta}_g$ ne se font pas par une multiplication de matrices mais à l'aide d'opérateurs calculant en ligne uniquement les éléments H_{ij} nécessaires. En effet, le stockage entier de la matrice est impossible ; il nécessiterait par exemple 256 exaoctets pour la reconstruction tomographique d'un volume de 2048^3 voxels à partir de 2048 projections sur un plan de 2048^2 pixels détecteur. Même si des symétries existent afin de réduire son stockage, l'utilisation de moteur de calcul comme les GPUs s'avèrent plus efficaces que d'effectuer de nombreux accès en lecture à une matrice gigantesque stockée en mémoire disque car dépassant la capacité de la mémoire RAM.

Si les méthodes d'inversion décrites en section 2.1 sont génériques, les opérateurs associés à $\boldsymbol{H}$ et $\boldsymbol{H}^t$ eux sont spécifiques à l'application car liés directement à l'instrument $\boldsymbol{H}$. Ainsi $\boldsymbol{H}$ et $\boldsymbol{H}^t$ correspondent pour la reconstruction tomographique à une

paire de projection/rétroprojection $\mathcal{P}/\mathcal{R}$ et pour les imageurs utilisés en radioastronomie à des convolutions $\mathcal{C}$. Dans ce chapitre, nos contributions pour l'accélération de $\mathcal{P}/\mathcal{R}$ puis de $\mathcal{C}$ sont exposées.

3.1 Reconstruction tomographique

Dans cette section, les différentes paires de projection/rétroprojection et l'importance de la dualité de ces deux opérateurs sont tout d'abord exposées. Leurs accélérations sur GPU et FPGA sont ensuite présentées.

3.1.1 Paires de projection/rétroprojection

Découlant de la linéarisation de la loi de Beer-Lambert (voir eq. 2.12), l'opération de projection $\mathcal{P}$ correspond au calcul de l'intégrale linéaire du volume discrétisé $f(\mathtt{xn}, \mathtt{yn}, \mathtt{zn})$ le long des rayons X associés aux mesures $g(\mathtt{un}, \mathtt{vn}, \mathtt{phi})$ des différents détecteurs du tomographe. L'opération de rétroprojection $\mathcal{R}$ correspond au dépôt des mesures $g(\mathtt{un}, \mathtt{vn}, \mathtt{phi})$ dans le volume le long des rayons X reliant le détecteur à la source. A noter que cette opération ne correspond pas à l'inverse de la projection : $\mathcal{R} \neq \mathcal{P}^{-1}$. Ainsi, les méthodes analytiques comme l'agorithme Feldkamp [Feldkamp 1984] qui utilisent lors de la reconstruction seulement l'opérateur de rétroprojection, l'appliquent sur des données de projection préalablement filtrées afin d'éliminer les artefacts en étoile apparaissant avec le rétroprojecteur seul. Les méthodes itératives MBIR permettant de s'affranchir des limites des méthodes analytiques (voir chap. 2), font nécessairement appel à l'expression discrétisée de cette paire d'opérateurs définie par :

$$\left\{\begin{array}{lcccc} \mathcal{P} & : & IR^{N_{\mathtt{xn}}\cdot N_{\mathtt{yn}}\cdot N_{\mathtt{zn}}} & \longrightarrow & IR^{N_{\mathtt{un}}\cdot N_{\mathtt{vn}}\cdot N_{\mathtt{phi}}} \\ \mathcal{R} & : & IR^{N_{\mathtt{un}}\cdot N_{\mathtt{vn}}\cdot N_{\mathtt{phi}}} & \longrightarrow & IR^{N_{\mathtt{xn}}\cdot N_{\mathtt{yn}}\cdot N_{\mathtt{zn}}} \end{array}\right.$$
$$\text{vérifiant} \qquad \left\{\begin{array}{lcl} \forall \boldsymbol{f}, \quad \mathcal{P}(\boldsymbol{f}) & = & \boldsymbol{H}\boldsymbol{f} \\ \forall \boldsymbol{g}, \quad \mathcal{R}(\boldsymbol{g}) & = & \boldsymbol{H}^t\boldsymbol{g}, \end{array}\right. \tag{3.1}$$

avec $\boldsymbol{H}$ la matrice système modélisant l'instrument.

Pour la constitution d'une paire de projection/rétroprojection, plusieurs modèles décrits en détails en annexe A existent : *ray-driven*, *voxel-driven*, *distance-driven* [De Man 2004] et *separable footprint* [Long 2010]. Ces modèles se différencient par la modélisation des voxels et des détecteurs qu'ils utilisent. Les modèles *voxel-driven* et *ray-driven*, les plus simplistes, ne prennent en compte que les centres du voxel ou du détecteur ; le modèle *distance-driven* [De Man 2004] utilise les limites de chaque voxel et détecteur définit en quatre points pris dans le plan orthogonal à l'axe majeur (voir définition en 1.1) ; le modèle à empreinte séparable, le plus élaboré, utilise les huit sommets des voxels et les quatre sommets des détecteurs sans un coût calculatoire trop important grâce à sa séparation du problème en deux : dans le plan

transverse, quatre sommets sont pris en compte, et dans le plan axial deux sommets sont pris en compte pour une empreinte axiale rectangulaire ou quatre pour une forme trapézoïdale. Avec cette interprétation du problème direct, les modèles *voxel-driven* et *ray-driven* peuvent être qualifiés de 1D, les modèles distance-driven de 2D et le modèle à empreinte séparable de 3D.

Les modèles simples *ray-driven* et *voxel-driven* sont les plus répandus du fait de leur coût calculatoire moins important. Mais c'est sous une forme hybride avec une paire P_{RD}/P_{VD} formée d'un projecteur *ray-driven* et d'un rétroprojecteur *voxel-driven* qu'ils sont employés afin d'éviter les écueils des rétroprojecteurs *ray-driven* $\mathcal{R}_{RD}$ et des projecteurs *voxel-driven* $\mathcal{R}_{VD}$. En effet, ces deux opérateurs introduisent non seulement des artefacts hautes fréquences [De Man 2002] mais aboutissent à des implémentations peu efficaces dues à une accumulation des valeurs ne se faisant pas sur l'indice de la boucle externe. Par exemple, pour la projection *voxel-driven*, la boucle externe est en $\texttt{xn}, \texttt{yn}, \texttt{zn}$ et les accumulations dans une donnée de projection $\mathcal{P}(\texttt{un}, \texttt{vn}, \texttt{phi})$ se font à ainsi des itérations différentes. Cela impliquera de nombreux accès en écriture concurrents dans la mémoire par les différents *threads* $\texttt{xn}, \texttt{yn}, \texttt{zn}$ lors d'une parallélisation sur GPU. A l'inverse, pour la rétroprojection *voxel-driven* $\mathcal{R}_{VD}$, les accumulations se font lors de la même itération de la boucle externe, c'est à dire séquentiellement par un seul *thread* GPU. Ces accumulations séquentielles se feront alors localement dans des registres et éviteront les conflits d'écriture entre *threads*.

3.1.2 Dualité des opérateurs $\mathcal{P}$ et $\mathcal{R}$

Si les opérateurs $\mathcal{P}_{RD}$ et $\mathcal{R}_{VD}$ sont un bon compromis entre précision du modèle et rapidité d'exécution, ils possèdent cependant une faiblesse théorique lors de leur utilisation par des algorithmes itératifs. En effet, ce projecteur et ce rétroprojecteur modélisent chacun différemment la matrice $\boldsymbol{H}$: $\mathcal{P}(\boldsymbol{f}) = \boldsymbol{H}_{\mathcal{P}} f$ et $\mathcal{R}(\boldsymbol{g}) = \boldsymbol{H}^t_{\mathcal{R}} \boldsymbol{g}$ avec $H_P \neq H_{\mathcal{R}}$. Les opérateurs sont alors dits non adjoints [Zeng 2000, Lee 2011, Arcadu 2016, Lou 2019] car ne respectent plus la condition de dualité définie ci-dessous :

$$\forall \boldsymbol{f}, \forall \boldsymbol{g}, < \boldsymbol{g} \cdot \mathcal{P}(\boldsymbol{f}) >=< \mathcal{R}(\boldsymbol{g}) \cdot \boldsymbol{f} >, \tag{3.2}$$

avec $< \cdot >$ le produit scalaire.

Si la paire est non duale, elle peut toutefois être qualifiée de valide [Zeng 2000] si la garantie de convergence est assurée par l'absence de valeur propres négatives dans $H^t_{\mathcal{R}} H_{\mathcal{P}}$. En pratique, cette non dualité voire la non validité ne pose pas de problème notoire de convergence des algorithmes car les effets sont perceptibles seulement après un très grand nombre d'itérations pour la majorité des méthodes itératives. Une paire non duale $\mathcal{P}_{RD}/\mathcal{R}_{VD}$ peut même apporter des bénéfices sur la qualité de reconstruction par rapport à des paires duales comme la suppression d'artefacts en anneaux rencontrés avec une paire $\mathcal{P}_{Siddon}/\mathcal{R}_{Siddon}$ [Zeng 1997]. Pour toutes

ces raisons, les *toolkits* tels qu'Astra [Palenstijn 2017], RTK [Rit 2014] ou TIGRE [Biguri 2016] utilisent des paires non duales $\mathcal{P}_{RD}/\mathcal{R}_{VD}$. Toutefois, cette sous optimalité et non garantie de convergence entraîne une tendance depuis quelques années à préférer une paire duale basée sur les modèles *distance-driven* ou *separable footprint* qui sont par ailleurs plus précis que les modèles *ray-driven* ou *voxel-driven*. A noter que des modèles basés sur des fonctions de base B-splines pour le volume et les projections existent également comme le *spline-driven* [Momey 2015] ou celui basé sur une convolution de B-splines [Savanier 2020] ; ils permettent ainsi de créer également des paires duales et aboutissent à une formulation explicite des coefficients d'interpolation sans une inflation du coût calculatoire.

Lors de la thèse de Camille Chapdelaine [Chapdelaine 2019], les utilisations des paires duale $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ du modèle *Separable Footprint* et non duale $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ avec un projecteur *ray-driven* par échantillonnage régulier et un rétroprojecteur *voxel-driven* avec interpolation linéaire ont été comparées. Si pour un algorithme TV classique [Ongie 2018], la différence de convergence observée pour les deux paires est quasi négligeable, pour l'algorithme Gauss-Markov-Potts, elle est plus notable comme illustré sur la Fig. 3.1.

Outre ces différences de convergence, l'utilisation d'une paire non duale peut s'avérer être un verrou pour certaine technique d'accélération algorithmique. Ainsi afin de pouvoir utiliser la méthode bayésienne variationnelle (`VBA`) pour l'algorithme de Gauss-Markov-Potts, le calcul de la diagonale de matrices $\boldsymbol{A}$ et $\boldsymbol{B}$ de grande taille est requis :

$$\left\{\begin{array}{lcl} \forall j,\, A_j &=& \left[\boldsymbol{H}^t\boldsymbol{V}_\varepsilon^{-1}\boldsymbol{H}\right]_{jj} \\ \forall i,\, B_i &=& \left[\boldsymbol{H}\boldsymbol{V}_f\boldsymbol{H}^t\right]_{ii}, \end{array}\right. \tag{3.3}$$

avec $\boldsymbol{V}_\varepsilon = \mathrm{diag}[\boldsymbol{v}_\varepsilon]$ et $\boldsymbol{V}_f = \mathrm{diag}[\boldsymbol{v}_f]$ correspondant respectivement à la variance des incertitudes et du volume estimé. La mise à jour de ces paramètres de loi de la taille des données N_g et N_f quand des modèles non stationnaires sont utilisés, se fait à chaque itération globale des méthodes bayésiennes hiérarchiques. Tels qu'exprimées en 3.3, $\boldsymbol{A}$ et $\boldsymbol{B}$ requièrent pour le calcul de chacun de leurs éléments diagonaux une projection et une rétroprojection respectivement sur des volumes élémentaires $e_i^{(i')} = \delta(i-i'), \forall i'$ et des données élémentaires $e_j^{(j')} = \delta(j-j'), \forall j$. Afin de contourner ce coût calculatoire rédhibitoire, l'équation 3.3 peut être ré-exprimée sous la forme :

$$\left\{\begin{array}{lcccl} \forall j,\, A_j &=& \displaystyle\sum_i H_{ij}^2[\boldsymbol{v}_\varepsilon^{-1}]_j &=& \mathcal{R}^{(2)}(V_\varepsilon^{-1}) \\ \forall i,\, B_i &=& \displaystyle\sum_j [H^t]_{ij}^2[\boldsymbol{v}_f]_i &=& \mathcal{P}^{(2)}(V_f)\,, \end{array}\right. \tag{3.4}$$

avec $\mathcal{R}^{(2)}$ et $\mathcal{P}^{(2)}$ des opérateurs de rétroprojection et de projection ayant leurs coefficients au carré. Le coût pour le calcul entier de $\boldsymbol{A}$ et $\boldsymbol{B}$ est alors équivalent à celui d'une opération respectivement de rétroprojection et de projection. Mais cette technique d'accélération algorithmique n'est possible qu'avec l'emploi d'une paire duale.

Dans le cas contrainte, les calculs de A_j et B_j seraient faussées et entraîneraient une instabilité pour la convergence de l'algorithme. Ces effets de raccourci calculatoire oubliant la non dualité de la paire employée sont observés également pour des méthodes plus simples. Par exemple, sans régularisation et avec un bruit stationnaire, si lors du calcul du pas optimal de l'algorithme de descente de gradient l'approximation 3.5 faisant appel à l'opérateur de projection $\mathcal{P}$ est faite , alors au bout d'un certain nombre d'itérations, une divergence de l'algorithme peut apparaître comme nous l'avons déjà observé.

$$\alpha = \frac{\|\Delta J(\boldsymbol{f}^{(n)})\|_2^2}{\left(\Delta J(\boldsymbol{f}^{(n)})\right)^t \boldsymbol{H}^t \boldsymbol{H} \Delta J(\boldsymbol{f}^{(n)})} \simeq \frac{\|\Delta J(\boldsymbol{f}^{(n)})\|_2^2}{\|\mathcal{P}\left(\Delta J(\boldsymbol{f}^{(n)})\right)\|_2^2} \tag{3.5}$$

Ce verrou des calculs des matrice $\boldsymbol{A}$ et $\boldsymbol{B}$ levé par l'utilisation d'une paire duale, nous a permis ainsi d'effectuer des reconstructions avec la méthode `VBA` sur des données réelles 3D [C_{2019c}] ce que nous n'avions pu faire qu'en 2D au préalable [C_{2016a}].

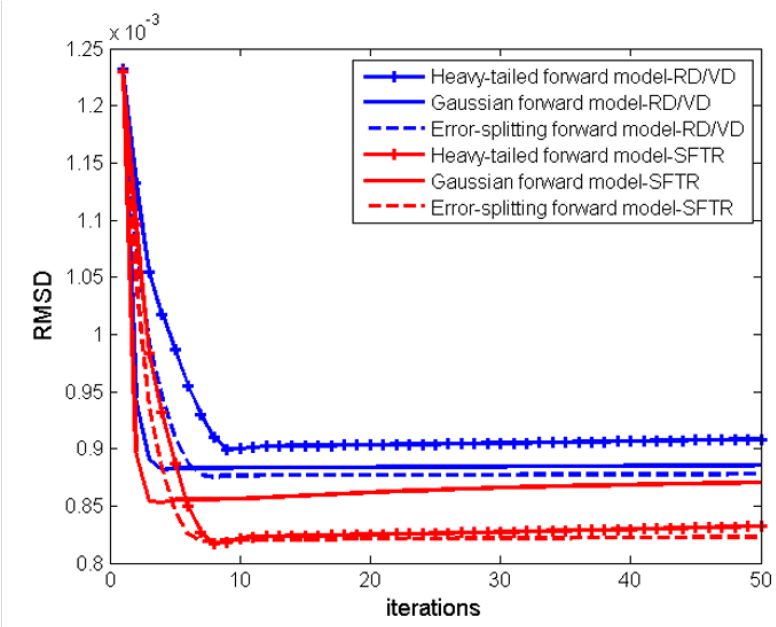


FIGURE 3.1 – Évolution en fonction des itérations du *Root Mean Square Difference* (RMSD) entre les volumes reconstruits par l'agorithme de Gauss-Markov-Potts à partir de 300 projections et une reconstruction de référence de l'IQI (voir Fig. 2.2) faite à partir de 2400 projections ; Différents modèles d'incertitudes (loi gaussienne, à queue lourde ou *error-splitting*) et paires de projection/rétroprojection (non duale $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ en bleu et à empreinte séparable SFTR en rouge) sont ainsi comparés ; *source : [Chapdelaine 2019].*

3.1.3 Accélération sur GPU

Nos premiers travaux sur l'accélération des opérateurs de projection et rétroprojection ont porté sur la paire non duale $\mathcal{P}_{RD}/\mathcal{R}_{VDL}$ car offrant le meilleur compromis entre qualité et vitesse d'exécution. Motivés par les bénéfices théoriques d'une paire duale, nous avons depuis proposé une nouvelle parallélisation sur GPU de la paire $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ et comparé ses performances avec celles des autres paires de projection/rétroprojection.

Paire non duale $\mathcal{P}_{RD}/\mathcal{R}_{VDL}$

Avant l'émergence de CUDA, les implémentations pour la reconstruction tomographique étaient soumises au cadre rigide imposé par le pipeline graphique conçu pour le rendu 3D. Le principe de l'utilisation des textures [Cabral 1994] a été étendu par F. Xu et K. Mueller [Xu 2007] pour la projection et la rétroprojection 3D en tomographie CT. Mais c'est bel et bien l'apparition de CUDA qui a révolutionné l'accélération de la reconstruction tomographique.

Dans un premier temps, les efforts de parallélisation sur GPU se sont focalisés sur le rétroprojecteur afin d'accélérer les méthodes de reconstruction par rétroprojection filtrée [Scherl 2007, Rohkohl 2009, Zinsser 2013]. L'algorithme $\mathcal{R}_{VDL}$ a été plébiscité car exprimant un fort potentiel de parallélisation, le calcul de chaque voxel étant indépendant. Par ailleurs, l'interpolation bilinéaire requise lors de l'accès aux données de projection est avantageusement accélérée grâce aux textures 2D des GPUs : les calculs d'interpolation sont effectués de manière matérielle et la latence des accès mémoire réduite grâce à un cache 2D.

Dans un second temps, afin de pouvoir accélérer les méthodes itératives, le projecteur a été à son tour accéléré avec une littérature très riche à ce sujet. L'algorithme de Siddon [Siddon 1985] a été parmi les premiers à être parallélisé bénéficiant notamment des optimisations d'implémentation déjà existantes sur CPU [Jacobs 1998, Han 1999] basées sur un calcul incrémental des paramètres de la droite des rayons traversant le volume comme par exemple dans les implémentations de [Xu 2010a] et [Nguyen 2015]. D'autres travaux se sont concentrés sur la réduction des branchements conditionnels existants dans l'algorithme originel de Siddon et responsables de divergence de *threads* : [Gao 2012] parcourt des rayons selon l'axe dit « majeur » afin que lors de la traversée d'une colonne de voxels, seulement deux intersections au maximum soient à calculer ; [Thompson 2014] améliore cette technique en factorisant certains calculs indépendants de la coordonnée axiale ; [Xiao 2012] convertit les instructions conditionnelles en opérations arithmétiques et logiques simples. Mais parmi les projecteurs *ray-driven*, les projecteurs basés sur l'interpolation du volume sont ceux qui s'adaptent le mieux à l'architecture des GPUs avec un emploi efficace des textures 2D ou 3D. Ainsi dans les parallélisations GPU de l'algorithme $\mathcal{P}_{Joseph}$ [Joseph 1982], les rayons traversent le volume via une boucle régulière selon l'axe « majeur » et échantillonnent le volume en accédant efficacement

à une texture 3D ou à une collection de textures 2D [Knaup 2008, Dittmann 2017]. L'échantillonnage du volume peut également se faire le long du rayon avec un pas régulier $\mathcal{P}_{Regular}$ (voir 1.1). Afin de rendre plus régulier les calculs et synchroniser finement les différents *threads* d'un bloc calculant mutuellement plusieurs rayons adjacents, [Chou 2011b] prend le même nombre d'échantillons du volume le long du rayon ; [Xie 2015] définit de même son projecteur *Fixed Sampling Number Projection* (FSNP) associé à un rétroprojecteur $\mathcal{R}_{VD}$. Si les performances de ces deux dernières stratégies sont intéressantes, la fréquence d'échantillonage du volume variera avec la longueur de l'intersection du rayon avec le champs de vue ; les segments « courts » auront un pas d'échantillonnage plus petit que les segments « longs » .

Mes premiers travaux au L2S se sont focalisés sur la parallélisation sur GPU d'une paire non duale en géométrie conique. Au GPI, la projection et la rétroprojection se faisaient avant cela via une multiplication de matrice, approche non tenable pour des volumes de plus de 256^3 voxels. Un rétroprojecteur $\mathcal{R}_{VDL}$ inspiré de mes travaux de thèse en Tomographie TEP à géométrie parallèle [R_{2009}], a été ainsi parallélisé avec la recherche de performances optimisées. Pour le projecteur, des différents algorithmes évalués, nous avons opté pour $\mathcal{P}_{Regular}$ qui échantillonne, grâce à la texture 3D des GPUs, le volume le long du rayon avec un pas constant (voir annexe 1.1). Cette paire $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ a ainsi permis une accélération d'algorithmes bayésiens de reconstruction 3D sur données réelles de grande taille [C_{2010}]. Par la suite, de nouveaux développements ont abouti au dépôt du logiciel TomoGPI [L_{2014}] (voir annexe C). Les performances de cette paire $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ comparées récemment avec ceux des *toolkits* Astra [Ast 2012, Palenstijn 2017] et RTK [Rit 2014] sont exposées dans le Tab. 3.1. Elles démontrent l'efficacité de nos implémentations tout à fait au niveau de l'état de l'art dans un domaine très compétitif.

	TomoGPI (s)	Astra (s)	RTK (s)
$\mathcal{P}_{Regular}$	5.34	6.74 ($\times$ 1.3)	41.6 ($\times$ 7.8)
$\mathcal{R}_{VDL}$	2.60	5.49 ($\times$ 2.1)	5.63 ($\times$ 2.2)

TABLE 3.1 – Temps d'exécution mesurés sur $\texttt{GPU}_{volta-Titan}$ pour les *kernels* GPU (sans transfert mémoire CPU-GPU) des projecteurs *ray-driven* $\mathcal{P}_{Regular}$ et rétroprojecteurs *voxel-driven* $\mathcal{R}_{VD}$ de TomoGPI comparés à ceux d'Astra et de RTK ; jeu de données $[1024 \times 1024^2; 1024^3]$; Pour plus de détails, se référer au rapport de stage V. Samy (2018).

Afin d'obtenir un facteur supplémentaire d'accélération, nous avons utilisé le format de stockage des réels en demi-précision (*half*) sur 16 bits au lieu des classiques 32 bits en simple précision (*single*) ou 64 bits en double précision. En effet, depuis 2015 avec CUDA 7.5, Nvidia autorise le stockage des données dans ce format qui permet de réduire ainsi par deux l'empreinte mémoire par rapport au format *single* utilisé plus

haut. Dans les *kernels* de calcul , les données à la lecture sont « décompressés » au format *single*, les calculs effectués en *single* et les données à l'écriture « compressés » au format *half*. Comme illustré Tab. 3.2, nous avons obtenu ainsi une accélération d'un facteur d'accélération 1.29 et 1.26 respectivement pour $\mathcal{P}_{Regular}$ et $\mathcal{R}_{VDL}$. A noter que les temps présentés en Tab. 3.2 incluent le temps de transfert de données dont nous pouvons faire abstraction ici car ils sont entièrement masqués grâce aux *streams* présentés en 4.2.2. Les accélérations obtenues sont donc dûs uniquement à l'accélération des *kernels* de calcul. Elles s'expliquent par un accès plus rapide à la mémoire globale par les coeurs de calcul. En effet, les données accédées sont deux fois moins volumineuses ; il y a donc moins de transactions mémoire et les caches mémoires peuvent contenir deux fois plus de données.

	single float	*half float*
$\mathcal{P}_{Regular}$	11.55	8.97 *(× 1.29)*
$\mathcal{R}_{VDL}$	5.36	4.25 *(× 1.26)*

TABLE 3.2 – Temps de traitement (transfert CPU-GPU inclus) sur $\texttt{GPU}_{maxwell}$ des opérateurs avec stockage des données en simple ou demi précision flottante ; jeu de données $[1024 \times 1024^2; 1024^3]$.

Paires duales

Des travaux s'attaquant aux handicaps exposés précédemment des projecteurs *voxel-driven* $\mathcal{P}_{VDL}$ [Du 2017] ou des rétroprojecteurs *ray-driven* $\mathcal{R}_{Siddon}$ [Gao 2012, Thompson 2014, Park 2014, Nguyen 2015] permettent d'avoir des paires duales *ray-driven* ou *voxel-driven* avec une accélération satisfaisante sur GPU. Mais, les paires *distance-driven* ou *separable footprint* également duales avec en prime un modèle plus précis peuvent bénéficier d'une accélération efficace sur GPU. S'inspirant de la version sans branchement conditionnel de la paire *distance-driven* de [Basu 2006], différentes implémentations sur GPU ont été proposées [Schlifske 2016, Liu 2017] ; les intégrales du volume $\boldsymbol{f}$ ou des données $\boldsymbol{g}$ sur le plan commun sont préalablement calculées pour ensuite être accédées dans la mémoire de texture des GPUs ; le bénéfice du cache 2D et de l'interpolateur bilinéaire câblé en matériel de ces textures rend ces implémentations très efficaces.

Parmi toutes les paires duales existantes, nous avons préféré la paire $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ car la jugeant plus précise. L'équipe de J. Fessler a proposé une première parallélisation de cette paire [Wu 2011], améliorée par la suite [Liu 2017] avec une accélération supplémentaire d'un facteur 2. Ces deux implémentations possèdent une boucle en `phi` d'appel de trois *kernels* avec des données intermédiaires de la taille du volume $\boldsymbol{f}$ ou des données de projection $\boldsymbol{g}$. Si la taille de $\boldsymbol{f}$ et $\boldsymbol{g}$ dépassent la capacité de la mémoire des cartes GPUs, ces implémentations requièrent à chaque itération `phi` le transfert de ces données de grande taille. Pour s'affranchir de cette limitation due à une empreinte mémoire trop importante, nous avons proposé une implémen-

tation avec un seul *kernel*. Dans la première version du projecteur [C$_{2018a}$], les axes majeurs, x ou y en fonction de l'angle `phi`, définissent chacun un *kernel*. Dans une deuxième version [C$_{2019a}$], ces deux kernels ont été fusionnés en définissant une coordonnée commune de parcours du rayon, technique inspirée de [Dittmann 2017]. Par ailleurs, la deuxième version a été optimisée notamment en factorisant les calculs de l'empreinte transverse F_{trans} avec des *threads* projettant sur une colonne axiale de détecteurs, ou rétroprojettant une colonne axiale de voxels. La mémoire locale (*shared memory*) a été utilisée afin que le nombre de registres par *threads* ne pénalisent pas les performances. Les performances obtenues (voir Tab. 3.3) sont comparables à celle de la première implémentation de l'équipe de J. Fessler [Wu 2011] mais moins bonnes que leur deuxième version [Liu 2017]. Notre implémentation lève toutefois le verrou de la boucle en `phi` des *kernels* et a ainsi rendu possible sans difficulté sa parallélisation multi-GPU [C$_{2019a}$] avec la même stratégie que celle mise en place pour notre paire non duale.

	TomoBayes (s)	[Wu 2011] (s)		[Liu 2017] (s)	
$\mathcal{P}_{SF}$	11.91	12.0	($\times$ 1.15)	7.2	($\div$ 1.46)
$\mathcal{R}_{SF}$	6.71	7.7	($\times$ 1.00)	4.6	($\div$ 1.65)

TABLE 3.3 – Temps de reconstruction sur $\texttt{GPU}_{kepler}$ mesuré de notre implémentation GPU de la paire $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ de TomoBayes, et comparé à ceux issus des publications de l'équipe de J. Fessler ; jeu de données $[984 \times 88 * 64; 512^2 * 64]$.

Performances comparées entre paires

Afin de comparer les performances entre les différentes paires, la métrique *Giga Updates Per Second* (**GUPS**) est couramment utilisée [Chou 2011a]. Elle correspond au nombre d'accumulations effectuées par seconde lors de la mise à jour des projections ou des voxels ; pour le projecteur $\mathcal{P}$ et le rétroprojecteur $\mathcal{R}$, elle se calcule de la manière suivante :

$$\begin{cases} GUPS_{\mathcal{P}} &= \dfrac{N_g * N_{\text{voxels traversés}}}{1024^3 * \text{Temps}} \\ GUPS_{\mathcal{R}} &= \dfrac{N_f * N_{\texttt{phi}}}{1024^3 * \text{Temps}}, \end{cases} \tag{3.6}$$

avec $N_{\text{voxels traversés}}$, le nombre de voxels traversés par le rayon qui est pris égal de manière approchée à la dimension transverse du volume $N_{\texttt{xn}}$, et Temps, le temps de traitement en seconde. Par ailleurs, nous avons mesuré pour le projecteur et le rétroprojecteur le nombre d'échantillons $\mathbf{N_{ech}}$ accédés respectivement dans le volume $\boldsymbol{f}$ et dans les données de projection $\boldsymbol{g}$. La vitesse $\mathbf{V_{ech}}$ indiquant le nombre d'échantillons accédés par seconde permet ainsi de relativiser la métrique **GPUS** en prenant en compte la précision de l'opérateur. Par exemple, $\mathcal{R}_{VDL}$ accède pour chaque voxel mis à jour à quatre échantillons lors de l'interpolation bilinéaire alors

que $\mathcal{R}_{SF}$ accède en moyenne à 8.86 échantillons, correspondant typiquement à une empreinte trapézoïdale transverse de largeur 4 et à une empreinte rectangulaire axiale de largeur comprise entre 2 et 3. A l'aide des outils de profilage de Nvidia, nous avons mesuré la complexité des différents algorithmes : **TFLOP** correspond au nombre de téra opérations flottantes exécutées ; **FLOP/texel** exprime ce nombre d'opérations par éléments de texture (*texel*) accédés et donne ainsi une idée de l'intensité arithmétique de chaque algorithme ; **Eff.** indique la puissance de calcul effectivement utilisée en pourcentage par rapport à la puissance crête théorique du GPU.

Opérateur	Temps (s)	**GPUS**	$\mathbf{N_{ech}}$ (10^{12})	$\mathbf{V_{ech}}$ (10^{12} ech./s)	**TFLOP**	**FLOP /texel**	**Eff.** (%)
$\mathcal{P}_{Regular}$	7.43	232.4	9.48	1.28	9.65	8.14	9.9
$\mathcal{P}_{SF}$	61.9	27.9	6.22	0.10	102.7	16.5	12.1
$\mathcal{P}_{Joseph/tex3D}$	5.14	336.2	3.96	0.77	3.47	3.51	4.9
$\mathcal{P}_{Joseph/tex2D}$	3.42	505.3	3.96	1.16	2.48	2.51	5.3
$\mathcal{R}_{VDL}$	1.37	752.9	4.40	3.23	1.72	1.57	9.2
$\mathcal{R}_{SF}$	33.8	30.3	9.74	0.29	145.2	14.9	31.3

TABLE 3.4 – Efficacités des opérateurs mesurés sur $\texttt{GPU}_{volta-Titan}$; jeu de données $[1024 \times 1536 * 1152; 1024^3]$.

Du Tab. 3.4, nous pouvons faire les observations suivantes :

- Pour toutes les paires, l'opérateur de rétroprojection est beaucoup plus rapide que celui de projection. L'écart est très marqué pour la paire non duale où le $P_{Regular}$ demande 5.5 fois plus de temps que $\mathcal{R}_{VDL}$ pour cette configuration des données. Cela provient du coût calculatoire beaucoup plus grand du projecteur avec 9.65 TFLOP contre 1.72 TFLOP pour le rétroprojecteur. Le calcul des équations des droites des différents rayons est plus lourd que celui seulement des coordonnées de projection `un`, `vn` pour le rétroprojecteur.
- Le projecteur de Joseph est plus rapide que $\mathcal{P}_{Regular}$ pour ses deux versions $\mathcal{P}_{Joseph/tex3d}$ et $\mathcal{P}_{Joseph/tex2d}$ faisant appel respectivement à des textures 3D ou 2D pour l'accès au volume. Cette différence s'explique tout simplement par le plus faible nombre d'échantillons pris le long du rayon par ce projecteur. En vitesse relative V_{ech}, $\mathcal{P}_{Regular}$ est le plus rapide.
- La paire *separable footprint* est bien plus lente que la paire non duale ; un facteur de 8.3 est observé pour le projecteur et près de 25 pour le rétroprojecteur. En vitesse relative V_{ech}, l'écart est d'un ordre de magnitude entre projecteurs ($\mathcal{P}_{Regular}/\mathcal{P}_{SF}$) et rétroprojecteurs ($\mathcal{R}_{VDL}/\mathcal{R}_{SF}$). Une des raisons à cela est que la paire SF est beaucoup plus calculatoire ; les FLOP/texel sont par exemple de 14.9 pour son rétroprojecteur soit près de dix fois plus que $\mathcal{R}_{VDL}$.

Nous comparons dans le Tab. 3.5 les performances de nos implémentations des paires

non duale $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ et duale $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ avec celle de la paire $\mathcal{P}_{DD}/\mathcal{R}_{DD}$ par [Liu 2017]. La paire *distance-driven* a des performances intermédiaires entre les deux autres. Elle reste beaucoup plus lente que la paire non duale : $\div 21.4$ pour la rétroprojection et $\div 2.3$ pour la projection ; mais elle est près de trois fois plus rapide que la paire *separable footprint*. Cela s'explique en partie par son modèle un peu plus simple. Si en axial, les deux modèles ont une empreinte quasi équivalente rectangulaire, en transverse le modèle DD a une empreinte équivalente à un rectangle, alors que le modèle SF a une empreinte plus précise en forme de trapèze. Mais cela n'explique pas l'importance de cet écart en performance. Une raison supplémentaire est sa meilleure adéquation avec l'architecture GPU grâce notamment à la version sans branchement conditionnelle [Basu 2006]. La paire SF reste quant à elle beaucoup plus calculatoire comme nous avons pu le constater dans le Tab. 3.4.

	distance-driven $\mathcal{P}_{DD}/\mathcal{R}_{DD}$	*separable footprint* $\mathcal{P}_{SF}/\mathcal{R}_{SF}$	*unmatched* $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$
Projecteur	17.65	6.2 ($\div 2.8$)	41.07 ($\times$ 2.3)
Rétroprojecteur	17.93	7.93 ($\div 2.3$)	384.4 ($\times$ 21.4)

TABLE 3.5 – Performances (GUPS) sur $\texttt{GPU}_{maxwell}$ issu des publications de l'équipe de B. De Man [Liu 2017] pour la paire *distance-driven*, et comparées à celles mesurées de nos implémentations des paires *unmatched* et *separable footprint* ; jeu de données $[1024 \times 1024^2; 1024^3]$ pour $\mathcal{P}_{DD}/\mathcal{R}_{DD}$, et $[1024 \times 1536 * 1280; 1024^2 * 640]$ pour $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ et $\mathcal{P}_{SF}/\mathcal{R}_{SF}$.

3.1.4 Accélération sur FPGA

Suite à l'émergence des GPUs, un faible nombre d'architectures sur FPGA pour la reconstruction tomographique ont été proposées. Ces architectures ont été conçues en VHDL ou Verilog pour le rétroprojecteur $\mathcal{R}_{VD}$ [Florian 2011, Leeser 2002, Gac 2008] ou la paire complète *separable footprint* $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ [Kim 2012]. Malheureusement, les performances sur FPGA sont de manière générale moins bonnes que celles sur GPUs malgré les efforts de développement plus importants que requiert cette conception d'une architecture sur mesure. L'apparition dans les années 2010 d'outils de synthèse de haut niveau plus matures facilitant la conception des architectures sur FPGA ont donné un second souffle à la reconstruction tomographique accélérée sur FPGA : [Xu 2010b] utilise l'HDL Impulse C pour concevoir un rétroprojecteur ; [Chen 2012] obtient une performance de 1.08 GUPS pour $\mathcal{P}_{Siddon}$ et 0.23 GUPS pour $\mathcal{R}_{Siddon}$ sur des Virtex VI (Xilinx) avec l'outil AutoESL HLS ; [Choi 2016] et [Wen 2020] utilisent Vivado HLS pour l'implémentation d'algorithmes itératifs avec une paire de Siddon accélérée sur des FPGAs de Xilinx. Ces deux derniers travaux obtiennent respectivement une vitesse de reconstruction de 20 GUPS avec 4 Virtex VI et 30 GUPS avec une Zynq UltraScale+ pour leurs deux opérateurs $\mathcal{P}_{Siddon}$ et $\mathcal{R}_{Siddon}$. Une utilisation minutieuse de la mémoire BRAM sur puce des FPGAs maximisant la localité des données est mise en place grâce à des études *offline* des accès mémoire.

[Choi 2016] qualifie son approche de *ray-driven voxel-tile* : les pipelines ont chacun la charge des calculs d'un bloc ou tuile de voxels pour une collection de rayons adjacents traités les uns après les autres. Ces bonnes performances, comparées à celles obtenues sur GPU (voir Tab. 3.4) permettent ainsi au FPGA de redevenir compétitif.

La thèse de Maxime Martelli [Martelli 2019] en collaboration avec Thalès avait pour objectif d'explorer le potentiel des nouveaux outils HLS pour l'accélération de la simulation radar. Conjointement à ce contexte industriel, nous avons évalué ces outils pour le domaine de la tomographie comme cas d'étude pour la méthodologie de conception sur outils HLS, ou plus exactement VHLS (voir 1.1.2), proposée dans le cadre de cette thèse. Nous nous sommes focalisés sur l'algorithme de rétroprojection $\mathcal{R}_{VD}$. Les premiers résultats obtenus sur une carte bas de gamme $\texttt{FPGA}_{\texttt{Cyclone-V}}$ ont valu un prix au concours Intel HPC à l'équipe d'étudiants que j'ai encadrée (voir section 3 du chapitre activité scientifique) . Ils ont ensuite été extrapolés sur la gamme des Arria 10 et publiés dès la première année de cette thèse [R_{2018a}]. Depuis, dans le cadre de la thèse de Daouda Diakite démarrée en 2019, nous avons pu évaluer les performances de notre architecture sur les cartes $\texttt{FPGA}_{Arria-10}$ [C_{2020a}] et plus récemment sur $\texttt{FPGA}_{Stratix-10}$. Dans le Tab. 3.6, nous les comparons à celles obtenues sur CPU et GPU. Si les performances sur FPGA sont supérieures à celle obtenues sur un CPU ($\times 40$), elles restent inférieures à celles des GPUs ($\div 116$). La technologie FPGA reste handicapée par une plus faible fréquence ($\div 8$) et une plus faible densité ($\div 112$) de *Processing Elements* (PE). Cependant, en terme d'efficacité de calcul mesurée en cycles par opérations élémentaires (i.e mises à jour d'un voxel) pour chaque PE, le FPGA est plus efficace ($\times 7.72$) : 2.23 contre 17.3 cycles/op/PE pour le GPU.

Accélérateur*	GFLOPS	Temps (ms)	GUPS	N_{PE}	Freq (Mhz)	Efficacité (Cycles/ op/PE)	Puissance (W)
$\texttt{CPU}_{xeon-tomo}$	500	66 000	0.06	6	2 900	267.4	47
$\texttt{GPU}_{pascal-1080Ti}$	10 615	14	285.7	3 584	1 481	17.3	237
$\texttt{GPU}_{pascal-TX2}$	665	253	15.8	256	1 300	19.6	12.9
$\texttt{FPGA}_{Stratix-10}$	9 200	1 620	2.47	32	185	2.23	
$\texttt{FPGA}_{Arria-10}$	1 366	5 340	0.75	32	150	5.98	9.9
$\texttt{FPGA}_{Cyclone-V}$	34	16 900	0.24				

TABLE 3.6 – Temps d'exécution de la rétroprojection 3D $\mathcal{R}_{VD}$ sur CPU/GPU/FPGA (avec interpolation au plus proche voisin sur CPU et FPGA) ; jeu de données $[256 \times 256^2; 256^3]$.

Les conclusions tirées des résultats de la Tab. 3.6 font écho à celles de mes travaux de thèse [R_{2009}] où une architecture décrite en langage matériel (VHDL) faisant appel à un cache mémoire 3D Adaptatif et Prédictif [Mancini 2004] avait été conçue pour

* Caractéristiques des processeurs et FPGAs utilisés détaillées en annexe B

la rétroprojection 3D (Tomographie TEP à géométrie parallèle) ; les performances brutes obtenues à l'époque sur une Virtex 2 Pro (0.03 GUPS) étaient de deux ordres de magnitude inférieures à celles du premier GPU avec CUDA (1.84 GUPS), cependant l'efficacité de calcul des pipelines atteignant 1.7 cycles/op/PE était proche de l'efficacité idéale de 1 cycle/op/PE. La grande différence, dix années plus tard, est que la conception a fait cette fois-ci appel aux outils de synthèse de haut niveau. Ces outils HLS se révèlent donc pertinents car permettent la conception d'une architecture avec la même efficacité pour un temps de développement drastiquement réduit par rapport au flot de conception classique. La méthodologie d'implémentation se rapproche en effet à celle employée pour le développement logiciel (voir thèse [Martelli 2019]).

3.2 Déconvolution

Après une brève description de l'opérateur de convolution $\mathcal{C}$ employé en déconvolution, nos contributions pour l'accélérer sur GPU en faisant appel au calcul et stockage de données avec la précision demi-flottante ou bien à l'aide des *tensor cores* seront exposées.

3.2.1 Opérateur de convolution $\mathcal{C}$

Dans le problème inverse de déconvolution, $\boldsymbol{H}$ et $\boldsymbol{H}^t$ correspondent à des convolutions notées respectivement $\mathcal{C}$ et $\mathcal{C}'$; leurs noyaux de convolution h et h' sont symétriques. Il n'y a donc pas de problématiques liées à la dualité comme c'est le cas en reconstruction tomographique. Par ailleurs, ces deux opérateurs enchaînés peuvent être fusionnés en une seule convolution $\mathcal{C}''$ de noyau $h'' = h * h'$. Quand il est calculé dans le domaine image, cette opération a des caractéristiques adaptés à l'architecture des GPUs : (i) massivement parallélisable, chaque pixel pouvant être traité indépendamment des autres, (ii) forte localité de ses accès mémoire dans la fenêtre du noyau de convolution, et (iii) opérations essentiellement constituées de multiplication-accumulations faisant ainsi tourner à plein régime les coeurs de calcul.

3.2.2 Calcul et stockage en demi-flottant

Afin d'accélérer la déconvolution opérée par les imageurs en radioastronomie, nous avons exploré le potentiel d'accélération du format flottant demi-précision (voir 3.1.3) pour le stockage de données mais également pour le calcul [C$_{2018b}$]. En effet, depuis l'architecture Pascal, les opérations en *half* sont autorisées et peuvent se faire à un débit deux fois plus important qu'en *single*.

Cette étude a tout d'abord fait le constat que les différentes librairies comme cuDNN ou CUBLAS redoutablement efficaces pour la convolution employée dans les réseaux

de neurones pour l'extraction de *features*, ne le sont pas lorsqu'une seule convolution sur une seule image est effectuée. Comme illustré dans la Fig. 3.2 (a), nos codes « naïfs » écrits en CUDA à l'instar des codes PRCF [Perrot 2016] ou ceux de la librairie ArrayFire [Yalamanchili 2015] se sont ainsi révélés plus efficaces que les librairies de Nvidia pour des noyaux de petites tailles. A partir d'une certaine taille du noyau (35 dans le cas étudié), le calcul dans l'espace de Fourier présente une complexité arithmétique moindre que le calcul direct dans le domaine de l'image ; la librairie cuFFT de Nvidia obtient alors naturellement de meilleures performances. Lorsque le calcul est effectué en demi-précision, l'accélération devient intéressante pour des noyaux de taille moyenne ($\times$1.5 pour 40) ou de grande taille ($\times$3.0 pour 100) mais au prix d'une perte assez importante en précision comme illustré Fig. 3.2 (b) et (c). Comme expérimenté en reconstruction tomographique (cf. section 3.1.3), une stratégie mixte stockant les données en demi-précision mais effectuant les calculs en simple précision dénommé *mixed* dans la Fig. 3.2 (b) et (c), s'avère un compromis intéressant avec une accélération effective à partir d'une taille de noyau supérieur à 50 ($\times$2 pour 100) pour une erreur de calcul de 10^{-4}.

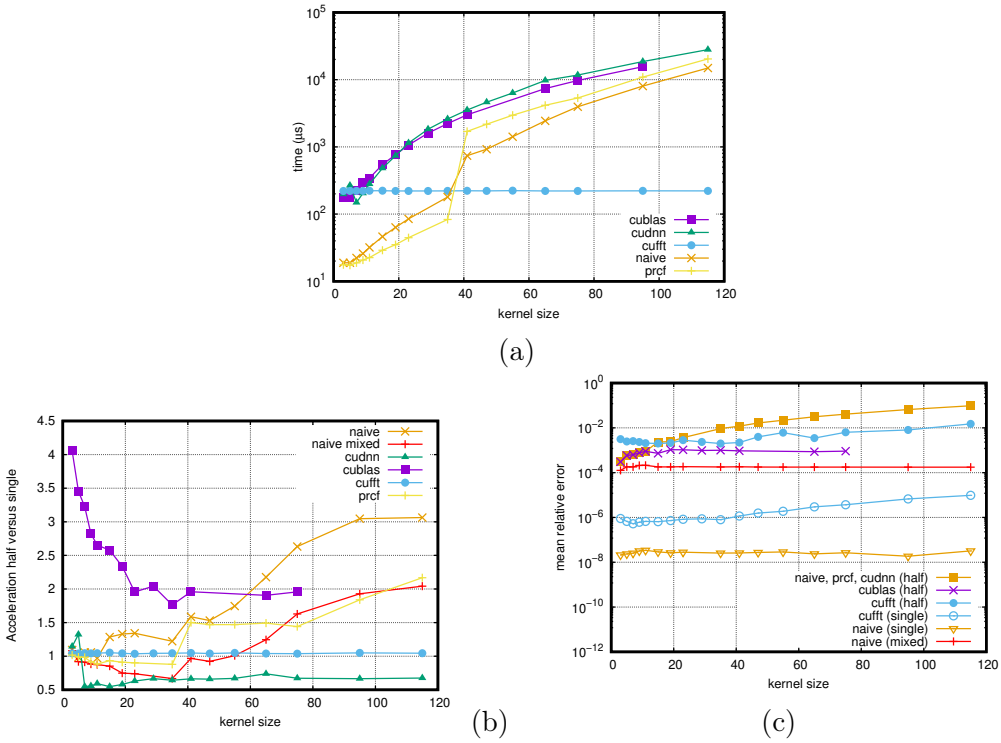


FIGURE 3.2 – Convolution 2D effectuée sur $\texttt{GPU}_{volta-Titan}$ sur des images de taille 512*152 avec différentes tailles de noyau en abscisse : (a) temps de calcul en flottant simple précision ; (b) accélération avec les calculs effectués en demi-précision par rapport aux calculs effectués en simple précision ; (c) erreur de calcul par rapport à une implémentation sur CPU en double précision.

Nous avons ensuite utilisé la demi-précision pour un algorithme de déconvolution par descente de gradient avec régularisation quadratique sur des données simulées pour un réseau d'antennes en radioastronomie ; la convolution a été faite dans ce cas par cuFFT car le noyau de convolution est de la taille de l'image. Nous tirons les mêmes conclusions que l'étude sur l'opérateur $\mathcal{C}$. Comme illustré Fig. 3.2.2, le calcul en demi-précision dégrade trop la précision des résultats notamment lors du calcul du pas d'optimal pour la descente du gradient, pour être utilisé aussi directement sans des précautions fines sur la dynamique des valeurs. La stratégie mixte offre encore une fois le meilleur compromis car ne dégrade pratiquement pas la convergence de l'algorithme tout en offrant un levier potentiel d'accélération.

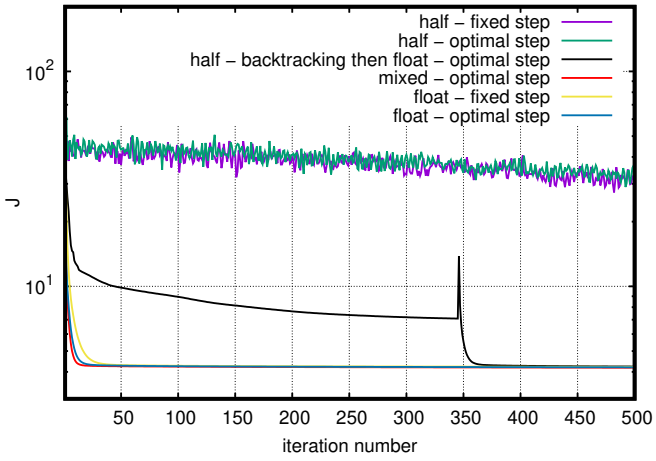


FIGURE 3.3 – Déconvolution pour un imageur en radioastronomie : convergence d'un algorithme de descente de gradient avec un pas fixe ou optimal pour différentes précisions de calcul sur $\texttt{GPU}_{volta-Titan}$.

3.2.3 Utilisation des *tensor cores*

Les *tensor cores* disponibles depuis l'architecture Volta des GPUs de Nvidia, sont des ressources de calcul additionnelles destinées à l'accélération de la multiplication et accumulation de matrices : $D = AB + C$. Ils utilisent une précision mixte avec une multiplication faite sur 16 bits et une accumulation sur 16 ou 32 bits. Ils ont été notamment conçus afin d'accélérer les réseaux de neurones convolutifs. ; toutes les convolutions 2D permettant d'extraire des *features* sur les très grandes bases d'images peuvent en effet être exprimées sous forme de multiplication de matrices [Chetlur 2014, Anderson 2017]. Cependant, cette approche est adaptée seulement pour une série de filtres 2D appliquée sur une collection d'images, et s'avère inefficace pour une convolution appliquée sur une seule image (voir Fig. 3.2) comme c'est le cas dans les problèmes de déconvolution.

Dans le cadre des travaux de thèse de Mickael Seznec, un algorithme *im2tensor* exprimant la convolution comme une multiplication de matrices est proposé afin de bénéficier ainsi du potentiel d'accélération des *tensor cores*. Comme illustré sur la Fig. 3.4 (a), son principe général est de multiplier une matrice K^T formée des coefficients du filtre avec un tensor 3D S formé de recopies de l'image 2D I, puis d'effectuer une opération de réduction sommation sur les diagonales du tensor 3D résultant P. Les premiers résultats de cet algorithme ont été exposés en conférence [C$_{2020d}$] ; cet algorithme *im2tensor* sera présenté de manière plus approfondie dans un futur article en ce urs de rédaction. Comme illustré sur la Fig. 3.4 (b), l'accélération obtenue par rapport aux approches classiques faisant appel à un code CUDA « naïf » , à la librairie cuFFT pour un calcul dans Fourier ou à la librairie cuDNN, est effective pour des noyaux de tailles moyennes (entre 40 et 50 pour cette taille d'image).

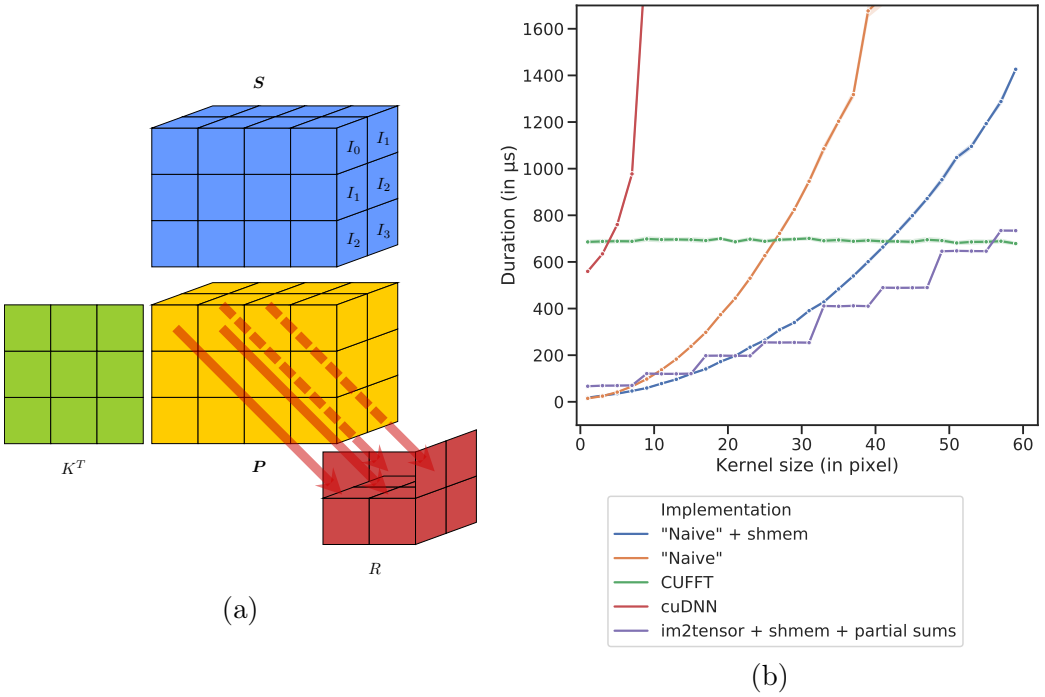


FIGURE 3.4 – Algorithme *im2tensor* : (a) principe général et (b) accélération sur $\texttt{GPU}_{volta-Xavier}$

3.3 Conclusion

L'accélération sur cibles GPU ou FPGA des opérateurs utilisés dans les algorithmes itératifs est un champ de recherche toujours très actif qui ne peut se contenter de l'appel à des librairies génériques.

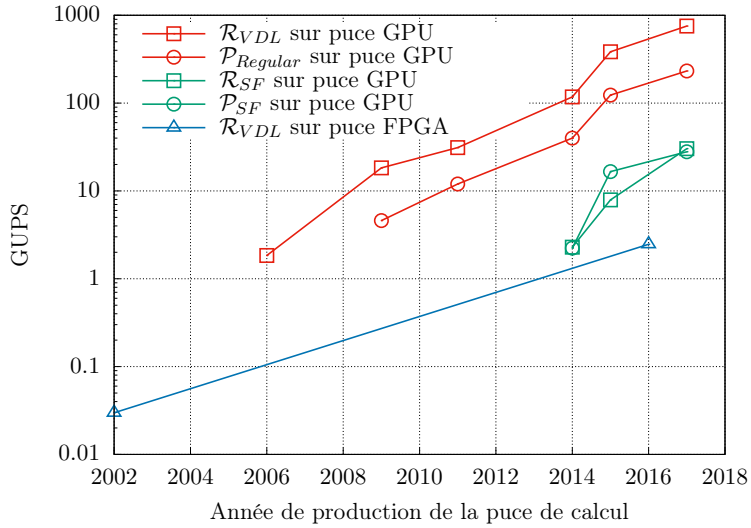


FIGURE 3.5 – Évolution de la vitesse de calcul mesurée en GUPS pour la paire duale $\mathcal{P}_{SF}/\mathcal{R}_{SF}$ (en vert) et la paire non duale $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ (en rouge) sur GPU et pour $\mathcal{R}_{VDL}$ sur FPGA (en bleu) ; les mesures ont été effectuées sur les puces suivantes : $\texttt{GPU}_{testla-G80}$, $\texttt{GPU}_{tesla}$, $\texttt{GPU}_{fermi}$, $\texttt{GPU}_{kepler}$, $\texttt{GPU}_{maxwell}$, $\texttt{GPU}_{volta-Titan}$, $\texttt{FPGA}_{Virtex-2-pro}$ et $\texttt{FPGA}_{Stratix-10}$.

Pour les opérateurs très spécifiques de projection $\mathcal{P}$ et rétroprojection $\mathcal{R}$ en reconstruction tomographique, la définition des modèles de l'instrument $\boldsymbol{H}$ est intimement liée à l'architecture de calcul. Nous avons ainsi mené nos travaux de recherche dans l'optique de trouver le meilleur compromis entre qualité théorique de l'algorithme et efficacité sur la cible technologique. Nous avons mis en avant l'importance de la dualité de la projection/rétroprojection qui ne doit pas systématiquement être « sacrifiée » afin d'obtenir une adéquation optimale à l'architecture des GPUs. La sous optimalité et la non garantie de convergence qu'elle implique, a ainsi fait émerger l'emploi plus large de paires duales aux modèles plus précis. Comme illustré sur la Fig. 3.3 comparant les performances d'une paire duale ($\mathcal{P}_{SF}/\mathcal{R}_{SF}$) et d'une paire non duale ($\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$), un ordre de magnitude sépare encore ces deux paires. Malgré ce retard, les paires duales bénéficient tout autant de la constante augmentation de la puissance de calcul des GPUs ; ainsi avec un gain d'un ordre de magnitude observé lors de ces cinq dernières années, leur coût calculatoire devient de moins en moins handicapant. Le contexte d'utilisation des algorithmes itératifs orientera donc le choix entre ces deux paires ; en routine clinique ou sur les lignes de production où le CND est employé, les paires non duales rapides pourront être utilisées pour satisfaire les fortes contraintes de temps de traitement ; à l'inverse pour le développement de méthodes de reconstruction dans les laboratoires de recherche, la paire duale aux propriétés théoriques solides sera à privilégier. A noter qu'une

stratégie mixte peut être mise en place faisant appel lors des premières itérations à une paire non duale pour une descente de gradient rapide, puis à une paire duale afin de garantir la convergence finale.

En reconstruction tomographique, nous avons également exploré le potentiel d'accélération des FPGAs pour lesquels les outils de synthèse de haut niveau ont donné une seconde chance. En effet, le choix de la cible s'est très rapidement restreint dans ce domaine aux seuls GPUs fabriqués par Nvidia. Les performances que nous avons obtenues jusqu'à présent sont pour le rétroprojecteur de deux ordres de magnitude inférieures à celle obtenues sur GPU. Toutefois, cet écart est le même que celui observé 12 ans plutôt lors de mes travaux de thèse avec une architecture sur mesure conçue avec un flot de conception classique. En ce sens, la technologie FPGA a progressé entre ces deux études en réduisant le temps d'implémentation. Grâce à un flot de conception se rapprochant de plus en plus d'un flot de développement logiciel, cette technologie est en bonne voie d'être adoptée par un public plus large que les seuls numériciens.

Même pour un opérateur générique comme la convolution, le développement de codes « experts » reste une approche intéressante dans certains cas spécifiques d'utilisation. Les librairies de Nvidia ne proposent en effet une accélération proche de l'optimal que pour les contextes applicatifs les plus répandus comme notamment l'apprentissage par réseaux de neurones. Cela laisse donc le champ à des propositions d'implémentations adaptées à des cas d'utilisation non couverts par ces librairies. Dans nos travaux actuels, nous nous intéressons ainsi à la convolution pour une seule image avec des noyaux de tailles moyennes, typiquement employée en déconvolution pour la radioastronomie. Dans ce contexte, nous explorons l'utilisation sur GPU des *tensor cores* et des unités de calcul en demi-précision flottant. Ces ressources de calcul plus « exotiques » que ses classiques coeurs en simple ou double précision flottante, nous ont permis d'avoir des premiers résultats d'accélération.

Chapitre 4

Calcul distribué de la boucle itérative

Sommaire

Dans le chapitre précédant, la parallélisation des opérateurs s'est faite sur des GPUs ou FPGAs qui ne sont en réalité que des co-processeurs pilotés par un hôte ; ce dernier centralise in fine les calculs et les données. A l'intérieur des nombreuses itérations des algorithmes d'inversion de données, cette centralisation peut être plus ou moins marquée. Dans ce chapitre, nous exposerons nos travaux pour l'accélération de toute la boucle itérative. Après une présentation de l'architecture matérielle des serveurs multi-GPU, des parallélisations mono et multi GPU seront proposées pour la reconstruction tomographique et la déconvolution appliquée à la radioastronomie.

4.1 Architecture matérielle des serveurs multi-GPU

Après avoir présenté brièvement les niveaux de parallélisation d'un *supercomputer* (*personnal* ou *HPC*), nous exposerons les technologies de bus de communication essentielles à une bonne performance sur ces serveurs. Leur bande passante limitée se révèle en effet un frein potentiel à l'accélération.

4.1.1 Niveaux de parallélisation

En calcul haute performance, trois niveaux de parallélisation sont employés afin d'accélérer une application sur un serveur de calcul multi-noeuds : (i) à gros grains sur

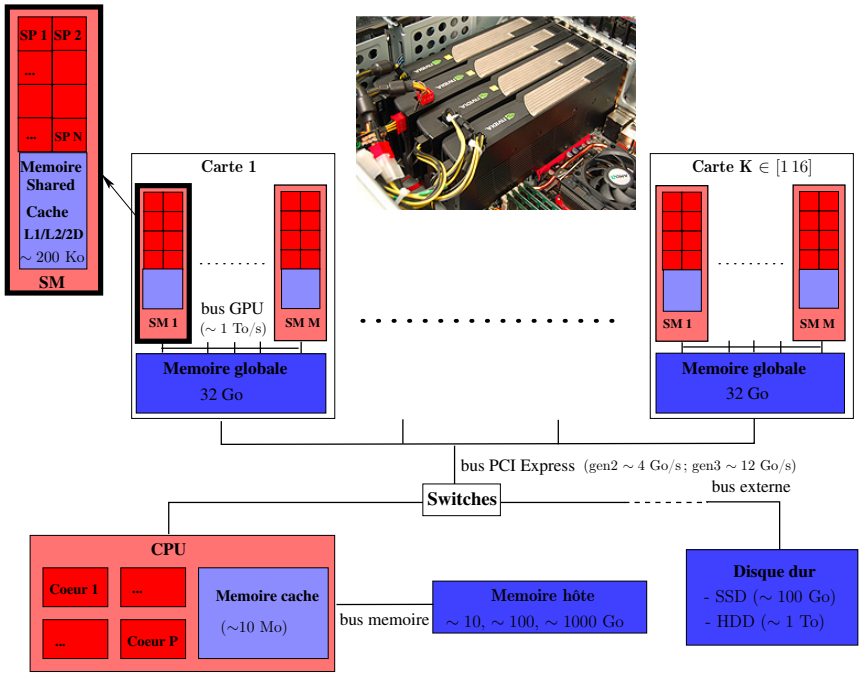


FIGURE 4.1 – Architecture d'un serveur de calcul multi-GPU avec K cartes.

les différents noeuds du serveur, (ii) à grains moyens sur les différentes cartes GPU de chaque noeud, et (iii) à grains fins sur les unités de calcul flottant de chaque puce accélératrice. Conjointement à une parallélisation à grains fins des opérateurs tels que $\mathcal{P}/\mathcal{R}$ et $\mathcal{C}$, nos travaux cherchent à obtenir grâce à une parallélisation à grains moyens une accélération supplémentaire idéalement proportionnelle aux nombres de cartes GPUs. La parallélisation à gros grains faisant appel à une programmation de type MPI, fera l'objet de travaux futurs dans un cadre collaboratif avec Atos-Bull (projet SKA).

Nous nous sommes focalisés sur les serveurs mono-noeud multi-GPU dont l'architecture est illustrée Fig. 4.1 et pour lesquels le nombre de cartes GPUs peut varier de 1 à 16. Nous avons principalement utilisé les serveurs acquis par le laboratoire L2S, $\mathtt{S}_{bebe}$ et $\mathtt{S}_{bebe2}$ possédant respectivement 8 et 10 GPUs (voir annexe B).

4.1.2 Bus de communication

Le ou les CPUs hôtes sont reliés aux cartes GPU via un bus à l'aide d'éventuelles commutateurs (*switches*). La technologie de bus la plus répandue est le PCI express qui offre un débit de l'ordre de 4 Go/s pour la génération 2 et de 12 Go/s pour la génération 3 (mesures effectuées sur nos serveurs $\mathtt{S}_{bebe}$ et $\mathtt{S}_{bebe2}$). Les différentes cartes GPU peuvent également communiquer entre elles via ces bus PCIe. Cependant ces bus se révélant un potentiel goulot d'étranglement, des bus aux débits plus

importants sont disponibles sur les serveurs HPC comme notamment la technologie de bus NVLink conçue par Nvidia ; sur l'architecture Ampère, un GPU peut ainsi gérer un canal de communication atteignant les 600 Go/s. Si ce bus a été conçu pour accélérer également les communications entre l'hôte et les cartes GPUs, en pratique celles-ci se font très majoritairement encore avec des bus PCIe ; en effet, à part les processeurs Power d'IBM, rares sont les processeurs compatibles avec le bus propriétaire NVLink.

4.2 Parallélisation mono-GPU de la boucle itérative

4.2.1 Architecture logicielle de la boucle itérative

Dans les implémentations des méthodes d'inversion de données, la boucle itérative principale est généralement exécutée sur l'hôte avec les calculs de $\boldsymbol{H}\boldsymbol{f}$ ou $\boldsymbol{H}^t\boldsymbol{\delta}_g$ déportés sur le co-processeur. Ce découpage logiciel a l'avantage de faciliter les développements des méthodes exprimées dans un langage de haut niveau comme Matlab avec les opérateurs accélérés appelés via une API. Cette stratégie a cependant deux désavantages, freins à l'accélération : (i) la centralisation des données sur l'hôte qui sera source de nombreux transferts de données entre l'hôte et les cartes GPUs, et (ii) les calculs de la boucle itérative exécutés sur l'hôte qui ne bénéficient pas de l'accélération offerte par les GPUs. Le recouvrement des temps de communication est une solution qui permet de minimiser l'impact des transferts mémoire tout en conservant une architecture logicielle centralisée sur l'hôte. Cependant, afin de se prémunir totalement de ces deux freins à l'accélération et tendre vers des performances optimales, la boucle itérative et les données seront amener à migrer pour résider entièrement sur les cartes GPUs.

4.2.2 Recouvrement des temps de transfert

Lorsque les données sont centralisées sur l'hôte, à chaque appel aux opérateurs, des transferts mémoires seront nécessaires pour que les cartes GPUs puissent récupérer les données d'entrée puis délivrer à l'hôte les résultats de ses calculs. Le Tab. 4.1 expose pour la reconstruction tomographique la proportion des temps de communication par rapport aux temps total de traitement pour les trois opérateurs $\mathcal{P}/\mathcal{R}/\mathcal{L}$ utilisés dans un algorithme $\texttt{DG}_{\texttt{regQ,opt}}$ de descente de gradient avec régularisation quadratique et calcul du pas d'optimal. Cette proportion de temps « perdu » est fonction de l'intensité arithmétique I_a (voir 1.2.2) de l'opérateur ; à noter que les accès mémoire pris en compte pour le calcul de cette I_a ne sont plus comme au chapitre 1 ceux en mémoire GPU mais ceux en mémoire CPU. Ainsi, si le projecteur $\mathcal{P}$ est relativement peu affecté (12%), les performances de l'opérateur laplacien $\mathcal{L}$ s'effondre avec 95% de temps consacré uniquement aux communications de données.

	calcul	transfert montant	transfert descendant
$\mathcal{P}_{Regular}$	88.0%	6.0%	6.0%
$\mathcal{R}_{VDL}$	71.1%	16.9%	12.1%
$\mathcal{L}$	5.0%	28.1%	66.9%

TABLE 4.1 – Distribution du temps de traitement pour les opérateurs $\mathcal{P}_{Regular}$, $\mathcal{R}_{VDL}$ et $\mathcal{L}$ mesurées sur $\texttt{GPU}_{maxwell}$; jeu de données $[1024 \times 1024^2; 1024^3]$.

Le traitement simultané de plusieurs flux de données, appelés *streams* dans la terminologie CUDA, permet de masquer partiellement ou totalement les temps de communication entre le CPU et le GPU. Les cartes GPU ont en effet la capacité de gérer en même temps que les calculs effectués par les *kernels*, deux transferts mémoire, un descendant (CPU vers GPU) et un montant (GPU vers CPU). Pour le rétroprojecteur $\mathcal{R}$ par exemple, nous avons ainsi découper le volume en sous-volumes chacun constitué de quelques coupes axiales, puis gérer la synchronisation de ces multiples flux comme illustré Fig. 4.2.

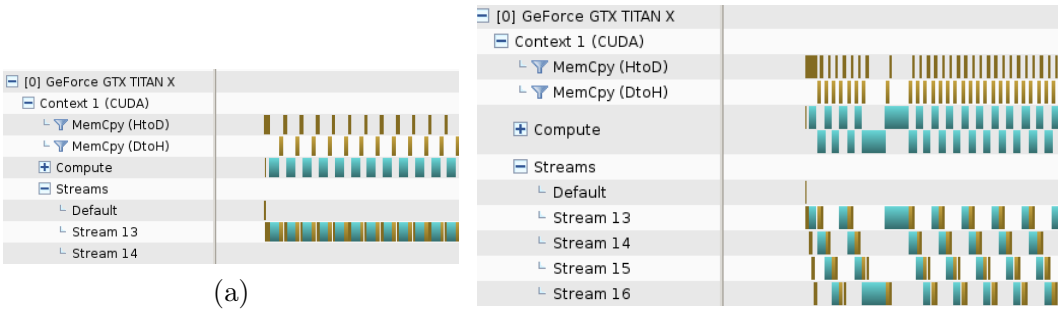


FIGURE 4.2 – Frise chronologique de l'exécution de la rétroprojection ; sans *streams* (a), les temps de calculs (en vert) sont entrecoupés de temps de transfert mémoire (en marron clair et foncé) *Host to Device* et *Device to Host* ; avec *streams* (b), les transferts mémoire sont masqués et le GPU calcule à plein régime ; exécution sur $\texttt{GPU}_{maxwell}$; jeu de données $[1024 \times 1024^2; 1024^3]$.

Avec ces *streams*, nous avons ainsi pu obtenir une accélération pour le projecteur $\mathcal{P}_{Regular}$ et le rétroprojecteur $\mathcal{R}_{VDL}$ respectivement de 1.25 et 1.41 (voir Tab. 4.2). Toutefois, nous pouvons constater qu'aucune accélération n'est obtenue pour le laplacien $\mathcal{L}$; les temps de transfert sont trop prépondérants (95%, cf. Tab. 4.1) pour permettre leur recouvrement par les temps de calcul. Par ailleurs, lors d'une parallélisation multi-GPU, les opérateurs $\mathcal{P}_{Regular}$ et $\mathcal{R}_{VDL}$ pourront se retrouver dans une telle situation ; la bande passante du bus étant partagée par les différentes cartes GPUs, les temps de transferts augmenteront et ne seront potentiellement plus recouvrables par les temps de calcul. Les *streams* apportent donc qu'une solution partielle à la suppression de ces temps « morts » de communication.

	sans stream (s)	avec streams (s)	
$\mathcal{P}_{Regular}$	14.41	11.55	*(× 1.25)*
$\mathcal{R}_{VDL}$	7.60	5.36	*(× 1.41)*
$\mathcal{L}$	3.06	3.07	*(× 1.00)*

TABLE 4.2 – Temps de traitement des opérateurs $\mathcal{P}_{Regular}$, $\mathcal{R}_{VDL}$ et $\mathcal{L}$ sur $\texttt{GPU}_{maxwell}$; jeu de données $[1024 \times 1024^2; 1024^3]$.

4.2.3 Boucle itérative résidant sur la carte GPU

Avec une boucle itérative résidant sur l'hôte, de nombreux calculs annexes de la boucle itérative comme ceux de la norme L2 ou de la différence entre données (i.e., $\boldsymbol{\delta}_g = \boldsymbol{g} - \hat{\boldsymbol{g}}$) ne sont pas déportés sur les GPUs ; d'une complexité calculatoire mineure par rapport aux opérateurs majeurs, leur accélération sur les GPUs n'est pas jugées prioritaires et ils restent ainsi exécutés sur le CPU du hôte. Toutefois, après accélération des opérateurs, ils peuvent prendre une part importante du temps de traitement global comme nous pouvons le constater Tab. 4.3 où ils représentent ainsi près d'un tiers du temps de calcul pour la version $\texttt{Mem}_{CPU}$.

	$\mathcal{P}_{Regular}(2)$ (%)	$\mathcal{R}_{VDL}$ (%)	$\mathcal{L}(3)$ (%)	Autres (%)	**Total** (s)	
$\texttt{Mem}_{CPU}$[1 GPU]	49.6	10.2	12.1	28.1	47.1	
$\texttt{Mem}_{GPU(half)}$[1 GPU]	78.3	18.3	2.2	1.2	21.9	*(× 2.15)*

TABLE 4.3 – Temps de traitement par itération des différents étapes de calcul de l'algorithme $\texttt{DG}_{\texttt{regQ,opt}}$ mesurés sur $\texttt{S}_{bebe}[\texttt{GPU}_{maxwell}]$ pour une boucle itérative résidant sur le hôte ou le GPU ; jeu de données $[1024 \times 1024^2; 1024^3]$.

Une méthode d'inversion résidant entièrement sur le co-processeur GPU est alors appropriée afin de lever ce frein. Lors de ce portage de toute la boucle itérative, certains algorithmes d'optimisation avec des vitesses théoriques de convergence supérieures pourront au final présenter une vitesse d'exécution moindre que des algorithmes aux calculs plus adaptés à une parallélisation GPU ; nous l'avons ainsi constaté pour une application de flot optique dense où une descente de gradient conjugué est pénalisée par les nombreux calculs de norme L2 requérant des opérations de réduction par sommation dont la parallélisation sur GPU n'est pas totalement massive [C$_{2020b}$].

En reconstruction tomographique, notre implémentation 100% GPU noté $\texttt{Mem}_{GPU}$ a permis de rééquilibrer les temps de calcul (voir Tab. 4.3). Les calculs « autres » ne représentent plus qu'un pourcent du temps total. Nous pouvons également remarquer que le temps de calcul du laplacien $\mathcal{L}$ a baissé passant de 30% à 2% du

temps global. En effet, les données résidant également sur la carte GPU, les temps de transferts que les *streams* ne permettaient pas de masquer ont disparus. Notons qu'afin de pouvoir faire résider toutes les données sur la mémoire de la carte GPU (limitée à 12 Go sur la carte $\texttt{GPU}_{maxwell}$), nous avons dû utiliser un stockage en demi-précision flottante (voir 3.1.3). Cela a permis ainsi de diviser par deux la taille des données. Mais cette technique a ses limites et lorsque des données de plus grandes tailles doivent être traitées, le stockage des volumes et projections ($\boldsymbol{f}^{(n)}$, $\boldsymbol{f}^{(n+1)}$, $\boldsymbol{g}$) et autres variables intermédiaires comme les paramètres ($\boldsymbol{\theta}_D$,$\boldsymbol{\theta}_f$) dans les méthodes bayésiennes hiérarchiques oblige à centraliser les données sur l'hôte ; la mémoire des cartes GPU est utilisée dans ce cas comme une simple mémoire tampon. Afin de nous abstraire de cette centralisation des données, nous avons chercher à distribuer l'ensemble des données sur les mémoires des K cartes qui offrent réunies un plus large espace mémoire. Par exemple, sur le serveur $\texttt{S}_{bebe2}[\texttt{GPU}_{maxwell}]$, nous avons ainsi $10 * 12 = 120$ Go à notre disposition.

4.3 Parallélisation multi-GPU de la boucle itérative

Dans cette section, nous proposons des parallélisations multi-GPU pour la reconstruction tomographique et la déconvolution appliquée à la radioastronomie. Nous nous focaliserons sur les dépendances de données intrinsèques à ces méthodes d'inversion. Elles ne permettent pas en effet une répartition triviale des données sur les mémoires des différentes cartes GPUs.

Dans la suite, nous utiliserons comme métrique d'efficacité de la parallélisation sur un serveur avec K cartes GPUs :

$$A_K = \frac{\text{Temps}_{1\,GPU}}{\text{Temps}_{K\,GPUs}} \tag{4.1}$$

et

$$\eta = \frac{A_K}{K}. \tag{4.2}$$

4.3.1 Reconstruction tomographique

Une première stratégie de parallélisation multi-GPU avec une boucle itérative résidant sur le CPU et basée sur une centralisation des données sur l'hôte est d'abord présentée. Une deuxième stratégie basée sur une distribution des données avec une communication *peer2peer* (p2p) des données entre GPUs est ensuite exposée.

Centralisation des données sur l'hôte

Pour la parallélisation multi-GPU, nous avons conservé pour chaque opérateur le même schéma de parallélisation que celui à grains fins utilisé sur les puces GPU. De

cette manière, pour le projecteur $\mathcal{P}_{Regular}$, la distribution des calculs sur les K cartes GPUs s'est faite selon les détecteurs $\mathrm{D}_{\mathtt{un,vn,phi}}$. Chaque GPU est alors responsable de la projection du volume sur une partie du plan de détecteurs comme illustré sur la Fig. 4.3. A l'inverse, pour le rétroprojecteur $\mathcal{R}_{VDL}$ et le laplacien $\mathcal{L}$, la distribution des calculs sur les K cartes GPUs s'est faite selon les voxels $\mathrm{V}_{\mathtt{xn,yn,zn}}$. Chaque GPU est alors responsable de la rétroprojection des données sur un sous-volume comme illustré sur la Fig. 4.4. Les premiers résultats de cette parallélisation ont été publiées à une conférence [C$_{2011a}$] mais les nouvelles optimisations et résultats obtenus par la suite n'ont pas fait l'objet de publication à ce jour.

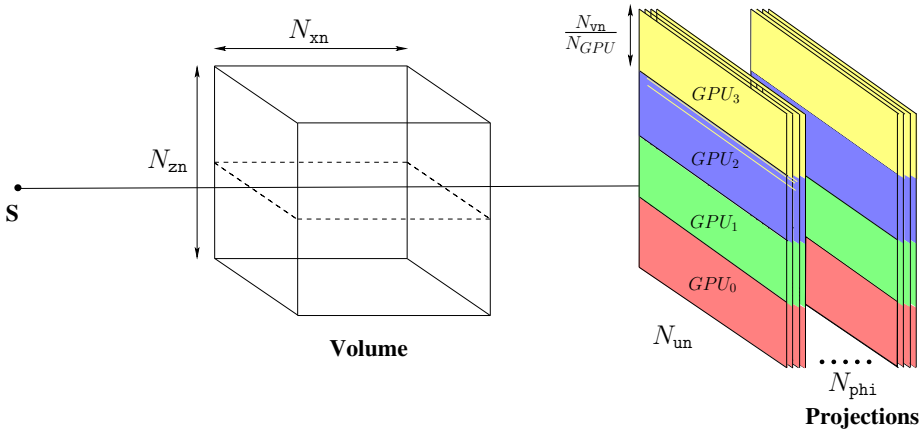


FIGURE 4.3 – Parallélisation multi-GPU de la projection.

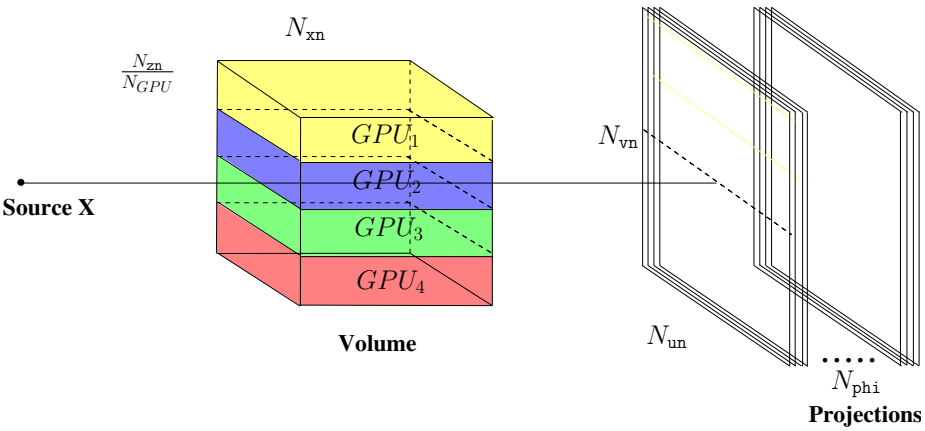


FIGURE 4.4 – Parallélisation multi-GPU de la rétroprojection.

Les performances obtenues sur le serveur $\mathtt{S}_{bebe}$ avec 8 GPUs sont exposées Tab. 4.4 pour chaque opérateur. Nous pouvons constater que l'efficacité n'est pas optimale ; $\mathcal{P}_{Regular}$ et $\mathcal{R}_{VDL}$ obtiennent une accélération respectivement d'un facteur 6.6 (i.e.

$\eta = 0.82$) et 3.1 (i.e. $\eta = 0.39$) malgré l'utilisation des *streams* ; $\mathcal{L}$ obtient un facteur d'accélération encore plus faible de 1.8 (i.e. $\eta = 0.23$) à cause de temps de transfert beaucoup trop prédominants.

	sans stream		avec streams	
	temps (s)	A_8	temps (s)	A_8
$\mathcal{P}_{Regular}$	2.66	5.4 *(η=0.68)*	1.76 *($\times$ 1.51)*	6.6 *(η=0.82)*
$\mathcal{R}_{VDL}$	1.93	3.9 *(η=0.49)*	1.73 *($\times$ 1.11)*	3.1 *(η=0.39)*
$\mathcal{L}$	1.67	1.8 *(η=0.23)*	1.67 *($\times$ 1.00)*	1.8 *(η=0.23)*

TABLE 4.4 – Efficacités de parallélisation des opérateurs $\mathcal{P}_{Regular}$, $\mathcal{R}_{VDL}$ et $\mathcal{L}$ sur $\mathtt{S}_{bebe}[\mathtt{GPU}_{maxwell}]$; jeu de données $[1024 \times 1024^2; 1024^3]$.

Cette efficacité de parallélisation est dépendante de la bande passante du bus de communication (BP_{bus}) ; plus exactement elle est fonction du ratio entre la puissance de calcul des GPUs (P_a) et BP_{bus}. Les performances mesurées avec différentes associations de cartes et de serveur en Tab. 4.5 l'illustrent bien. Le serveur $\mathtt{S}_{bebe}$ avec un bus PCIe gen2 (BP_{bus} =4 Go/s) couplé aux cartes $\mathtt{GPU}_{tesla}$ permet une bonne efficacité de parallélisation ($\eta > 0.95$) mais avec les cartes $\mathtt{GPU}_{maxwell}$ plus puissantes offrent une efficacité non optimale. Le serveur $\mathtt{S}_{bebe2}$ avec un bus PCIe gen3 trois plus rapide (BP_{bus} =12 Go/s) couplé à ces mêmes cartes $\mathtt{GPU}_{maxwell}$ présente des performances nettement améliorées ($\eta > 0.9$).

	$\mathcal{P}_{Regular}$		$\mathcal{R}_{VDL}$	
	A_N	η	A_N	η
$\mathtt{S}_{bebe}[\mathtt{GPU}_{tesla}]$	$A_8 =$ 7.72	0.97	$A_8=$ 7.63	0.95
$\mathtt{S}_{bebe}[\mathtt{GPU}_{maxwell}]$	$A_8 =$ 6.58	0.82	$A_8=$ 3.10	0.39
$\mathtt{S}_{bebe2}[\mathtt{GPU}_{maxwell}]$	$A_{10} =$ 9.6	0.96	$A_{10}=$ 9.0	0.9

TABLE 4.5 – Efficacités de parallélisation des opérateurs $\mathcal{P}_{Regular}$, $\mathcal{R}_{VDL}$ sur différents serveurs multi-GPU ; jeu de données $[1024 \times 1024^2; 1024^3]$.

Les performances de ces opérateurs parallélisés sur plusieurs GPUs et utilisés par un algorithme $\mathtt{DG}_{\mathtt{regQ,opt}}$ de descente de gradient avec régularisation quadratique et calcul du pas d'optimal sont exposées Tab. 4.6. Comme nous pouvons le constater les performances obtenues sont décevantes avec une accélération de l'algorithme de seulement un facteur 2 avec 8 GPUs. Cela est tout simplement dû aux calculs « autres »toujours résidants sur l'hôte ; leurs temps de calcul sont invariants et par conséquent deviennent prépondérants dans la boucle itérative. Ces résultats appellent donc à migrer l'ensemble des calculs de la boucle itérative sur les GPUs.

	$\mathcal{P}_{Regular}(2)$ (%)	$\mathcal{R}_{VDL}$ (%)	$\mathcal{L}(3)$ (%)	Autres (%)	**Total** (s)
$\texttt{Mem}_{CPU}$[1 GPU]	49.6	20.2	30.1	28.1	47.1
$\texttt{Mem}_{CPU}$[8 GPU]	15.9	6.6	21.6	55.9	23.6 *(× 2.00)*

TABLE 4.6 – Temps de traitement par itération des différentes étapes de calcul de l'algorithme $\texttt{DG}_{\texttt{regQ,opt}}$ mesurés sur $\texttt{S}_{bebe}[\texttt{GPU}_{maxwell}]$ avec données centralisée sur CPU et une parallélisation sur 1 ou 8 GPUs ; jeu de données $[1024 \times 1024^2; 1024^3]$.

Distribution des données et communication p2p GPUs

Afin de pouvoir faire résider l'ensemble des calculs et des données sur les GPUs, le découpage des données doit être repensé. En effet, contrairement à la géométrie d'acquisition avec faisceaux parallèles, en géométrie conique, les supports de $\mathcal{P}$ et $\mathcal{R}$ ne coïncident pas avec le découpage des données utilisé en Fig. 4.3 et 4.4. Si on découpe le volume V et les plans de détecteurs D en K tranches axiales, respectivement notées V_k et D_k et les données associés $\boldsymbol{f}_{Vk}$ et $\boldsymbol{g}_{Dk}$, alors le support de $\mathcal{P}(\boldsymbol{f}_{Vk})$ est plus grand que D_k. Symétriquement, le support de $\mathcal{R}(\boldsymbol{g}_{Dk})$ est plus grand que V_k. La Fig. 4.5 illustre cette non coïncidence. En géométrie parallèle (a), la projection sur D_k nécessite uniquement le sous volume V_k et la rétroprojection sur V_k nécessite uniquement le sous plan D_k. En géométrie conique (b), la projection sur D_k nécessite plus que le seul sous volume V_k (en vert) ; il faut également accéder en lecture aux tranches du dessous V_{k-1} (jaune) et du dessus V_{k+1} (rouge).

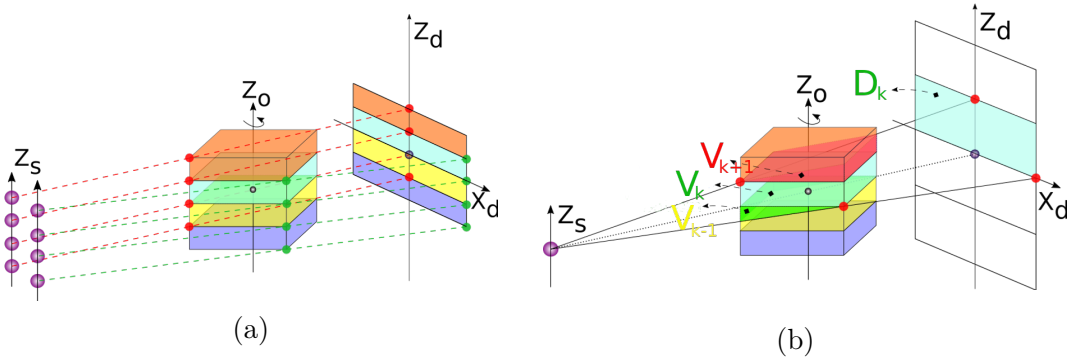
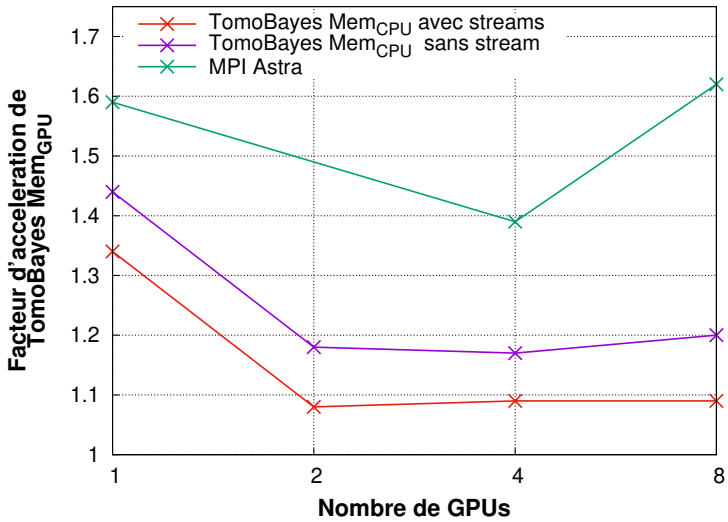


FIGURE 4.5 – Projections de sous-volumes pour une géométrie parallèle (a) et conique (b).

Afin de conserver le découpage des données utilisé en Fig. 4.3 et 4.4, nous avons mis en oeuvre des transferts p2p entre GPUs. Ainsi avant chaque opérateur, le GPU k échange des données avec les GPUs $k-1$ et $k+1$: des tranches de sous volumes avant la rétroprojection $\mathcal{R}$, et des tranches de sous plans de détecteurs avant la projection $\mathcal{P}$. Nous avons comparé les performances obtenues avec une version multi-GPU d'Astra [Palenstijn 2017] (branche git MPI). La parallélisation d'Astra

correspond à une stratégie différente avec le calcul de projections partielles faites à partir du seul sous volume V_k stocké par chaque GPU ; la projection finale sur les rayons traversant plusieurs volume V_k est alors calculé à l'aide d'une étape de réduction par sommation avec l'échange des projections partielles calculées par les différents GPUs. Il est à noter que la branche git principale d'Astra [Ast 2012] est mono-GPU avec centralisation des données sans *streams* ; Cette version MPI est en effet déclarée comme expérimentale car elle requiert des efforts importants de compilation pour pouvoir l'exécuter sur un serveur en particulier. Nous avons effectué ce travail afin de pouvoir comparer nos performances avec celle d'Astra sur un même jeu de données et des mêmes serveurs. A notre connaissance, excepté Astra, les autres *toolkits* multiGPU ne proposent pas de décentralisation de données ; Tigre [Biguri 2016] par exemple est basé sur une centralisation des données sur l'hôte avec *streams*.



	TomoBayes			MPI Astra
	Mem$_{GPU}$	Mem$_{CPU}$	Mem$_{CPU}$	
`HtH` (mn)	2.34	2.56 ×1.09	2.80 ×1.20	3.79 ×1.62
`DG` (mn)	2.55	17.07 ×6.70	17.18 ×6.74	3.84 ×1.51

FIGURE 4.6 – Reconstructions pour 100 itérations exécutées sur $\mathtt{S}_{bebe2}[\mathtt{GPU}_{maxwell}]$ et $\mathtt{S}_{zay}[\mathtt{GPU}_{volta-V100a}]$; (haut) facteurs d'accélération obtenues par TomoBayes avec un stockage $\mathtt{Mem}_{GPU}$ pour la boucle `HtH` par rapport au stockage $\mathtt{Mem}_{CPU}$ (avec et sans *streams*) et la parallélisation MPI d'Astra ; (bas) temps de traitement pour la boucle `HtH` et l'algorithme `DG` de descente de gradient sans régularisation.

Les accélérations obtenues par notre logiciel TomoBayes avec décentralisation des données par rapport à nos versions centralisées avec et sans *streams*, et par rapport à la version MPI d'Astra sont exposées sur le graphe de la Fig. 4.5. Nous avons

utilisé pour ces tests une simple boucle itérative $\boldsymbol{H}^t\boldsymbol{H}$ alternant projections et rétroprojections. Avec 8 GPUs, l'accélération par rapport à notre version centralisée avec *streams* est de 1.09. Le résultat le plus marquant est que lorsque nous utilisons une boucle itérative complète comme par exemple une descente de gradient, le facteur d'accélération est de 6.7. La limitation constatée en 4.3.1 est ainsi levée. Par ailleurs, nous avons des performances du même ordre de grandeur qu'Astra avec une accélération d'un facteur 1.5 pour cette configuration de données. Ces premiers résultats ont été publiés à la conférence RTSCA [C$_{2020c}$] et seront prochainement valorisés par la rédaction d'un article de journal.

4.3.2 Déconvolution pour la radioastronomie

La déconvolution des hypercubes en radioastronomie (voir 2.3.1) peut recourir à une régularisation à la fois spatiale et spectrale ; l'apparition d'une dépendance de données complique alors la distribution des données pour la mise en oeuvre d'une parallélisation multi-GPU. Cette problématique a été abordée lors de ma délégation CNRS au laboratoire Lagrange et a fait l'objet des travaux d'Olivier Pérard au L2S que j'ai encadré lors de son stage de M2 poursuivi par un CDD pour le développement du logiciel DeconvSKA.

Quand la régularisation spectrale s'en mêle

Avec la double régularisation spatiale et spectrale, le critère $J(\boldsymbol{f})$ à minimiser présente une expression simplifiée de cette forme :

$$J(\boldsymbol{f}) = \sum_{l=1}^{L} \|\tilde{\boldsymbol{I}}_l - \boldsymbol{H}_l\boldsymbol{I}_l\|_2^2 + \mu_I \sum_{l=1}^{L} \|\boldsymbol{D}_I\boldsymbol{I}_l\|_2^2 + \mu_S \sum_{i,j=1,1}^{N^2} \|\boldsymbol{D}_S\boldsymbol{S}_{i,j}\|_2^2,$$

avec

$$\begin{cases} \boldsymbol{f} & : \text{l'hypercube du ciel décomposé en } L \text{ images } \boldsymbol{I}_l \text{ ou } N^2 \text{ spectres } \boldsymbol{S}_{i,j} \\ \boldsymbol{g} & : \text{l'hypercube } \textit{dirty} \text{ à déconvoluer composé de } L \text{ images } \textit{dirty } \tilde{\boldsymbol{I}}_l \\ \boldsymbol{H} & : \text{la matrice instrument décomposée en } L \text{ sous matrices } \boldsymbol{H}_l \\ \mu_I, \mu_S & : \text{les paramètres scalaires de régularisation spatiale et spectrale} \\ \boldsymbol{D}_I, \boldsymbol{D}_S & : \text{des opérateurs linéaires appliqués à une image ou à un spectre} \end{cases} \tag{4.3}$$

En utilisant un algorithme DG de pas α, la solution sera recherchée itérativement :

$$\boldsymbol{f}^{(m+1)} = \boldsymbol{f}^{(m)} - \alpha \nabla J_I(\boldsymbol{f}^{(m)}) - \alpha \nabla J_S(\boldsymbol{f}^{(m)}),$$

avec

$$\begin{cases} \nabla J_I(\boldsymbol{f}) &= \displaystyle\sum_{l=1}^{L} -\frac{1}{2}\boldsymbol{H}_l^t(\tilde{\boldsymbol{I}}_l - \boldsymbol{H}_l\boldsymbol{I}_l) + \mu_I \boldsymbol{D}_I^t \boldsymbol{D}_I \boldsymbol{I}_l \\ \nabla J_S(\boldsymbol{f}) &= \mu_S \displaystyle\sum_{i,j=1,1}^{N^2} \boldsymbol{D}_S^t \boldsymbol{D}_S \boldsymbol{S}_{i,j}. \end{cases} \tag{4.4}$$

Pour un élément de l'hypercube $f_{i,j,l}$, cela correspondra à :

$$f_{i,j,l}^{(m+1)} = f_{i,j,l}^{(m)} - \alpha\big[\nabla J_I(\boldsymbol{I}_l)\big]_{i,j} - \alpha\big[\nabla J_S(\boldsymbol{S}_{i,j})\big]_l,$$

avec

$$\begin{cases} \nabla J_I(\boldsymbol{I}_l) &= -\frac{1}{2}\boldsymbol{H}_l^t(\tilde{\boldsymbol{I}}_l - \boldsymbol{H}_l\boldsymbol{I}_l) + \mu_I \boldsymbol{D}_I^t \boldsymbol{D}_I \boldsymbol{I}_l \\ \nabla J_S(\boldsymbol{S}_{i,j}) &= \mu_S \boldsymbol{D}_S^t \boldsymbol{D}_S \boldsymbol{S}_{i,j}. \end{cases} \tag{4.5}$$

Dans la suite, l'application de la matrice $\boldsymbol{H}_l$ se fera par un opérateur de convolution $\mathcal{C}$. De même, les opérateurs linéaires utilisés pour les régularisations se feront par convolution. Pour la convolution correspondant à $D_S^t D_S$, nous noterons son noyau h et son rayon R.

Parallélisation multi-GPU

Dans le cas où aucune régularisation spectrale n'est appliquée, les calculs pour une image I_l du ciel sont indépendants de ceux effectués pour les autres images ; la parallélisation multi-GPU revient alors à L déconvolutions d'images pouvant être distribuées sur les K cartes GPU en suivant un simple parallélisme de tâches : chaque GPU_k est chargé de déconvoluer une collection de $L_{GPU} = \dfrac{L}{K}$ images *dirty* $\tilde{\boldsymbol{I}}_{GPU\,k} = \left\{\tilde{\boldsymbol{I}}_{k*L_{GPU}}, \tilde{\boldsymbol{I}}_{k*L_{GPU}+1} \ldots \tilde{\boldsymbol{I}}_{k*L_{GPU}+k-1}\right\}$. La distribution des données de l'hypercube se fait ainsi simplement selon les longueurs d'onde comme illustré Fig. 4.7 (a).

Quand la régularisation spectrale est appliquée, le GPU_k aura alors besoin d'images stockées sur des GPUs voisins. Par exemple pour D_S pris égal à un laplacien de noyau [-1 2 -1] qui correspondra à un rayon R égal à 2 pour $D_S^t D_S$, le GPU_k aura besoin

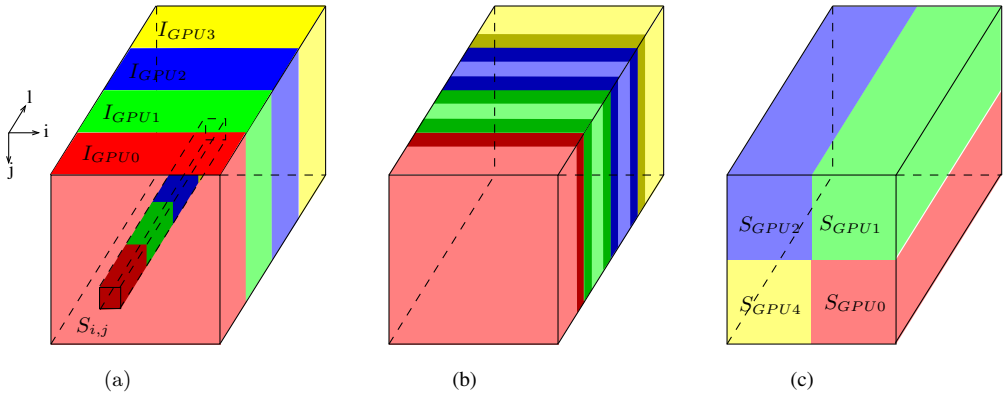


FIGURE 4.7 – Distribution des données sur 4 GPUs pour la déconvolution d'un hypercube : (a) selon la dimension spectrale ; (b) selon la dimension spectrale avec échange d'images entre GPU voisins (couleur foncé) ; (c) selon la dimension spatiale.

d'accéder à deux images chez chacun de ses voisins GPU_{k-1} et GPU_{k+1} comme illustré en Fig. 4.7 (b). Un transfert de données entre GPUs permettra dans ce cas de calculer $\nabla J_S(\boldsymbol{f})$ tout en conservant une distribution spectrale des données. Pour des valeurs de R plus grandes, la collection d'images à stocker sur chaque GPU_k ($I_{GPU\,k}$ + les images voisines) pourra se révéler trop grande et dépasser la capacité des mémoires GPUs. Pour des valeurs de R extrêmes, une réorganisation de la distribution des données sera nécessaire afin de calculer $\nabla J_S(\boldsymbol{f})$; elle se fera selon la dimension spatiale avec chaque GPU_k chargé de ce calcul pour une collection de spectres $S_{GPU\,k}$ comme illustré Fig. 4.7 (c). Dans ce cas, à chaque itération, la distribution des données sur les différents GPUs alternera entre un découpage dans la dimension spectrale (Fig. 4.7-a) pour calculer $\nabla J_I(\boldsymbol{f})$, et un découpage dans la dimension spatiale (Fig. 4.7-c) pour calculer $\nabla J_S(\boldsymbol{f})$; cette stratégie sera coûteuse en temps de transferts de données. Nous proposons dans la suite une solution à ce cas limite avec des premiers résultats.

Convolution spectrale distribuée : premiers résultats

Nous avons abordé cette problématique en souhaitant conserver une distribution spectrale des données lors du calcul de $\nabla J_S(\boldsymbol{f})$ sans transfert de données massif avant ou après la convolution. Dans la suite, le spectre $\left[\nabla J_S(\boldsymbol{f})\right]_{i,j}$ sera noté $\mu_S * \boldsymbol{\psi}_{i,j}$. Pour calculer $\nabla J_S(\boldsymbol{f})$, il faudra ainsi effectuer N^2 convolutions 1D,

$$\forall\, i,j,\ \ \boldsymbol{\psi}_{i,j} = \boldsymbol{S}_{i,j} * h \tag{4.6}$$

qui dans le domaine direct avec R impaire s'exprimeront par

$$\forall\, l,\ \ \boldsymbol{\psi}_{i,j}(l) = \sum_{r=-R}^{R} \boldsymbol{S}_{i,j}(l+r) * h(-r). \tag{4.7}$$

Dans le schéma de distribution des calculs proposé, une parallélisation à grains fins sur les coeurs des GPUs se fera selon la dimension spatiale ; nous détaillons dans la suite le calcul de chaque spectre $\boldsymbol{\psi}_{i,j}$ qui sera le même pour toutes les coordonnées i, j. Au début et tout au long des calculs, chaque GPU k ne possède qu'un sous-spectre de $\boldsymbol{S}_{i,j}$ correspondant à l'intervalle de longueur d'onde $[k*L_{GPU}, k*L_{GPU}+k-1]$ comme illustré Fig. 4.7 (a). Ces données d'entrée ne bougeront pas du GPU k ; ce sont les sommes partielles $\mathbf{Sum}_{i,j}$ calculées au cours des itérations r de la convolution (voir eq. 4.7) qui transiteront au besoin de GPUs en GPUs. A chaque itération r, deux étapes de calcul et une de transfert de données entre GPUs se succéderont. Ces étapes pour le calcul de $\psi_{i,j}(l)$ sont illustrées sur la Fig. 4.8 et détaillées ci-dessous :

1. sur le GPU k', le coefficient $h(-r)$ du noyau de convolution est appliqué à $\boldsymbol{S}_{i,j}(l-r)$, puis cette valeur $\boldsymbol{S}_{i,j}(l-r)*h(-r)$ est accumulée à la somme partielle courante $\mathbf{Sum}^{up}_{i,j}(l-r)$.
2. sur le GPU k'', le coefficient $h(r)$ du noyau de convolution est appliqué à $\boldsymbol{S}_{i,j}(l+r)$, puis cette valeur $\boldsymbol{S}_{i,j}(l+r)*h(r)$ est accumulée à la somme partielle courante $\mathbf{Sum}^{down}_{i,j}(l+r)$.
3. les sommes partielles résultantes $\mathbf{Sum}^{up}_{i,j}(l-r+1)$ et $\mathbf{Sum}^{down}_{i,j}(l+r-1)$ sont transférées aux GPUs voisins ($k'+1$ ou $k''-1$) si leur indice de stockage change d'intervalle de longueur d'onde, i.e. $l-r+1=(k'+1)$ ou $l+r-1=(k''-1)$.

Au bout de R itérations, les sommes partielles montantes $\mathbf{Sum}^{up}_{i,j}(l)$ et descendantes $\mathbf{Sum}^{up}_{i,j}(l)$ arriveront dans le GPU k gérant leur longueur d'onde, i.e. $l \in [k*L_{GPU}, k*L_{GPU}+k-1]$. Leur somme correspondra alors à $\psi_{i,j}(l)$.

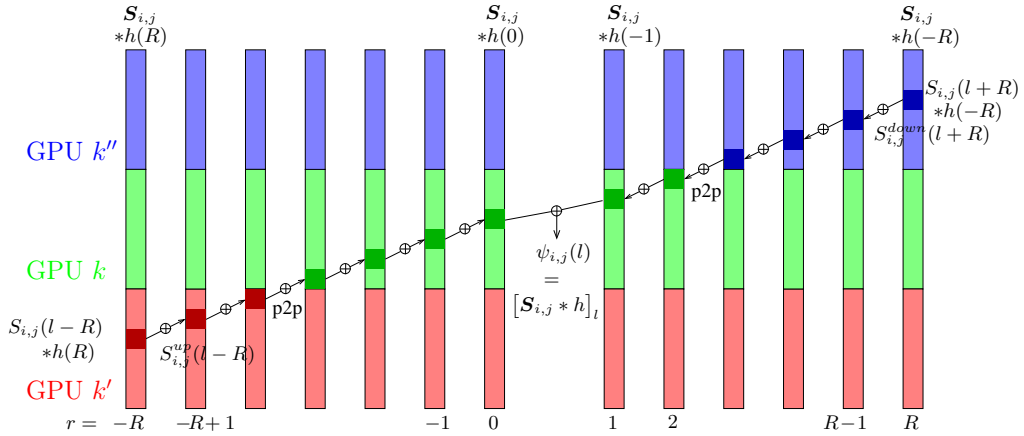
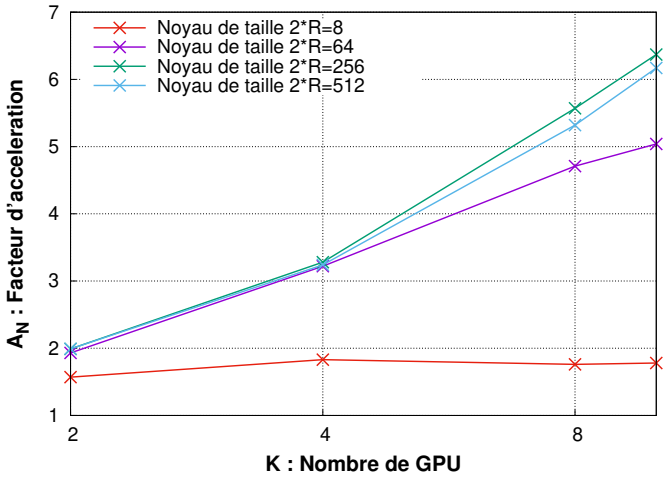


FIGURE 4.8 – Convolution spectrale distribuée sur un serveur multi-GPU : chaque GPU (rouge, vert, bleu) à la charge d'un sous spectre d'entrée $\boldsymbol{S}^k_{i,j}$ et de sortie $\boldsymbol{\psi}^k_{i,j}$.

Les résultats d'accélération obtenus sont exposés Tab. 4.8. L'efficacité de parallélisation à grains moyens est bonne pour des noyaux de taille assez grande ; une accélération d'un facteur 7 est ainsi obtenu avec 8 GPUs sur $\texttt{S}_{zay}$ pour un hypercube de taille 1024^3 et un noyau de taille 512. Comparé à une convolution exécutée sur Matlab sur $\texttt{CPU}_{xeon-bebe2}$ (28 coeurs physiques actifs), une accélération d'un facteur 75 est ainsi atteinte. Une évaluation plus approfondie de cette méthode doit encore être réalisée en exécutant l'algorithme itératif entier (eq. 4.5). Il sera notamment utile de la comparer avec une méthode plus classique de parallélisation multi-GPU qui consisterait à redistribuer les données selon la dimension spatiale (Fig. 4.7-c), lancer ensuite une convolution avec un *kernel* GPU plus efficace (toutes les données sont alors disponibles pour la parallélisation à grains fins), et enfin redistribuer les données selon la dimension spectrale (Fig. 4.7-a). Des premiers tests montrent en effet que de tels *kernels* sont deux fois plus efficaces que notre *kernel* actuel. La question est de savoir si ces transferts mémoire pré et post-traitement permettront à la méthode « classique » de conserver cet avantage. Par ailleurs, des améliorations pourront être apportées à notre schéma de parallélisation en intensifiant les calculs dans le *kernel* avec plus d'itérations r traitées avant chaque transfert inter-GPU. Cela permettrait ainsi d'effectuer des accumulations dans des registres et non systématiquement dans la mémoire globale comme c'est actuellement le cas.



	Matlab sur $\texttt{CPU}_{xeon-bebe2}$	$\texttt{S}_{zay}[8\,\texttt{GPU}_{volta-V100a}]$	$\texttt{S}_{bebe2}[10\,\texttt{GPU}_{maxwell}]$
Temps (s)	119	1.58 *(× 75.5)*	3.22 *(× 30.1)*
A_N		A_8=6.97	A_{10}=7.87
η		0.87	0.79

FIGURE 4.9 – Accélération de la convolution spectrale parallélisée sur serveurs multi-GPU pour un hypercube de taille 1024^3 ; (haut) facteurs d'accélération obtenues sur $\texttt{S}_{bebe2}$ par rapport à une version mono-GPU pour différentes tailles du noyau de convolution ; (bas) temps de traitement obtenus sur Matlab et comparés à ceux obtenus sur $\texttt{S}_{zay}$ et $\texttt{S}_{bebe2}$ pour un noyau de taille $2 * R = 512$.

4.4 Conclusion

Afin d'éviter une centralisation des données sur l'hôte, nos travaux sur la distribution des calculs sur serveur multi-GPU de la boucle itérative des algorithmes d'inversion de données cherchent à les stocker sur les mémoires des cartes GPUs ; réunies, ces mémoires offrent une capacité de stockage équivalente à celle de l'hôte. Cette stratégie s'appuie sur les communications haut débit offertes par les bus inter-GPU (NVLink) et évite les communications bas débit des bus reliant l'hôte aux GPUs (PCIe). Pour une efficacité la plus optimale possible, elle est basée sur un découpage des données adapté aux deux niveaux de parallélisme ; pour le parallélisme à grains fins, une forte localité des accès mémoire est recherchée afin de rendre intense les *kernels* de calcul exécutés sur les milliers de coeurs des GPUs ; pour le parallélisme à grains moyens, une indépendance des jeux de données confiés à chaque carte GPU est attendue. Mais comme nous l'avons constaté en reconstruction tomographique et en déconvolution appliquée à la radioastronomie des dépendances existent entre ces jeux de données. Des échanges de données entre GPU spécifique à chaque problématique ont ainsi été mis en place. Nos résultats d'accélération en tomographie sont tout à fait au niveau de l'état de l'art et nos premiers résultats en radioastronomie sont encourageants.

Chapitre 5

Perspectives à court et moyen terme

Sommaire

Ce chapitre dresse des perspectives à court et moyen terme à mes travaux de recherche. Ils s'appuieront sur mes collaborations actuelles avec des collègues du GPI (principalement François Orieux et Charles Soussen) et des partenaires industrielles (Thalès TRT, GE Healthcare, Safran, Atos Bull) ou académiques (Lagrange, IETR, IRISA, Obs Paris). Par ailleurs, des nouvelles collaborations seront recherchées à l'échelle locale comme l'initiative de plateforme TOMX sur le plateau de Saclay, nationale ou internationale via le projet SKA. La reconstruction tomographique et la radioastronomie restera au coeur de mes travaux futurs à court et moyen terme, et les perspectives ouvertes par les nouveaux outils de l'IA pour l'accélération des algorithmes seront explorées suite aux travaux initiés dans la thèse de Mickael Seznec.

5.1 Reconstruction tomographique

5.1.1 Exploration algorithmique et architecturale pour la paire $\mathcal{P}/\mathcal{R}$

Dans le cadre des travaux de thèse de Daouda Diakite, la recherche de l'accélération sur cibles FPGA pour la reconstruction tomographique sera poursuivie. L'objectif est d'affiner la méthodologie de conception avec les outils HLS d'Intel élaborée lors

de la thèse de Maxime Martelli. La thèse de Daouda Diakite se focalise sur des algorithmes posant des difficultés de parallélisation sur GPU notamment à cause de divergence de branches. Une étude hors ligne du taux de réutilisation des données en fonction des paramètres des boucles sur les voxels et les détecteurs sera réalisée ; elle permettra de rendre plus efficace l'utilisation de la mémoire BRAM par l'architecture générée par les outils HLS. Les performances sur FPGA seront comparées à celles obtenues sur cibles GPU avec une utilisée plus systématique des *roofline models*. Une parallélisation multi-FPGA sera par ailleurs explorée. Ces travaux seront particulièrement utiles pour préparer ceux portant sur le radiotélescope SKA.

5.1.2 Logiciel TomoBayes

Les développements liés au logiciel TomoBayes (voir annexe C) seront poursuivis avec la volonté que son accès soit ouvert. Pour une diffusion plus large, un accord doit être trouvé entre le L2S et Safran, les deux co-propriétaires de ce logiciel. Une intégration de modules (méthodes bayésiennes, paires $\mathcal{P}/\mathcal{R}$ accélérées sur GPUs) dans des *toolkits* ayant déjà une communauté d'utilisateurs reste la stratégie de valorisation recherchée. Depuis la fin de la thèse de C. Chapdelaine, le L2S continue ses développements sur ce logiciel. Dans une prochaine version (v3), sera ajoutée la parallélisation multi-GPU avec une distribution des données sur les différents cartes du GPU qui évite les transferts de données vers le CPU à chaque itération [C_{2020c}]. L'accélération sur FPGA de la rétroprojection [C_{2020a}] sera également ajoutée. Par ailleurs, la géométrie d'acquisition conique sera généralisée avec une matrice de projection par angle d'acquisition afin de reconstruire les données en imagerie interventionnelle (C-arm) dans le cadre d'une collaboration avec GE Healthcare (Buc). Les spécifications de ces futurs développements et de son impact sur les paires de projection/rétroprojection ont été effectuées par Jean Cohen lors de son parcours recherche à CentraleSupélec que je co-encadre avec Charles Soussen.

5.2 SKA, un projet HPC pionnier pour des communautés en adéquation

Le défi calculatoire que constitue le radiotélescope SKA (*Square Kilometer Array*) [project SKA 2016, Acero 2018] illustré Fig. 5.1 motive mon implication grandissante en radioastronomie. Il nécessite des travaux de recherche interdisciplinaires ; il est ainsi source de nombreuses collaborations pour le GPI : DeconvSKA, soumission du projet ANR Dark-era, la thèse ExaSKA démarrée en décembre 2019 et le projet ROHSA-GPU.

5.2.1 DeconvSKA

La collaboration avec le laboratoire Lagrange portant sur la parallélisation multi-GPU de la déconvolution en radioastronomie sera poursuivie. Comme indiqué dans le chapitre précédant, l'évaluation de la méthode de parallélisation est à approfondir.

Nous cherchons actuellement les moyens financiers permettant à Olivier Pérard de poursuivre ses travaux dans cette voie.

5.2.2 Projet ANR Dark-era

Après le résumé du projet Dark-era, la motivation du GPI ainsi que son le rôle au sein de ce consortium sont présentés.

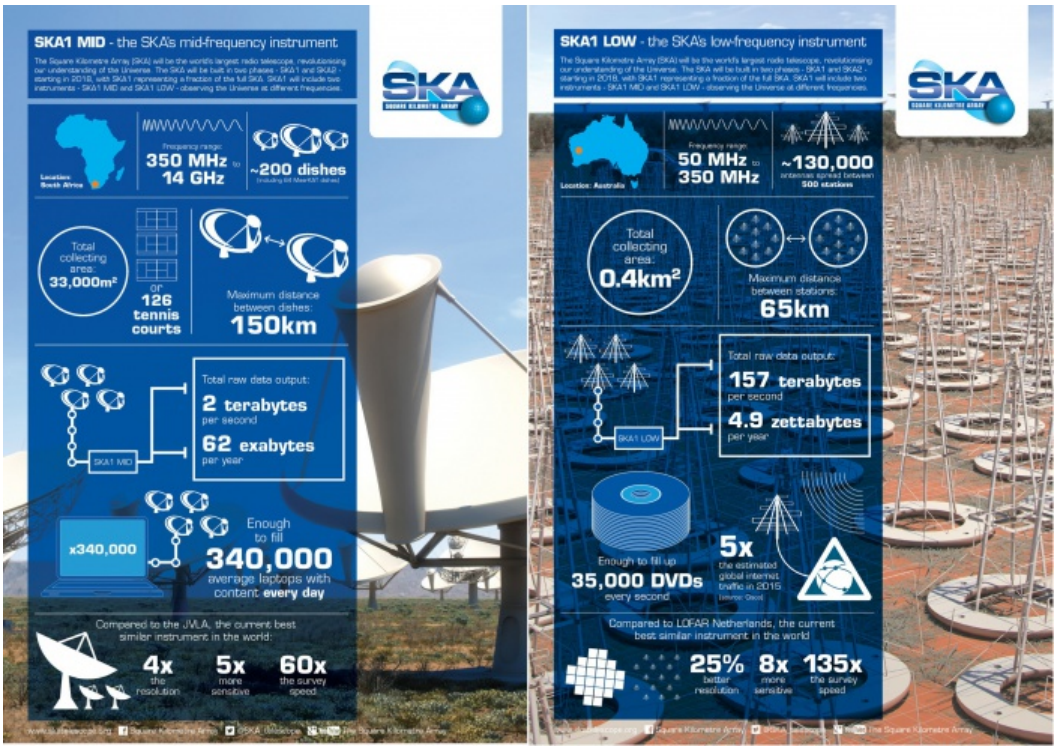


FIGURE 5.1 – Le radiotélescope SKA

Remarque : le projet ANR Dark-era a passé la phase 1 lors de sa soumission en 2019 et 2020. Nous sommes en attente de la réponse pour la phase 2. L'implication forte que requiert la coordination du montage d'un projet ANR comme Dark-era a été possible grâce au temps recherche dégagé lors de ma délégation CNRS. L'élaboration du projet a été initié par le projet SKALLAS (PEPS astro informatique du CNRS) dont je suis également porteur. Il a nécessité des pré-études comme par exemple un stage en collaboration entre le L2S et l'IETR afin d'explorer le potentiel de l'API Dask pour la parallélisation multi-noeuds de la déconvolution [Morooka 2019].

Résumé du projet PRC déposé

Le radiotélescope SKA nécessitera des supercalculateurs à fortes contraintes techniques. Le *Science Data Processor* (SDP) chargé de produire les images multidimen-

sionnelles du ciel devra exécuter en temps réel une chaîne d'algorithmes complexes à partir de données provenant de télescopes ayant un débit de plusieurs Tb/s sans aucune capacité de stockage. Le SDP devra également être le plus écologique possible avec un budget énergétique de seulement 1 MWatt pour 250 pétaflops. De telles exigences en matière d'énergie et de calcul impliquent que le SDP soit une architecture innovante orientée flux de données et hétérogène, basée sur un système HPC standard combiné à un FPGA ou à une architecture *manycore* de type GPUs ou MPPA de Kalray. Un défi crucial consiste à évaluer les performances, tant en temps qu'en énergie, des nouveaux algorithmes de flux de données scientifiques complexes sur des infrastructures de calcul complexes qui n'existent pas encore, ce qui est difficilement possible sans des méthodes de co-conception efficaces et des outils de prototypage rapide.

SimSDP est l'outil de prototypage rapide pour les applications en flux de données de type SKA développée par Dark-era. Grâce à une approche mixte originale basée sur l'exécution et la simulation, SimSDP a pour but de fournir des analyses précoces en termes d'utilisation de la mémoire, de latence, de débit et de consommation d'énergie. Suivant une approche d'Adéquation Algorithme Architecture, SimSDP s'appuiera sur un modèle de flux de données de l'algorithme et un modèle de l'architecture cible. SimSDP est bâti sur deux outils existants : PREESM et SimGrid comme illustré Fig. 5.2. PREESM évalue avec précision les performances d'un seul nœud hétérogène ; SimGrid simule avec précision les communications entre les nœuds. Ainsi, l'association de PREESM et de SimGrid permettra des simulations fiables de systèmes HPC hétérogènes à grande échelle.

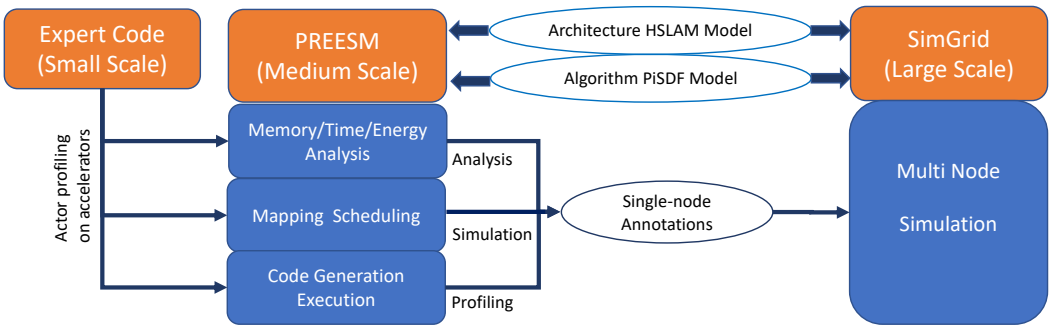


FIGURE 5.2 – Outil de prototypage rapide SimSDP.

Via SimSDP, l'espace des algorithmes et des architectures seront explorés dans le contexte de SKA. La nouvelle génération de pipelines d'imagerie en radioastronomie comme DDFacet sera décrite à un niveau d'abstraction élevé, adapté au ciblage de tout système HPC hétérogène multinoeuds que SKA pourrait choisir à l'avenir. Ensuite, plusieurs configurations d'architecture SDP (nombre de nœuds, type d'accélérateurs) et plusieurs configurations d'algorithmes SDP seront explorées conjoin-

tement grâce aux simulations à grande échelle offertes par SimSDP. En outre, des prototypes SDP sur MPPA et sur FPGA conçus par synthèse de haut niveau (HLS) et mis en place au radiotélescope NenuFAR de Nançay seront développés et profilés sur des jeux de données à petite échelle en *streaming* afin d'évaluer le potentiel des accélérateurs *low-power* alternatifs à l'architecture GPU classique. Les performances sur GPU/MPPA/FPGA seront transmises au SimSDP via des annotations sur les graphes de flux de données. Les prototypes SDP seront également comparés aux simulations SimSDP sur des ensembles de données à moyenne échelle afin d'évaluer les nouvelles fonctionnalités de SimSDP.

Dark-era est un consortium regroupant des compétences complémentaires en informatique, traitement du signal et astronomie, avec douze membres permanents provenant des équipes de SimGrid et de PREESM (IETR), du GPI au L2S et de deux équipes de radioastronomie aux Obs. de Paris et de la Côte d'Azur. Les résultats préliminaires obtenus par ce consortium en collaboration avec Atos-Bull dans le cadre du projet CNRS SKALLAS seront poursuivis. Deux doctorants travailleront sur l'association des outils PREESM et Simgrid dans SimSDP, et deux post-docs seront recrutés. Avec le soutien de SKA-France, ce projet de 4 ans vise à promouvoir les contributions françaises à SKA telles que DDFacet et à être une force de proposition pour SKA computing. Enfin, Dark-era entend être le terreau de nouvelles collaborations internationales notamment à travers le réseau européen et international Rising STARS.

Exploration des accélérateurs *low power*

Outre la participation à la construction d'un simulateur d'un supercalculateur exaflopique possédant une architecture hétérogène et l'apport d'une contribution française à la communauté *SKA computing*, la motivation du GPI pour la participation à ce projet est l'exploration des accélérateurs *low power* comme les FPGAs.
Les conclusions préliminaires du *Critical Design Review* [SDP consortium 2018] du consortium SKA SDP ont conduit à choisir le coprocesseur GPU pour la plateforme de prototypage des performances (*P3 architecture*). En effet, les performances compétitives et la maturité de cette technologie vieille de 13 ans en font un standard dans le domaine du HPC. Elle a été assez rapidement et largement adoptée par la communauté de la radioastronomie à différents niveaux : le corrélateur CSP [Broekema 2018], les algorithmes de déconvolution (WSCLEAN [Offringa 2014] utilisant IDG [Veenboer 2017], OMP [Belle 2017], MEM [Cárcamo 2018]) ou l'imageur DDFacet [Hugo 2014]. Cette architecture *many cores* offre des ressources d'accélération additionnelles à ces milliers de coeurs avec ses unités de calcul demi-flottant ou ses *tensor cores* comme nous l'avons déjà décrit dans ce manuscrit.

Des accélérateurs alternatifs aux processeurs GPUs présentent toutefois une meilleure efficacité énergétique et à cet titre ne peuvent pas être mis hors jeu pour le choix final de l'accélérateur sur le futur supercalculateur SDP. Dark-era vise à cibler deux

solutions alternatives : le *many cores* MPPA Kalray et le FPGA. L'IETR étudie les performances du Kalray sur le pipeline CSP depuis 2014 en collaboration avec AUT (Nouvelle-Zélande) [Hascoet 2015], et sur la calibration [Sourbier 2018] depuis 2017. Le FPGA est une architecture matérielle offrant un plus grand degré de liberté que les architectures basées sur des processeurs comme le CPU, le GPU ou le MPPA avec une conception dédiée aux algorithmes. Cependant, les flots de synthèse habituels nécessitent une expertise matérielle et un long temps de mise en œuvre comme nous l'avons vu au chapitre 1. L'une des promesses des outils HLS émergeants [Nane 2016] est de rendre le développement de FPGA accessible aux ingénieurs logiciels avec des implémentations matérielles générées à partir de langages de programmation logiciels comme C, C++ ou OpenCL. Ensuite, la conception du FPGA peut être optimisée progressivement par l'intégration de blocs matériels. Les solutions FPGA dédiées sont généralement plus performantes que les solutions GPU en utilisant toute la puissance de calcul disponible et en évitant l'encombrement de la mémoire. L'équipe Astron a ainsi réalisé avec succès une conception HLS sur FPGA pour la radioastronomie [Veenboer 2019]. L'outil SimSDP évaluera le gain de performance en utilisant le FPGA et le MPPA à l'intérieur des nœuds HPC à l'échelle SKA.

Découpage en tâches et rôle du GPI

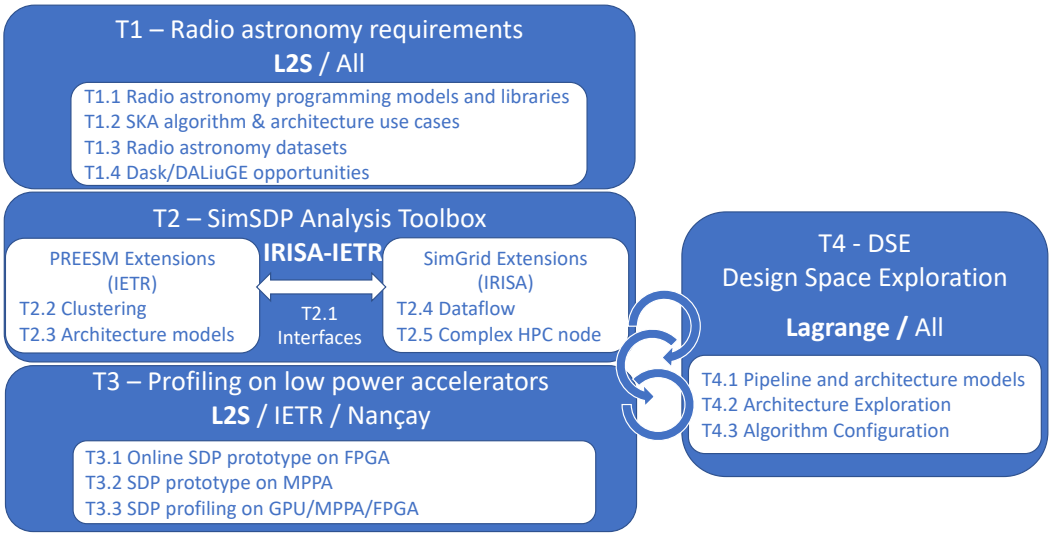


FIGURE 5.3 – Découpage en tâches du projet Dark-era.

L'organisation en tâches du projet Dark-era est illustrée Fig. 5.3. Dans la Tâche T1, les chercheurs en traitement du signal et en informatique conjointement avec les radioastronomes définissent les besoins en développements logicielles spécifiques au domaine de la radioastronomie ; la Tâche T2 se focalise sur la construction de l'outil SimSDP de prototypage rapide ; la Tâche T3 évalue les architectures *low power*

et fournit des noyaux de calcul optimisées ainsi que des *feedbacks* de performances nécessaires à l'outil SimSDP ; enfin, la Tâche T4 effectue l'exploration architecturale et algorithmique à l'aide de SimSDP sur des cas d'études définis dans la Tâche T1. Le GPI est le coordinateur principal du projet et est ainsi responsable de la tâche T1. Il est également responsable de la tâche T3 mettant son expertise duale en problèmes inverses et architectures de calcul au service du projet. Dans ce contexte, un démonstrateur sur FPGA pour le radiotélescope NenuFAR [Zarka 2015] sera conçu par le GPI en collaboration avec l'observatoire de Nançay.

5.2.3 Projet ExaSKA

La thèse de Nicolas Monnier débutée en décembre 2019 en collaboration avec Atos Bull et l'observatoire de Paris, englobe plus largement l'ensemble du pipeline du radiotélescope SKA afin de proposer une parallélisation multi-GPU et multi-noeuds. Une étude en amont de la chaîne de traitement est en cours ayant pour objectif de décloisonner les étapes du pipeline et faire émerger des leviers d'accélération sur cette architecture possédant trois niveaux de parallélisation : (i) gros grains sur les différents noeuds du serveur, (ii) grains moyens avec les différentes cartes GPU de chaque noeud et (iii) grains fins avec les milliers de coeurs de chaque GPU.

5.2.4 Projet ROHSA-GPU

La collaboration du GPI avec Marc-Antoine Miville-Deschênes (CEA) et Antoine Marchal (CITA à Toronto), initiée par François Orieux, sera poursuivie afin d'accélérer la méthode ROHSA pour le démélange de gaz interstellaires (voir section 2.3.2). Ils font l'objet des travaux en parcours recherche de Jérémy Besson que j'encadre ; ce dernier effectuera un stage de 4 mois à l'automne 2020 au L2S puis 6 mois au laboratoire CITA pour développer de manière plus approfondie ses travaux.

5.3 Utilisation des outils de l'IA

Dans le cadre des travaux de thèse de Mickael Seznec, les outils matériels et méthodologiques de l'IA sont utilisés comme outils d'accélération. Les travaux sur la convolution $\mathcal{C}$ portée sur *tensor cores* (voir 3.2.3) seront ainsi poursuivis. Parallèlement, des travaux sont en cours utilisant des réseaux de neurones pour accélérer sur systèmes embarqués des algorithmes entiers de flot optique dense. L'idée est d'utiliser les réseaux de neurones pour mimer les algorithmes traditionnels de flot optique dense. Cette inférence potentiellement plus rapide que des algorithmes difficilement parallélisables sur GPU pourrait ainsi apporter un bon compromis entre qualité et vitesse de traitement. La phase d'apprentissage est faite à partir de données générées par les algorithmes traditionnels. Cette approche pourra par la suite se généraliser à d'autres algorithmes du traitement d'image.

5.4 Conclusion

L'approche A^3 « globale » développée au sein du GPI est par essence proche des algorithmes ; elle est basée sur une expertise partagée au sein d'une équipe de recherche à la fois sur les algorithmes et les architectures de calcul. Cela explique ma volonté de ne pas m'écarter à moyen terme des deux champs applicatifs présentés tout au long de ce manuscrit : la reconstruction tomographique et la radioastronomie. Ces domaines de recherche sont dynamiques avec des problématiques se renouvelant sans cesse ; l'accélération des calculs restera encore longtemps une des clefs afin de faire émerger des avancées majeures pour ces deux champs applicatifs. Conjointement à cette recherche applicative, les concepts algorithmiques et architecturaux mis en oeuvre doivent servir à une recherche plus fondamentale se situant à la frontière de l'informatique et du traitement du signal. L'expertise développée pourra par exemple se transcrire en guide de bonnes pratiques d'accélération afin de généraliser les travaux à d'autres domaines. La collaboration avec l'IETR dans le cadre du projet SKA est en ce sens tout particulièrement enrichissante pour mes travaux en A^3 ; elle me permet en effet d'interagir avec une équipe de recherche travaillant sur une méthodologie A^3 basée sur le développement d'outils comme PREESM complémentaire à l'approche A^3 que je développe au sein du GPI.

Annexes

Annexe A

Modèles de projection et rétroprojection

Dans cette présentation des principaux modèles de projection et rétroprojection utilisés en reconstruction tomographique, les opérateurs sont décrits de manière succinctes afin de permettre analyses et discussions sur leurs potentiels de parallélisation sur cibles GPU ou FPGA. Chaque algorithme modélise différemment la composante purement géométrique de la matrice système $\boldsymbol{H}$ dont les éléments H_{ij} correspondent à la sensibilité du détecteur i défini par $D_{\texttt{un,vn,phi}}$ au voxel j défini par $V_{\texttt{xn,yn,zn}}$. Chaque modèle définit implicitement une approximation d'un modèle analytique idéal qui calculerait le volume d'intersection entre le parallélépipède élémentaire délimitant le voxel j avec le faisceau conique reliant la source ponctuelle de rayon X au détecteur rectangulaire i comme illustré en Fig. A.1.

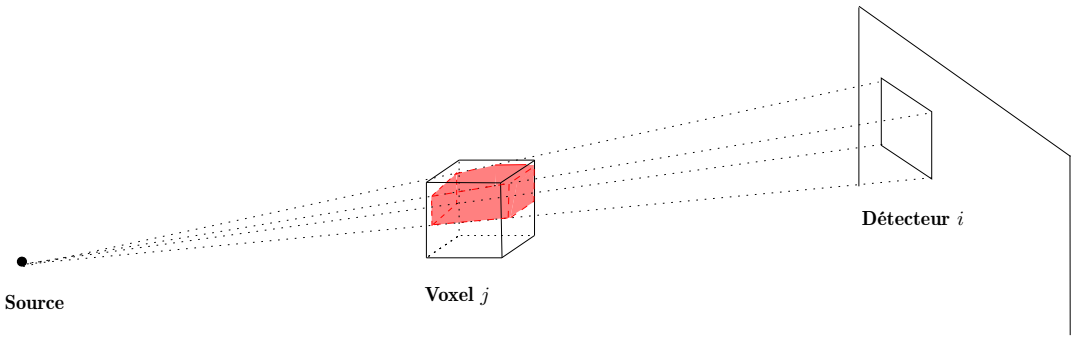


Figure A.1 – Modèle analytique d'intersection entre les volumes du voxel i et du faisceau conique reliant la source au détecteur j.

Dans la suite de cette annexe, les coordonnées x, y, z dans l'espace volume et u, v, ϕ dans celui des projections seront exprimées de manière discrète afin de correspondre aux indices de stockage des données dans les mémoires des systèmes de traitement numérique. Par exemple, à la coordonnée physique u est associée la coordonnée discrète un :

$$un = \frac{u}{\delta_u} + un_0, \tag{A.1}$$

avec δ_u la longueur d'un détecteur dans la dimension u, et un_0 un offset utilisé afin d'avoir des indices mémoire positifs. Par ailleurs, pour chaque coordonnée discrète

est associé une valeur entière $\mathtt{un_e}$ et une valeur décimale ε_u afin d'exprimer les opérations d'interpolation :

$$un \quad = un_e + \varepsilon_u. \tag{A.2}$$

Ces coordonnées sont notées $\mathtt{un}$ lors leur utilisation en indices de boucle. Par ailleurs, les boucles externes utilisées pour la distribution en *threads* lors d'une parallélisation massive sur GPU, apparaissent en encadré : $\boxed{\text{for } \mathtt{un}}$.

1.1 Modélisation *ray-driven*

Pour calculer les éléments H_{ij}, ce modèle est basé sur le lancer du rayon partant de la source et passant par le centre du détecteur i. Plusieurs algorithmes existent pour calculer l'intersection du rayon i avec les voxels j qu'il rencontre. Ils diffèrent principalement par la manière de parcourir le volume le long de ce rayon : (i) boucle fonction des intersections du rayon avec la grille du volume pour $\mathcal{P}_{Siddon}$ (ii) boucle avec un pas constant sur la droite définie par le rayon pour $\mathcal{P}_{Regular}$, et (iii) boucle sur un axe dit « majeur » ($\vec{x}$ ou $\vec{y}$) pour $\mathcal{P}_{Joseph}$.

$\mathcal{P}_{Siddon}$: Projecteur de Siddon

Le projecteur de Siddon [Sidon 1985] définit la contribution $H_{i,j}$ d'un voxel $V_{\mathtt{xn,yn,zn}}$ dans le détecteur $D_{\mathtt{un,vn,phi}}$ comme proportionnelle à la longueur d'intersection $l_{V,R}$ du parallélépipède élémentaire délimitant le voxel avec le rayon $R_{\mathtt{un,vn,phi}}$ le traversant comme illustré dans le cas 2D en Fig. A.2 (a) :

$$\mathcal{P}_{Siddon}(\mathtt{un,vn,phi}) = \sum_{\mathtt{xn,yn,zn}} l_{V,R}(\mathtt{un,vn,phi;xn,yn,zn}) * f(\mathtt{xn,yn,zn}). \tag{A.3}$$

$\mathcal{P}_{Regular}$: Projecteur *Regular Sampling*

Le projecteur *Regular Sampling* intègre le volume linéairement le long du rayon comme illustré en Fig. A.2 (b) :

$$\mathcal{P}_{Regular}(\mathtt{un,vn,phi}) = \delta_k \sum_{\mathtt{k}} f(M_{\mathtt{k}}) = \delta_k \sum_{\mathtt{k}} f(xn_{\mathtt{k}}, yn_{\mathtt{k}}, zn_{\mathtt{k}}), \tag{A.4}$$

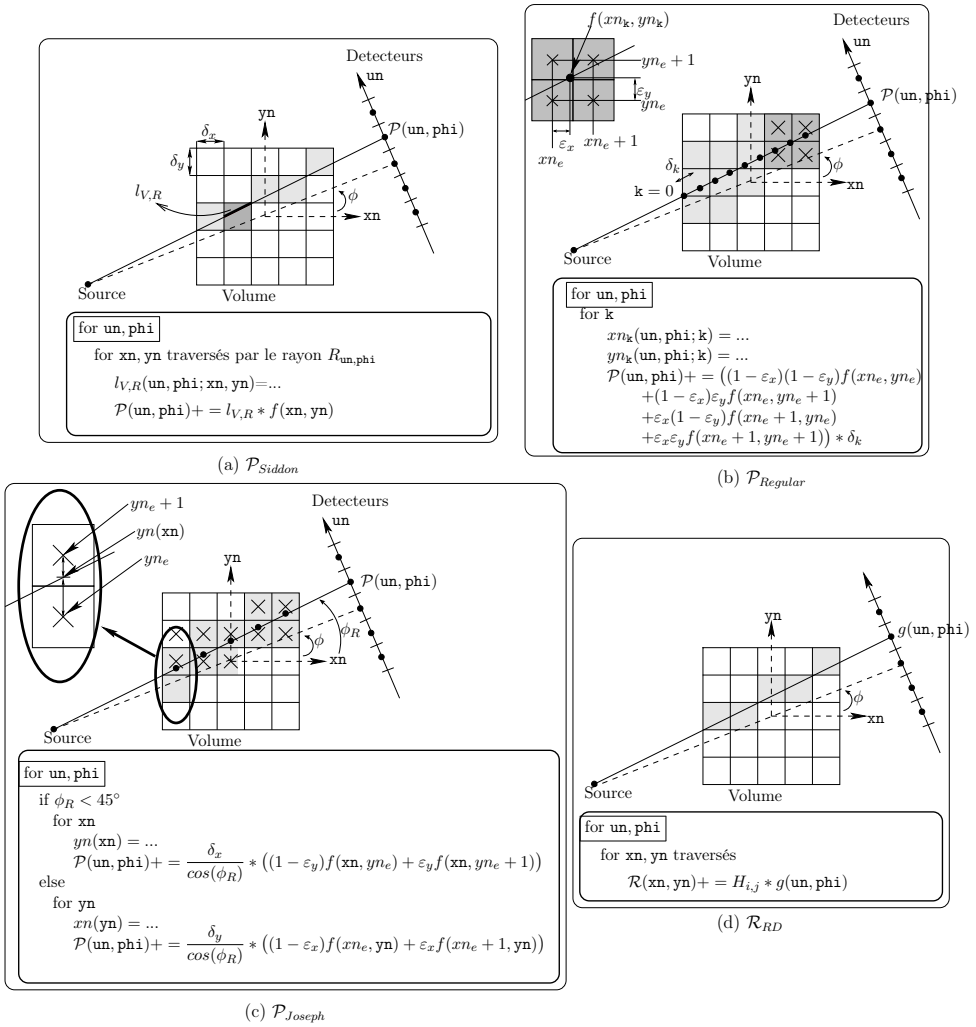
avec $M_{\mathtt{k}}$, les échantillons du volume le long du rayon $R_{\mathtt{un,vn,phi}}$, et δ_k la longueur physique du pas d'échantillonnage. La valeur de $f(xn_{\mathtt{k}}, yn_{\mathtt{k}})$ est ensuite calculée par interpolation bilinéaire dans le cas 2D et par interpolation trilinéaire dans le cas 3D. H_{ij} correspond ainsi à ces coefficients d'interpolation.

$\mathcal{P}_{Joseph}$: Projecteur de Joseph

L'algorithme de Joseph [Joseph 1982] calcule pour chaque rayon $R_{\mathtt{un,vn,phi}}$ l'intégrale linéaire du volume selon l'axe le moins incliné par rapport au rayon, appelé axe

majeur comme illustré sur la figure A.2 (c). Si l'angle $\phi_R = (\overrightarrow{R}_{\mathtt{un,vn,phi}}, \overrightarrow{\mathtt{xn}})$ est inférieur à 45° alors l'axe majeur est l'axe $\overrightarrow{\mathtt{xn}}$ sinon l'axe $\overrightarrow{\mathtt{yn}}$; l'autre axe est appelé mineur. Pour un angle `phi` où `xn` est l'axe majeur, ce projecteur a pour expression :

$$\mathcal{P}_{Joseph}(\mathtt{un},\mathtt{vn},\mathtt{phi}) = \frac{\delta_x}{cos(\phi_R)} \sum_{\mathtt{xn}} f\big(\mathtt{xn}, yn(\mathtt{un},\mathtt{phi};\mathtt{xn}), zn(\mathtt{un},\mathtt{vn},\mathtt{phi};\mathtt{xn})\big). \tag{A.5}$$



(a) $\mathcal{P}_{Siddon}$

(b) $\mathcal{P}_{Regular}$

(c) $\mathcal{P}_{Joseph}$

(d) $\mathcal{R}_{RD}$

FIGURE A.2 – Algorithmes 2D de projection : $\mathcal{P}_{Siddon}$ (a), $\mathcal{P}_{Regular}$ (b) et $\mathcal{P}_{Joseph}$ (c) ; Algorithme 2D de rétroprojection *ray-driven* $\mathcal{R}_{RD}$ (d).

Une interpolation linéaire selon l'axe mineur est effectuée (axe yn sur la figure A.2-c). Dans le cas 3D, une interpolation bilinéaire est effectuée sur le plan orthogonal à l'axe majeur. H_{ij} correspond ainsi aux coefficients d'interpolation associés.

$\mathcal{R}_{RD}$: Rétroprojecteur *ray-driven*

Les rétroprojecteurs *ray-driven* duaux des projecteurs $\mathcal{P}_{Siddon}$, $\mathcal{P}_{Regular}$ et $\mathcal{P}_{Joseph}$, déposent le long de chaque rayon la valeur $g(\mathtt{un}, \mathtt{vn}, \mathtt{phi})$ dans les voxels rencontrés. Ils possèdent donc une boucle principale sur les rayons $\mathrm{R}_{\mathtt{un,vn,phi}}$ comme illustré en Fig. A.2 (d) :

$$\mathcal{R}_{RD}(\mathtt{xn}, \mathtt{yn}, \mathtt{zn}) = \sum_{\mathtt{un,vn,phi}} \sum_{\mathtt{xn,yn,zn}} h(\mathtt{xn}, \mathtt{yn}, \mathtt{zn}; \mathtt{un}, \mathtt{vn}, \mathtt{phi}) * g(\mathtt{un}, \mathtt{vn}, \mathtt{phi}) \quad \text{(A.6)}$$

1.2 Modélisation *voxel-driven*

Pour calculer les éléments H_{ij}, le modèle *voxel driven* se base sur le lancer du rayon partant de la source et passant par le centre du voxel j. La sensibilité du détecteur i est alors fonction des coordonnées un, vn de ce rayon et correspond à des coefficients d'interpolation.

P_{VDL} : Projecteur *Voxel driven* avec Interpolation bilinéaire

Pour chaque angle de projection phi, à partir des coordonnées un, vn correspondant à la projection du centre du voxel xn, yn, zn sur le plan de détecteurs, une interpolation des projections est réalisée :

$$\begin{aligned}\mathcal{P}_{VDL}(\mathtt{un}, \mathtt{vn}, \mathtt{phi}) \quad = \quad & \sum_{\mathtt{xn,yn,zn}} h_{interp}\big(un(\mathtt{phi}; \mathtt{xn}, \mathtt{yn}), vn(\mathtt{phi}; \mathtt{xn}, \mathtt{yn}, \mathtt{zn}); \mathtt{un}, \mathtt{vn}\big) * \\ & f(\mathtt{xn}, \mathtt{yn}, \mathtt{zn}),\end{aligned} \quad \text{(A.7)}$$

avec $h_{interp}(un, vn; \mathtt{un}, \mathtt{vn})$ les coefficients d'interpolation. Parmi les techniques d'interpolation les plus utilisées, nous pouvons citer :

- Interpolation bilinéaire, entre les quatre détecteurs les plus proches du point $(un, vn, \mathtt{phi})$ sur la rangée de projection $(un_e, vn_e, \mathtt{phi})$, $(un_e + 1, vn_e, \mathtt{phi})$, $(un_e, vn_e + 1, \mathtt{phi})$ et $(un_e + 1, vn_e + 1, \mathtt{phi})$, comme illustré en Fig. A.2(d) ;
- Plus proches voisins (NN : Nearest Neighborhood). La valeur de la projection est celle correspondante au détecteur de coordonnées entières les plus proches de $un(\mathtt{xn}, \mathtt{yn}, \mathtt{phi})$ et $vn(\mathtt{xn}, \mathtt{yn}, \mathtt{phi})$.

$\mathcal{R}_{VDL}$: Rétroprojecteur *Voxel Driven* avec interpolation bi-Linéaire

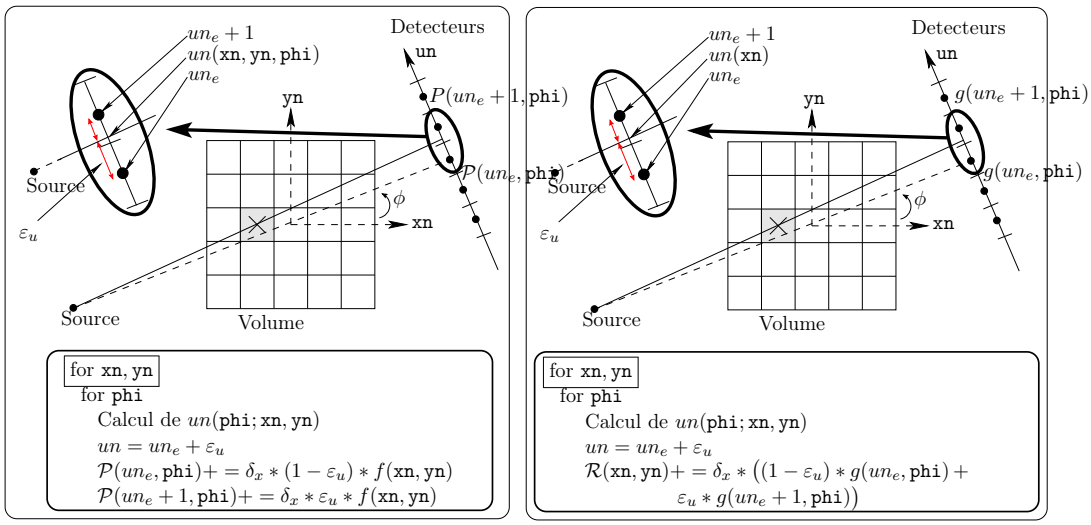
Le rétroprojecteur voxel-driven avec interpolation bilinéaire $\mathcal{R}_{VDL}$, dual de $\mathcal{P}_{VDL}$, cherche pour chaque voxel (xn, yn, zn), ses $N_{\mathtt{phi}}$ projections correspondantes $g_{interp}(un, vn, \mathtt{phi})$

obtenues par interpolation dans le plan de détecteurs, pour ensuite les sommer selon `phi` :

$$\mathcal{R}_{VDL}(\mathtt{xn},\mathtt{yn},\mathtt{zn}) = \sum_{\mathtt{phi}} g_{interp}\big(un(\mathtt{phi};\mathtt{xn},\mathtt{yn}), vn(\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}), \mathtt{phi}\big) \quad \text{(A.8)}$$

Les projections $g_{interp}\big(un(\mathtt{phi};\mathtt{xn},\mathtt{yn}), vn(\mathtt{phi};\mathtt{xn},\mathtt{y},\mathtt{zn}), \mathtt{phi}\big)$ sont obtenues après interpolation bilinéaire :

$$\begin{array}{rcll} g_{interp}(un, vn, \mathtt{phi}) & = & (1-\varepsilon_u)*(1-\varepsilon_v) & * \quad g(un_e, vn_e, \mathtt{phi}) \\ & + & \varepsilon_u*(1-\varepsilon_v) & * \quad g(un_e+1, vn_e, \mathtt{phi}) \\ & + & (1-\varepsilon_u)*\varepsilon_v & * \quad g(un_e, vn_e+1, \mathtt{phi}) \\ & & \varepsilon_u*\varepsilon_v & * \quad g(un_e+1, vn_e+1, \mathtt{phi}). \end{array} \quad \text{(A.9)}$$



(a) $\mathcal{P}_{VDL}$ (b) $\mathcal{R}_{VDL}$

FIGURE A.3 – Algorithmes de projection $\mathcal{P}_{VDL}$ (a) et rétroprojection $\mathcal{R}_{VDL}$ (b) *voxel driven* avec Interpolation linéaire dans le cas 2D

1.3 Modélisation *distance driven*

Contrairement aux modèles *voxel-driven* ou *ray-driven*, la paire *distance-driven* [De Man 2004] ne modélise par les voxels ou les détecteurs uniquement par leur centre mais par leurs limites dans le plan orthogonal à l'axe majeur (voir définition en 1.1). Les limites des voxels et des détecteurs dans ce plan commun est utilisé à la fois par le projecteur et le rétroprojecteur. Il permet de définir ainsi une paire *matched*.

Les équations qui suivent sont exprimées dans le cas 2D où les limites des pixels $P_{\texttt{xn,yn}}$ et des détecteurs $D_{\texttt{un,phi}}$ sont placées dans un axe commun comme illustré sur la Fig. A.4. Par ailleurs, elles correspondent uniquement aux angles ϕ_R définissant `xn` comme axe majeur. Pour les autres angles définissant `yn` comme axe majeur, les équations s'obtiennent en permutant le rôle des coordonnées `xn` et `yn`. A noter que pour les deux opérateurs, le même facteur d'échelle $\frac{\delta_x}{cos(\phi_R)}$ que pour $\mathcal{P}_{Joseph}$ est utilisé pour prendre en compte la longueur de traversée plus ou moins grande du rayon dans le voxel, fonction de l'angle ϕ_R entre le rayon et l'axe majeur de la grille cartésienne du volume.

$\mathcal{P}_{DD}$: Projecteur *Distance Driven*

Pour un angle ϕ_R définissant `xn` comme axe majeur, le projecteur est défini dans le cas 2D par :

$$\mathcal{P}_{DD}(\texttt{un},\texttt{phi}) = \frac{\delta_x}{cos(\phi_R)} * \sum_{\texttt{xn}} \frac{1}{yn_H(\texttt{xn}) - yn_B(\texttt{xn})} * \int_{yn_B(\texttt{xn})}^{yn_H(\texttt{xn})} f(\texttt{xn}, yn')\, dyn', \tag{A.10}$$

avec $yn_B(\texttt{un},\texttt{phi};\texttt{xn})$ et $yn_H(\texttt{un},\texttt{phi};\texttt{xn})$, les coordonnées yn des limites du détecteur $D_{\texttt{un,phi}}$ sur l'axe « commun » orthogonal à l'axe majeur.

$\mathcal{R}_{DD}$: Rétroprojecteur *Distance Driven*

Pour un angle ϕ_R définissant `xn` comme axe majeur, le rétroprojecteur, dual de P_{DD}, est défini dans le cas 2D par :

$$\mathcal{R}_{DD}(\texttt{xn},\texttt{yn}) = \frac{\delta_x}{cos(\phi_R)} * \sum_{\texttt{phi}} \frac{1}{un_H(\texttt{xn}) - un_B(\texttt{xn})} * \int_{un_B(\texttt{xn})}^{un_H(\texttt{xn})} g(un', \texttt{phi})\, dun', \tag{A.11}$$

avec $un_B(\texttt{phi};\texttt{xn},\texttt{yn})$ et $un_H(\texttt{phi};\texttt{xn},\texttt{yn})$, les coordonnées un des limites du pixel $P_{xn,yn}$ sur l'axe « commun » orthogonal à l'axe majeur.

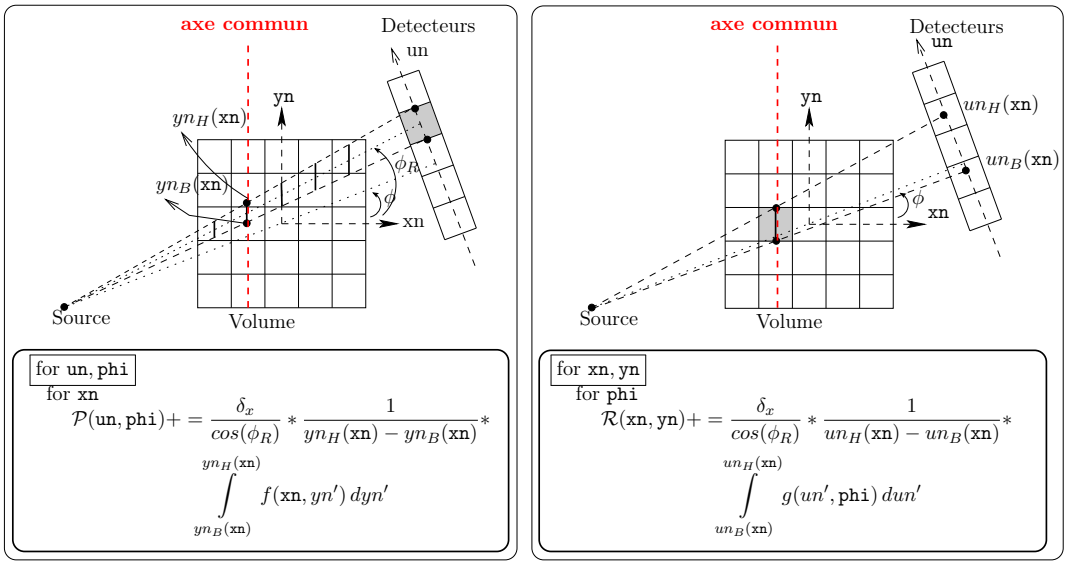


FIGURE A.4 – Algorithmes de projection $\mathcal{P}_{DD}$ (gauche) et rétroprojection *distance-driven* $\mathcal{R}_{DD}$ (droite) dans le cas 2D.

1.4 Modélisation *separable footprint*

Le modèle à empreinte séparable [Long 2010] définit la contribution $H_{i,j}$ comme proportionnelle à une empreinte $F(\mathtt{un},\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn})$ du voxel $V_{\mathtt{xn,yn,zn}}$ sur le détecteur $D_{\mathtt{un,vn,phi}}$. Cette empreinte F pour expression :

$$F(\mathtt{un},\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) \quad = \quad \int\limits_{D_{\mathtt{un,vn}}} e(un', vn')[\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}]\, dun'\, dvn', \tag{A.12}$$

avec $e(un', vn')[\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}]$, la projection du volume élémentaire $V_{\mathtt{xn,y,zn}}$ sur le plan de détecteurs à l'angle $\mathtt{phi}$. Le calcul de l'intégrale de $e(un', vn')[\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}]$ est approchée par une séparation en composantes transverse $F_{trans}(\mathtt{un},\mathtt{phi};\mathtt{xn},\mathtt{yn})$ et axiale $F_{ax}(\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn})$:

$$\begin{aligned} F(\mathtt{un},\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) \quad = \quad & l_\psi(\mathtt{phi};\mathtt{xn},\mathtt{yn}) * l_\theta(\mathtt{un},\mathtt{vn}) * \\ & \int\limits_{\mathtt{un}-0.5}^{\mathtt{un}+0.5} e_{trans}(un')[\mathtt{phi};\mathtt{xn},\mathtt{yn}]\, dun' \; * \\ & \int\limits_{\mathtt{vn}-0.5}^{\mathtt{vn}+0.5} e_{ax}(vn')[\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}]\, dvn', \end{aligned} \tag{A.13}$$

avec $l_\psi(\mathtt{phi};\mathtt{xn},\mathtt{yn})$ et $l_\theta(\mathtt{un},\mathtt{vn})$ des facteurs d'échelle, et $e_{trans}(un')[\mathtt{phi};\mathtt{xn},\mathtt{yn}]$ et $e_{ax}(vn')[\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}]$, les projections du voxel $V_{\mathtt{xn,yn,zn}}$ sur les axes transverse $\vec{un}$

et axial $\vec{vn}$. Ces empreintes sont modélisées par une fonction trapézoïdale en transverse et rectangulaire en axial dont les sommets correspondent aux projections des sommets du voxel. Pour une modélisation plus précise, une forme trapézoïdale en axial peut être préférée même si en pratique, elle n'apporte pas une amélioration notable de la qualité de reconstruction [Long 2010].

$\mathcal{P}_{SF}$: Projecteur *Separable Footprint*

Le projecteur à empreinte séparable est défini par :

$$\begin{aligned}\mathcal{P}_{SF}(\mathtt{un},\mathtt{vn},\mathtt{phi}) &= \sum_{\mathtt{xn}}\sum_{\mathtt{yn}}\sum_{\mathtt{zn}} F(\mathtt{un},\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) f(\mathtt{xn},\mathtt{yn},\mathtt{zn}) \\ &= l_\theta(\mathtt{un},\mathtt{vn}) \sum_{\mathtt{xn}}\sum_{\mathtt{yn}} l_\psi(\mathtt{phi};\mathtt{xn},\mathtt{yn}) F_{trans}(\mathtt{un},\mathtt{phi};\mathtt{xn},\mathtt{yn}) * \\ &\quad \sum_{\mathtt{zn}} F_{ax}(\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) f(\mathtt{xn},\mathtt{yn},\mathtt{zn}).\end{aligned} \tag{A.14}$$

La Fig. A.5 (haut) illustre la projection d'un rayon $R_{\mathtt{un},\mathtt{vn},\mathtt{phi}}$ à travers le volume sur le plan de détecteurs. Lors d'une parallélisation sur GPU, elle pourra être associée à un *thread*. Pour chaque pixel $(\mathtt{xn},\mathtt{yn})$ traversé par le rayon dans le plan transverse, trois étapes de calcul apparaissent :

1. Calcul de l'empreinte transverse $F'_{trans}(\mathtt{un},\mathtt{phi};\mathtt{xn},\mathtt{yn}) = l_\psi * F_{trans}$;
2. Boucle en $\mathtt{zn}$ sur les voxels dont l'empreinte axiale est non nulle, $\mathcal{P}'_{ax} = \sum_{\mathtt{zn}_{\min}}^{\mathtt{zn}_{\max}} F_{ax}(\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) f(\mathtt{xn},\mathtt{yn},\mathtt{zn})$;
3. Accumulation, $\mathcal{P}(\mathtt{un},\mathtt{vn},\mathtt{phi}) += l_\theta * F'_{trans}(\mathtt{un},\mathtt{phi};\mathtt{xn},\mathtt{yn}) * \mathcal{P}'_{ax}$.

$\mathcal{R}_{SF}$: Rétroprojecteur *Separable Footprint*

Le rétroprojecteur à empreinte séparable est le dual du projecteur $\mathcal{P}_{SF}$. Les données de projections $g(\mathtt{un},\mathtt{vn},\mathtt{phi})$ contribuant au voxel $V_{\mathtt{xn},\mathtt{yn},\mathtt{zn}}$ sont accumulées après pondération par l'empreinte du voxel dans les détecteurs $D_{\mathtt{un},\mathtt{vn},\mathtt{phi}}$:

$$\begin{aligned}\mathcal{R}_{SF}(\mathtt{xn},\mathtt{yn},\mathtt{zn}) &= \sum_{\mathtt{phi}}\sum_{\mathtt{vn}}\sum_{\mathtt{un}} F(\mathtt{un},\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) g(\mathtt{un},\mathtt{vn},\mathtt{phi}) \\ &= \sum_{\mathtt{phi}}\sum_{\mathtt{un}} l_\psi(\mathtt{phi};\mathtt{xn},\mathtt{yn}) F_{trans}(\mathtt{un},\mathtt{phi};\mathtt{xn},\mathtt{yn}) * \\ &\quad \sum_{\mathtt{vn}} l_\theta(\mathtt{un},\mathtt{vn}) F_{ax}(\mathtt{vn},\mathtt{phi};\mathtt{xn},\mathtt{yn},\mathtt{zn}) g(\mathtt{un},\mathtt{vn},\mathtt{phi}).\end{aligned} \tag{A.15}$$

La Fig. A.5 (bas) illustre ainsi la rétroprojection d'un voxel $V_{\mathtt{xn},\mathtt{yn},\mathtt{zn}}$ qui pourra être associé à un *thread* lors d'une parallélisation GPU. Les coordonnées $\mathtt{un}$ et $\mathtt{vn}$ pour lesquelles l'empreinte respectivement transverse et axiale est non nulle déterminent

les bornes $[\mathtt{un}_{\mathtt{min}}, \mathtt{un}_{\mathtt{max}}]$ et $[\mathtt{vn}_{\mathtt{min}}, \mathtt{vn}_{\mathtt{max}}]$ des boucles sur `un` et `vn`. Trois étapes de calcul apparaissent sous la double boucle en `phi` et en $\mathtt{un} \in [\mathtt{un}_{\mathtt{min}}, \mathtt{un}_{\mathtt{max}}]$:

1. Boucle en `vn`, $\mathcal{R}_{ax,\mathtt{phi}} = \sum_{\mathtt{vn}_{\mathtt{min}}}^{\mathtt{vn}_{\mathtt{max}}} F_{ax}(\mathtt{vn}, \mathtt{phi}; \mathtt{xn}, \mathtt{yn}, \mathtt{zn}) * g(\mathtt{un}, \mathtt{vn}, \mathtt{phi})$;
2. Calcul de l'empreinte transverse $F'_{trans}(\mathtt{un}, \mathtt{phi}; \mathtt{xn}, \mathtt{yn}) = l_\psi * F_{trans}$;
3. Accumulation, $\mathcal{R}(\mathtt{xn}, \mathtt{yn}, \mathtt{phi}) += F'_{trans}(\mathtt{un}, \mathtt{phi}; \mathtt{xn}, \mathtt{yn}) * \mathcal{R}_{ax,\mathtt{phi}}$.

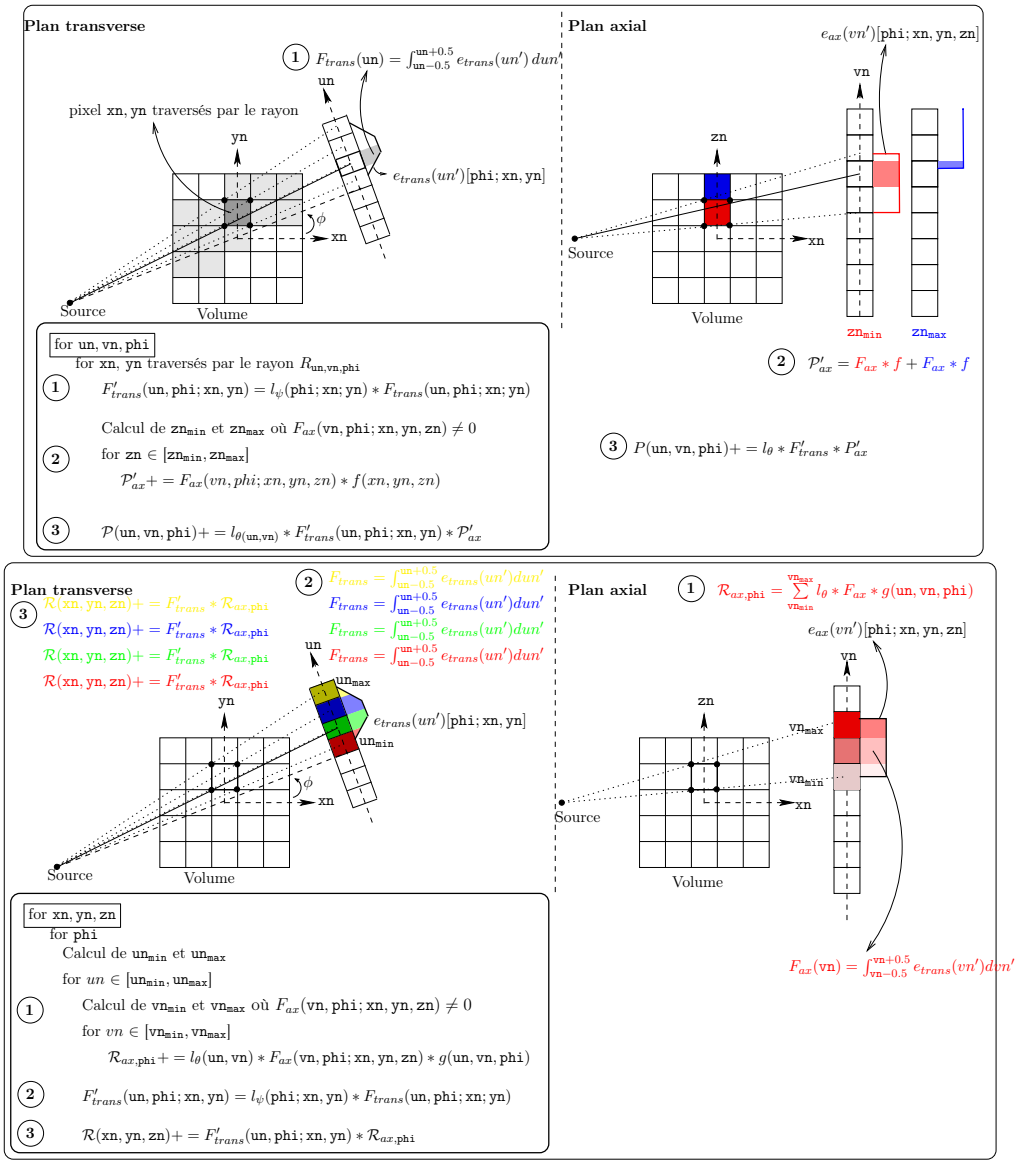


FIGURE A.5 – Algorithmes de projection $\mathcal{P}_{SF}$ (haut) et rétroprojection $\mathcal{R}_{SF}$ (bas) à empreinte séparable dans le cas 3D.

Annexe B

Serveurs et cartes accélératrices utilisées

Cartes GPU

Les différentes cartes GPU utilisées sont décrites dans le Tab. B.1.

Nom	**Carte**	**Architecture** [Comp. Cap.]	**Année**	**N$_{coeurs}$**	**Freq.** (GHz)	**TFlops**	**Mem** (Go)
Cartes connectées à un PC							
GPU$_{tesla-G80}$	GTS 8800	Tesla [1.0]	2006	96	1.2	0.23	0.64
GPU$_{tesla}$	Tesla C1060	Tesla [1.3]	2009	240	1.30	0.62	4
GPU$_{fermi}$	Tesla C2050	Fermi [2.0]	2011	448	1.15	1.03	4
GPU$_{kepler}$	Tesla K80	Kepler [3.7]	2014	2496	0.87	4.37	12
GPU$_{maxwell}$	Titan X	Maxwell [5.2]	2015	3072	1.07	6.60	12
GPU$_{pascal-1080Ti}$	1080Ti	Pascal [6.1]	2017	3584	1.48	10.6	11
GPU$_{volta-Titan}$	Titan V	Volta [7.0]	2017	5120	1.45	14.9	12
GPU$_{volta-V100a}$	V100-NVlink	Volta [7.0]	2017	5120	1.53	15.7	32
GPU$_{volta-V100b}$	V100-PCIe	Volta [7.0]	2017	5120	1.38	14.1	32
GPU$_{turing}$	RTX 2080 Ti	Volta [7.5]	2018	4352	1.63	14.2	11
Systèmes embarqués							
GPU$_{pascal-TX2}$	Jetson TX2	Pascal [6.2]	2016	256	1.46	0.75	8
GPU$_{volta-Xavier}$	Jetson Xavier	Volta [7.2]	2018	512	1.38	1.41	32

Table B.1 – Cartes GPU utilisées.

Serveurs multi-GPU

Les serveurs multi-GPU utilisés sont principalement ceux du laboratoire L2S : S$_{bebe}$ et S$_{bebe2}$. Mais afin de participer à la mutualisation des ressources de calcul, ces tests sont à présent effectués sur les ressources externes à la disposition du L2S (voir page web du pôle calcul scientifique du L2S). Le L2S a ainsi accès sur le plateau de Saclay aux mésocentres Moulon et Saclay-IA, et au niveau national à la plateforme de calcul Jean Zay. Les caractéristiques des noeuds multi-GPU utilisés sont décrites dans le Tab. B.2.

Nom	Mésocentre	Noeud	Cartes GPU	Bus	
				CPU-GPU	inter-GPU
$\texttt{S}_{bebe}$	L2S	machine mono-noeud bebe (2010)	8 $\texttt{GPU}_{tesla}$ ou 8 $\texttt{GPU}_{maxwell}$	PCIe gen2	PCIe gen2
$\texttt{S}_{bebe2}$	L2S/GEOPS	machine mono-noeud bebe2 (2016)	10 $\texttt{GPU}_{maxwell}$	PCIe gen3	PCIe gen3
$\texttt{S}_{zay}$	IDRIS	gpu_p2 de la machine Jean Zay	8 $\texttt{GPU}_{volta-V100a}$	PCIe gen3	NVLink
$\texttt{S}_{ruche}$	Moulon	gpu[1-5] de la machine ruche	4 $\texttt{GPU}_{volta-V100b}$	PCIe gen3	PCIe gen3
$\texttt{S}_{lab-ia}$	Saclay-IA	n[101-102] de la machine Lab-IA	4 $\texttt{GPU}_{volta-V100a}$	PCIe gen3	NVLink

TABLE B.2 – Serveurs multi-GPU utilisés.

Cartes FPGA

Les caractéristiques des différentes cartes FPGA utilisées sont décrites dans le Tab. B.3. La carte bas de gamme $\texttt{FPGA}_{Cyclone-V}$ nous a permis d'avoir nos premiers résultats qui ont été poursuivis avec la carte de moyenne gamme $\texttt{FPGA}_{Arria-10}$ à laquelle le laboratoire LAL nous a donné accès. Par ailleurs, nous accédons à la carte haut de gamme $\texttt{FPGA}_{Stratix-10}$ via le *DevCloud* d'Intel.

FPGA	Puce	Carte	Année	Logique (k LE)	DSP		Prix
					$\mathbf{N_{DSP}}$	TFlops	
$\texttt{FPGA}_{Stratix-10}$	SX 2800	Intel PAC	2016 (puce) 2019 (carte)	2 753	5 760	9.2	autour de 10 k €
$\texttt{FPGA}_{Arria-10}$	GX 1150	Attila (Reflex)	2014 (puce) 2015 (carte)	1 150	1 518	1.37	entre 2 et 4 k €
$\texttt{FPGA}_{Cyclone-V}$	5CSEA	DE1 SoC (Terasic)	2014 (carte) 2012 (puce)	85	87	0.035	entre 0.2 et 0.3 k €
$\texttt{FPGA}_{Virtex-2-pro}$		Avnet	2002 (puce)				

TABLE B.3 – Cartes FPGA utilisées.

Processeurs CPU

Des Xeons d'Intel sont utilisés comme CPU de référence lors d'études comparatives :

- $\texttt{CPU}_{xeon-tomo}$ est un mono-xeon composé de 6 coeurs @2.9 Hz (modèle E5-2667) ; utilisé pour la comparaison de performances de la rétroprojection sur CPU, GPU et FPGA (section 3.1.4).
- $\texttt{CPU}_{xeon-bebe2}$ est un bi-xeon composé de 2*14 coeurs @2.6 Ghz (modèle E5-2690V4) ; utilisé pour la comparaison de performances de la convolution spectrale sur serveur multi-GPU (section 4.3.2).

Annexe C

Logiciels de reconstruction tomographique du GPI

Depuis 2008, le GPI développe pour son usage interne ou celui de ses partenaires industrielles, des logiciels de reconstruction tomographique 3D (voir Fig. C.1). La valorisation de ces logiciels s'est faite en fonction du contexte des différents projets menés (TomoX, TomoGPI et TomoBayes) avec une finalité industrielle ou une politique de diffusion libre des logiciels.

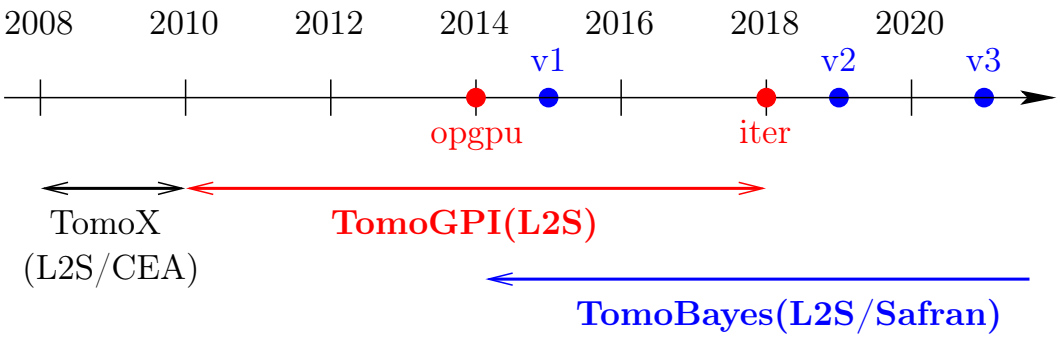


Figure C.1 – Logiciels du GPI pour la reconstruction tomographique.

3.1 Découpage en trois modules

Un logiciel de reconstruction tomographique nécessite trois modules : (i) une librairie de gestion des données et de la géométrie du scanner, (ii) une librairie d'accélération des opérateurs de calcul (projections, rétroprojection, convolution 3D), et (iii) des méthodes itératives de reconstructions faisant appel aux deux libraires de plus bas niveau. Le GPI apporte son expertise sur les deux derniers modules.

3.2 Contexte industriel ou « libre »

TomoX (L2S/CEA)

Ces premiers travaux effectués dans le cadre du projet DIGITEO TomoX (2008-2009) en collaboration avec le L2S et le CEA-List ont permis la mise en oeuvre d'une première version d'accélération sur serveur multi-GPU d'algorithmes bayésiens appliqués sur des données réelles 3D [C_{2010}]. Les opérateurs $\mathcal{P}$ et $\mathcal{R}$ ont été développés en CUDA avec une interface (fonctions mex) leur permettant d'être appelés par les algorithmes itératifs depuis Matlab.

TomoGPI (L2S)

La motivation du GPI pour la diffusion libre de ses logiciels est triple : (i) rendre accessible ses codes dans la philosophie de la reproductibilité numérique des résultats publiés [Stodden 2013], (ii) gagner en visibilité et catalyser les collaborations avec d'autres acteurs académiques ou industriels du domaine, et (iii) offrir un logiciel utilisable par une large communauté d'utilisateurs. Le dernier objectif le plus ambitieux requiert un solide support technique afin d'offrir une qualité de service aux utilisateurs avec notamment un premier module logiciel s'adaptant à plusieurs contextes de reconstruction (configuration du scanner, format des données...), ou l'organisation de formations destinées aux futurs utilisateurs. Le premier module des logiciels du GPI sont adaptés seulement aux jeux de données de ses partenaires industrielles (Safran, Carestream Dental ou CEA-List) et requiert trop d'effort pour s'adapter aux besoins des diverses communautés d'utilisateurs en imagerie médicale ou en CND (source ponctuelle ou synchrotrons). C'est pourquoi le GPI s'est donné comme objectif d'apporter des briques de base à des *toolkits* déjà mis à la disposition tels qu'Astra [Ast 2012] ou RTK [Rit 2014]. Le GPI a ainsi déposé deux codes ouverts : opgpuTomoGPI [L_{2014}] qui contient le code C/CUDA développé par le L2S des opérateurs $\mathcal{P}_{Regular}/\mathcal{R}_{VDL}$ accélérés sur GPU, et iterTomoGPI [L_{2018}], *plugin* Astra en Matlab des méthodes itératives bayésiennes hiérarchiques développées dans le cadre de la thèse de Li Wang et le post-doc de Mircea Dumitru.

TomoBayes (L2S/Safran)

Au cours de la collaboration du L2S avec Safran (2014-2019) pour la reconstruction de pièces aéronautiques, le logiciel TomoBayes [$L_{2015-2019}$] a été développé. Il a été le support des études menées par Thomas Boulay (post-doc) et Camille Chapdelaine (thèse). La première version (v1) déposée en 2015 est constituée d'un logiciel complet avec les trois modules tels que décrits en 3.1. Le deuxième module fait appel à la librairie libre du L2S, opgpuTomoGPI [L_{2014}]. La deuxième version (v2) a apporté : (i) la paire duale de projection et rétroprojection $\mathcal{P}_{DD}/\mathcal{R}_{DD}$ accélérée sur GPU, et (ii) des méthodes itératives bayésiennes intégrant des informations *a priori* sur les pièces à imager.

Bibliographie

[Acero 2018] F. Acero*et al.* French ska white book. arXiv.org/astro-ph, 2018. hal-01686223. (Cité en page 70.)

[Ammanouil 2019] R. Ammanouil, A. Ferrari, D. Mary, C. Ferrari et F. Loi. *A parallel multispectral deconvolution.* MNRAS, 2019. arXiv:1905.08468. (Cité en page 32.)

[Anderson 2017] Andrew Anderson, Aravind Vasudevan, Cormac Keane et David Gregg. *Low-memory GEMM-based convolution algorithms for deep neural networks*, 2017. (Cité en page 49.)

[Arcadu 2016] Filippo Arcadu, Marco Stampanoni et Federica Marone. *On the crucial impact of the coupling projector-backprojector in iterative tomographic reconstruction.* arXiv preprint arXiv :1612.05515, 2016. (Cité en page 37.)

[Ast 2012] *ASTRA Toolbox*, 2012. www.astra-toolbox.com. (Cité en pages 41, 62 et 92.)

[Basu 2006] Samit Basu et Bruno De Man. *Branchless distance driven projection and backprojection - art. no. 60650Y.* Proceedings of SPIE - The International Society for Optical Engineering, 02 2006. (Cité en pages 42 et 45.)

[Beck 2009] Amir Beck et Marc Teboulle. *A fast iterative shrinkage-thresholding algorithm for linear inverse problems.* SIAM journal on imaging sciences, vol. 2, no. 1, pages 183–202, 2009. (Cité en page 25.)

[Belle 2017] J. V. Belle, R. Armstrong et J. Gain. *Accelerated Deconvolution.* JAI, 2017. 10.1142/S225117171750009X. (Cité en page 73.)

[Biguri 2016] Ander Biguri, Manjit Dosanjh, Steven Hancock et Manuchehr Soleimani. *TIGRE : a MATLAB-GPU toolbox for CBCT image reconstruction.* Biomedical Physics & Engineering Express, vol. 2, no. 5, page 055010, sep 2016. (Cité en pages 38 et 62.)

[Bollaert 2008] Thomas Bollaert. Catapult synthesis : A practical introduction to interactive c synthesis, pages 29–52. Springer Netherlands, Dordrecht, 2008. (Cité en page 11.)

[Boulay 2013] Thomas Boulay. *Algorithm development for NCTR function - Parallel Computing application on GPU cards.* Theses, Université Paris Sud - Paris XI, Octobre 2013. (Cité en page 26.)

[Boyd 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato et Jonathan Eckstein. *Distributed optimization and statistical learning via the alternating direction method of multipliers.* Foundations and Trends® in Machine Learning, vol. 3, no. 1, pages 1–122, 2011. (Cité en page 25.)

[Broekema 2018] P. Chris Broekema, J. Jan David Mol, R. Nijboer, A.S. van Amesfoort, M.A. Brentjens, G. Marcel Loose, W.F.A. Klijn et J.W. Romein. *Cobalt : A GPU-based correlator.* ASTRON COMPUT, 2018. arXiv:1801.04834. (Cité en page 73.)

[Cabral 1994] Brian Cabral, Nancy Cam et Jim Foran. *Accelerated Volume Rendering and Tomographic Reconstruction Using Texture Mapping Hardware.* In Proceedings of the 1994 Symposium on Volume Visualization, VVS '94, page 91–98, New York, NY, USA, 1994. Association for Computing Machinery. (Cité en page 40.)

[Canis 2011] Andrew Canis, Jongsok Choi, Mark Aldham, Victor Zhang, Ahmed Kammoona, Jason H. Anderson, Stephen Brown et Tomasz Czajkowski. *LegUp : High-Level Synthesis for FPGA-Based Processor/Accelerator Systems.* In Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays, FPGA '11, page 33–36, New York, NY, USA, 2011. Association for Computing Machinery. (Cité en page 11.)

[Chambolle 2011] Antonin Chambolle et Thomas Pock. *A first-order primal-dual algorithm for convex problems with applications to imaging.* Journal of Mathematical Imaging and Vision, vol. 40, no. 1, pages 120–145, 2011. (Cité en page 25.)

[Chambolle 2015] Antonin Chambolle et Ch Dossal. *On the convergence of the iterates of the "Fast Iterative Shrinkage/Thresholding Algorithm".* Journal of Optimization theory and Applications, vol. 166, no. 3, pages 968–982, 2015. (Cité en page 25.)

[Chapdelaine 2019] Camille Chapdelaine. *Bayesian iterative reconstruction methods for 3D X-ray Computed Tomography.* Theses, Université Paris-Saclay, Avril 2019. (Cité en pages 28, 29, 30, 38 et 39.)

[Chapman 2007] Barbara Chapman, Gabriele Jost et Ruud van der Pas. Using openmp : Portable shared memory parallel programming. MIT, 2007. (Cité en page 10.)

[Chen 2012] Jianwen Chen, Jason Cong, Ming Yan et Yi Zou. *FPGA-Accelerated 3D Reconstruction Using Compressive Sensing.* In Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, FPGA '12, page 163–166, New York, NY, USA, 2012. Association for Computing Machinery. (Cité en page 45.)

[Chen 2015] Long Chen. *Iterative reconstruction methods for the reduction of metal artifact and dose in dental CT.* Theses, Université Paris Sud - Paris XI, F ?rier 2015. (Cité en page 29.)

[Chetlur 2014] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro et Evan Shelhamer. *cuDNN : Efficient Primitives for Deep Learning*, 2014. (Cité en page 49.)

[Choi 2016] Y. Choi et J. Cong. *Acceleration of EM-Based 3D CT Reconstruction Using FPGA.* IEEE Transactions on Biomedical Circuits and Systems, vol. 10, no. 3, pages 754–767, June 2016. (Cité en pages 45 et 46.)

[Chou 2011a] Cheng-Ying Chou, Yi-Yen Chuo, Yukai Hung et Weichung Wang. *A fast forward projection using multithreads for multirays on GPUs in medical*

image reconstruction. Medical Physics, vol. 38, no. 7, pages 4052–4065, 2011. (Cité en page 43.)

[Chou 2011b] Cheng-Ying Chou, Yi-Yen Chuo, Yukai Hung et Weichung Wang. *A fast forward projection using multithreads for multirays on GPUs in medical image reconstruction.* Medical Physics, vol. 38, no. 7, pages 4052–4065, 2011. (Cité en page 41.)

[Chouzenoux 2010] Emilie Chouzenoux. *Majorize-Minimize algorithms for stepsize determination. Application to the resolution of inverse problems.* Theses, Ecole Centrale de Nantes (ECN), D ?embre 2010. (Cité en page 22.)

[Chu 2013] Ning Chu. *Bayesian approach in acoustic source localization and imaging.* Theses, Université Paris Sud - Paris XI, Novembre 2013. (Cité en page 26.)

[Coussy 2008] Philippe Coussy, Cyrille Chavet, Pierre Bomel, Dominique Heller, Eric Senn et Eric Martin. Gaut : A high-level synthesis tool for dsp applications, pages 147–169. Springer Netherlands, Dordrecht, 2008. (Cité en page 11.)

[Cárcamo 2018] M. Cárcamo, P.E. Román, S. Casassus, V. Moral et F.R. Rannou. *Multi-GPU for radio astronomy.* Astron. Comput., 2018. arXiv:1703.02920. (Cité en page 73.)

[Dabbech 2018] A Dabbech, A Onose, A Abdulaziz, R A Perley, O M Smirnov et Y Wiaux. *Convex optimization from VLA data.* MNRAS, 2018. arXiv:1710.08810. (Cité en page 32.)

[De Man 2002] Bruno De Man et S Basu. *Distance-driven projection and backprojection.* In Nuclear Science Symposium Conference Record, 2002 IEEE, volume 3, pages 1477–1480. IEEE, 2002. (Cité en page 37.)

[De Man 2004] Bruno De Man et Samit Basu. *Distance-driven projection and backprojection in three dimensions.* Physics in medicine and biology, vol. 49, no. 11, page 2463, 2004. (Cité en pages 36 et 84.)

[Demoment 1989] G. Demoment. *Image reconstruction and restoration : overview of common estimation structures and problems.* IEEE Trans. Acoust. Speech Signal Process., vol. 37, pages 2024–2036, 1989. (Cité en page 24.)

[Dittmann 2017] Jonas Dittmann et Randolf Hanke. *Simple and efficient raycasting on modern GPU's read-and-write memory for fast forward projections in iterative CBCT reconstruction.* In The 14th International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, 2017. (Cité en pages 41 et 43.)

[Du 2017] Yi Du, Gongyi Yu, Xincheng Xiang et Xiangang Wang. *GPU accelerated voxel-driven forward projection for iterative reconstruction of cone-beam CT.* Biomedical engineering online, vol. 16, no. 1, page 2, 2017. (Cité en page 42.)

[Dumitru 2016] Mircea Dumitru. *A Bayesian approach for periodic components estimation for chronobiological signals.* Theses, Université Paris-Saclay, Mars 2016. (Cité en page 28.)

[Esser 2009] Ernie Esser. *Applications of Lagrangian-based alternating direction methods and connections to split Bregman.* CAM report, vol. 9, page 31, 2009. (Cité en page 25.)

[Feldkamp 1984] LA Feldkamp, LC Davis et JW Kress. *Practical cone-beam algorithm.* JOSA A, vol. 1, no. 6, pages 612–619, 1984. (Cité en page 36.)

[Florian 2011] Pfanner Florian, Michael Knaup et Marc Kachelrieß. *High performance parallel backprojection on FPGA.* In Fully3D, 2011. (Cité en page 45.)

[Gac 2008] Nicolas Gac, Stéphane Mancini, Michel Desvignes et Dominique Houzet. *High speed 3D tomography on CPU, GPU, and FPGA.* EURASIP Journal on Embedded systems, vol. 2008, page 5, 2008. (Cité en page 45.)

[Gao 2012] Hao Gao. *Fast parallel algorithms for the X-ray transform and its adjoint.* Medical physics, vol. 39, no. 11, pages 7110–7120, 2012. (Cité en pages 40 et 42.)

[Garsden 2015] H. Garsden et al. *LOFAR sparse image reconstruction.* A&A, 2015. arXiv:1406.7242. (Cité en page 32.)

[Goldstein 2009] Tom Goldstein et Stanley Osher. *The split Bregman method for L1-regularized problems.* SIAM journal on imaging sciences, vol. 2, no. 2, pages 323–343, 2009. (Cité en page 25.)

[Han 1999] Guoping Han, Zhengrong Liang et Jiangsheng You. *A fast ray-tracing technique for TCT and ECT studies.* In Nuclear Science Symposium, 1999. Conference Record. 1999 IEEE, volume 3, pages 1515–1518. IEEE, 1999. (Cité en page 40.)

[Hascoet 2015] Julien Hascoet, Jean-Francois Nezan, Andrew Ensor et Benoit Dupont de Dinechin. *Implementation of FFT Algorithm onto MPPA.* In DASIP, 2015. hal-01238833. (Cité en page 74.)

[Högbom 1974] J. A. Högbom. *Aperture synthesis.* A&A Supplement Series, vol. 15, page 417, 1974. (Cité en page 32.)

[Hugo 2014] Benjamin Hugo, Oleg Smirnov, Cyril Tasse et James Gain. *A multi-threaded CPU and GPU-based facet imager*, 2014. Git repository. (Cité en page 73.)

[Idier 2001] Jérôme Idier. Approche bayésienne pour les problèmes inverses. Hermès Science Publications, 2001. (Cité en pages 21 et 23.)

[Jacobs 1998] Filip Jacobs, Erik Sundermann, Bjorn De Sutter, Mark Christiaens et Ignace Lemahieu. *A fast algorithm to calculate the exact radiological path through a pixel or voxel space.* CIT. Journal of computing and information technology, vol. 6, no. 1, pages 89–94, 1998. (Cité en page 40.)

[Joseph 1982] Peter M. Joseph. *An Improved Algorithm for Reprojecting Rays through Pixel Images.* vol. 1, no. 3, pages 192–196, Nov. 1982. (Cité en pages 40 et 80.)

[Khairy 2019] Mahmoud Khairy, Amr G. Wassal et Mohamed Zahran. *A Survey of Architectural Approaches for Improving GPGPU Performance, Programmability and Heterogeneity.* Journal of Parallel and Distributed Computing, vol. 127, pages 65–88, Mai 2019. (Cité en page 17.)

[Khronos 2011] Khronos. *The OpenCL specification v1.1*, 2011. (Cité en page 10.)

[Kim 2012] J. K. Kim, J. A. Fessler et Z. Zhang. *Forward-Projection Architecture for Fast Iterative Image Reconstruction in X-Ray CT.* IEEE Transactions on Signal Processing, vol. 60, no. 10, pages 5508–5518, 2012. (Cité en page 45.)

[Kirk 2016] David B Kirk et W Hwu Wen-Mei. Programming massively parallel processors : A hands-on approach. Morgan kaufmann, 2016. (Cité en page 17.)

[Knaup 2008] M. Knaup et M. Kachelriess. *GPU-Based Parallel-Beam and Cone-Beam GPU Forward- and Backprojection using CUDA.* In MIC, 2008. (Cité en page 41.)

[Lee 2011] Nam-Yong Lee et Yong Choi. *Theoretical Investigation on an Unmatched Backprojector for Iterative Reconstruction in Emission Computed Tomography.* Journal of The Korean Physical Society - J KOREAN PHYS SOC, vol. 59, 08 2011. (Cité en page 37.)

[Leeser 2002] Miriam Leeser, Srdjan Coric, Eric Miller, Haiqian Yu et Marc Trepanier. *Parallel-Beam Backprojection : an FPGA Implementation Optimized for Medical Imaging.* 2002. ACM/SIGDA, 2002. (Cité en page 45.)

[Liu 2017] Rui Liu, Lin Fu, Bruno De Man et Hengyong Yu. *GPU-Based Branchless Distance-Driven Projection and Backprojection.* IEEE Transactions on Computational Imaging, vol. PP, pages 1–1, 03 2017. (Cité en pages 42, 43 et 45.)

[Long 2010] Yong Long, Jeffrey A Fessler et James M Balter. *3D forward and back-projection for X-ray CT using separable footprints.* IEEE transactions on medical imaging, vol. 29, no. 11, pages 1839–1850, 2010. (Cité en pages 36, 85 et 86.)

[Lou 2019] Y. Lou, S. Park, F. Anis, R. Su, A. A. Oraevsky et M. A. Anastasio. *Analysis of the Use of Unmatched Backward Operators in Iterative Image Reconstruction With Application to Three-Dimensional Optoacoustic Tomography.* IEEE Transactions on Computational Imaging, vol. 5, no. 3, pages 437–449, 2019. (Cité en page 37.)

[Mancini 2004] Stéphane Mancini et Nicolas Eveno. *An IIR based 2D adaptive and predictive cache for image processing.* In XIX conference on Design of Circuits and Integrated Systems (DCIS'2004), Bordeaux, France, 2004. (Cité en pages 18 et 46.)

[Martelli 2019] Maxime Martelli. *High-Level Approach for the Acceleration of Algorithms on CPU/GPU/FPGA Heterogeneous Architectures. Application to Radar Qualification and Electromagnetic Listening Systems.* Theses, Université Paris-Saclay, D ?embre 2019. (Cité en pages 18, 26, 46 et 47.)

[Martinez 2018] C. Martinez, J. A. Fessler, M. Desco et M. Abella. *Statistical Image Reconstruction with Sample-Based Beam-Hardening compensation for X-ray CT*. In The Fifth International Conference on Image Formation in X-Ray Computed Tomography, pages 11–14, 2018. (Cité en page 29.)

[Momey 2015] F. Momey, L. Denis, C. Burnier, E. Thiebaut, J. Becker et L. Desbat. *Spline Driven : High Accuracy Projectors for Tomographic Reconstruction From Few Projections*. IEEE Transactions on Image Processing, vol. 24, no. 12, pages 4715–4725, 2015. (Cité en page 38.)

[Morooka 2019] F. Morooka. *Dask description of a deconvolution*. Rapport technique, INSA, 2019. Report & Oral. (Cité en page 71.)

[Nane 2016] Razvan Nane, Vlad Mihai Sima, Christian Pilato, Jongsok Choi, Blair Fort, Andrew Canis, Yu Ting Chen, Hsuan Hsiao, Stephen Dean Brown, Fabrizio Ferrandi, Jason Helge Anderson et Koen Bertels. *A Survey of FPGA HLS Tools*. IEEE T COMPUT AID D, 2016. Open Access. (Cité en page 74.)

[Nguyen 2015] Van-Giang Nguyen et Soo-Jin Lee. *Parallelizing a matched pair of ray-tracing projector and backprojector for iterative cone-beam CT reconstruction*. IEEE Transactions on Nuclear Science, vol. 62, no. 1, pages 171–181, 2015. (Cité en pages 40 et 42.)

[Nuyts 2013] Johan Nuyts, Bruno De Man, Jeffrey A Fessler, Wojciech Zbijewski et Freek J Beekman. *Modelling the physics in the iterative reconstruction for transmission computed tomography*. Physics in medicine and biology, vol. 58, no. 12, page R63, 2013. (Cité en page 29.)

[Offringa 2014] A. R. Offringa, B. McKinley, N. Hurley-Walker, F. H. Briggs, R. B. Wayth, D. L. Kaplan, M. E. Bell, L. Feng, A. R. Neben, J. D. Hughes et et al. *wsclean : radioastronomy imager implementation*. MNRAS, 2014. arXiv:1407.1943. (Cité en page 73.)

[Ongie 2018] Greg Ongie, Naveen Murthy, Laura Balzano et Jeffrey A. Fessler. *A Memory-Efficient Algorithm for Large-Scale Sparsity Regularized Image Reconstruction*. In The Fifth International Conference on Image Formation in X-Ray Computed Tomography, pages 20–23, 2018. (Cité en pages 30 et 38.)

[Palenstijn 2017] Willem Palenstijn, Jeroen Bédorf, Jan Sijbers et Kees Batenburg. *ASTRA toolbox*. ASCI, 2017. open access. (Cité en pages 38, 41 et 61.)

[Park 2014] Hyeong-Gyu Park, Yeong-Gil Shin et Ho Lee. *A fully GPU-based ray-driven backprojector via a ray-culling scheme with voxel-level parallelization for cone-beam CT reconstruction*. Technology in cancer research & treatment, pages tcrt–2012, 2014. (Cité en page 42.)

[Pelcat 2014] M. Pelcat, K. Desnos, J. Heulot, C. Guy, J. Nezan et S. Aridhi. *Preesm : A dataflow-based rapid prototyping framework for simplifying multicore DSP programming*. In 2014 6th European Embedded Design in Education and Research Conference (EDERC), pages 36–40, 2014. (Cité en page 18.)

[Perrot 2016] Gilles Perrot, Stéphane Domas et Raphaël Couturier. *An optimized GPU-based 2D convolution implementation.* Concurrency and Computation : Practice and Experience, vol. 28, no. 16, pages 4291–4304, Novembre 2016. (Cité en page 48.)

[Petrisor 2011] Teodora Petrisor, Eric Lenormand, Remi Barrere et Michel Barreteau. *SpearDE : Plateforme de développement de chaînes de parallélisation sur architectures distribuées hétérogènes.* 11 2011. (Cité en page 18.)

[project SKA 2016] project SKA. *https://www.youtube.com/watch?v=w_q6kB2nCdw*, 2016. (Cité en page 70.)

[Putnam 2008] A. Putnam, D. Bennett, E. Dellinger, J. Mason, P. Sundararajan et S. Eggers. *CHiMPS : A C-level compilation flow for hybrid CPU-FPGA architectures.* In 2008 International Conference on Field Programmable Logic and Applications, pages 173–178, 2008. (Cité en page 11.)

[Radon 1917] J. Radon. *Uber die Bestimmung von Funktionen durch ihre Integralwerte langs gewisser Mannigfaltigkeiten.* Ber. Sachs. Akad. Wissenschaft. Leipzig Math. Phys. Kl., no. 69, pages 262–267, 1917. (Cité en page 22.)

[Rau 2011] U. Rau et T. J. Cornwell. *A multi-scale multi-frequency deconvolution.* A&A, 2011. arXiv:1106.2745. (Cité en page 32.)

[Rit 2014] S Rit, M Vila Oliva, S Brousmiche, R Labarbe, D Sarrut et G C Sharp. *The Reconstruction Toolkit (RTK), an open-source cone-beam CT reconstruction toolkit based on the Insight Toolkit (ITK).* Journal of Physics : Conference Series, vol. 489, no. 1, page 012079, 2014. (Cité en pages 38, 41 et 92.)

[Robert 1997] Christian Robert. Simulations par la méthode mcmc. Economica, 1997. (Cité en page 24.)

[Robert 2005] Christian P. Robert et George Casella. Monte carlo statistical methods (springer texts in statistics). Springer-Verlag, Berlin, Heidelberg, 2005. (Cité en page 24.)

[ROCCC 2013] ROCCC. *Overview of ROCCC 2.0 : a C to VHDL compilation toolset*, 2013. (Cité en page 11.)

[Rohkohl 2009] C Rohkohl, B Keck, H Hofmann et Joachim Hornegger. *Technical Note : RabbitCT-an open platform for benchmarking 3D cone-beam reconstruction algorithms.* Medical physics, vol. 36, pages 3940–4, 09 2009. (Cité en page 40.)

[Sanders 2010] Jason Sanders et Edward Kandrot. CUDA by example : An introduction to general-purpose GPU programming. Addison-Wesley Professional, 2010. (Cité en page 17.)

[Sauer 1993] Ken Sauer et Charles Bouman. *A local update strategy for iterative reconstruction from projections.* IEEE Transactions on Signal Processing, vol. 41, no. 2, pages 534–548, 1993. (Cité en page 28.)

[Savanier 2020] M. Savanier, C. Riddell, Y. Trousset, E. Chouzenoux et J.C. Pesquet. *A Matched CBCT Projector-Backprojector Based on the Convolution of B-splines.* In 6th International Conference on Image Formation in X-Ray Computed Tomography, Regensburg, Germany, Ao ?t 2020. (Cité en page 38.)

[Scherl 2007] H. Scherl, B. Keck, M. Kowarschik et J. Hornegger. *Fast GPU-Based CT Reconstruction using the Common Unified Device Architecture (CUDA).* In 2007 IEEE Nuclear Science Symposium Conference Record, volume 6, pages 4464–4466, 2007. (Cité en page 40.)

[Schlifske 2016] Daniel Schlifske et Henry Medeiros. *A fast GPU-based approach to branchless distance-driven projection and back-projection in cone beam CT.* In Despina Kontos et Thomas G. Flohr, editeurs, Medical Imaging 2016 : Physics of Medical Imaging, volume 9783, pages 742 – 749. International Society for Optics and Photonics, SPIE, 2016. (Cité en page 42.)

[SDP consortium 2018] SDP consortium. *Critical Design Review.* Rapport technique, oct. 2018. sdp-cdr-documentation. (Cité en page 73.)

[Siddon 1985] Robert L Siddon. *Fast calculation of the exact radiological path for a three-dimensional CT array.* Medical physics, vol. 12, no. 2, pages 252–255, 1985. (Cité en page 40.)

[Sidon 1985] R.L. Sidon. *Fast calculation of the exact radiological path for a three-dimensional CT array.* Med. Phys., vol. 12, no. 2, pages 252–255, 1985. (Cité en page 80.)

[Silva 2013] Bruno Silva, An Braeken, Abdellah Touhafi et Erik D'Hollander. *Performance Modeling for FPGAs : Extending the Roofline Model with High-Level Synthesis Tools.* International Journal of Reconfigurable Computing, vol. 2013, 11 2013. (Cité en page 18.)

[Sorel 2004] Yves Sorel. *SynDEx : System-Level CAD Software for Optimizing Distributed Real-Time Embedded Systems.* Journal ERCIM News, vol. 59, pages 68–69, 2004. (Cité en page 18.)

[Sourbier 2018] Nicolas Sourbier, Jean-François Nezan, Cyril Tasse et Julien Hascoet. *Calibration Algorithms onn MPPA.* In IEEE SIPS, 2018. hal-01900350. (Cité en page 74.)

[Stodden 2013] Victoria Stodden, Peixuan Guo et Zhaokun Ma. *Toward Reproducible Computational Research : An Empirical Analysis of Data and Code Policy Adoption by Journals.* PLOS ONE, vol. 8, no. 6, pages 1–8, 06 2013. (Cité en page 92.)

[Thompson 2014] William Thompson, M, Lionheart et William RB. *GPU Accelerated Structure-Exploiting Matched Forward and Back Projection for Algebraic Iterative Cone Beam CT Reconstruction.* In The Third International Conference on Image Formation in X-Ray Computed Tomography, 2014. (Cité en pages 40 et 42.)

[TIMA 2012] TIMA. *AUGH : Autonomous and User Guided High-level synthesis*, 2012. (Cité en page 11.)

[Veenboer 2017] B. Veenboer, M. Petschow et J. W. Romein. *Image-Domain Gridding on Graphics Processors*. In IPDPS, 2017. open science. (Cité en page 73.)

[Veenboer 2019] Bram Veenboer et John W. Romein. *Radio-Astronomical Imaging : FPGAs vs GPUs*. In Euro-Par, pages 509–521, 2019. (Cité en page 74.)

[Volkov 2016] Vasily Volkov. *Understanding Latency Hiding on GPUs*. Theses, University of California Berkeley, 2016. (Cité en pages 14 et 17.)

[Wang 2017] Li Wang. *Fast and Accurate 3D X ray Image Reconstruction for Non Destructive Test Industrial Applications*. Theses, Université Paris-Saclay, D ?embre 2017. (Cité en page 28.)

[Wen 2020] S. Wen et G. Luo. *FPGA-accelerated Automatic Alignment for Three-dimensional Tomography*. In 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), pages 172–176, 2020. (Cité en page 45.)

[Wiaux 2009] Y. Wiaux, L. Jacques, G. Puy, A. Scaife et P. Vandergheynst. *Compressed sensing for radio interferometry*. MNRAS, vol. 395, 2009. arXiv:0812.4933. (Cité en page 32.)

[Wienke 2012] Sandra Wienke, Paul Springer, Christian Terboven et Dieter an Mey. *OpenACC—first experiences with real-world applications*. In European Conference on Parallel Processing, pages 859–870. Springer, 2012. 10.1007/978-3-642-32820-6_85. (Cité en page 10.)

[Williams 2009] Samuel Williams, Andrew Waterman et David Patterson. *Roofline : An Insightful Visual Performance Model for Multicore Architectures*. Commun. ACM, vol. 52, no. 4, page 65–76, Avril 2009. (Cité en page 16.)

[Wu 2011] Meng Wu et Jeffrey A Fessler. *GPU acceleration of 3D forward and backward projection using separable footprints for X-ray CT image reconstruction*. In Proc. Intl. Mtg. on Fully 3D Image Recon. in Rad. and Nuc. Med, volume 6, page 021911. Citeseer, 2011. (Cité en pages 42 et 43.)

[Xiao 2012] Kai Xiao, Danny Chen, X Hu et Bo Zhou. *Efficient implementation of the 3D-DDA ray traversal algorithm on GPU and its application in radiation dose calculation*. Medical physics, vol. 39, pages 7619–25, 12 2012. (Cité en page 40.)

[Xie 2015] Lizhe Xie, Yining Hu, Bin Yan, Lin Wang, Benqiang Yang, Wenyuan Liu, Libo Zhang, Limin Luo, Huazhong Shu et Yang Chen. *An Effective CUDA Parallelization of Projection in Iterative Tomography Reconstruction*. PLOS ONE, vol. 10, no. 11, pages 1–17, 11 2015. (Cité en page 41.)

[Xu 2007] Fang Xu et Klaus Mueller. *Real-time 3D computed tomographic reconstruction using commodity graphics hardware*. Physics in Medicine and Biology, vol. 52, no. 12, pages 3405–3419, 2007. (Cité en page 40.)

[Xu 2010a] F. Xu. *Fast implementation of iterative reconstruction with exact ray-driven projector on GPUs.* Tsinghua Science and Technology, vol. 15, no. 1, pages 30–35, 2010. (Cité en page 40.)

[Xu 2010b] J. Xu, N. Subramanian, A. Alessio et S. Hauck. *Impulse C vs. VHDL for Accelerating Tomographic Reconstruction.* In 2010 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, pages 171–174, 2010. (Cité en page 45.)

[Yalamanchili 2015] Pavan Yalamanchili, Umar Arshad, Zakiuddin Mohammed, Pradeep Garigipati, Peter Entschev, Brian Kloppenborg, James Malcolm et John Melonakos. ArrayFire - A high performance software library for parallel computing with an easy-to-use API. AccelerEyes, Atlanta, 2015. (Cité en page 48.)

[Yali 2015] Moein Pahlavan Yali. *FPGA-Roofline : An Insightful Model for FGPA-based Hardware Acceleration in Modern Embedded Systems.* PhD thesis, Virginia Polytechnic Institute and State University, 2015. (Cité en page 18.)

[Zarka 2015] P. Zarka*et al. NenuFAR : Instrument description and science case.* In 2015 International Conference on Antenna Theory and Techniques (ICATT), pages 1–6, 2015. (Cité en page 75.)

[Zeng 1997] G. L. Zeng, Yi Weng et G. T. Gullberg. *Iterative reconstruction with attenuation compensation from cone-beam projections acquired via nonplanar orbits.* IEEE Transactions on Nuclear Science, vol. 44, no. 1, pages 98–106, 1997. (Cité en page 37.)

[Zeng 2000] Gengsheng L. Zeng et Grant T. Gullberg. *Unmatched Projector/Backprojector Pairs in an Iterative Reconstruction Algorithm.* IEEE Transactions on medical imaging, vol. 19, no. 5, pages 548–555, May 2000. (Cité en page 37.)

[Zheng 2014] Yuling Zheng. *Fast variational Bayesian algorithms and their application to large dimensional inverse problems.* Theses, Université Paris Sud - Paris XI, D ?embre 2014. (Cité en page 24.)

[Zinsser 2013] Timo Zinsser et Benjamin Keck. *Systematic Performance Optimization of Cone-Beam Back-Projection on the Kepler Architecture.* In Fully3D committee, editeur, Proceedings of the 12th Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, page 225–228, 2013. (Cité en page 40.)

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Adéquation algorithme architecture pour l'accélération de méthodes d'inversion de données en grande dimension

Mots clés : Adéquation Algorithme Architecture (A^3), calcul parallèle, GPU, FPGA, synthèse de haut niveau, problème inverse, reconstruction tomographique, radioastronomie.

Résumé : L'amélioration constante de la résolution des instruments parallèlement à la complexité grandissante des méthodes de reconstruction basées sur des modèles de plus en plus précis, s'accompagne d'un besoin croissant en puissance de calcul. Les cartes accélératrices constituées de puces GPUs ou FPGAs sont une opportunité pour réduire ce fossé technologique existant entre les systèmes d'acquisition et de reconstruction. Dans le contexte particulier de la résolution de problèmes inverses mal posés, mes travaux de recherche en adéquation algorithme architecture au sein du GPI visent à prendre en compte en amont de la définition des méthodes, les limitations et le potentiel des architectures d'accélération. Après une présentation des parallélismes offerts par les architectures GPU et FPGA, le contexte algorithmique des méthodes bayésiennes et leur application en reconstruction tomographique y sont exposés ; cet HDR se focalise ensuite sur l'accélération des opérateurs de projection/rétroprojection et de convolution utilisés en tomographie et radioastronomie, puis sur la parallélisation de toute la boucle itérative des méthodes d'inversion afin de les faire résider entièrement sur des serveurs multi-GPUs. Enfin, les perspectives à court et moyen terme de ces travaux sont exposées.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France