

UNIVERSITÉ DE MONTRÉAL

DÉVELOPPEMENT D'UN SCHEMA DE CALCUL PRISMATIQUE GÉNÉRALISÉ
PARALLÈLE EN TRANSPORT DÉTERMINISTE HÉTÉROGÈNE 3-D

MARC-ANDRÉ LAJOIE
DÉPARTEMENT DE GÉNIE PHYSIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE NUCLÉAIRE)
MAI 2017

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

DÉVELOPPEMENT D'UN SCHÉMA DE CALCUL PRISMATIQUE GÉNÉRALISÉ
PARALLÈLE EN TRANSPORT DÉTERMINISTE HÉTÉROGÈNE 3-D

présentée par : Lajoie Marc-André

en vue de l'obtention du diplôme de : Philosophiæ Doctor

a été dûment acceptée par le jury d'examen constitué de :

M. HÉBERT Alain, Doctorat, président

M. MARLEAU Guy, Ph. D., membre et directeur de recherche

M. FÉVOTTE François, Ph. D., membre et codirecteur de recherche

M. ÉTIENNE Stéphane, Doctorat, membre

M. DAHMANI Mohamed, Ph. D., membre externe

DÉDICACE

*Grâce à Marie-Eve,
À Murielle,
Pour Jeanette...*

REMERCIEMENTS

Le financement de cette thèse est fourni par le Fonds Québécois de Recherche en Nature et Technologies (FQRNT) et par Électricité de France (EDF). Cette dernière met aussi à notre disposition l'expertise et les ressources informatiques qui ont été nécessaires à son achèvement. L'auteur remercie sincèrement les deux organismes pour leur soutien.

La collaboration avec Électricité de France aura été intrinsèque au déroulement du projet. L'entreprise était intéressée par l'élaboration d'un outil de validation de sa chaîne de calcul, et le cadre de développement de DRAGON a fourni un support de base adéquat pour arriver à cet objectif. EDF a fourni à la fois un accès à une expertise de haut niveau en calcul haute performance et en comportement neutronique des réacteurs, en plus d'un accès à des capacités de calcul parallèle à grande échelle. Leur support aura été essentiel au déroulement de ces travaux, et je les en remercie. Mon séjour dans leurs bureaux de Clamart aura été mémorable de par les expériences, les apprentissages et les personnes rencontrées.

Je tiens aussi à sincèrement remercier d'abord et avant tout mes co-directeurs, Guy Marleau et François Févotte, pour leur aide, leur expertise, leur support, et *surtout* leur patience. Sans leur soutien constant – malgré mes trop nombreux relâchements – je n'y serais jamais arrivé.

Un merci tout spécial à Tanguy Courau pour le travail préalable sans lequel ce projet n'aurait jamais pu débiter.

Merci à tous ceux dont j'aurai eu l'occasion de croiser le chemin à Paris (et à Clamart), dans le désordre : Bruno Lathuilière, Pierre Guérin, Laurent Plagne, Sethy Montan, Salli Moustafa, Wilfried Kischenmann, Angélique Ponçot, Éléonore Bavoil, Fannie Meyer, Flora Lucet-Sanchez, Émile Razafindrakoto, Bruno Quinnez, Frank Hülsemann, Jean-Sylvestre Lamy, Hadrien Leroyer, Mark Zweers, Thierry Fouquet, Joel LeMer, Alexis Jinaphanh et j'en passe. L'expérience aurait été tout autre sans les discussions, rencontres, sorties et cafés. Merci.

Merci à tous mes collègues et professeurs de l'IGN à Montréal (toujours dans le désordre) Nicolas Martin, Romain LeTellier, Ève-Lyne Pelletier, Jean-Sébastien Duquette, Majid Fassi-Fehri, Mehdi Mahjoub, Hem Prabha Raghav, Tarek Benguedouar, Alexandre Landry, Geneviève Harrisson, Emmanuel St-Aubin, et j'en oublie.

Merci à ma famille, mes frères, ma mère qui m'ont épaulé et soutenu.

Merci aussi à mes employeurs depuis ces dernières années, CGI et KPMG, qui m'auront laissé

le temps de finir la rédaction de cette thèse à temps perdu.

Merci finalement et tout spécialement à Marie-Eve, pour sa patience stupéfiante et son soutien indéfectible, pour avoir sauté dans le vide avec moi, pour les expériences, les hauts et les bas, sans lesquels le tout aurait été vachement moins intéressant !

RÉSUMÉ

Ce projet a pour but d'améliorer la performance des algorithmes de résolution de l'équation du transport du flux neutronique dans les réacteurs nucléaires. La méthode des caractéristiques (MoC) en particulier, étant de plus en plus largement utilisée en raison de ses nombreux avantages intrinsèques (traitement direct de l'anisotropie de la diffusion, nombre de régions arbitrairement large, etc.) reste cependant très lourde en termes de nombre d'opérations (et donc de temps d'exécution), et de besoins mémoire ou d'accès disque.

Le but de cette thèse est de proposer une généralisation d'un algorithme d'analyse des géométries prismatiques qui permet le traitement de géométries non-uniformes selon l'axe de projection, mais qui restent cependant prismatiques par morceaux le long de celui-ci. Ce traitement prismatique ouvre par ailleurs des avenues de parallélisation de l'algorithme MoC, qui sont aussi explorées dans ce contexte.

L'algorithme a été validé et vérifié comme étant fonctionnel sur des cas-test qui représentent des situations réelles, à petite et grande échelle. Le haut d'un crayon de combustible a été simulé afin de valider et d'explorer les propriétés numériques de l'algorithme, tout comme un assemblage flambé, afin d'étudier les avantages informatiques à plus grande échelle de la méthode améliorée.

En particulier, les résultats montrent une réduction du volume du fichier de traçage (et donc des besoins en mémoire) d'environ 4 ordres de grandeur et du temps nécessaire pour résoudre l'équation de transport sur la géométrie globale d'un facteur 10 – avant l'introduction du calcul parallèle, par rapport à l'algorithme 3-D traditionnel. En outre, on observe les mêmes résultats (à quelques exceptions dues au niveau de la précision numérique), ainsi que les mêmes caractéristiques de convergence lors de l'exécution.

L'analyse des performances de la parallélisation de l'algorithme permet de conclure qu'il s'agit d'une avenue prometteuse. Les résultats montrent une diminution supplémentaire du temps de calcul nécessaire de 60% dans le module de calcul du flux lorsque 4 fils d'exécution sont utilisés, et suggèrent une scalabilité intéressante si cette méthode est couplée à des algorithmes plus traditionnels.

Des axes de parallélisation supplémentaires sont finalement suggérés, de même que l'intégration plus à fond de certaines méthodes de projection afin d'améliorer la versatilité et la performance de l'algorithme développé dans cette thèse.

ABSTRACT

The goal of this project is to improve the performance of the algorithms used for the solution of the transport equation for the neutron flux in nuclear reactors. Of particular interest is the Method of Characteristics (MoC), used more and more widely due to its numerous advantages (native treatment of scattering anisotropy, arbitrarily large number of spatial regions, etc.) but still hindered by its comparatively large computing requirements, both in terms of number of operations (and hence execution time), and memory requirements or disk accesses.

This particular project proposes the generalization of an algorithm for a prismatic representation of geometries that enables the consideration of non-uniform geometries along the projection axis, but that still keep a piece-wise axial uniformity. This new prismatic representation also opens up new avenues for parallel computing inside the MoC algorithm, that are also explored in this project.

The algorithm itself has been validated and verified for test-cases representing real assembly configurations, both on small and larger scales. The top of a fuel pin is therefore simulated to explore the numerical properties of the algorithm, together with a buckled assembly to study the larger scale computing advantages of the improved method.

In particular, the results show a reduction in the size of the tracking file of around 4 orders of magnitude (and hence the memory requirements), and of about 10 times in terms of computing time required to solve the transport equation when comparing with the traditional 3-D solution algorithm – before the introduction of parallel computing. The numerical results are the same (within the precision due to numerical instability), and the same convergence characteristics are observed over the course of execution.

The parallelization performances of the algorithm show a very good longer term potential. The results indicate an additional 60% reduction in the computing time required to solve the transport problem when using 4 execution threads, and suggests a potential scalability when coupled with other more traditional parallelization options.

Some additional parallelization avenues are suggested, together with the integration of additional improvements to the projection methods and algorithms to improve further upon the performance and flexibility of the solution that is developed in this thesis.

TABLE DES MATIÈRES

DÉDICACE	iii
REMERCIEMENTS	iv
RÉSUMÉ	vi
ABSTRACT	vii
TABLE DES MATIÈRES	viii
LISTE DES TABLEAUX	xi
LISTE DES FIGURES	xii
LISTE DES SIGLES ET ABRÉVIATIONS	xiii
LISTE DES ANNEXES	xiv
CHAPITRE 1 INTRODUCTION	1
1.1 Mise en contexte et travaux antérieurs	1
1.2 Problématique	1
1.3 Structure du document	3
CHAPITRE 2 TRANSPORT NEUTRONIQUE	4
2.1 Interactions neutrons-matière	4
2.2 Équation du transport neutronique	4
2.2.1 Le terme source général	6
2.2.2 Problème critique	8
2.3 Codes déterministes et discrétisation	10
2.3.1 Formalisme multigroupe	10
2.3.2 Expansion en harmoniques sphériques	12
CHAPITRE 3 MÉTHODES NUMÉRIQUES EN TRANSPORT	14
3.1 Survol	14
3.2 Méthodes différentielles	15
3.2.1 Méthode des harmoniques sphériques	15

3.2.2	Méthode aux ordonnées discrètes	15
3.3	Méthode des caractéristiques	16
3.3.1	Présentation	16
3.3.2	Description	17
3.3.3	Schéma d'intégration	17
3.3.4	Traçage	20
3.3.5	Normalisation	23
3.3.6	Conditions frontières	24
3.4	Schéma numérique de résolution du système	25
3.4.1	Résolution spatiale	26
3.4.2	Méthodes d'accélération	27
3.5	Calcul de coeur	28
CHAPITRE 4 REPRÉSENTATION PRISMATIQUE ET TRAÇAGE		30
4.1	Définition	30
4.2	Applicabilité	30
4.3	Formalisme	31
4.3.1	Séparation de la géométrie en sections uniformes	32
4.4	Projection de la géométrie	35
4.4.1	Création de la géométrie projetée	35
4.4.2	Matrice de correspondance	38
4.4.3	Exemple	39
4.4.4	Extrusion et concaténation	40
4.4.5	Calcul des volumes et surfaces	42
4.5	Développement Logiciel	45
4.5.1	Projection	45
4.5.2	Résolution	45
4.5.3	Gestion de la mémoire	46
CHAPITRE 5 CALCUL PARALLÈLE		47
5.1	Calculateurs cibles	47
5.2	Options de parallélisation	48
5.2.1	SIMD	48
5.2.2	Mémoire partagée	49
5.2.3	Mémoire distribuée	50
5.3	Objectifs	51
5.3.1	Accélération	51

5.3.2	Scalabilité	51
5.3.3	Intensité arithmétique	52
5.3.4	Équilibrage des charges	53
5.4	Travaux antérieurs	54
5.5	Identification des gisements de parallélisme	55
5.5.1	Résolution Jacobi vs Gauss-Seidel	56
5.5.2	Boucle sur les groupes	57
5.5.3	Parallélisation par angle ou par direction	58
5.5.4	Parallélisation sur les lignes extrudées	59
CHAPITRE 6 RÉSULTATS		63
6.1	Simulation d'un embout de crayon	63
6.1.1	Description du problème	63
6.1.2	Résultats et discussion	68
6.1.3	Validation	69
6.1.4	Convergence Spatiale	73
6.2	Simulation d'un assemblage flambé	75
6.2.1	Description du problème	75
6.2.2	Discussion	82
6.2.3	Comparaison et performance	84
6.3	Calcul parallèle	85
6.3.1	Cas uniforme	86
6.3.2	Assemblage	87
6.4	Sommaire	89
CHAPITRE 7 CONCLUSION		90
7.1	Avenues futures	91
RÉFÉRENCES		92
ANNEXES		96

LISTE DES TABLEAUX

Tableau 4.1	Table de correspondance 2-D \rightarrow 3-D	41
Tableau 5.1	Architecture Ivanoe	48
Tableau 6.1	Composition isotopique des mélanges	64
Tableau 6.2	Table de correspondance 2-D \rightarrow 3-D pour l'embout d'un crayon . . .	67
Tableau 6.3	Erreurs sur les surfaces et les volumes calculés	68
Tableau 6.4	Comparaison des solveurs MoC 3-D et prismatique	70
Tableau 6.5	Validation – Composition uniforme	72
Tableau 6.6	Validation – Géométrie uniforme	73
Tableau 6.7	Validation – Géométrie non-uniforme	73
Tableau 6.8	Convergence spatiale - résultats	74
Tableau 6.9	Résultats assemblage déformé	80
Tableau 6.10	Résultats assemblage déformé-vide	83
Tableau 6.11	<i>Speedup</i> et efficacité pour une géométrie uniforme	86
Tableau 6.12	<i>Speedup</i> et efficacité pour un assemblage déformé	88

LISTE DES FIGURES

Figure 3.1	Traçage d'une cellule 2-D dans la direction $\hat{\Omega}$	21
Figure 4.1	Découplage des dépendances angulaires en φ et θ	33
Figure 4.2	Projection d'une géométrie prismatique 3-D→2-D « superdiscrétisée »	34
Figure 4.3	Bout de crayon	36
Figure 4.4	Combinaison de plans 2-D	37
Figure 4.5	Numérotation des régions sur la ligne	41
Figure 4.6	Extrusion de la ligne	44
Figure 5.1	Traitement parallèle des paquets en extrusion	60
Figure 6.1	Section de crayon	65
Figure 6.2	Combinaison des plans de projection et numérotation partielle	66
Figure 6.3	Traçage sur la géométrie projetée	71
Figure 6.4	Convergence Spatiale - résultats	74
Figure 6.5	Assemblage flambé – vue de côté	76
Figure 6.6	Crayon cylindrique non-déformé	78
Figure 6.7	Crayon cylindrique déformé	78
Figure 6.8	Superposition de crayons cylindriques	79
Figure 6.9	Équivalence entre les cellules régulières et rectangularisées	79
Figure 6.10	Cellules cartésiennes normales et déformées	81
Figure 6.11	Cellules cartésiennes projetées	82
Figure 6.12	Flambage exagéré	83
Figure 6.13	<i>Speedup</i> et efficacité pour une géométrie uniforme	87

LISTE DES SIGLES ET ABRÉVIATIONS

ACA	<i>Algebraic Collapsing Acceleration</i>
ACR	<i>Advanced CANDU Reactor</i>
ASA	<i>Asymptotic Synthetic Acceleration</i>
CANDU	Canada deutérium uranium
DD	<i>Diamond Differencing</i>
DSA	<i>Diffusion Synthetic Acceleration</i>
EDF	Electricité de France
FQRNT	Fonds québécois de recherche en nature et technologie
GMRES	<i>Generalized Minimal Residual</i>
LS	<i>Linear Surface</i>
MoC	<i>Method of Characteristics</i>
MPI	<i>Message Passing Interface</i>
P_{ij}	Méthode des probabilités de collision
P_N	Méthode des harmoniques sphériques
PVM	<i>Parallel Virtual Machine</i>
REB	Réacteur à eau bouillante
REP	Réacteur à eau sous pression
SC	<i>Step characteristic</i>
SCR	<i>Self-Collision Rebalancing</i>
SIMD	<i>Single Instruction – Multiple Data</i>
SLC	<i>Simplified Linear Characteristic</i>
S_N	Méthode aux ordonnées discrètes
SOR	<i>Successive Over Relaxation</i>
SP_N	Méthode <i>Simplified P_N</i>
TMT	<i>Track Merging Technique</i>

LISTE DES ANNEXES

ANNEXE I - FICHIERS D'ENTRÉE	96
ANNEXE II - TAUX DE RÉACTION	129

CHAPITRE 1 INTRODUCTION

1.1 Mise en contexte et travaux antérieurs

Une des plus importantes préoccupations de l'industrie électro-nucléaire est d'assurer la sûreté et la sécurité des installations nucléaires. Afin de prédire le comportement d'un réacteur dans toute situation, la connaissance de la distribution des neutrons dans le cœur est primordiale. Cette distribution est dictée par l'équation de transport de Boltzmann.

La méthode utilisée actuellement pour résoudre cette équation repose sur des approximations de modélisation (ex : opérateur de diffusion) afin de simplifier le calcul, avec les incertitudes qui les accompagnent. Les programmes informatiques développés pour simuler le comportement du cœur du réacteur sont dits à « deux niveaux » : Un premier calcul (calcul de cellule) est effectué sur une cellule représentative du réacteur (un ou quelques crayons de combustible – assemblage ou cellule – souvent limité à deux dimensions) pour établir le flux neutronique dans chaque région spatiale. Une homogénéisation pondérée des sections efficaces macroscopiques est ensuite effectuée à partir du flux calculé, et celles-ci sont par la suite transférées à un second algorithme. Ce dernier se charge de résoudre une équation de diffusion (opérateur de transport dégradé) à l'échelle du cœur complet (généralement en géométrie 3-D), d'où on peut déterminer la distribution globale du flux et donc de la puissance dans le réacteur (Glasstone et Sesonske, 1994).

1.2 Problématique

La résolution numérique directe de l'équation de transport neutronique à l'échelle d'un réacteur nucléaire de puissance a longtemps été considérée trop ardue pour être entreprise dans un temps non-prohibitif, en raison notamment de la quantité énorme d'informations à stocker et d'opérations à effectuer. Cette situation est toutefois en train de changer en raison du développement des calculateurs pétaflopiques et de la possibilité de recourir à des calculs massivement parallèles.

Une nouvelle méthode est donc proposée ici. Elle se base sur une combinaison d'une parallélisation à grande échelle afin de tirer profit des capacités des supercalculateurs, avec un traitement prismatique de la géométrie. Ce dernier permettra notamment de réduire le nombre d'opérations à effectuer, mais surtout les besoins mémoire nécessaires à la résolution

de l'équation. Ces deux aspects devraient permettre la modélisation utilisant la théorie du transport sur l'ensemble du cœur du réacteur, et ce, en trois dimensions. De plus, l'utilisation de la méthode des caractéristiques pour résoudre l'équation de transport comporte l'avantage d'être efficace pour les problèmes à grande échelle, en plus d'être mieux adaptée à la parallélisation que d'autres méthodes concurrentes.

Le standard MPI (*Message Passing Interface*) (Gropp *et al.*, 1999) permet de gérer les réseaux de calcul parallèles à grande échelle. Les deux implémentations du standard les plus répandues, disponibles en source libre, sont MPICH, développée par le Argonne National Laboratory, et LAM/Open MPI, développée par un consortium de laboratoires universitaires et le Los Alamos National Laboratory (Gabriel *et al.*, 2004). Une fois intégré au code DRAGON (Marleau *et al.*, 2006) et avec le traitement de la méthode des caractéristiques prismatiques, le parallélisme devrait permettre la simulation à l'échelle du cœur complet du réacteur, court-circuitant ainsi la seconde étape du calcul décrite plus haut.

L'utilisation d'un tel schéma de calcul sur le cœur complet du réacteur aurait plusieurs avantages. Ne comportant pratiquement pas d'approximations physiques, il permettrait d'établir un outil étalon très adapté à la validation et à l'amélioration des outils de calcul industriels, à la maîtrise des incertitudes, ainsi qu'à l'analyse physique des phénomènes complexes de couplage entre neutronique, thermohydraulique et thermomécanique. Le projet devrait avoir un impact important sur l'industrie nucléaire, et intéressera de nombreux acteurs dans le milieu. Il est d'ailleurs effectué en collaboration avec Électricité de France, qui y voit une opportunité de disposer d'outils de référence pour la validation de sa chaîne de calcul et l'amélioration de ses outils de simulation des cœurs de centrales.

Des travaux antérieurs ont par ailleurs été effectués par différentes équipes dans l'optique de résoudre cette problématique. Particulièrement d'intérêt dans le cadre de ce travail, le travail de thèse de Sciannandrone (2015), qui propose une résolution parallèle de l'équation de transport dans les géométries prismatiques, en se basant sur les caractéristiques traversant séquentiellement les mêmes milieux – et en particulier les mêmes surfaces. Dans le même contexte, les travaux de G. Gunow et Smith (2016) mettent en oeuvre le traçage en ligne, réduisant les besoins mémoire contre davantage d'opérations CPU. Finalement, les travaux préliminaires de l'auteur en lien avec cette thèse se doivent aussi d'être mentionnés, présentant une version fonctionnelle de l'algorithme suggéré ici (Lajoie *et al.*, 2014), mais ne faisant qu'effleurer la partie sur le calcul parallèle.

Cet aspect couvrant la parallélisation du projet est aussi effectué en proche collaboration avec le département Sinetics d'EDF, en particulier avec les groupes I27-I28 (Simulation du

comportement neutronique des cœurs et du cycle du combustible) et I23 (Analyse et modèles numériques), ce dernier ayant une expertise particulière en calcul haute performance très poussée.

1.3 Structure du document

Afin d'élaborer sur les aspects fondamentaux du projet, la première partie de ce document présente l'équation de transport à résoudre (Chapitre 2), ainsi que la méthode des caractéristiques qui est utilisée pour y arriver (Chapitre 3). On présente par la suite la simplification prismatique du processus d'analyse géométrique, ainsi que les principaux avantages qu'elle apporte à notre méthodologie (Chapitre 4). Puis, les améliorations de la parallélisation des algorithmes déjà disponibles dans le code DRAGON sont exposées (Chapitre 5). Après avoir présenté les avenues choisies et les problèmes inhérents à chacune des options et des développements effectués, nous exposons les résultats obtenus (Chapitre 6). Ceux-ci seront répartis selon l'aspect prismatique, l'aspect parallèle, puis la conjugaison des deux. Finalement, nous concluons et présentons des avenues futures de recherche qui permettraient d'améliorer davantage les méthodes décrites dans cette thèse.

CHAPITRE 2 TRANSPORT NEUTRONIQUE

2.1 Interactions neutrons-matière

Les interactions des neutrons avec le milieu dans lequel ils évoluent varient des collisions élastiques ou inélastiques, à la capture radiative et à la fission. Chacune de ces possibilités d'interaction est reprise par le concept de section efficace, qui représente la probabilité linéaire pour un neutron se déplaçant dans un milieu de subir une interaction d'un certain type. La section efficace macroscopique (Σ) peut dès lors être utilisée pour exprimer le facteur exponentiel d'atténuation ($e^{-\Sigma_i L}$) dû à un type i d'interaction, d'un faisceau de neutrons en fonction de la distance L parcourue dans le milieu. Dans ce cas, la section efficace macroscopique totale est la somme de toutes les sections partielles pour le milieu considéré.

2.2 Équation du transport neutronique

Le point central de la majorité des problèmes de neutronique est le flux neutronique. Celui-ci est établi en résolvant l'équation de transport, ou une de ses formes dérivées. Cette équation est dérivée des travaux de Ludwig Boltzmann portant sur la dynamique des gaz.

En supposant que les neutrons dans le réacteur n'interagissent pas entre eux, et qu'ils se déplacent en ligne droite entre les collisions avec les noyaux, la population neutronique à l'intérieur d'un réacteur évoluera en fonction du temps (t) dans un espace de phases à six dimensions : trois coordonnées spatiales de position ($\vec{r} = x\vec{i} + y\vec{j} + z\vec{k}$) et trois représentant la vitesse du neutron ($\vec{V}_n = \frac{d\vec{r}}{dt}$).

En s'inspirant de la présentation de Hébert (2009), nous pouvons définir les quantités suivantes :

- $V_n = \|\vec{V}_n\|$,
- $\hat{\Omega} = (\sqrt{1 - \mu^2} \cos \varphi, \sqrt{1 - \mu^2} \sin \varphi, \mu)$, où μ est le cosinus de l'angle polaire θ , et l'azimuth est φ . On peut établir la relation $\vec{V}_n = V_n \hat{\Omega}$,
- $n(\vec{r}, V_n, \hat{\Omega}, t)$, la densité de population neutronique, avec $n(\vec{r}, V_n, \hat{\Omega}, t) d^2\Omega d^3r dV_n$ la grandeur physique représentant le nombre de neutrons dans le volume d^3r autour de \vec{r} , avec une vitesse V_n dans un intervalle dV_n , dans la direction $\hat{\Omega}$, dans un intervalle de $d^2\Omega$, au temps t .

L'équation de bilan pourra s'écrire en considérant le nombre de neutrons produits et le nombre de neutrons perdus, dans l'intervalle de temps $[t, t + \Delta t]$. La variation du nombre de neutrons est donnée par :

$$\left[n(\vec{r}, V_n, \hat{\Omega}, t + \Delta t) - n(\vec{r}, V_n, \hat{\Omega}, t) \right] d^3r d^2\Omega dV_n \quad (2.1)$$

De la même façon, le nombre de neutrons s'échappant de l'hypervolume $d^3r d^2\Omega dV_n$ (centré autour de $\vec{r}, V_n, \hat{\Omega}$) est donné pendant l'intervalle de temps Δt par un terme de migration (*streaming*) :

$$\vec{\nabla} \cdot \hat{\Omega} V_n n(\vec{r}, V_n, \hat{\Omega}, t) d^3r d^2\Omega dV_n \Delta t \quad (2.2)$$

et le nombre de collisions entre les neutrons et le milieu, peut être représenté comme :

$$\Sigma(\vec{r}, V_n) V_n n(\vec{r}, V_n, \hat{\Omega}, t) d^3r d^2\Omega dV_n \Delta t \quad (2.3)$$

Finalement, le nombre de nouveaux neutrons émis, à la fois par diffusion et par fission est donné par un terme source général que nous présenterons plus en détails dans la prochaine section :

$$Q(\vec{r}, V_n, \hat{\Omega}, t) d^3r d^2\Omega dV_n \Delta t \quad (2.4)$$

où l'on définit :

- $\Sigma(\vec{r}, V_n)$, la section efficace totale macroscopique, que nous supposons indépendante de $\hat{\Omega}$ (milieu isotrope), et de t , afin d'alléger la notation,
- $\vec{\nabla} \cdot \hat{\Omega} V_n n(\vec{r}, V_n, \hat{\Omega}, t)$, le bilan des neutrons migrant à travers le volume considéré.

Nous introduirons ici le concept de flux neutronique, $\phi(\vec{r}, V_n, \hat{\Omega}, t)$ ¹. Notons que celui-ci n'est pas un flux proprement dit (comme on dirait d'un flux de chaleur) mais est plutôt défini, à l'aide de la densité de population neutronique $n(\vec{r}, V_n, \hat{\Omega}, t)$, comme :

$$\phi(\vec{r}, V_n, \hat{\Omega}, t) = V_n n(\vec{r}, V_n, \hat{\Omega}, t) \quad (2.5)$$

1. Certains auteurs utilisent φ et ϕ pour identifier le flux angulaire et le flux intégré. Nous nous en remettons ici aux dépendances des fonctions pour faire cette distinction.

Cela nous permet de réécrire l'équation globale de bilan comme :

$$\frac{n(\vec{r}, V_n, \hat{\Omega}, t + \Delta t) - n(\vec{r}, V_n, \hat{\Omega}, t)}{\Delta t} = Q(\vec{r}, V_n, \hat{\Omega}, t) - \vec{\nabla} \cdot \hat{\Omega} \phi(\vec{r}, V_n, \hat{\Omega}, t) - \Sigma(\vec{r}, V_n) \phi(\vec{r}, V_n, \hat{\Omega}, t) \quad (2.6)$$

que l'on peut exprimer, en prenant la limite lorsque $\Delta t \rightarrow 0$, sous la forme :

$$\frac{1}{V_n} \frac{\partial}{\partial t} \phi(\vec{r}, V_n, \hat{\Omega}, t) + \hat{\Omega} \cdot \vec{\nabla} \phi(\vec{r}, V_n, \hat{\Omega}, t) + \Sigma(\vec{r}, V_n) \phi(\vec{r}, V_n, \hat{\Omega}, t) = Q(\vec{r}, V_n, \hat{\Omega}, t) \quad (2.7)$$

où l'on a utilisé l'identité $\vec{\nabla} \cdot (\hat{\Omega} \phi(\vec{r}, \hat{\Omega})) = \hat{\Omega} \cdot \vec{\nabla} \phi(\vec{r}, \hat{\Omega}) + \phi(\vec{r}, \hat{\Omega}) \vec{\nabla} \cdot \hat{\Omega}$, avec $\vec{\nabla} \cdot \hat{\Omega} = 0$. Cette équation est dite la forme différentielle de l'équation de transport.

2.2.1 Le terme source général

Dans le cas général, le terme source mentionné précédemment doit tenir compte de toutes les sources de neutrons pouvant entrer dans le volume de contrôle considéré. Celles-ci proviennent dans le cas qui nous intéresse, de trois types de sources en particulier :

- Les sources de diffusion : des neutrons d'autres directions et d'autres énergies subissant des collisions pour se retrouver à l'énergie et dans la direction qui nous concerne ;
- Les sources de fission : des neutrons absorbés par un noyau fissile qui, ayant fissionné, émet de nouveaux neutrons – ceux-ci pouvant apparaître sur-le-champ, ou à retardement ;
- Les sources externes où la quantité de neutrons générée ne dépend pas de la population de neutrons.

Afin de simplifier la notation, nous utilisons ici un changement de variables où nous avons remplacé la dépendance en V_n par l'énergie $E = \frac{m_n V_n^2}{2}$ du neutron se déplaçant à la vitesse V_n .

Dans l'équation (2.7), le terme maintenant devenu $Q(\vec{r}, E, \hat{\Omega}, t)$ est le terme comprenant tant les sources de neutrons provenant de la fission et de la diffusion (de $\hat{\Omega}'$ vers $\hat{\Omega}$, et de E' vers E , de toutes les origines ($\forall E' \in [0, E_{max}], \forall \hat{\Omega}' \in 4\pi$)), que les sources extérieures. Cependant, ces dernières n'interviennent généralement pas dans les cas qui nous intéressent, et reposent

hors du contexte de ce document. Nous les représentons donc par un terme très général $S_{\text{ext}}(\vec{r}, E, \hat{\Omega}, t)$, et redirigeons le lecteur curieux vers des sources plus appropriées comme Hébert (2009) et Lewis et Miller Jr. (1993).

Pour les sources de diffusion, nous devons considérer les neutrons débutant avec une énergie E' et une direction $\hat{\Omega}'$, pour toutes les valeurs de E' , et pour toutes les directions $\hat{\Omega}'$, qui subissent soit une collision, soit une réaction (n, xn) (un neutron incident et x neutrons émis). Cette source de nouveaux neutrons est exprimée par :

$$Q_s(\vec{r}, E, \hat{\Omega}, t) = \int_0^\infty \int_{4\pi} \tilde{\Sigma}_S(\vec{r}, E \leftarrow E', \hat{\Omega} \leftarrow \hat{\Omega}') \phi(\vec{r}, E', \hat{\Omega}', t) d^2\Omega' dE' \quad (2.8)$$

où nous utilisons $\tilde{\Sigma}_S$ comme la section efficace différentielle macroscopique de diffusion (*scattering*), de manière à ce que $\tilde{\Sigma}_S(\vec{r}, E \leftarrow E', \hat{\Omega} \leftarrow \hat{\Omega}') d^2\Omega' dE'$ inclue les neutrons incidents avec l'énergie E' (dans l'intervalle dE') sortant de la collision avec l'énergie E (d'où $E \leftarrow E'$) et dans la direction $\hat{\Omega}'$ (dans l'intervalle $d^2\Omega'$) ressortant dans la direction $\hat{\Omega}$ (d'où $\hat{\Omega} \leftarrow \hat{\Omega}'$). Celle-ci inclut aussi les termes pour les réactions (n, xn) , comme étant : $\tilde{\Sigma}_s = \Sigma_s + 2\Sigma_{n,2n} + 3\Sigma_{n,3n} + \dots$. Notons ici que si les milieux qui composent le domaine spatial sont isotropes, la section efficace de diffusion est une fonction seulement de l'angle $\hat{\Omega} \cdot \hat{\Omega}'$.

La production de neutrons par fission étant isotrope, leur distribution est donnée par la somme des neutrons prompts et retardés, moyennés sur 4π :

$$Q_f(\vec{r}, E, \hat{\Omega}, t) = \frac{1}{4\pi} [Q_f(\vec{r}, E, t) + Q_{\text{ret}}(\vec{r}, E, t)] \quad (2.9)$$

On peut développer les sources isotropes et promptes, en fonction du taux d'émission $\nu\Sigma_f$ et du spectre d'émission χ_j :

$$Q_f(\vec{r}, E, t) = \sum_{j=1}^J \chi_j(E) \int_0^\infty \nu\Sigma_{f,j}(\vec{r}, E') \phi(\vec{r}, E', t) dE' \quad (2.10)$$

où l'on a utilisé

- $\phi(\vec{r}, E, t)$, le flux intégré, défini comme $\phi(\vec{r}, E, t) = \int_{4\pi} \phi(\vec{r}, E, \hat{\Omega}, t) d^2\Omega$,
- J , le nombre d'isotopes fissiles, chacun représenté par $j = 1 \dots J$,

- $\nu\Sigma_{f,j}$, le taux de production de neutrons par fission de l'isotope j , avec $\Sigma_{f,j}$, sa section efficace macroscopique de fission, et ν le nombre moyen de neutrons émis par fission,
- $\chi_j(E)$, le spectre d'émission des neutrons, c'est-à-dire la probabilité qu'un neutron émis par l'isotope j d'avoir une énergie E , à dE près.

Les neutrons retardés étant émis avec un profil énergétique différent et à un moment subséquent par rapport aux neutrons prompts, ils sont représentés comme suit :

$$Q_{\text{ret}}(\vec{r}, E, t) = \sum_{i=1}^N \tilde{\chi}_i(E) \lambda_i C_i(\vec{r}, t) \quad (2.11)$$

où on a :

- N isotopes précurseurs de neutrons retardés ;
- $\tilde{\chi}_i$ le spectre d'émission des neutrons retardés ;
- λ_i , la constante de désintégration du précurseur i ;
- C_i , la concentration du précurseur i .

On peut finalement écrire le terme source total et général, de la manière suivante :

$$Q(\vec{r}, E, \hat{\Omega}, t) = \int_0^\infty \int_{4\pi} \tilde{\Sigma}_S(\vec{r}, E \leftarrow E', \hat{\Omega} \cdot \hat{\Omega}') \phi(\vec{r}, E', \hat{\Omega}', t) d^2\Omega' dE' + \frac{1}{4\pi} \sum_{i=1}^N \tilde{\chi}_i(E) \lambda_i C_i(\vec{r}, t) + \frac{1}{4\pi} \sum_{j=1}^J \chi_j(E) \int_0^\infty \nu \Sigma_{f,j}(\vec{r}, E') \phi(\vec{r}, E', t) dE' + S_{\text{ext}}(\vec{r}, E, \hat{\Omega}, t) \quad (2.12)$$

2.2.2 Problème critique

Deux types de problèmes sont considérés dans le cadre de la neutronique. Le premier concerne les problèmes dits « à source », et sont surtout utilisés dans le cadre de calculs de radioprotection. Dans le cadre du stockage de déchets, par exemple, on fixe une source, connue pour le problème, et on calcule le flux sur la géométrie complète du problème.

Les problèmes dits de « cinétique » s'inscrivent dans ce contexte. En appliquant une discrétisation de la variable temporelle, on peut arriver à suivre l'évolution du flux en fonction des changements isotopiques du milieu et de la présence des précurseurs de neutrons retardés évoqués dans l'équation (2.11). Par la suite, on résout une suite de problèmes à source, en gardant cette considération pour les neutrons retardés qui fait en sorte que le terme source

prend une autre forme que celle que nous allons développer. Dans certaines situations, on utilise par ailleurs l'expression « cinétique ponctuelle » pour évoquer le fait que l'on modélise le réacteur complet seulement en fonction de la variable temporelle, en « moyennant » les variables spatiales et énergétique.

Le second type est celui qui nous intéresse davantage dans le cadre des calculs de réacteurs, les calculs dits « critiques ». Dans ce cadre, on tente d'analyser un système dont la réaction en chaîne de fission s'auto-entretient dans une certaine mesure, généralement dans un état quasi-stationnaire.

En règle générale, on utilise l'approximation quasi-statique, qui implique que l'on considère l'équation en mode stationnaire, les variations temporelles étant traitées le cas échéant, par exemple, en utilisant des méthodes dérivées de la théorie des perturbations. L'équation en mode stationnaire est donc exprimée comme suit, en laissant tomber les dépendances à la variable temporelle :

$$\hat{\Omega} \cdot \vec{\nabla} \phi(\vec{r}, E, \hat{\Omega}) + \Sigma(\vec{r}, E) \phi(\vec{r}, E, \hat{\Omega}) = Q(\vec{r}, E, \hat{\Omega}) \quad (2.13)$$

où nous avons aussi laissé tomber le terme contenant les neutrons retardés pour les intégrer au spectre des fissions promptes dans la source stationnaire.

Par contre, comme le système simulé n'est généralement pas à l'équilibre parfait entre production et élimination de neutrons à un moment donné, nous devons introduire dans les équations ci-dessus un terme restaurant cet équilibre afin de pouvoir permettre au modèle de s'appliquer à une géométrie qui est généralement non-critique. Nous transformons donc l'équation statique en problème à valeur propre en ajoutant un facteur k_{eff} au terme de fission.

Ce dernier, appelé facteur de multiplication effectif de neutrons, représente une proportion nécessaire entre le nombre de neutrons produits (fissions) et ceux perdus (absorption et fuites), afin de conserver l'approximation stationnaire – c'est en quelque sorte la valeur propre du système. Sa valeur varie autour de 1, et on considère que le réacteur est *sur-critique* si $k_{\text{eff}} > 1$, alors que la population de neutrons augmente dans le temps, qu'il est *critique* pour $k_{\text{eff}} = 1$, alors que le nombre de neutrons reste stable, et qu'il est *sous-critique* lorsque $k_{\text{eff}} < 1$, dans quel cas la population de neutrons diminue dans le temps. Il vient en quelque sorte modifier ν , le nombre de neutrons produits par fission. Sa valeur représente un ratio du nombre de neutrons produits à chaque génération par celui de la génération précédente.

Le terme source devient donc :

$$Q(\vec{r}, E, \hat{\Omega}) = \int_0^\infty \int_{4\pi} \tilde{\Sigma}_S(\vec{r}, E \leftarrow E', \hat{\Omega} \cdot \hat{\Omega}') \phi(\vec{r}, E', \hat{\Omega}') d^2\Omega' dE' \\ + \frac{1}{4\pi k_{\text{eff}}} \sum_{j=1}^J \chi_j(E) \int_0^\infty \nu \Sigma_{f,j}(\vec{r}, E') \phi(\vec{r}, E') dE' \quad (2.14)$$

Même avec cette approximation quasi-stationnaire, il reste très ardu de résoudre cette équation analytiquement sauf pour des cas extrêmement simples. On utilisera donc généralement des approximations et des méthodes numériques afin d'isoler la quantité d'intérêt, c'est-à-dire le flux ϕ , qui devient le vecteur propre du système, et de déterminer la valeur propre, k_{eff} .

2.3 Codes déterministes et discrétisation

La résolution de cette équation sur l'ensemble de la géométrie considérée permet de déterminer le flux neutronique. Par contre, la résolution analytique est généralement beaucoup trop compliquée sauf sur des géométries particulièrement simples, et avec des hypothèses simplificatrices concernant les dépendances énergétiques des sections efficaces. Des méthodes ont donc été développées afin de résoudre numériquement cette équation.

Ces méthodes déterministes reposent sur des approximations de modélisation supplémentaires permettant de discrétiser les variables spatiales, angulaires et énergétiques. Nous présentons ici le formalisme multigroupe pour la variable énergétique ainsi que les expansions en harmoniques sphériques réelles pour le traitement de la variable angulaire. Le traitement spatial est exploré plus en détails dans la présentation de la méthode des caractéristiques dans le chapitre 3.

2.3.1 Formalisme multigroupe

La discrétisation multigroupe consiste en une séparation du continuum énergétique, en G groupes d'énergie. On considère alors que les neutrons sont monoénergétiques dans chacun des groupes – on peut donc représenter la distribution énergétique par une fonction continue par parties. On transforme donc les intégrales en énergie de l'équation (2.14) en sommation sur chacun des groupes d'énergie considérés. Notons aussi l'utilisation de E_0 comme borne supérieure d'énergie plutôt que ∞ . Une valeur de $E_0 \sim 20$ MeV permet généralement de couvrir l'ensemble des énergies des neutrons dans un réacteur à fission, surtout thermique.

On doit par conséquent redéfinir les quantités utilisées dans l'équation de transport précédemment, en les moyennant sur l'intervalle énergétique du groupe considéré, ici le flux et la section efficace :

$$\phi_g(\vec{r}, \hat{\Omega}) = \int_{E_g}^{E_{g-1}} \phi(\vec{r}, E, \hat{\Omega}) dE \quad (2.15)$$

$$\Sigma_g(\vec{r}) = \frac{1}{\phi_g(\vec{r})} \int_{E_g}^{E_{g-1}} \Sigma(\vec{r}, E) \phi(\vec{r}, E) dE \quad (2.16)$$

où nous remarquons la pondération de Σ_g par ϕ_g , qui permet de conserver le taux de réaction. Nous pouvons donc écrire la forme différentielle de l'équation de transport multigroupe à l'état stationnaire :

$$\hat{\Omega} \cdot \vec{\nabla} \phi_g(\vec{r}, \hat{\Omega}) + \Sigma_g(\vec{r}) \phi_g(\vec{r}, \hat{\Omega}) = Q_g(\vec{r}, \hat{\Omega}) \quad (2.17)$$

et nous avons défini le terme source multigroupe comme :

$$\begin{aligned} Q_g(\vec{r}, \hat{\Omega}) &= \sum_{h=1}^G \int_{4\pi} \tilde{\Sigma}_{S,g \leftarrow h}(\vec{r}, \hat{\Omega} \cdot \hat{\Omega}') \phi_h(\vec{r}, \hat{\Omega}') d^2\Omega' \\ &+ \frac{1}{4\pi k_{\text{eff}}} \sum_{j=1}^J \chi_{j,g} \sum_{h=1}^G \nu \Sigma_{f,j,h}(\vec{r}) \phi_h(\vec{r}) \end{aligned} \quad (2.18)$$

avec :

$$\chi_{j,g} = \int_{E_g}^{E_{g-1}} \chi_j(E) dE \quad (2.19)$$

$$\nu \Sigma_{f,j,g}(\vec{r}) = \frac{1}{\phi_g(\vec{r})} \int_{E_g}^{E_{g-1}} \nu \Sigma_{f,j}(\vec{r}, E) \phi(\vec{r}, E) dE \quad (2.20)$$

$$\tilde{\Sigma}_{S,g \leftarrow h}(\vec{r}, \hat{\Omega} \cdot \hat{\Omega}') = \frac{1}{\phi_h(\vec{r})} \int_{E_g}^{E_{g-1}} \int_{E_h}^{E_{h-1}} \tilde{\Sigma}_S(\vec{r}, E \leftarrow E', \hat{\Omega} \cdot \hat{\Omega}') \phi(\vec{r}, E') dE' dE \quad (2.21)$$

Cette formulation offre l'avantage indéniable de simplifier et d'accélérer énormément les calculs déterministes, tout en restant générale dans le contexte où le nombre de groupes est

arbitraire (variant d'environ une dizaine à plusieurs milliers). Par contre, la présence de résonances étroites dans les données des sections efficaces peut causer un phénomène appelé « autoprotection » de par lequel le flux (réel) est fortement atténué à l'intérieur même d'un groupe d'énergie, alors que le flux ne présente pas cette dépression dans le modèle multi-groupe. Une nouvelle étape est donc nécessaire dans la chaîne de calcul de cellule afin de tenir compte de ces effets. On y calcule des nouvelles sections efficaces moyennées, dites « autoprotégées », qui seront ensuite utilisées par le code de réseau DRAGON (Hébert et Marleau, 1991).

2.3.2 Expansion en harmoniques sphériques

Finalement, afin de pouvoir traiter des effets d'anisotropie de la diffusion, on développe le terme source en harmoniques sphériques réelles, à l'aide d'une expansion en polynômes de Legendre (Lewis et Miller Jr., 1993) tronquée à l'ordre L , de manière à obtenir :

$$Q_g(\vec{r}, \hat{\Omega}) = \sum_{l=0}^L \frac{2l+1}{4\pi} \sum_{m=-l}^l \mathcal{R}_l^m(\hat{\Omega}) Q_{l,g}^m(\vec{r}) \quad (2.22)$$

avec les moments de la source $Q_{l,g}^m(\vec{r}) = [\Sigma_{s,g \leftarrow h,l}(\vec{r}) \phi_{l,g}^m(\vec{r}) + S_{l,g}^m(\vec{r})]$, où $S_{l,g}^m(\vec{r})$ contient la source à la fois de fission et externe. On a utilisé les fonctions associées de Legendre :

$$\mathcal{R}_l^m(\hat{\Omega}) = \begin{cases} \sqrt{(2 - \delta_{m,0}) \frac{(l-|m|)!}{(l+|m|)!}} \mathcal{P}_l^{|m|}(\mu) \cos(m\varphi) & \text{si } m \geq 0 \\ \sqrt{2 \frac{(l-|m|)!}{(l+|m|)!}} \mathcal{P}_l^{|m|}(\mu) \sin(|m|\varphi) & \text{si } m < 0 \end{cases} \quad (2.23)$$

où, ayant déterminé les polynômes de Legendre par récurrence :

$$\begin{aligned} P_0(\mu) &= 1 \\ P_1(\mu) &= \mu \\ P_{l+1}(\mu) &= \frac{1}{l+1} [(2l+1)\mu P_l(\mu) - lP_{l-1}(\mu)] \end{aligned} \quad (2.24)$$

on a défini les fonctions de Legendre associées, \mathcal{P}_l^m , comme :

$$\mathcal{P}_l^m(\mu) = (1 - \mu^2)^{m/2} \frac{d^m}{d\mu^m} P_l(\mu). \quad (2.25)$$

À l'aide de ces définitions, on peut définir les moments du flux et de la source comme :

$$\phi_{l,g}^m(\vec{r}) = \int_{4\pi} \mathcal{R}_l^m(\hat{\Omega}) \phi_g(\vec{r}, \hat{\Omega}) d^2\Omega \quad (2.26)$$

$$S_{l,g}^m(\vec{r}) = \int_{4\pi} \mathcal{R}_l^m(\hat{\Omega}) S_g(\vec{r}, \hat{\Omega}) d^2\Omega \quad (2.27)$$

ainsi que les coefficients de l'expansion en polynômes de Legendre des sections efficaces de diffusion :

$$\Sigma_{s,g\leftarrow h,l}(\vec{r}) = \int_{-1}^1 \Sigma_{s,g\leftarrow h}(\mu) P_l(\mu) d\mu \quad (2.28)$$

De manière analogue à l'équation (2.22), on peut aussi exprimer le développement du flux angulaire comme :

$$\phi_g(\vec{r}, \hat{\Omega}) = \sum_{l=0}^L \frac{2l+1}{4\pi} \sum_{m=-l}^l \phi_{l,g}^m(\vec{r}) \mathcal{R}_l^m(\hat{\Omega}) \quad (2.29)$$

Cette expansion permet, dans les méthodes qui peuvent en tirer profit, le traitement de milieux qui présentent de forts effets d'anisotropie, notamment pour un traitement adéquat du modérateur, ou pour des calculs de radioprotection. C'est le cas de la méthode des caractéristiques, qui sera présentée au chapitre 3.

En règle générale, la troncature de cette expansion se limite au cas isotrope ($L = 0$) ou linéairement anisotrope ($L = 1$), sauf pour des cas très particuliers. Ici, nous conservons cependant le cadre général.

CHAPITRE 3 MÉTHODES NUMÉRIQUES EN TRANSPORT

3.1 Survol

Les méthodes numériques déterministes (par opposition aux méthodes stochastiques) utilisées pour la résolution de l'équation de transport, présentée au chapitre 2, peuvent être catégorisées selon trois axes, en fonction de la forme de l'équation de transport sur laquelle elles reposent :

- la formulation intégrale (méthode des probabilités de collision, ou P_{ij}) ;
- la formulation différentielle (méthodes P_N , SP_N , S_N) ;
- la formulation caractéristique (méthode des caractéristiques, ou MoC).

La méthode des probabilités de collision (P_{ij}) est généralement bien adaptée à de petits problèmes pouvant présenter des géométries hétérogènes et non-structurées. En supposant des sources isotropes, on dérive, à partir de l'équation de transport intégrale, des relations donnant la probabilité pour un neutron émis dans la région i de subir sa première interaction dans la région j , et ce, pour l'ensemble des régions spatiales. La solution du flux se structure alors autour de la résolution d'un système linéaire faisant intervenir cette matrice des P_{ij} et les sources. La méthode est souvent utilisée comme première étape du calcul de réacteur, permettant d'obtenir les sections efficaces macroscopiques utilisées dans le calcul de cœur, mais reste fortement limitée pour les plus gros problèmes par la taille de la matrice.

Les méthodes reposant sur la forme différentielle tirent parti d'une discrétisation de la variable angulaire, soit par des expansions en harmoniques sphériques (P_N , SP_N), ou par une discrétisation directe de la variable angulaire dans des directions $\hat{\Omega}_i$ (méthode des ordonnées discrètes - S_N).

Cette discrétisation angulaire se conjugue aussi à un traitement de la variable spatiale, qui distinguera les méthodes utilisées entre elles. Un abus de langage courant associe le traitement spatial utilisant les différences diamant, couramment utilisées, avec « la » méthode S_N , mais on peut certainement inclure dans la grande famille des méthodes aux ordonnées discrètes les méthodes nodales, mais aussi les méthodes des caractéristiques, courtes et longues. Les méthodes utilisées dans ce contexte sont notamment intéressantes en raison de schémas de différences qui sont partagés avec MoC, et des méthodes d'accélération communes. Nous survolons ces caractéristiques communes dans les sections 3.2 et 3.3.

En particulier, la méthode des caractéristiques s’articule donc autour d’un balayage angulaire pondéré commun avec celui des méthodes S_N , mais pouvant utiliser les structures de données et les schémas de calcul développés pour P_{ij} . MoC présente l’avantage à la fois de pouvoir être utilisée dans des géométries non-structurées, et de requérir moins de mémoire que les probabilités de collisions sur une géométrie équivalente, en plus de pouvoir être utilisée à toutes les étapes du calcul de réseau. De surcroît, MoC permet nativement le traitement des sources anisotropes. Nous nous intéressons ici à la méthode des caractéristiques dites longues, que nous présentons plus en détails dans la section 3.3.

3.2 Méthodes différentielles

3.2.1 Méthode des harmoniques sphériques

La méthode des harmoniques sphériques (P_N) se base sur un développement limité du flux sur une base d’harmoniques sphériques. Cette méthode produit des équations couplées qui deviennent rapidement lourdes. Ainsi, on lui préfère généralement une simplification, la méthode SP_N (*Simplified P_N*), dans laquelle l’expansion se fait plutôt sur une base incomplète basée sur des polynômes de Legendre, qui simplifie la solution, sans toutefois résoudre explicitement l’équation de transport. Cette méthode est utilisée notamment pour des calculs de coeur complet, permettant une précision accrue vis-à-vis des codes de diffusion.

3.2.2 Méthode aux ordonnées discrètes

L’idée de base des méthodes de la famille S_N , ou méthodes aux ordonnées discrètes, repose sur une discrétisation du flux angulaire, en résolvant le problème selon des directions précises. Celles-ci sont données par des quadratures angulaires bien définies, et la solution du flux se concrétise autour d’un balayage spatial dans chacune des directions. La méthode aux différences finies n’est utilisée à grande échelle que sur des géométries structurées et requiert des accélérations performantes pour en assurer la convergence.

L’idée de base reste de développer $\phi(\hat{\Omega})$ en une série de $\phi(\hat{\Omega}_q)$ en choisissant un nombre fini de directions $\hat{\Omega}_q$, chacune associée à un poids ω_q , de manière à ce que :

$$\int_{4\pi} d^2\Omega = \sum_q \omega_q \hat{\Omega}_q = 4\pi, \quad (3.1)$$

et donc, dans le cas général :

$$\int_{4\pi} f(\hat{\Omega}) d^2\Omega = \int_0^\pi \int_0^{2\pi} f(\theta, \varphi) d\varphi d\theta = \sum_q \mathcal{W}_q f(\theta_q, \varphi_q) = \sum_q \mathcal{W}_q f(\hat{\Omega}_q) \quad (3.2)$$

On en retire donc une série d'équations couplées représentant l'équation de transport (2.17), discrétisée angulairement :

$$\hat{\Omega}_q \cdot \vec{\nabla} \phi_g(\vec{r}, \hat{\Omega}_q) + \Sigma_g(\vec{r}) \phi_g(\vec{r}, \hat{\Omega}_q) = Q_{g,q}(\vec{r}, \hat{\Omega}_q) \quad (3.3)$$

où les intégrales angulaires sont évaluées par l'équation (3.2).

Il importe généralement de porter une attention particulière au choix de la quadratures qui est utilisée. Mentionnons le caractère « level-symmetric » de certaines d'entre elles, qui permettent de préserver la symétrie entre les octants, ainsi qu'aux relations de conservation des particules pour certaines méthodes. Notons en passant que les quadratures sont généralement définies non pas par les angles θ et φ mais par les trois cosinus directeurs associés.

Par ailleurs, prenons ici le soin de distinguer les *méthodes* S_N de l'approximation qui consiste à choisir des ordonnées discrètes pour la résolution. La première consiste en une *méthode de collocation*, s'appliquant à une géométrie généralement structurée qu'on résoud en balayant le maillage selon les directions choisies. La seconde consiste en une série d'approximations concernant les quadratures (les directions ainsi que les poids associés) et certains schémas de différences qui y sont utilisés. Ceux-ci sont certes utilisés par la *méthode* S_N , mais aussi par la méthode des caractéristiques. Nous les survolons dans la section suivante.

3.3 Méthode des caractéristiques

3.3.1 Présentation

La méthode des caractéristiques repose sur une technique mathématique permettant la transformation de l'équation aux dérivées partielles (2.7) en plusieurs équations différentielles ordinaires, le long de segments quelconques dans l'espace. Dans le cas de la théorie du transport neutronique, comme les neutrons se déplacent en ligne droite entre les collisions, les segments *caractéristiques* seront des lignes droites traversant l'espace.

Originellement proposée en transport neutronique par Askew (1972) en lien avec WIMS, puis rapidement portée dans CACTUS (Halsall, 1980), la méthode des caractéristiques est aujourd'hui utilisée par une multitude de codes, et a notamment été implantée en 3-D par

Suslov (1993) dans le code MCGG3D, ainsi que dans DRAGON dans sa version cyclique (2-D) par Roy (1998), puis généralisée par Le Tellier (2006). Le code APOLLO, développé au CEA (Sanchez et Chetaine, 2000) propose aussi son implémentation à l'aide du module TDT. Dans le contexte du développement dans un logiciel libre comme le code DRAGON, il est aussi intéressant de mentionner le code OpenMoC, développé au MIT (Boyd *et al.*, 2014), qui incorpore dans son architecture même l'idée du parallélisme, plutôt que d'y adapter un code existant.

Les développements en lien avec MoC se sont dans un premier temps généralement concentrés sur les méthodes d'accélération numériques, que nous aborderons dans la section 3.4.2. Les dernières années, notamment avec le développement rapide de codes en 3-D, ont vu les besoins en temps de calcul décupler – d'où l'intérêt croissant vers les méthodes de calcul parallèle, et la plupart des codes mentionnés dans le paragraphe précédent proposent déjà leur implémentation parallèle, notamment APOLLO (Sciannandrone, 2015), MICADO (Févotte et Lathuilière, 2013), et DRAGON (Dahmani et Roy, 2005). Nous aborderons plus en détails ces aspects dans le chapitre 5.

3.3.2 Description

Sous sa forme *caractéristique*, l'équation de Boltzmann (2.7) mono-cinétique (on omettra les indices g de groupe) peut s'exprimer, le long d'un segment orienté dans la direction $\hat{\Omega}$, comme :

$$\frac{d}{ds}\phi(\vec{r} + s\hat{\Omega}, \hat{\Omega}) + \Sigma(\vec{r} + s\hat{\Omega})\phi(\vec{r} + s\hat{\Omega}, \hat{\Omega}) = Q(\vec{r} + s\hat{\Omega}, \hat{\Omega}) \quad (3.4)$$

Pour résoudre l'équation (3.4) sur l'ensemble des segments, la méthode des caractéristiques utilise une procédure de traçage, qui consiste à générer un ensemble Υ de lignes d'intégration \vec{T} . Chacune des lignes est définie uniquement et entièrement par un point de départ \vec{p} et une orientation $\hat{\Omega}$. En pratique, on définit Υ selon un ensemble de plans $\Pi_{\hat{\Omega}}$ de normale $\hat{\Omega} \in 4\pi$, sur lesquels on définit les points de départ \vec{p} des lignes. En utilisant la coordonnée s pour indiquer le déplacement sur la ligne d'intégration, on peut représenter les positions orientées dans l'espace comme $\vec{p} + s\hat{\Omega}$.

3.3.3 Schéma d'intégration

La résolution du flux s'effectue à l'aide du flux moyen sur un segment, défini comme :

$$\bar{\phi}_k(\vec{T}) = \frac{1}{L_k(\vec{T})} \int_0^{L_k(\vec{T})} \phi(\vec{r}_k + s\hat{\Omega}, \hat{\Omega}) ds \quad (3.5)$$

où on a utilisé L_k , la longueur de l'intersection entre la ligne caractéristique et la région k , et où \vec{r}_k est défini comme le point de l'intersection de la ligne caractéristique avec la frontière entrante de la région k (d'où la relation $\vec{r}_{k+1} = \vec{r}_k + L_k\hat{\Omega}$).

En prenant l'intégrale selon s de (3.4) le long du segment, on trouve la relation :

$$\phi_{k+1}(\vec{T}) - \phi_k(\vec{T}) + \Sigma_k L_k \bar{\phi}_k(\vec{T}) = L_k \bar{Q}_k(\vec{T}) \quad (3.6)$$

où on suppose que la section efficace est constante le long du segment et où la source moyenne sur un segment est : $\bar{Q}_k(\vec{T}) = \frac{1}{L_k} \int_0^{L_k} Q(\vec{r}_k + s\hat{\Omega}, \hat{\Omega}) ds$.

On voit donc que l'évaluation de l'intégrale de l'équation (3.4) requiert une dépendance spatiale concernant le terme source. Trois hypothèses sont généralement utilisées (Le Tellier et Hébert, 2006) : des sources constantes (*step characteristic*), un schéma diamant, ou encore des sources linéaires.

Step characteristic

L'hypothèse la plus répandue est celle du *step characteristic* (SC), qui suppose que le terme source est constant sur chaque segment de la trajectoire, s'exprimant comme :

$$Q(\vec{r}_k + s\hat{\Omega}, \hat{\Omega}) = Q_k(\hat{\Omega}) \quad \forall s \in [0, L_k] \quad (3.7)$$

où $Q_i(\hat{\Omega})$ est la source angulaire dans la région i , qu'on peut calculer comme :

$$Q_i(\hat{\Omega}) = \sum_{l=0}^L \frac{2l+1}{4\pi} \sum_{m=-l}^l \mathcal{R}_l^m(\hat{\Omega}) [\Sigma_{s,i}^l \phi_{l,i}^m + S_{l,i}^m] \quad (3.8)$$

En définissant le parcours optique comme $\tau_k(\vec{T}) = \Sigma_k L_k(\vec{T})$ sur le segment k (avec Σ_k la section efficace totale de la région correspondante), on utilisera un facteur intégrant $e^{-\tau_k(\vec{T})}$ pour résoudre analytiquement l'équation (3.4). On obtient alors :

$$\phi_{k+1}(\vec{T}) = \phi_k(\vec{T})\alpha^{SC} + Q_k(\hat{\Omega}) \quad (3.9)$$

$$\bar{\phi}_k(\vec{T}) = \phi_k(\vec{T})\frac{\beta^{SC}}{L_k} + Q_k(\hat{\Omega})\frac{\gamma^{SC}}{L_k} \quad (3.10)$$

où :

$$\begin{aligned} \alpha^{SC} &= e^{-\tau_k(\vec{T})} \\ \beta^{SC} &= \frac{1 - e^{-\tau_k(\vec{T})}}{\Sigma_k} \\ \gamma^{SC} &= \frac{L_k}{\Sigma_k} \left[1 - \frac{1 - e^{-\tau_k(\vec{T})}}{\tau_k(\vec{T})} \right] \end{aligned} \quad (3.11)$$

Schéma diamant

Il est aussi pertinent de mentionner le schéma diamant (DDO), qui a été abondamment exploré dans le cadre des méthodes S_N , et pour MoC par Suslov (1993). Ce schéma représente le flux moyen sur un segment comme :

$$\bar{\phi}_k(\vec{T}) = \frac{\phi_{k+1}(\vec{T}) + \phi_k(\vec{T})}{2} \quad (3.12)$$

ce qui mène à des relations similaires à celles présentées aux équations (3.9) et (3.10), mais où les coefficients sont donnés par :

$$\begin{aligned} \alpha^{DD} &= \frac{2 - \tau_k}{2 + \tau_k} \\ \beta^{DD} &= \frac{2L_k}{2 + \tau_k} \\ \gamma^{DD} &= \frac{L_k^2}{2 + \tau_k} \end{aligned} \quad (3.13)$$

Ce schéma est parfois plus précis que SC, tout en étant aussi compatible au niveau algorithmique. Il reste toutefois moins répandu car il est susceptible de générer des flux négatifs.

Schéma linéaire

Finalement, le schéma linéaire suppose une représentation linéaire de la source dans chaque région, de la forme :

$$Q(r_k + s\hat{\Omega}, \hat{\Omega}) = \bar{Q}_k(\vec{T}) + \left(s - \frac{L_k}{2}\right) Q_k^1(\hat{\Omega}) \quad (3.14)$$

La manière de déterminer la pente Q_k^1 de la source mène à différents schémas, qui peut inclure DD (pente proportionnelle au flux) ou SC (pente nulle).

Dans le cas général, cette approximation permet d'obtenir de meilleures représentations spatiales du flux, qui permettent l'utilisation d'un maillage plus grossier et de meilleurs temps de calcul. Toutefois, il importe de bien imposer certaines contraintes sur la représentation des pentes des sources afin de garantir la positivité (des sources et des flux) et la conservation des neutrons (et donc des sources par direction) du schéma.

Ce schéma requiert cependant des manipulations supplémentaires afin d'imposer la conservation, comme le font Santandrea et Sanchez (2002b) dans le schéma LS (*Linear Surface*), qui nécessite l'évaluation et le stockage des moments du flux pour chaque frontière de la région de calcul. Finalement, il est aussi intéressant de mentionner le schéma SLC (*Simplified Linear Characteristics*) (Le Tellier et Hébert, 2006), qui repose sur la continuité des sources dans des régions matérielles successives, qui peuvent correspondre à différentes régions du maillage de calcul. Plus récemment, les schémas SC et DDO du code DRAGON ont été généralisés aux sources linéaires en 2-D (Hébert, 2011, 2016).

3.3.4 Traçage

Peu importe le schéma choisi, en utilisant le formalisme des caractéristiques, on arrive à transformer les quantités intégrées sur le volume en intégrales sur l'ensemble du traçage Υ . La méthode des caractéristiques (tout comme la méthode des probabilités de collision d'ailleurs) repose donc sur un opérateur de sommation sur le traçage. Pour une fonction f quelconque, définie sur une région i de volume V_i , cet opérateur peut être représenté comme suit :

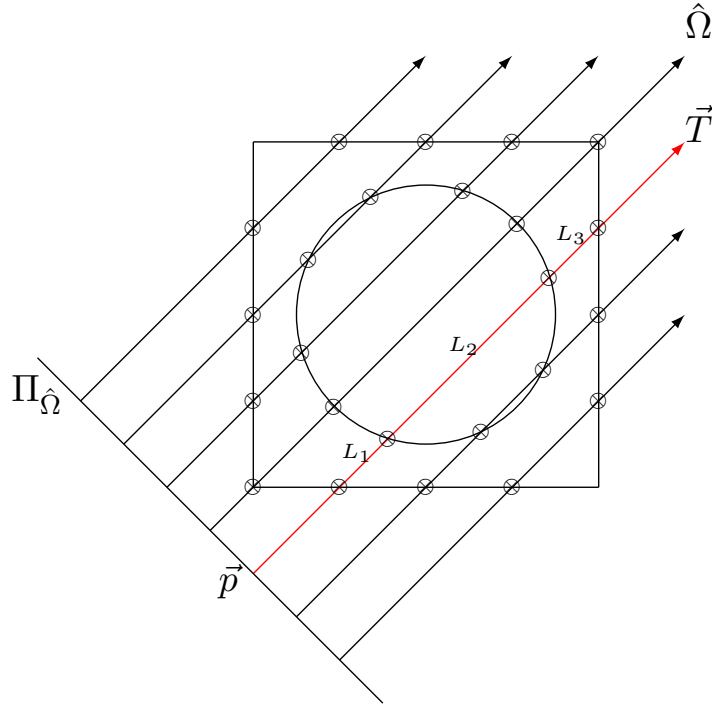


Figure 3.1 Traçage d'une cellule 2-D dans la direction $\hat{\Omega}$

$$\begin{aligned}
 \mathcal{F}_i &= \int_{V_i} \int_{4\pi} f(\vec{r}, \hat{\Omega}) d^2\Omega d^3r \\
 &= \int_{\Upsilon} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) f(\vec{T}) ds d^4T \\
 &= \frac{1}{4\pi} \int_{4\pi} \int_{\Pi_{\hat{\Omega}}} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) f(\vec{T}) ds d^2p d^2\Omega
 \end{aligned} \tag{3.15}$$

où on a utilisé la fonction caractéristique $\mathcal{W}_i(s, \vec{T})$ qui vaut 1 si le point s de la caractéristique \vec{T} est contenu dans la région i et 0 sinon.

En vertu de la formulation (3.15), les moments du flux (équation (2.26)) dans chacune des régions i du domaine, exprimés comme :

$$\phi_{l,i}^m = \frac{1}{V_i} \int_{V_i} \int_{4\pi} \mathcal{R}_l^m(\hat{\Omega}) \phi(\vec{r}, \hat{\Omega}) d^2\Omega d^3r \tag{3.16}$$

sont réécrits sous la forme :

$$\phi_{l,i}^m = \frac{1}{V_i} \int_{\Upsilon} \int_{-\infty}^{\infty} \mathcal{R}_l^m(\hat{\Omega}) \phi(\vec{p} + s\hat{\Omega}, \hat{\Omega}) \mathcal{W}_i(s, \vec{T}) ds d^4T \quad (3.17)$$

où on a transformé l'intégrale sur le volume en une intégrale sur le traçage Υ . En pratique, l'opérateur de l'équation (3.15) calcule les indices régionaux i et les longueurs L_k des segments dans chacune des régions traversées par chacune des lignes \vec{T} (voir figure 3.1).

On peut ainsi remplacer les intégrales $\int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) ds$ par $\sum_k \delta_{i,k} L_k$. Ceci nous permet donc de réécrire l'équation (3.17) comme :

$$\phi_{l,i}^m = \frac{1}{V_i} \int_{\Upsilon} \sum_k \delta_{i,k} L_k(\vec{T}) \mathcal{R}_l^m(\hat{\Omega}) \bar{\phi}(\vec{T}) d^4T \quad (3.18)$$

En pratique, comme il n'est pas envisageable d'effectuer une intégration purement analytique sur le traçage complet, on utilise une quadrature numérique comme celle des méthodes S_N de l'équation (3.1), à laquelle on ajoute des trajectoires parallèles afin de couvrir l'ensemble de l'espace dans chaque direction. On définit donc en plus des directions discrètes, une densité surfacique de trajectoires sur chaque plan $\Pi_{\hat{\Omega}_q}$ (de normale $\hat{\Omega}_q$), chaque trajectoire discrète (uniquement définie par un point $\vec{p}_{q,n}$ sur $\Pi_{\hat{\Omega}_q}$) étant aussi pondérée par un poids $\sigma_{q,n}$, de sorte que :

$$\int_{\Pi_{\hat{\Omega}_q}} d^2p = \sum_n \sigma_{q,n} = 1. \quad (3.19)$$

On remplace donc l'intégrale sur le traçage de l'équation (3.18) par une quadrature (pondérée par ω_q et $\sigma_{q,n}$) sur les trajectoires (individuellement définies comme $\vec{T}_{q,n}$), de manière à obtenir une évaluation numérique de l'équation (3.18), de la manière suivante :

$$\phi_{l,i}^m = \frac{1}{V_i} \sum_q \omega_q \sum_n \sigma_{q,n} \sum_k \delta_{i,k} \mathcal{R}_l^m(\hat{\Omega}_q) L_k(\vec{T}_{q,n}) \bar{\phi}(\vec{T}_{q,n}) \quad (3.20)$$

Finalement, à l'aide de ces moments du flux et de l'équation (2.29), on peut retrouver le flux angulaire. En pratique, un opérateur balaie chacune des lignes comme en témoignent les équations (3.9) et (3.10), et on met à jour les moments des sources et du flux à chaque itération interne.

Systeme matriciel

En réintégrant le formalisme multigroupe, on peut combiner les équations (3.8), (3.9), (3.10)

et (3.18), afin d'obtenir un système matriciel de g équations couplées, à résoudre pour calculer les moments du flux, de la forme :

$$\vec{\Phi}^g = \mathbf{T}^g \left(\vec{S}^g + \Sigma_s^g \vec{\Phi}^g \right) \quad (3.21)$$

où

- $\vec{\Phi}^g$ est un vecteur contenant les moments du flux $\phi_{l,i,g}^m$ pour chaque région dans le groupe g ;
- \mathbf{T}^g est une matrice d'intégration sur le traçage Υ , représentant un balayage le long des segments connectant les différentes régions entre elles ;
- \vec{S}^g est un vecteur contenant les moments des sources $S_{l,i,g}^m$ de fission et externes pour chaque région et chaque groupe, tels que définis à la section 2.3.2 ; et
- Σ_s^g est une matrice diagonale (en espace) contenant les moments $\Sigma_{s,g \leftarrow h,i}^l$ des sections efficaces de diffusion pour chaque région et chaque groupe.

On voit que la résolution de ce système nécessite un processus itératif sur les $\bar{\phi}$ et le terme source, en plus des conditions frontières, pour obtenir les moments du flux. Celui-ci permet aussi de calculer le flux intégré, utilisé pour déterminer les quantités d'intérêt comme les taux de réaction. Davantage de détails sur la méthode itérative et sa performance sont donnés dans la section 5.5.1.

3.3.5 Normalisation

En particulier, si on utilise $f \equiv 1$ dans l'équation (3.15), on obtient une estimation du volume à l'aide du traçage. En ne prenant que les intégrales sur les plans et les lignes d'intégration, on obtient le volume de la région i , angulairement dépendant (on n'intègre pas sur $\hat{\Omega}$ dans l'équation (3.22)), ou encore une évaluation numérique de celui-ci, moyenné sur 4π (équation (3.23)) :

$$\tilde{V}_{i,\hat{\Omega}} = \int_{\Pi_{\hat{\Omega}}} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) ds d^2p \quad (3.22)$$

$$\tilde{V}_i = \frac{1}{4\pi} \int_{4\pi} \int_{\Pi_{\hat{\Omega}}} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) ds d^2p d^2\Omega \quad (3.23)$$

Ces volumes numériques peuvent être comparés aux volumes réels des régions V_i afin de

calculer l'erreur introduite par la discrétisation due au traçage. On peut réduire cette erreur en raffinant davantage le pas de traçage, et en utilisant une quadrature angulaire plus élevée. Cela a par contre comme désavantage de grossir (parfois substantiellement) le fichier de traçage, et de ralentir le solveur de flux, qui doit le lire à chaque itération. Par ailleurs, plus on augmente le nombre de lignes, plus les poids associés à chacune deviennent petits, ce qui peut engendrer des instabilités numériques.

On calcule donc des facteurs de normalisation, qui permettront d'améliorer l'évaluation des quantités calculées par l'opérateur (3.15). Cette normalisation vient en quelque sorte corriger partiellement l'erreur introduite par la discrétisation du traçage. L'ampleur de cette correction peut par ailleurs être utilisée afin d'évaluer la précision du traçage, où une correction importante pourrait indiquer qu'une densité insuffisante de lignes a été utilisée. Ce genre de situation peut être détecté sur la base de critères utilisant les différences entre le volume numérique et analytique de la région, et notamment son maximum, l'erreur RMS relative ou sa moyenne par direction.

De manière générale, on normalise chaque segment en fonction des volumes directionnels. Pour ce faire, on multiplie la longueur de chaque segment par un ratio entre le volume numérique angulaire et le volume « analytique » (en pratique, V_i est aussi un volume calculé numériquement, mais à l'aide des formules géométriques appropriées – nous utiliserons dans le reste du document l'appellation analytique pour désigner les volumes calculés de cette façon), de la manière suivante :

$$\tilde{L}_k = \frac{V_i}{\tilde{V}_{i,\Omega}} L_k \quad (3.24)$$

et c'est ce nouveau \tilde{L}_k qui est utilisé dans le processus de résolution. L'approche utilisant les volumes angulaires est généralement privilégiée, en particulier avec la méthode des caractéristiques, car on ne traite pas toujours les quantités intégrées angulairement, particulièrement dans les cas de diffusion linéairement anisotrope.

3.3.6 Conditions frontières

Mathématiquement, pour fermer le système, on doit avoir recours à des conditions frontières. La résolution du système sur une géométrie finie (un domaine \mathcal{D} bordé par une surface \mathcal{S}), nécessite, pour commencer le processus itératif, la connaissance du flux sur la frontière entrante (ϕ_1). Nous nous contenterons ici d'une discussion qualitative des conditions aux frontières

et de leur utilisation et référons le lecteur curieux à Sanchez *et al.* (2002). En référence à la géométrie située à l'extérieur du domaine, on note les différents types de conditions de symétrie à la frontières suivantes, selon que la cellule adjacente correspond à une translation ou à l'image miroir de la cellule considérée :

- **Translation**, ou périodique, de façon à ce que le flux sortant du domaine est transféré comme ré-entrant à l'autre extrémité du domaine.
- **Réflexion**, où le flux sortant du domaine est réfléchi par rapport à la normale au plan de la frontière au même endroit

De plus, venant s'additionner à ces conditions, on peut aussi imposer des conditions à la frontière. Selon que l'on simule exactement la réflexion ou la transmission de la trajectoire caractéristique explicitement ou pas, on a des conditions :

- **Directionnelles**, selon laquelle soit les neutrons (et donc la caractéristique) qui rencontrent la frontière du domaine sont simplement réfléchis à un angle symétrique. Ce type de conditions frontières n'a toutefois été appliqué qu'en 2-D pour la méthode des caractéristiques car il requiert un processus de traçage particulier, cyclique, dépendant de la topologie de la géométrie et permettant de connecter les lignes d'intégration, qui deviennent périodiques, de manière à ce que $\vec{r}_1 = \vec{r}_{K+1}$. Ces conditions sont utilisées dans le module **MOCC** de DRAGON3 ou dans le module **MCCGT** de DRAGON4 si l'option **TSPC** a été sélectionnée.
- **Isotropes**, selon laquelle une fraction d'albédo β des neutrons atteignant une frontière (flux scalaire) sont réémis de manière isotrope vers l'intérieur du domaine. En particulier, mentionnons les conditions d'albédo nul ($\beta = 0$), qui correspondent à un flux ré-entrant à la frontière nul, qui nous intéressera dans le cadre de la modélisation d'un réacteur complet. Dans le cas des conditions de translation mentionnées ci-haut, on parlera de coefficient de transmission plutôt que d'albédo. Ces options sont activées par le mot-clé **TISO** dans DRAGON.

3.4 Schéma numérique de résolution du système

Au final, le processus de résolution numérique de l'équation de transport se concrétise dans la plupart des méthodes autour de trois boucles d'itération imbriquées afin d'obtenir des convergences spatiales, sur le terme source de diffusion, et sur le terme source de fission :

Itérations de puissance Ce premier niveau d'itérations tente de converger sur la source

de fission. On met à jour la source de fission à chaque itération, qu'on considère fixe pour les deux niveaux inférieurs ;

Itérations multigroupe Cette boucle d'itération tente de converger sur la diffusion des neutrons à travers la plage énergétique. On considère ici le terme de source de fission connu. Comme les neutrons naissent généralement dans la région rapide et ralentissent presque uniformément jusque dans les régions énergétiques thermiques, la matrice de diffusion contenant les sections efficaces est en grande partie triangulaire inférieure, et les sources de diffusion dans les groupes les plus énergétiques ne dépendront que du flux dans le groupe lui-même ou dans les groupes précédents. Le processus de résolution est donc séquentiel jusqu'au groupe où apparaît le *up-scattering* (Siegel *et al.* (2008)) ;

Itérations internes ou spatiales (si requises) En considérant les sources de diffusion et de fission connues, on peut résoudre l'équation mono-cinétique spatialement. C'est ici qu'intervient la méthode des caractéristiques dans la résolution de l'équation de transport, qui utilise un schéma itératif pour converger sur les flux angulaires dans chacune des régions.

3.4.1 Résolution spatiale

En règle générale, pour un problème à n régions spatiales et à l surfaces extérieures, la méthode des probabilités de collision, historiquement utilisée pour résoudre les problèmes de transport neutronique, requiert le stockage et l'inversion d'une matrice pleine, de dimension $\frac{(n+1) \times (n+l+1)}{2}$ dans le cas typique (diffusion isotrope), mais qui peut être réduite à $\frac{n(n+1)}{2}$ selon les conditions frontières. Dans tous les cas, il s'agit d'un problème qui croît typiquement en $O(n^2)$ avec la taille du problème (Marleau (2001)). La méthode des caractéristiques quant à elle requiert la résolution d'un système à $[(n+l) \times (L+1)^2]$ inconnues (section 3.3.4), pour tous les moments du flux et les flux aux frontières développés à l'ordre L . Ce système, quant à lui, croît typiquement en $O(n)$ avec la taille. Dans ce contexte, on peut observer que la méthode des caractéristiques, bien que coûteuse en termes d'opérations de calcul à effectuer à cause de sa nature itérative pour la résolution spatiale du flux, peut traiter *a priori* des problèmes de dimensions spatiales plus étendues que la méthode des probabilités de collision, en plus de pouvoir traiter facilement l'anisotropie.

Chacune des itérations MoC internes mentionnées à la section précédente requiert un balayage de l'ensemble du traçage. La taille maximale du problème à traiter reste donc toutefois limitée par l'espace disque disponible pour le fichier de traçage, celui-ci pouvant devenir rapidement très volumineux. Afin de contourner cette limitation, le procédé de traçage en ligne (*inline*

tracking) peut être utilisé, par lequel les lignes d'intégration sont générées à la volée, au fur et à mesure qu'elles sont requises par le solveur plutôt que d'être stockées dans un fichier sur le disque dur (Dahmani *et al.* (2004)). Cependant, les implantations de MoC couramment utilisées dans le code Dragon ne tirent pas entièrement profit de cet avantage, puisque le module de *tracking* crée un fichier de lignes d'intégration stocké en mémoire pour être par la suite utilisé par le solveur MoC.

Ce processus requiert toutefois de recalculer les intersections à chaque fois qu'elles sont nécessaires, ce qui implique un certain nombre d'opérations additionnelles. Dans tous les cas, un compromis doit être fait entre le stockage (et les accès disque ou mémoire) et le calcul à la volée. Un point médian entre ces deux méthodes peut être imaginé impliquant le développement d'un processus de représentation prismatique de la géométrie qui fera l'objet du chapitre 4.

3.4.2 Méthodes d'accélération

Comme il a été mentionné, la méthode des caractéristiques repose sur un processus itératif dont la convergence spatiale peut être longue et ardue (par exemple pour des cas fortement diffusifs), demandant ainsi de fortes ressources informatiques. Il est donc important d'examiner la possibilité d'utiliser des méthodes d'accélération afin d'améliorer la vitesse de convergence des processus itératifs décrits ci-dessus. On classe ces méthodes selon deux catégories principales :

- Les méthodes **synthétiques**, (Santandrea et Sanchez, 2002a) qui reposent sur un pré-conditionnement d'une formulation matricielle du processus de résolution du flux, qui apporte une correction sur les valeurs estimées des flux. Ces méthodes regroupent les méthodes synthétiques de diffusion (*Diffusion Synthetic Acceleration – DSA*), les plus répandues, qui utilisent en général une expansion angulaire du flux pour approximer le comportement de l'équation de transport, comme dans les méthodes aux harmoniques sphériques (*Asymptotic Synthetic Acceleration – ASA*). On inclut aussi ici la méthode ACA (pour *Algebraic Collapsing Acceleration*), implantée dans DRAGON par Le Tellier (2006), qui elle repose sur une simplification de l'opérateur d'intégration sur le traçage.
- Les méthodes **itératives**, qui remplacent les itérations internes libres par des itérations forçant une convergence plus rapide. Les techniques employées pour ce faire sont multiples. La plus connue, et celle implantée dans DRAGON pour la méthode des caractéristiques, GMRES (Saad.Y., 2003), tente de minimiser le résidu à chaque ité-

ration interne, ce qui ne permet toutefois pas nécessairement d’obtenir la meilleure accélération. Elle peut être avantageuse lorsque combinée avec les méthodes de pré-conditionnement dans certaines conditions, mais l’avantage qui en est retiré dépend fortement du problème considéré (Dahmani *et al.* (2005)).

Les méthodes d’accélération synthétiques résolvent à chaque itération un système dégradé. Cette forme dite *corrective* de l’équation permet de trouver un flux correctif, qui, dans l’implantation ACA de DRAGON, se formule comme une simple addition sur les premiers moments du flux (flux scalaires) (Le Tellier et Hébert (2007)), et peut aussi être utilisée pour fournir une meilleure initialisation de ces mêmes flux (Le Tellier *et al.* (2006)).

Une technique de condensation de lignes (TMT - *Track Merging Technique*) (Wu et Roy, 2003) est aussi utilisée en conjugaison avec cette dernière et permet de regrouper les différentes lignes qui traversent les mêmes régions dans le même ordre, et d’accélérer encore la convergence. Finalement, mentionnons aussi la méthode SCR (*self-collision rebalancing*) (Wu et Roy, 1999), reposant sur une équivalence entre les termes sources de diffusion à l’intérieur du même groupe de la méthode des caractéristiques et celui de la méthode des probabilités de collision, qui a donné de bons résultats en conjonction avec GMRES et ACA (Le Tellier (2006)). Toutes ces méthodes d’accélération sont en général compatibles entre elles (sans toutefois garantir une meilleure convergence), et avec les techniques de traçage en ligne, et de projection prismatique de la géométrie (Dahmani *et al.*, 2008).

3.5 Calcul de coeur

La procédure de calcul actuelle, qui permet d’évaluer le flux et les taux de réaction, est dite multi-niveaux. À cet effet, un calcul de transport (tel que présenté dans les chapitres 2 et 3) sur une partie représentative du réacteur (une *cellule*), est effectué – généralement en 2-D – dont on tire des propriétés qui sont homogénéisées spatialement et condensées énergétiquement. Ces propriétés uniformisées sont par la suite utilisées pour un calcul à l’échelle du réacteur complet (en 3-D) à l’aide d’un opérateur de transport dégradé ou simplifié.

Cette procédure permet au final de simplifier considérablement le problème en comparaison à un calcul de transport par la méthode des caractéristiques dans des géométries non structurées, lorsqu’appliqué à l’échelle du coeur du réacteur complet. Cependant, comme toute approximation au niveau physique ou mathématique, elle entraîne un certain niveau d’imprécision supplémentaire dû à l’utilisation d’un opérateur dégradé, et à une représentation moins fidèle du système physique sous-jacent.

Notamment, l'introduction dans l'équation de transport de l'opérateur de diffusion à l'aide de la loi de Fick en remplacement de l'opérateur de transport a des conséquences sur la modélisation des phénomènes physiques. On peut noter par exemple les effets aux frontières du domaine dans des conditions de vide (section 3.3.6), où l'on doit inclure dans les modèles soit une distance d'extrapolation (Glasstone et Sesonske, 1994) afin d'éviter un flux nul à la frontière plutôt qu'un flux réentrant nul, soit une relation approximative pour représenter le flux entrant nul (Hébert, 2009, Sec 5.24).

L'approximation de diffusion a historiquement eu l'avantage indéniable de simplifier considérablement le problème mathématique à résoudre, ce qui dans un contexte numérique se traduit par un temps de calcul considérablement réduit, tout en tenant compte des phénomènes 3-D pouvant se produire dans le réacteur. Même avec l'introduction de la possibilité du calcul prismatique et parallèle dans les chapitres suivants, on ne peut envisager dans un avenir proche de remplacer en totalité l'utilisation des méthodes de diffusion, notamment pour simuler l'opération d'un réacteur. On peut dans un premier temps utiliser ces nouvelles méthodes dans un contexte de validation de nouvelles améliorations apportées au modèles de diffusion, puis les utiliser dans des études de cas statiques puis en évolution.

CHAPITRE 4 REPRÉSENTATION PRISMATIQUE ET TRAÇAGE

4.1 Définition

Les géométries des réacteurs nucléaires ont, du point de vue du calcul numérique, l'avantage de présenter – généralement – une certaine forme d'invariance selon un axe principal : vertical pour les réacteurs REP ou REB, horizontal pour les réacteurs CANDU ou ACR. Cet axe privilégié a historiquement permis des calculs assez précis et représentatifs de la physique sous-jacente pour les besoins du moment en utilisant des calculs 2-D sur une ou plusieurs coupes axiales. L'introduction des calculs 3-D permet une prise en compte de phénomènes physiquement plus complexes, au coût cependant d'un nombre d'opérations numériques d'autant plus élevé.

L'implantation de la méthodologie prismatique présentée ici permet de combiner les deux avantages. On obtient donc la précision de la méthode 3-D, tout en conservant des besoins en calcul réduits en tirant avantage numériquement de l'axe privilégié dans le réacteur. Cependant, on doit tenir compte du fait que l'aspect prismatique n'est généralement pas constant sur toute la hauteur du réacteur, avec des changements de composition et de géométrie.

4.2 Applicabilité

Une méthodologie de représentation prismatique de la géométrie déjà implantée dans le logiciel DRAGON par Le Tellier (2006) a été validée notamment dans la modélisation des barres d'acier de contrôle des réacteurs ACR. Cependant, elle ne permet que de représenter des géométries prismatiques uniformément sur toute la hauteur – c'est-à-dire équivalentes à des géométries 2-D comportant plusieurs coupes axiales. Comme cette méthode ne repose que sur une transformation géométrique, mentionnons tout de même qu'il était possible d'y faire varier la composition du milieu, pour ainsi tenir compte d'une certaine non-uniformité axiale.

Le complément de la méthode proposé ici tend à élargir le champ des géométries qu'il sera possible de traiter avec cette méthode, se rapprochant d'autant plus d'une réelle représentation tridimensionnelle. En faisant varier la géométrie axialement, on pourra alors représenter la plupart des éléments faisant partie du cœur de la plupart des réacteurs nucléaires.

Dans les faits, cette méthode implique une capacité de traitement pour toute géométrie cartésienne, qui peut inclure des raffinements cylindriques ou des crayons, en autant que tous sont orientés selon le même axe, qui devient alors l'axe de projection. Cela permet de modéliser plus efficacement certaines configurations de réacteurs comme :

- celles des CANDUs lorsque deux grappes de combustibles consécutives ne sont pas alignées angulairement ;
- des barres de contrôle partiellement insérées dans des REP ;
- les grilles ou les réflecteurs inférieurs et supérieurs encore dans les REP, qui ne nécessitent pas le même niveau de discrétisation que le combustible ou le modérateur.

L'avantage principal de cette nouvelle généralisation vient donc de la possibilité de modifier axialement la discrétisation en fonction de la hauteur. On pourra donc envisager, dans l'exemple d'une barre de contrôle partiellement insérée, de mailler plus grossièrement le trou d'eau laissé par son passage que la barre elle-même. Nous présenterons aussi plus loin des résultats concernant un assemblage dont la section centrale a été décalée par rapport à son centre, suite à un fluage dû à des flux neutroniques et thermiques, de même qu'aux contraintes mécaniques.

4.3 Formalisme

Afin de bien exposer la méthode, nous débuterons par une présentation du formalisme utilisé, se basant sur l'opérateur de traçage, ou *tracking*, qui a été introduit dans la section 3.3.4, que nous découplons d'abord pour en présenter la formulation s'appliquant aux géométries explicitement prismatiques, avant de développer les aspects de projection et d'extrusion propres à la méthode implantée ici.

Ainsi, l'opérateur de traçage présenté à l'équation (3.15) peut être simplifié sans perte de précision si on considère une géométrie globale dont tous les éléments présentent une invariance axiale selon un axe quelconque (par exemple, une géométrie 2-D, ou 3-D uniforme axialement). On désigne ici cet axe par la coordonnée z , mais une géométrie présentant une invariance selon x ou y peut aussi être traitée, en vertu d'une simple rotation du système d'axes. On peut dès lors décomposer l'intégration d'une fonction f quelconque sur les directions angulaires $\hat{\Omega}$ selon les angles azimutal (φ) et polaire ($\mu = \cos(\theta)$) comme :

$$\int_{4\pi} f(\hat{\Omega}) d^2\Omega \equiv \int_0^{2\pi} \int_{-1}^1 f(\varphi, \mu) d\mu d\varphi \quad (4.1)$$

Cette décomposition nous permet d'évaluer l'intégrale sur $\Pi_{\hat{\Omega}}$ en utilisant une base orthogonale $(I_{\perp}, I_{\parallel})_{\hat{\Omega}}$, de normale $\hat{\Omega}$. Sur cette nouvelle base, en se déplaçant le long de p_{\perp} sur I_{\perp} , on demeure dans le plan (x, y) de la géométrie projetée, alors qu'un déplacement le long de p_{\parallel} sur I_{\parallel} sera orthogonal à la fois à I_{\perp} et à $\hat{\Omega}$ (voir figure 4.1). On peut donc réécrire l'opérateur de traçage (équation (3.15)) comme :

$$\mathcal{F}_i = \frac{1}{4\pi} \int_{4\pi} \int_{\Pi_{\hat{\Omega}}} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) f(\vec{T}) ds d^2p d^2\Omega \quad (4.2)$$

$$= \frac{1}{4\pi} \int_0^{2\pi} \int_{-1}^1 \int_{I_{\perp}} \int_{I_{\parallel}} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) f(\vec{T}) ds dp_{\parallel} dp_{\perp} d\mu d\varphi \quad (4.3)$$

$$= \frac{1}{4\pi} \int_{-1}^1 \int_{I_{\parallel}} \left[\int_0^{2\pi} \int_{I_{\perp}} \int_{-\infty}^{\infty} \mathcal{W}_i(s, \vec{T}) f(\vec{T}) ds dp_{\perp} d\varphi \right] dp_{\parallel} d\mu \quad (4.4)$$

On observe par ailleurs que la variation des lignes en fonction seulement de p_{\perp} et φ se résume à un traçage 2-D conventionnel.

Au moment de la discrétisation par l'opérateur de traçage, on choisit des quadratures angulaires qui sont le produit d'une quadrature azimutale et polaire. On obtient ainsi un procédé de traçage qui peut être séparé en deux parties distinctes. Dans une première étape, on génère Υ_{\perp} qui considère seulement les variations selon φ et p_{\perp} dans une géométrie 2-D résultant de la projection de la géométrie globale dans un plan orthogonal à z . On génère par la suite les lignes d'intégration en 3-D à partir de Υ_{\perp} suivant μ . Ce traçage secondaire est toutefois grandement simplifié en tenant compte du fait que les régions du domaine qu'il doit traverser seront entièrement cartésiennes, et que les longueurs à considérer sont des multiples μ des longueurs déjà calculées dans Υ_{\perp} .

4.3.1 Séparation de la géométrie en sections uniformes

Cet algorithme de traitement des géométries invariantes axialement peut être poussé plus loin, pour considérer les géométries non-uniformes axialement. Celles-ci doivent cependant toujours présenter des invariances axiales au niveau plus local, à l'échelle de la cellule par exemple. Il s'agit donc de la relaxation d'une contrainte supplémentaire par rapport au traitement prismatique uniforme, qui reste d'ailleurs toujours compatible avec la nouvelle méthodologie implantée.

En pratique, on procède en commençant par une série de tests effectués par le module de traçage dans le but de vérifier que tous les éléments qui composent la géométrie possèdent

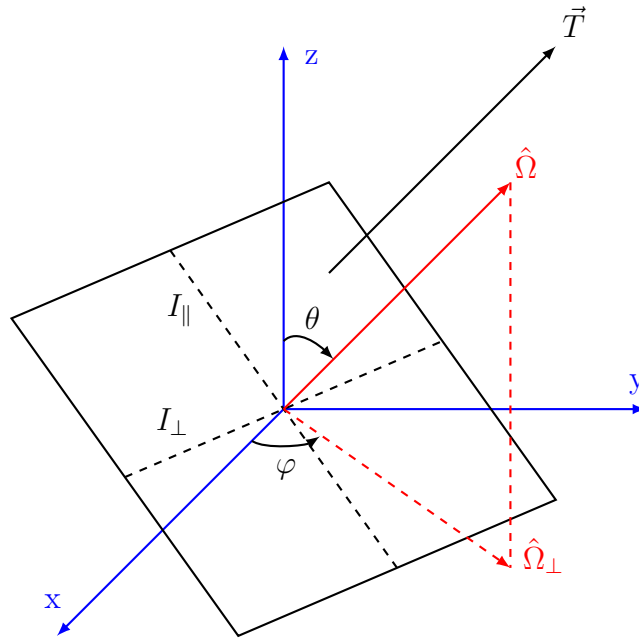


Figure 4.1 Découplage des dépendances angulaires en φ et θ ¹

une symétrie selon l'axe considéré. Une implantation de ce type a déjà été effectuée dans DRAGON avec succès par Le Tellier *et al.* (2008), mais ne considérait que les géométries avec une invariance axiale s'étendant sur toute la hauteur de la géométrie. La méthode que nous proposons implique d'étendre ce procédé pour permettre des variations géométriques selon l'axe principal (tant que celles-ci respectent la symétrie axiale). Deux approches peuvent être envisagées pour ce faire :

- Une décomposition selon plusieurs plans (x, y) , un pour chaque changement de géométrie selon l'axe de projection ;
- Une projection de l'ensemble de la géométrie dans un plan unique, « super-discrétisé », qui regroupe tous les éléments géométriques sur toute la hauteur du réacteur.

C'est cette dernière approche qui a été retenue, dans l'optique où l'on cherche principalement à minimiser les données partagées en mémoire lors du calcul, et que plusieurs Υ_{\perp} (requérant d'autant plus de capacité de stockage) devant être générés, la décomposition en plusieurs plans ne conviendrait pas. De plus, aux interfaces des différentes subdivisions en z , la connectivité des lignes d'intégration est loin d'être assurée, et dépendra de la hauteur à laquelle elle survient et de la quadrature polaire choisie.

On subdivise donc la géométrie en fonction de certaines hauteurs, qui correspondent chacune à un changement de la discrétisation, incluant au minimum un plan par cellule superposée.

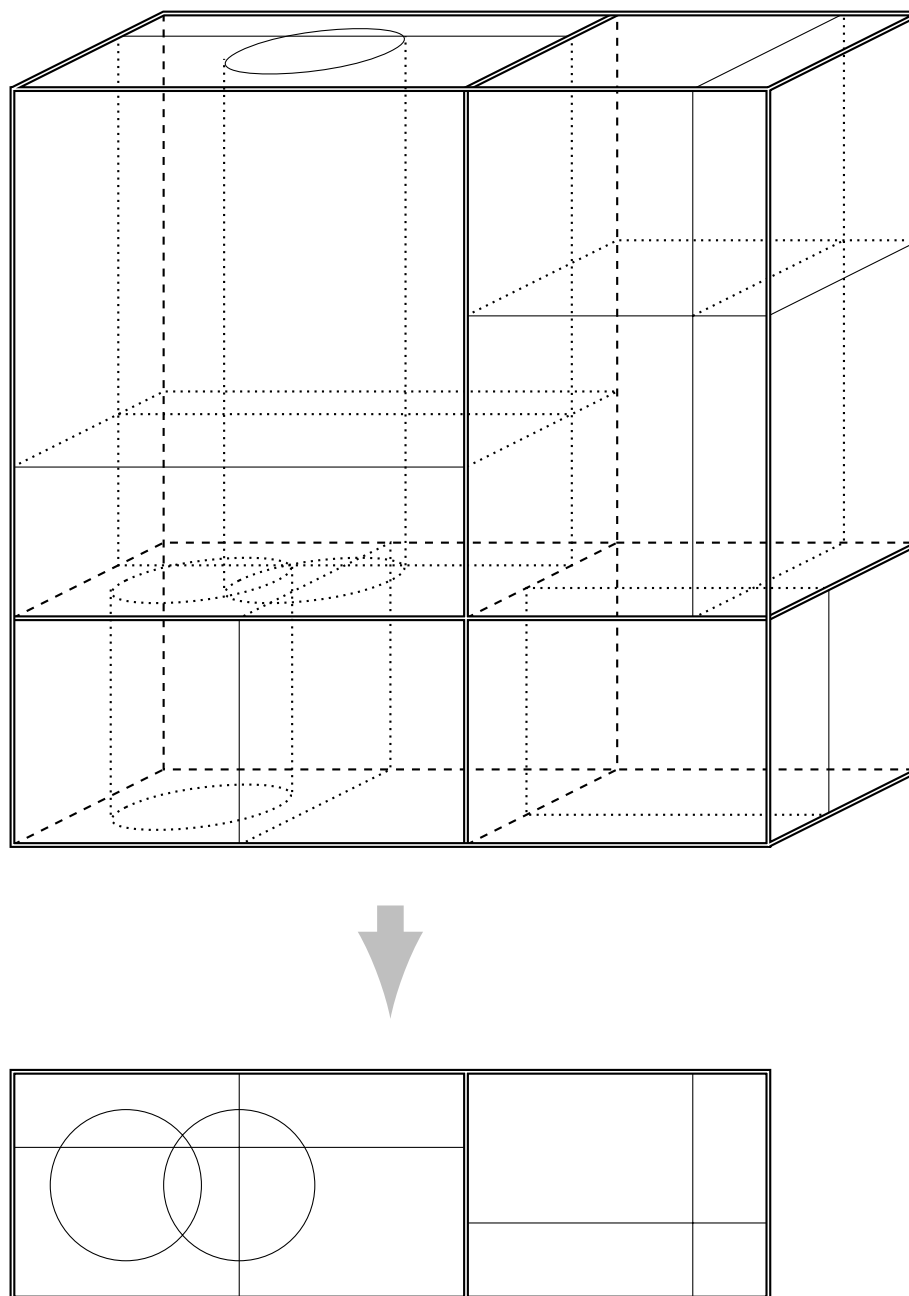


Figure 4.2 Projection d'une géométrie prismatique 3-D \rightarrow 2-D « superdiscretisée »

Chacun de ces plans correspond à une coupe/tranche radiale de la géométrie à cette hauteur, et contient les informations nécessaires pour conserver les informations de la géométrie originale. On peut voir ce processus en application aux figures 4.3 et 4.4 où nous illustrons le tout par un exemple plus concret : le bout d'un crayon de combustible.

La géométrie utilisée pour simuler un tel crayon est typiquement composée de deux cellules superposées. La première contient le crayon, sa gaine ainsi que le modérateur / caloporteur. La seconde est remplie du même modérateur/calporteur, discrétisée (ou pas) typiquement selon un maillage cartésien. Une telle géométrie représentée dans l'algorithme prismatique est composée de deux plans, un par cellule, qui sont par la suite combinés (voir la section 4.4.1) ainsi qu'une matrice de correspondance (voir la section 4.4.2) afin de ne perdre aucune information en lien avec la géométrie originale.

4.4 Projection de la géométrie

Les développements effectués dans cette thèse ont été incorporés dans une version de développement du code DRAGON4 (Marleau *et al.*, 2006).

Dans le cadre du développement de la méthode dans le module `NXT` de DRAGON, la première étape du traitement prismatique de la géométrie consiste donc, comme dans une géométrie normale, à analyser les éléments géométriques qui la composent, ainsi qu'à en calculer les volumes analytiques pour utilisation subséquente lors du processus de normalisation. On procède ensuite à la projection de la géométrie.

Une difficulté supplémentaire découle de la géométrie qui peut maintenant varier selon l'axe principal z . Plusieurs régions avec des géométries différentes vont donc être superposées dans la projection. Il est donc nécessaire d'élaborer une méthode permettant d'établir une correspondance entre les indices régionaux qui soit à la fois complète et assez simple pour être accessible rapidement par le solveur MoC.

4.4.1 Création de la géométrie projetée

Les deux principaux défis consistent à réaliser, *primo*, la projection de la géométrie 3-D dans un plan 2-D, pour lequel une correspondance parallélépipède-rectangle ou cylindre-cercle est effectuée, et *secundo*, la réalisation d'un schéma de numérotation des régions et surfaces de cette nouvelle géométrie, doublée d'un schéma d'indexation des régions, incluant

1. Image inspirée de Le Tellier *et al.* (2008)

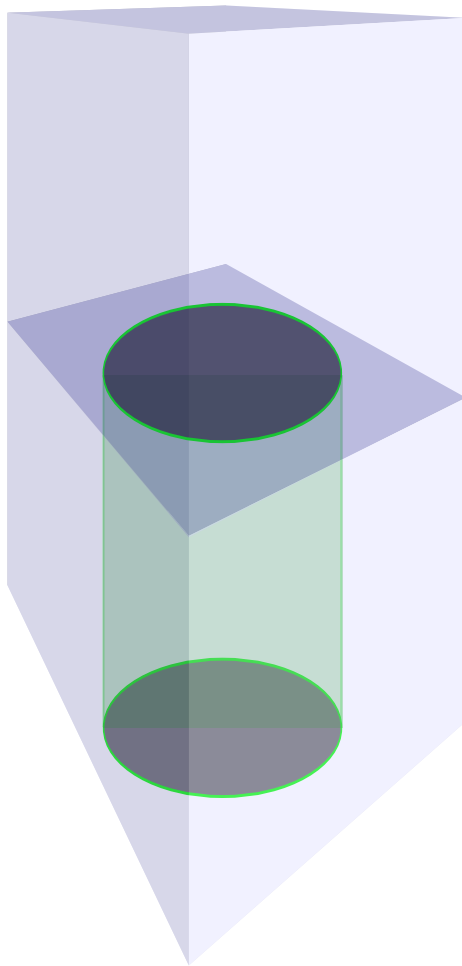


Figure 4.3 Bout de crayon

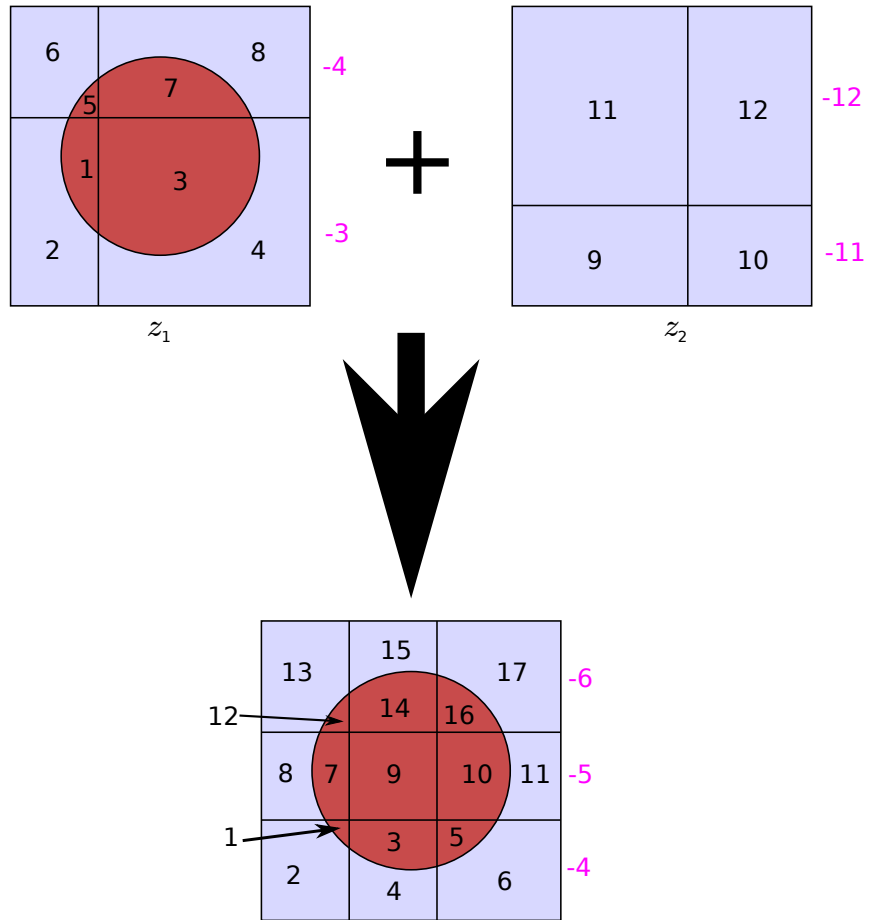


Figure 4.4 Combinaison de plans 2-D

la correspondance de la numérotation entre les géométries 3-D et la géométrie projetée en 2-D. Cela permet de préserver les indices régionaux initiaux lorsque vient le temps de reconstruire en extrusion les lignes d'intégration en 3-D.

La procédure visant à effectuer cette étape doit donc premièrement générer la géométrie « super-discrétisée » en deux dimensions. La méthode déjà implantée utilisait pour ce faire le plan initial ($z = 0$), qui contenait déjà toutes les informations sur la géométrie, et notait seulement les positions des plans subséquents.

À l'intérieur d'une boucle sur les cellules présentes dans le plan ($X - Y$), un balayage en z a été ajouté, pour tenir compte des discrétisations supplémentaires qui n'apparaissent pas originellement dans ce plan (voir figure 4.2). Pour l'instant, cette discrétisation se limite à n'importe quel maillage cartésien, couplé avec une ou plusieurs discrétisations annulaires du type CARCEL au niveau de la cellule.

La gestion des crayons vient cependant poser un problème supplémentaire, étant donné qu'ils se superposent à la géométrie de la cellule. Dans ce cas, le calcul des régions 2-D par les routines habituelles ne peut être appliqué. Par contre, cela ne vient poser un problème que pour la détermination de la précision du traçage, car la normalisation est effectuée à partir des volumes 3-D. Reste que le traçage, dans sa gestion habituelle, ne peut tenir compte de ces recouvrements géométriques. Des modifications ont donc été apportées afin de permettre la gestion de cas où la projection des crayons ne se recouvrent pas. Plusieurs crayons peuvent donc être décrits dans la géométrie, mais ne seront traitables que lorsqu'ils sont concentriques, ou qu'ils ne se recouvrent pas.

4.4.2 Matrice de correspondance

Une fois les propriétés géométriques de la projection obtenues, on utilise les routines de numérotation traditionnelles afin de fournir un repère cohérent pour la gestion de cette nouvelle géométrie, en plus de simplifier la gestion des surfaces qui ne sont pas externes à la géométrie globale, et des régions qui disparaissent dans certaines configurations.

Dans cette optique, la seconde étape consiste à créer une matrice de correspondance qui permet de reconstruire la géométrie originale sans perte d'information. Celle-ci sera de taille $[(N_Z + 1) \times (N_P + M_P)]$ où N_P et M_P sont respectivement le nombre de régions et le nombre de surfaces externes dans la géométrie projetée, et N_Z est le nombre total de plans perpendiculaires à z sur toute la hauteur de la géométrie ($N_Z = 5$ dans la figure 4.2). La première et la dernière ligne de cette matrice servent à stocker les correspondances pour les

surfaces supérieure et inférieure de la géométrie originale, alors que les autres correspondent aux indices régionaux (ou surfaciques) de la géométrie originale à la hauteur z correspondante. Il s'agira entre autres d'établir les limites en z de chacune des régions qui ne s'étendent pas sur toute la hauteur de la géométrie globale, ou même d'une cellule.

Dans ce cadre, un algorithme d'identification des régions et des surfaces boucle sur la hauteur en z afin de rétablir la correspondance géométrique entre les éléments surfaciques ou volumiques des deux géométries, qui s'incarne dans le remplissage de cette table de correspondance.

Au final, pour peu que la géométrie originale comporte un moindrement de variations selon l'axe de projection, cette nouvelle géométrie 2-D contiendra éventuellement plus de régions (et de surfaces) qu'à n'importe quelle hauteur le long de l'axe de projection, et le stockage des données pour ce faire a donc dû être modifié en conséquence.

4.4.3 Exemple

Un exemple simple de combinaison de géométries est présenté aux figures 4.3 et 4.4. Ici, la géométrie comprend deux plans de discrétisation différents, un contenant le crayon (z_1), et un comportant une discrétisation radiale différente dans le modérateur (z_2), au dessus. La numérotation des régions 3-D est indiquée sur la figure 4.4, et la matrice de projection correspondante est indiquée au tableau 4.1.

Un cas aussi simple présente l'avantage indéniable de comporter très peu de régions spatiales et de surfaces externes, et permet donc de simplifier l'analyse de la projection, ainsi que d'exposer le fonctionnement de l'algorithme de projection et de correspondance. On peut d'ores et déjà constater que la géométrie 2-D projetée est en elle-même, pour ce cas simple, plus complexe que chacun des deux plans individuellement. Mentionnons par ailleurs qu'il est donc possible de générer des projections comportant davantage de régions 2-D que la géométrie 3-D. Des géométries 2-D très complexes peuvent donc être générées rapidement, pour des géométries 3-D comportant un nombre élevé de variations axiales.

En revenant à l'exemple, on remarque donc qu'aux numéros identifiants de surface (négatifs), correspondent les surfaces équivalentes de la géométrie originale, ainsi que les régions. On peut donc observer qu'une région 3-D ne correspond plus à une seule et unique région 2-D, mais peut se retrouver à différentes positions, toujours adjacentes. Il s'agit là de frontières qui sont virtuelles, et correspondent à une division qui ne couvre pas toute la hauteur de la géométrie.

Mentionnons aussi que dans le tableau 4.1, deux colonnes supplémentaires sont nécessaires afin de tenir compte des surfaces extérieures dont la normale est colinéaire avec l'axe de projection, autrement dit, les surfaces du dessus et du dessous. Dans ce cas, les identifiants des surfaces supérieures et inférieures correspondant à des surfaces 2-D sont nuls, car ils représentent des arêtes de la géométrie originale, qui sont inutiles ici.

Les surfaces externes peuvent aussi contenir des frontières virtuelles, au même titre que les régions et les surfaces latérales, mais elles n'influencent pas le processus de concaténation, et l'extrusion seulement dans la mesure où deux lignes extrudées contribuent au même flux entrant ou sortant.

4.4.4 Extrusion et concaténation

Une fois le traçage 2-D généré, il ne suffit évidemment pas de l'analyser comme tel, si l'on désire obtenir un calcul de flux 3-D. Il faut donc, au niveau du solveur, effectuer une extrusion de chaque ligne 2-D, ainsi qu'une concaténation des segments aux frontières virtuelles.

C'est donc ici qu'intervient le traçage secondaire simplifié mentionné plus tôt, sur p_{\parallel} et μ , qui calcule les longueurs comme des simples multiples de $\frac{1}{\mu}$ des longueurs déjà calculées dans Υ_{\perp} , en fonction des intersections avec les plans z . Les longueurs en (x, y) étant déjà calculées, le processus restant s'apparente à celui sur une géométrie 2-D finie, dont les frontières seront les divisions z de la géométrie originale.

L'extrusion comporte donc deux composantes, soit une composante bouclant sur tous les angles polaires, et une seconde considérant le décalage sur p_{\parallel} , qui permet de retrouver la densité surfacique d'une quadrature 3-D classique. On retrouve cette manière de faire illustrée à l'algorithme 1.

Là où la méthodologie proposée diffère fondamentalement de l'implantation précédente concerne la recombinaison des segments. Au niveau du solveur, il s'agit de rétablir, à l'aide de la matrice de correspondance décrite plus haut, les intersections des lignes tri-dimensionnelles reconstituées avec les frontières régionales qui peuvent être « virtuelles » ou réelles, dépendant de la position dans la géométrie.

En pratique, on analyse les indices régionaux des lignes au moment de l'extrusion pour la reconstruction. Au moment de changer de région (qui correspond à un changement à l'intérieur d'une même ligne dans la matrice de correspondance dans le cas d'une surface verticale, ou un changement de ligne dans le cas du passage d'une surface horizontale), on compare

Tableau 4.1 Table de correspondance 2-D \rightarrow 3-D

Région 2-D	Surface 3-D inférieure	Région ou surface 3-D		Surface 3-D supérieure
		z_1	z_2	
...
-6	0	-4	-12	0
-5	0	-3	-12	0
-4	0	-3	-11	0
...
1	-17	1	9	-25
2	-18	2	9	-25
3	-19	3	9	-25
4	-20	4	9	-25
...
17	-24	8	12	-28

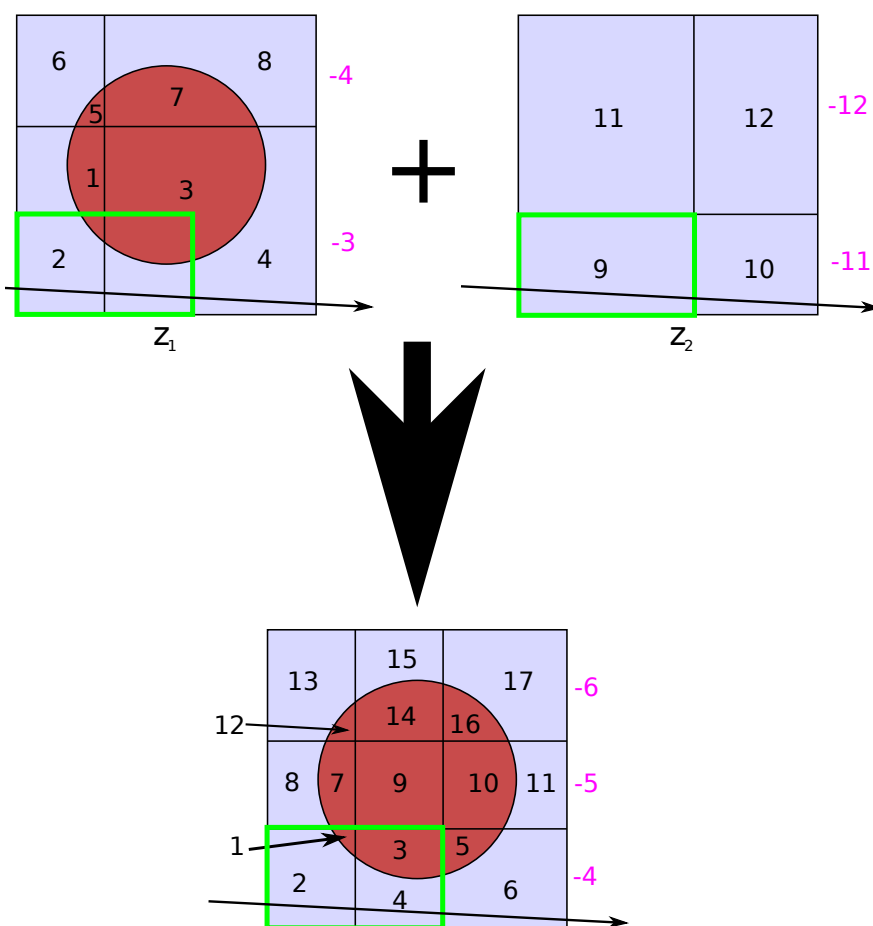


Figure 4.5 Numérotation des régions sur la ligne

```

Initialisation;
for Toutes les lignes 2-D do
  Lire la ligne 2-D;
  for Tous les angles polaires  $[0, \pi/2]$  do
    Calculer le poids;
    for Tous les décalages  $I_{\parallel}$  do
      for Angle polaire positif et négatif do
        Extruder la ligne et concaténer;
        Normaliser les longueurs;
        for Tous les groupes d'énergie non-convergés do
          Résoudre selon les relations de propagation;
        end
      end
    end
  end
end

```

Algorithme 1 : Extrusion et concaténation

les indices régionaux correspondant aux deux régions, et on recombine en additionnant les longueurs dans le cas où il s'avère qu'il n'y a pas de frontière dans la géométrie 3-D originale.

Ce processus est présenté à la figure 4.6, où on voit bien, par rapport à la géométrie donnée en exemple précédemment, les numéros de régions 2-D, en bleu, et les régions 3-D, en noir. Lorsque ces dernières se répètent consécutivement, on a alors la présence des frontières virtuelles, et les lignes doivent être concaténées, présenté par les frontières noires. On présente ici pour simplifier la figure qu'un nombre restreint d'angles polaires, et de décalages selon I_{\parallel} .

L'ensemble de ce processus d'extrusion est répété dans le solveur de flux à chaque itération interne, incluant la nouvelle comparaison des indices régionaux. Cette charge de calcul supplémentaire pourra dans certains cas ralentir chacune des itérations, mais pourra aussi les accélérer, si on considère le temps d'accès au disque pour la lecture dans le processus 3-D normal, qu'il n'est nécessaire de faire qu'une fois par ligne 2-D dans le cas d'une géométrie prismatique.

4.4.5 Calcul des volumes et surfaces

Une problématique apparaît aussi lors du calcul des surfaces et des volumes numériques (équation (3.23)). Cette étape sert à la fois à fournir un facteur de normalisation pour corriger partiellement l'erreur numérique provenant de la discrétisation due au processus de traçage, et à évaluer la précision de celui-ci par comparaison avec les valeurs analytiques, tel qu'élaboré

au chapitre 3.3.4. Le module de traçage peut sans problème calculer les volumes et surfaces analytiques (pour la géométrie globale) comme c'est le cas actuellement. Par contre, il ne peut calculer les volumes numériques, car l'analyse complète basée sur les lignes reconstituées n'est effectuée que dans un second temps.

Dans la géométrie prismatique projetée, la procédure actuelle de normalisation directe (en modifiant explicitement la longueur des lignes dans le module de traçage) ne peut prendre en compte les erreurs dues à la contribution en 3-D des lignes (comme le traçage se fait sur la géométrie projetée en 2-D). Il a donc fallu modifier cette stratégie afin de prendre en compte les erreurs de volumes en 3-D dans notre algorithme de traçage 2-D.

De plus, la création de la géométrie « super-discrétisée » engendre la création de régions virtuelles qui n'ont pas de signification dans le cadre de la géométrie originale, et qui peuvent être de très petites dimensions (c.f. les intersections des cercles à la figure 4.2). Dans le cas général, une telle région virtuelle appartient typiquement à des régions 3-D différentes en fonction de la position en z , et effectuer une normalisation différente sur les segments qui la traversent de celle pour la région avec laquelle elle sera recombinaisonnée au moment de l'extrusion pourrait mener à des résultats indésirables ou faussés.

Une normalisation sur ces volumes n'est donc pas souhaitable, d'autant plus qu'elle pourrait notamment engendrer des instabilités numériques du fait des petites quantités impliquées, qu'elle introduit une difficulté supplémentaire au niveau du calcul des volumes analytiques, en plus de n'apporter aucun avantage direct du point de vue de la normalisation.

On cherche donc à découpler le processus de normalisation du processus de traçage. Pour ce faire, on garde l'analyse géométrique (calcul des volumes et surfaces) déjà en place pour la géométrie originale, et on calcule les volumes numériques et les facteurs de normalisation au niveau de la reconstruction 3-D des lignes.

Ce schéma a pour avantage de garder les lignes non-déformées, ce qui peut être utile notamment à des fins de visualisation pour déterminer si toutes les régions ont bien été échantillonnées, et permet surtout de régler les problèmes liés aux régions virtuelles créées par la projection.

En pratique, l'appel au module `MCCGT` contient une première extrusion, indépendante de celles pratiquées par la suite dans le solveur de flux. Celle-ci permet de calculer initialement les volumes et surfaces numériques directionnels, et de les garder en mémoire pour réutilisation dans le module `FLU`.

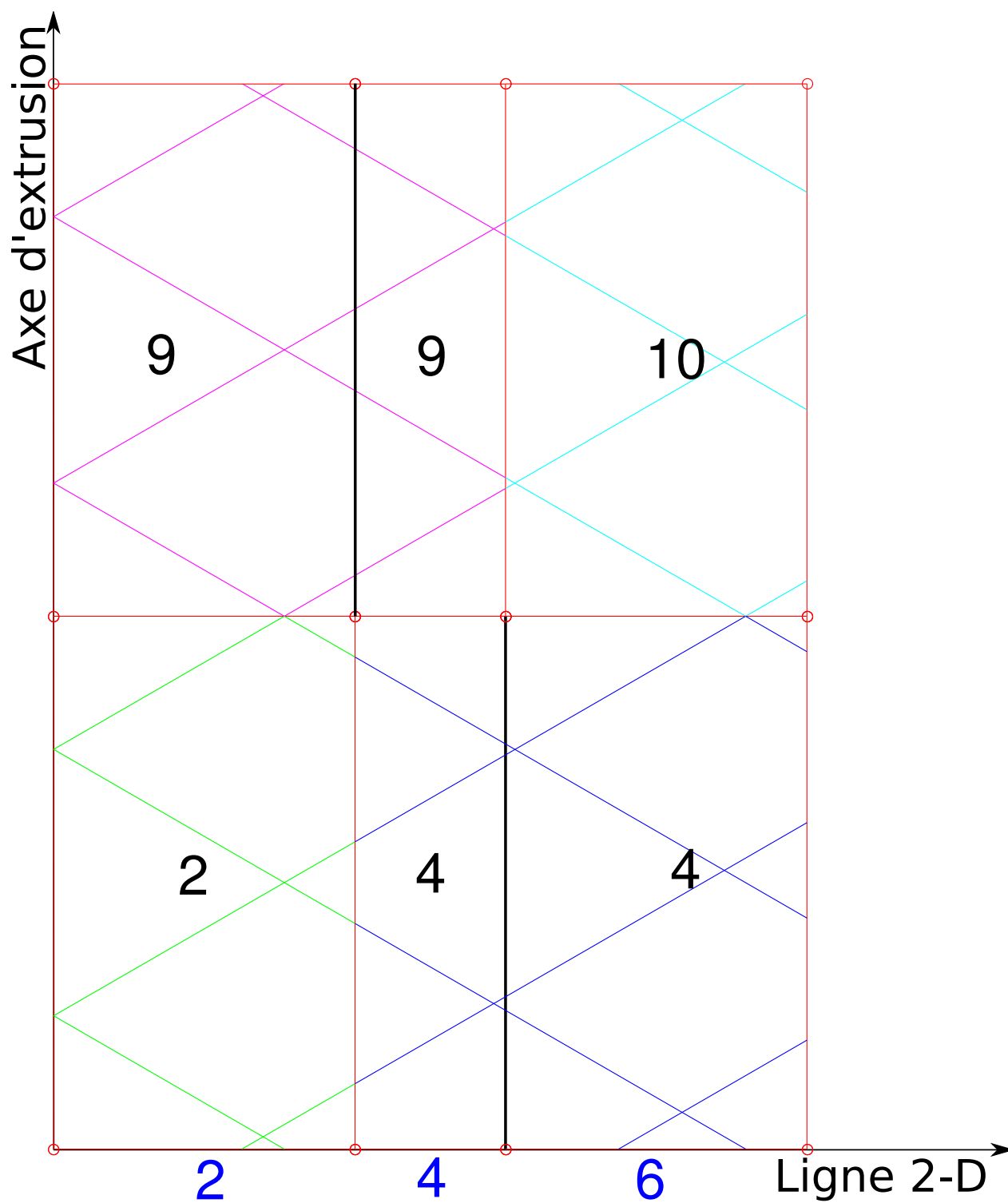


Figure 4.6 Extrusion de la ligne

Cette manière de faire permet aussi, lors d'un premier passage, une vérification plus simple du processus d'extrusion et de combinaison afin de détecter des erreurs, tout en ne coûtant en termes de temps de calcul qu'un passage sur le traçage et une reconstruction, qui sont effectuées plusieurs fois au niveau du solveur de toutes manières.

Du point de vue de l'utilisateur, la procédure reste entièrement transparente. Une géométrie 3-D est définie normalement, alors qu'un simple mot-clé `PRIZ` à l'appel du module `NXT:`, suivi d'une densité linéaire d'extrusion (correspondant à $I_{||}$), permet la projection et le traçage de la géométrie, ainsi que les options correspondantes dans les modules d'assemblage et de flux.

4.5 Développement Logiciel

Le développement logiciel s'articule logiquement selon deux axes, tels que présentés dans la structure du présent chapitre. Le premier s'oriente vers l'algorithme de projection lui-même, alors que le second s'oriente vers la résolution et l'extrusion.

4.5.1 Projection

Ce premier aspect doit tout d'abord s'assurer bien entendu de projeter la géométrie correctement, mais aussi et surtout, de ne perdre aucune information en lien avec la géométrie originale (3-D). Cet aspect est effectué dans le module `NXT:`.

Le fonctionnement normal de la première partie de ce module (`NXTACG`) n'est pas affecté, et procède au calcul des volumes et à l'analyse générale de la géométrie. L'appel au module `NXTPR3` est ensuite effectué, qui lui crée la géométrie projetée (incluant la matrice de projection), ainsi que la numérotation correspondante, et effectue le calcul des nouveaux volumes 2-D. Un appel au module de traçage des lignes d'intégration (`NXTTCG`) est par la suite effectué normalement, sur cette géométrie projetée.

Il importe ici de mentionner que les facteurs de normalisation qui sont calculés dans ce contexte ne s'appliquent qu'à la géométrie projetée, et ne serviront pas au calcul lui-même. Ils peuvent cependant être utiles afin d'estimer la précision du traçage.

4.5.2 Résolution

Le second aspect du développement logiciel effectué est plutôt en lien avec le module de résolution du flux lui-même, dans ce cas, `MCG:/MCCGT:`. Le module est appelé, dans l'ordre

normal d'exécution, avant l'appel au module FLU: lui-même, afin de calculer les nouveaux facteurs de normalisation au moyen d'une première extrusion, qui pourra être utile lors de développements parallèles subséquents (discutés dans le chapitre suivant), et qui peut être utilisée afin de valider l'extrusion elle-même dans un premier temps.

Par la suite, le module FLU: qui gère les itérations de puissance et internes discutées dans la section 3.4 fait appel au module spécialisé MCG:. À ce niveau, la modification de l'algorithme (séquentiel) intervient pour les différents angles d'extrusion, où à chaque changement de région, si la frontière est virtuelle (identifiée comme telle selon la matrice de projection où les deux indices régionaux seront identiques), la longueur des deux segments est additionnée.

4.5.3 Gestion de la mémoire

Les aspects de la gestion de la mémoire ne s'appliquent pratiquement en fait qu'au premier aspect décrit ci-dessus – la projection. À cet effet, la matrice de projection est créée incrémentalement au moment de la projection comme le nombre de permutations des plans et des régions orthogonaux à l'axe de projection croît exponentiellement, et il n'est pas envisageable dans l'optique d'une solution généraliste, d'envisager toutes les possibilités de prime abord. La même logique s'applique évidemment (dans une moindre mesure) aux autres quantités régionales ou surfaciques (albédos, sous-géométries, etc.)

CHAPITRE 5 CALCUL PARALLÈLE

Les ordinateurs récents présentent tous aujourd'hui des architectures permettant le traitement de plusieurs instructions simultanément. Le calcul parallèle nous permet de tirer profit de cette particularité afin de diminuer le temps d'exécution des programmes. En particulier, lorsque vient temps de s'attaquer à des problèmes à très grande échelle, cette parallélisation du code devient à toutes fins pratiques essentielle.

Dans les faits, en gardant en tête l'objectif éventuel de résoudre l'équation de transport à l'échelle d'un cœur de réacteur complet, en raison des dimensions spatiales du problème à traiter (nombre de régions) et malgré les méthodes d'accélération proposées, il est à prévoir que les coûts en termes de temps de calcul et de ressources mémoire demeureront excessifs si un algorithme séquentiel de résolution est considéré. C'est ici qu'intervient le calcul haute performance.

5.1 Calculateurs cibles

Dans le cadre de l'utilisation d'un solveur de neutronique, deux architectures sont à considérer dans un premier temps. Ainsi, on aura des postes de calcul scientifique, qui peuvent être utilisés pour des calculs de routine. Dans ce contexte, la machine type à la R&D d'EDF est la HP Z-600. Celle-ci comporte deux processeurs Intel Xeon E5620, basés sur l'architecture Westmere (Nehalem 32nm) quad-core, cadencés à 2.40GHz, doublés de 12 Go de mémoire vive.

Dans un deuxième temps, on considère un supercalculateur comme Ivanoe, qui comporte plusieurs types de noeuds, et dont l'architecture est présentée au tableau 5.1. En plus de ces processeurs CPU, le cluster possède aussi 34 noeuds composés de processeurs graphiques, mais qui n'ont qu'un intérêt limité dans le cadre du présent travail.

Un tel supercalculateur présente un intérêt particulier dans les études de sûreté ou de design, mais surtout dans des buts d'effectuer des calculs de validation des modèles dégradés utilisés en exploitation, en plus de fournir une base de développement pour les architectures futures qui sont amenées à être disponibles de manière plus répandue.

Tableau 5.1 Architecture Ivanoe

Noeuds de calcul standards

Nombre Noeuds	Processeurs				RAM
	CPU	CPU/Coeur	Cœurs/Noeud	Total	Qté
1382	Intel Xeon X5670 Westmere	2	6	12	24Go

Noeuds de calculs grande mémoire

Nombre Noeuds	Processeurs				RAM
	CPU	CPU/Coeur	Cœurs/Noeud	Total	Qté
16	Intel Xeon X5660 Westmere	2	6	12	128Go
8	Intel Xeon X7560 Nehalem	4	8	32	256Go
4	Intel Xeon X7560 Nehalem	4	8	32	512Go
1	Intel Xeon X7560 Nehalem	4	8	32	1To

5.2 Options de parallélisation

Dans ce contexte, il importe de bien distinguer les types de parallélisme qui peuvent être utilisés, et dans quel cadre il vaut mieux les appliquer. On distingue donc généralement trois niveaux de parallélisme dans les architectures des calculateurs modernes : les unités vectorielles (SIMD), le parallélisme à mémoire partagée, et celui à mémoire distribuée. Nous les survolons ici rapidement, et suggérons au lecteur curieux de se référer à Levesque et Wagenbreth (2010) pour une discussion plus approfondie.

5.2.1 SIMD

Le type de parallélisme du plus bas niveau qui soit est celui dit du SIMD - pour *Single Instruction / Multiple Data*. Ce type d'implémentation permet de gérer plusieurs fois la même instruction simultanément sur plusieurs éléments de vecteurs (on parle donc d'unités vectorielles, à l'intérieur même du coeur du processeur). Dans notre cas, comme l'élément fondamental du calcul MoC consiste dans le balayage progressif de chacune des lignes, et que le flux de sortie d'un élément est utilisé en entrée pour le suivant, ce type de parallélisme est d'un intérêt assez limité, d'autant plus que certains compilateurs (comme `ifort`) permettent maintenant une forme de vectorisation automatique de certaines instructions lorsque possible, au moment de la compilation.

Par contre, il serait potentiellement intéressant de le considérer dans le cadre des travaux de Sciannandrone (2015), qui modifie le stockage des fichiers de lignes pour regrouper celles qui traversent successivement les mêmes milieux. Dans ce cas, comme les opérations à effectuer

sont les mêmes sur chaque ligne, mais que la valeur à calculer change (typiquement la longueur), il pourrait éventuellement s'agir d'une option qui permettrait de mutualiser certaines opérations, et constitue certainement une avenue à éventuellement considérer.

5.2.2 Mémoire partagée

Le premier type de parallélisation qui nous intéresse dans le cadre de ce travail concerne donc un niveau un peu plus élevé, et permet de séparer les instructions à effectuer en différents *threads* (ou fils d'exécution) qui peuvent être exécutés simultanément. En particulier, cette méthode peut être avantageusement utilisée sur un ordinateur comprenant des processeurs multi-coeurs, ce qui est le cas de la plupart des architectures des processeurs sur lesquels sera amené à tourner le logiciel développé.

Une première parallélisation à ce niveau a comme avantage d'améliorer la performance des codes lorsqu'ils sont utilisés régulièrement, et ce, y compris sur des postes de travail standards. De plus, si l'algorithme de résolution parallèle est bien conçu et implanté de manière efficace, il permet de réduire le temps des calculs de routine utilisant le solveur sur une base régulière. Une parallélisation de ce type s'applique donc aux architectures multi-coeurs des processeurs récents, et pourra affecter à la fois les performances à l'utilisation routinière du code, tout comme la gestion des opérations à l'intérieur d'un même noeud d'un supercalculateur.

L'interface de programmation OpenMP (Dagum et Menon, 1998) permet la mise en place de ce type de parallélisation sans trop d'efforts et avec des modifications généralement mineures au code. De plus, les instructions sont généralement rédigées sous forme de commentaires, permettant une compilation normale avec les compilateurs traditionnels pour obtenir un code séquentiel (utilisé dans notre cas surtout à des fins de comparaison), et une compilation parallèle en utilisant les options appropriées des compilateurs récents.

Dans le contexte d'une parallélisation en mémoire partagée, tous les fils partagent un même environnement d'exécution, et cela permet l'utilisation de variables partagées aisément. De même des variables telles que des compteurs de boucle parallèle doivent être utilisées de manière distincte par les processus, et sont donc privées pour chacun des fils d'exécution. Par contre, ce partage de l'espace mémoire peut causer des conflits dans le cas où une allocation a été effectuée au préalable dans le code, pour des variables qui doivent être privées.

Dans le cadre du code DRAGON, la gestion de la mémoire avec FORTRAN est effectuée dynamiquement par des appels à des fonctions de la GANLIB (Roy et Hébert, 2000) reposant sur des *malloc* en langage C, dont les pointeurs sont passés sous forme d'entiers aux

routines sous-jacentes. Il faut donc, pour s'assurer que deux fils n'écrivent pas dans le même espace mémoire, allouer et gérer les blocs mémoire différemment. Pour ce faire, des directives pour le préprocesseur ont été incluses en fonction des options de compilations spécifiées par l'utilisateur, rendant le code toujours compatible avec une compilation séquentielle.

5.2.3 Mémoire distribuée

Lorsque vient le temps de passer à une échelle encore supérieure (nombre de régions plus élevé), l'utilisation de clusters s'avérera essentielle. Ceux-ci sont composés de plusieurs *noeuds*, qui possèdent chacun plusieurs processeurs, pouvant chacun exécuter plusieurs processus en parallèle (en mémoire partagée), comme il est présenté au tableau 5.1.

Dans ce contexte de parallélisation en mémoire distribuée, chaque noeud a son propre processus en exécution, et peut effectuer les instructions de la partie séquentielle du code si nécessaire. Les données partagées doivent l'être explicitement par les différents processus, à l'aide d'un protocole standard, qui peut être le *Parallel Virtual Machine – PVM* (Geist *et al.*, 1994), ou encore *Message Passing Interface – MPI* (Gropp *et al.*, 1999) par exemple, qui permettent de gérer ces échanges entre les différents noeuds.

Cet échange explicite des données entre les différents processus demande en revanche des modifications beaucoup plus importantes au code source original, et possiblement de revoir certains algorithmes si ceux-ci n'ont pas été pensés en fonction du calcul parallèle. De plus, les appels aux routines *MPI* explicites dans le code peuvent rendre ardue la compatibilité avec un compilateur séquentiel, et exiger des efforts de maintenance particuliers si une seule version doit être conservée.

Nous avons mentionné la gestion des échanges entre les différents noeuds de supercalculateurs tels Ivanoe, mais il importe de préciser qu'une parallélisation en mémoire distribuée peut aussi être adaptée à des architectures beaucoup plus hétérogènes, comme des réseaux de calcul distribués sur plusieurs PCs fonctionnant en parallèle à l'échelle d'un réseau local.

Dans un tel réseau, toutefois, la durée de ces échanges (tout comme les accès à des données stockées sur un disque) peuvent être non-négligeables en comparaison à la durée des opérations CPU. Pour cette raison, il est primordial, pour obtenir des performances et des accélérations (*speedups*) raisonnables, de gérer ces accès de manière à les minimiser. Ce problème est généralement moindre dans le cadre de supercalculateurs, mais représente tout de même un aspect important de l'optimisation du code, surtout en comparaison au temps d'accès à une mémoire partagée.

Finalement, il convient aussi de mentionner les possibilités de parallélisation dites hybrides, où si possible, on utilise une parallélisation en mémoire partagée à l'échelle d'un noeud, sur laquelle on superpose un second niveau de parallélisation qui tire parti des possibilités en mémoire distribuée. Nous discuterons brièvement de cette option dans le cadre du solveur prismatique à la section 5.5.

5.3 Objectifs

Dans le cadre du développement d'une extension parallèle aux solveurs de DRAGON, il importe de définir des objectifs à la fois de performance et d'échelle. Nous présenterons donc ici rapidement les notions qui permettent de quantifier les améliorations apportées au niveau du solveur.

5.3.1 Accélération

Dans un premier temps, l'objectif du calcul parallèle est bien entendu de réduire le temps pris afin d'effectuer un calcul en particulier. Dans ce cadre, il convient de parler d'accélération, ou de *speedup*. L'accélération S est donc définie très simplement comme le ratio entre les temps d'exécution séquentiel (T_{seq}) et parallèle (T_p) pour p processeurs :

$$S_p = \frac{T_{seq}}{T_p} \quad (5.1)$$

On parle aussi d'efficacité, où cette dernière est ramenée en fonction du nombre p de processus : $E_p = \frac{S_p}{p}$. L'efficacité représente une mesure d'utilisation des processus en calcul comparativement au temps nécessaire pour les échanges et à la synchronisation. Une efficacité de 1 est idéale, et un doublement du nombre d'unités de calcul entraîne dans ce cas une diminution de moitié du temps d'exécution.

5.3.2 Scalabilité

La scalabilité (de *to scale*) permet une mesure plus précise des possibilités de performance d'un code parallèle. Elle tient compte de la portion α du code qui doit rester séquentielle de par la conception de l'algorithme, pour calculer le temps d'exécution en fonction de p :

$$T(p) = \alpha \times T(1) + (1 - \alpha) \frac{T(1)}{p} \quad (5.2)$$

$$= T(1) \left(\alpha + \frac{1 - \alpha}{p} \right) \quad (5.3)$$

et on peut en tirer le *speedup* :

$$S = \frac{p}{1 + \alpha(p - 1)} \quad (5.4)$$

Cette équation est connue comme la loi d'Amdahl (Amdahl, 1967), et permet d'évaluer le gain maximal qui peut être gagné en augmentant le nombre de processus traitant la portion parallèle d'un code. Dans le cas qui nous intéresse, comme on ne considère pour le moment que la parallélisation du solveur de flux prismatique, la partie séquentielle compte donc les modules de gestion des bibliothèques, de traçage et d'analyse géométrique, ainsi que d'autoprotection des résonances si nécessaire. La loi d'Amdahl dicte donc qu'au delà d'une certaine extensibilité, il faudra aussi paralléliser ces parties du code afin d'obtenir des performances adéquates.

Dans le cadre qui nous concerne, on cherche non seulement à améliorer le temps de calcul de cas bien précis, mais on cherche surtout à augmenter la taille des problèmes qu'il est possible de traiter. Ainsi, outre le passage de la mémoire partagée à des calculateurs à mémoire distribuée (ou une parallélisation hybride), il est aussi nécessaire de considérer comment la taille du problème augmente. Dans la plupart des cas, à mesure que le nombre de régions augmente, le temps passé dans le solveur augmente, et ce, plus rapidement que dans la partie séquentielle. Cette propriété est expliquée par la loi de Gustafson (1988), et indique la propension à tirer profit du calcul parallèle pour traiter de plus gros cas dans le même temps, plutôt que pour accélérer un cas donné.

5.3.3 Intensité arithmétique

Afin de quantifier les considérations de bande passante énumérées plus tôt, il convient de parler aussi d'intensité arithmétique. On définit cette dernière comme le rapport entre le nombre d'opérations effectuées et la quantité de données transférées en mémoire. Une basse intensité impliquera donc un problème dont le goulot d'étranglement se situe dans la bande

passante de la mémoire (*memory-bound*), et on cherchera donc à diminuer la quantité de données à échanger et à communiquer. Dans le cas inverse, une haute intensité signifie que le(s) processeur(s) est utilisé à plein régime (*CPU-bound*) et une parallélisation supplémentaire pourrait être bénéfique.

De plus, bien que l'accès aux protocoles de communication entre les noeuds des ordinateurs massivement parallèles est plus adéquat que pour les réseaux composés de plusieurs ordinateurs, il s'agit souvent d'un facteur limitant pour la performance, particulièrement pour les applications qui ne sont pas foncièrement parallélisables (*embarassingly parallel* – comme les méthodes Monte-Carlo, par exemple).

Lors du processus de parallélisation il faut donc veiller à garder au minimum requis les échanges d'information entre les différents processus, ainsi que l'accès aux données partagées. Ces dernières pourront être séparées et réparties, autant que faire se peut, sur les différents noeuds de calcul afin d'optimiser la vitesse des opérations. Bien que la méthode des caractéristiques soit mieux adaptée à ce type de parallélisation que la méthode des probabilités de collision, il est évident que des quantités non-négligeables d'information doivent tout de même être échangées. La gestion et la réduction des opérations de communication entre les processus (*broadcast* et *reduce* en particulier, en mémoire distribuée, étant bloquantes) constitue donc un des aspects importants du travail à effectuer.

5.3.4 Équilibrage des charges

Dans tous les cas de calcul hautement parallèles, il surviendra un moment où certains processus seront terminés, alors que d'autres poursuivent leur exécution. Ce délai dans l'exécution du programme peut gravement pénaliser le temps de calcul parallèle, particulièrement s'il survient fréquemment. Dans ce contexte, il convient d'équilibrer adéquatement les charges (*load balancing*) entre les différents processus en cours d'exécution. Deux stratégies peuvent être adoptées :

- Une répartition déterministe, où un algorithme assigne différentes tâches à chaque processus, en tentant de répartir au mieux les tâches à accomplir.
- Une répartition statistique (ou *round-robin*) où chaque fois qu'un processus termine son exécution, il reçoit la prochaine liste de tâches à effectuer. Avec une quantité suffisante de tâches, l'équilibre est atteint de manière statistique.

De prime abord, l'opération atomique du processus de résolution du flux est le calcul de $\phi_{k+1}(\vec{T})$ en connaissant $\phi_k(\vec{T})$, la longueur $L_k(\vec{T})$ du segment et les sections efficaces du

milieu, sur toute la longueur de la *track*. Cette opération implique le calcul d'une exponentielle décroissante, qui peut être soit calculée, soit interpolée à partir de données tabulées (plus rapide). À partir de ces informations, il est possible d'évaluer la charge de calcul pour chaque *track* en fonction du nombre de segments qui la composent.

Dans une parallélisation par angle ou par direction, on peut s'attendre en première approximation à ce que la charge de calcul soit sensiblement équivalente (quoiqu'imparfaite), comme il s'agit de la même géométrie qui est analysée selon chaque direction, en utilisant la même densité surfacique pour la génération des *tracks*. Une meilleure gestion de la répartition des charges tend généralement à réduire les temps d'attente imposés par les processus de communication bloquants.

5.4 Travaux antérieurs

Comme il a été mentionné, la méthode des caractéristiques est particulièrement bien adaptée à une parallélisation (notamment en comparaison de l'autre méthode utilisée de manière routinière pour les calculs de transport en géométries non-structurées, la méthode des probabilités de collision), du fait que la plupart des opérations de base sont indépendantes les unes des autres et que l'accès à des données partagées est assez limité. Ces dernières seront principalement constituées du *tracking* ou des informations géométriques, des sections efficaces multigroupes par région, et finalement, des moments du flux calculés à l'itération précédente et à l'itération courante. Des efforts de parallélisation du code DRAGON ont déjà été effectués par le passé, notamment par Wu et Roy (2001) ou Dahmani (2006) qui ont utilisé MoC dans les modules MCI:/MCU:et MOCC:, et par Qaddouri *et al.* (1996) en utilisant la méthode des probabilités de collision.

Cependant, certaines de ces implémentations ne s'appliquaient qu'à des géométries restreintes (cellule unitaire ou supercellule), et avaient été implantées pour la plupart sur des réseaux informatiques non-dédiés. Des études ont été effectuées afin d'étudier l'extensibilité (*scalability*) de la parallélisation du code en augmentant le nombre de processeurs. Les caractéristiques de l'infrastructure réseau, comme une répartition entièrement locale de la mémoire (sur chaque ordinateur), ou un minimum d'interaction entre les processus rendaient particulièrement appropriée l'utilisation de PVM comme protocole de communication entre les processus, bien que MPI (plus récent) ait aussi été considéré Dahmani et Roy (2005).

Les résultats obtenus pour une parallélisation de la méthode des caractéristiques offrent des accélérations très intéressantes, et une extensibilité appropriée (Dahmani et Roy, 2006). En

utilisant ces techniques en conjugaison avec le *tracking* en ligne, on élimine virtuellement les contraintes reliées aux capacités mémoire, et on peut alors traiter des problèmes de dimensions spatiales très étendues – en supposant l'accès à une capacité de calcul et un algorithme d'accélération appropriés.

Toutefois, les techniques d'accélération mentionnées précédemment (particulièrement la plus efficace, ACA) et le solveur traitant les géométries prismatiques (uniformes) n'ont été implantés que dans le module MCGG: de DRAGON, sur lequel aucune parallélisation n'a été effectuée. C'est d'ailleurs sur cette base que la décision a été prise d'effectuer les développements logiciels dans la Version 4 du code, qui contient ce module. Cette version dispose aussi d'une méthode plus raffinée pour traiter l'autoprotection des résonances reposant sur la méthode des sous-groupes, et qui peut elle aussi utiliser la méthode des caractéristiques pour résoudre l'équation de ralentissement.

Dans le cadre du développement du solveur prismatique présenté au chapitre 4, les développements suivant les options de parallélisme ont été effectués sans se soucier du processus de *tracking* en ligne, et en gardant toute l'information sur les lignes d'intégration dans un fichier séquentiel binaire (2-D projeté – beaucoup moins volumineux qu'une représentation 3-D). On envisage ici les axes de parallélisation qui s'appliquent plus particulièrement à la géométrie prismatique, comme une première étape vers l'intégration du *tracking* en ligne, pour permettre l'extension à des géométries spatialement toujours plus étendues.

5.5 Identification des gisements de parallélisme

Les principaux défis concernant l'implantation du code sur une machine massivement parallèle proviennent de la gestion de la mémoire distribuée, généralement disponible en quantité limitée sur chacun des noeuds (ceci pourrait empêcher notamment le stockage local des fichiers de *tracking*, qui peuvent être très volumineux). Notons aussi les problèmes liés à la gestion et à la répartition des charges (*load balancing*) entre les différents noeuds.

Pour gérer ces problèmes en tenant compte de la géométrie prismatique, plusieurs axes de parallélisation ont été envisagés pour répartir les tâches entre les différents noeuds de calcul, incluant :

- par angle (φ en 2-D) dans une géométrie prismatique projetée ;
- par *track* 2-D dans la géométrie projetée ;
- par direction ($\hat{\Omega}$ en 3-D) dans une géométrie reconstituée ;
- par *track* 3-D reconstituée ;

- par groupe d'énergie ;
- par région spatiale.

Les travaux de Dahmani (2006) tendent à démontrer que dans certaines situations, une distribution des charges en *round robin* (où les *tracks* sont distribuées successivement aux processus dans l'ordre où elles sont générées) permet d'obtenir des accélérations intéressantes, meilleures que pour les cas où les *tracks* sont distribuées aux processeurs suivant les directions $\hat{\Omega}$. De plus, une discrétisation en $\hat{\Omega}$ (ou en φ dans la géométrie projetée) devrait être compatible avec une parallélisation du préconditionneur ACA. Des accélérations performantes peuvent aussi être obtenues dans les cas où une évaluation préalable du nombre d'opérations à effectuer sur chaque *track* est effectuée, afin de déterminer le cœur approprié auquel l'envoyer, de manière à répartir plus équitablement la charge totale (Wu et Roy (2001)).

5.5.1 Résolution Jacobi vs Gauss-Seidel

Dans le solveur prismatique considéré, la boucle parallélisable de plus bas niveau considère les différents groupes d'énergie. Celle-ci a effectivement été combinée à la boucle interne des itérations caractéristiques, ce que nous présentons ici.

Pour ce faire, nous considérerons le système matriciel présenté à la section 3.3.4, qui nous permet de développer le système multigroupe sur la base de l'équation (3.21). On sépare le terme source pour en extraire les sources de diffusion provenant des autres groupes ($g' \neq g$), laissant les sources de fission et externes :

$$\vec{S}^g = \vec{F}^g + \sum_{g' \neq g} \Sigma_s^{g \leftarrow g'} \vec{\Phi}^{g'} \quad (5.5)$$

ce qui permet de retrouver un système que l'on peut écrire de la manière suivante :

$$\mathbf{H}^g \vec{\Phi}^g = \mathbf{T}^g \left(\vec{F}^g + \sum_{g' \neq g} \Sigma_s^{g \leftarrow g'} \vec{\Phi}^{g'} \right), \quad (5.6)$$

où l'on a défini $\mathbf{H} = \mathbf{I} - \mathbf{T}\Sigma_{s,w}$, avec $\Sigma_{s,w}$ la matrice de section efficace de diffusion du groupe vers lui-même.

L'équation (5.6) nous permet d'envisager deux manières de résoudre le problème qui nous est présenté. La première, en utilisant la méthode de Gauss-Seidel ou les méthodes dérivées de

surrelaxation successive (*Successive overrelaxation – SOR*) (Young, 1954), nous permettent de manière générale de calculer le flux à l’itération $(i + 1)$ comme :

$$\mathbf{H}^g \vec{\Phi}^{g(i+1)} = \mathbf{T}^g \left(\vec{F}^g + \sum_{g' < g} \Sigma_s^{g \leftarrow g'} \vec{\Phi}^{g'(i+1)} + \sum_{g' > g} \Sigma_s^{g \leftarrow g'} \vec{\Phi}^{g'(i)} \right) \quad (5.7)$$

Dans ce cas, on remarque que chaque nouvelle itération $(i + 1)$ doit avoir en main tous les résultats de l’itération courante pour les groupes j précédents. On n’a alors d’autre choix que de procéder séquentiellement, et dans ce contexte une parallélisation n’est pas appropriée, car chaque itération doit posséder les informations relatives à la boucle en cours de traitement. On peut toutefois aussi minimiser l’espace mémoire requis en se limitant à un seul vecteur ϕ , car on peut mettre à jour les nouvelles valeurs calculées au fur et à mesure. Par contre, une manière de faire très proche, la méthode de Jacobi (Press *et al.*, 1994), nous permet d’avoir ces itérations dérivées de la manière :

$$\mathbf{H}^g \vec{\Phi}^{g(i+1)} = \mathbf{T}^g \left(\vec{F}^g + \sum_{g' \neq g} \Sigma_s^{g \leftarrow g'} \vec{\Phi}^{g'(i)} \right) \quad (5.8)$$

Cette méthode, bien que convergeant généralement plus lentement que la méthode de Gauss-Seidel (intuitivement, on le voit, comme on utilise des éléments du vecteur de flux plus « vieux », datant de l’itération précédente), a l’avantage d’offrir des possibilités de parallélisation intrinsèques.

5.5.2 Boucle sur les groupes

Les travaux de Dahmani (2006), tels qu’exposés par Le Tellier (2006) ont démontré que, dans une telle situation, les itérations multigroupe décrites dans la section 3.4, peuvent être confondues avec les itérations internes. Cette manière de procéder nous permet d’effectuer une simple boucle sur chacun des groupes d’énergie – et comme toutes les données composant ces boucles sont indépendantes, on peut aisément les paralléliser.

On considère donc le système itératif multigroupe suivant, aisément parallélisable :

$$\vec{\Phi}^{g(i+1)} = \mathbf{T}^g \left(\vec{F}^g + \sum_{g' \neq g}^G \Sigma_s^{g \leftarrow g'} \vec{\Phi}^{g'(i)} \right) \quad (5.9)$$

De surcroît, comme il s’agit d’une étape clé du solveur (les exponentielles présentées aux équations (3.9) et (3.10) y sont calculées), une parallélisation y sera d’autant plus efficace – pour autant qu’un nombre important de groupes est utilisé.

5.5.3 Parallélisation par angle ou par direction

Outre la boucle sur les groupes, la parallélisation par *track* 2-D ou par angle φ semblent *a priori* des voies très prometteuses. Mentionnons aussi qu’une répartition des processus parallèles en fonction des angles évite aussi les conflits au niveau de l’écriture des flux angulaires calculés à la fin de chaque itération, puisqu’un seul processus sera en charge de l’écriture d’un même moment du flux pour une région donnée.

Dans la nouvelle approche prismatique du solveur, au niveau de la parallélisation, les approches « traditionnelles » de parallélisation sur les lignes se transformeront en boucles parallèles sur les directions et les lignes 2-D. Il s’agit alors d’envoyer, une fois la projection et le *tracking* prismatique faits, une ligne 2-D par processus, qui se chargera de reconstruire la *track* en 3-D et de calculer le flux sur chaque segment associé. Comme il s’agit d’une étape cruciale du calcul MoC, et qu’elle doit être répétée à chaque itération on peut s’attendre dans ce cas à des gains importants en termes de performance. Ces possibilités ont été traitées abondamment par d’autres auteurs, tant en 2-D (Boyd III, 2014) qu’en 3-D (Dahmani et Roy, 2005) que sur des décompositions hybrides (Févotte et Lathuilière, 2013), et ne seront pas abordées en détails ici.

C’est d’ailleurs ici que viennent jouer les approches de programmation parallèle hybrides mentionnées plus tôt. On peut donc envisager un premier niveau de parallélisme au niveau de la boucle sur les groupes (en mémoire partagée), qui est englobé, un niveau plus haut, par une distribution des données par direction à traiter sur différents processeurs (en mémoire distribuée). Ces boucles parallèles seront donc imbriquées, améliorant d’autant la performance du code.

Par ailleurs, tel que discuté dans Le Tellier (2006) et Le Tellier *et al.* (2008), il serait avantageux d’éventuellement étendre la technique d’accélération ACA à un environnement parallèle à mémoire distribuée. Dans ce cadre, une partition du flux correctif mentionné au paragraphe 3.4.2 est effectuée et répartie sur p processus, qui en calculent les composantes partielles. On recombine (*reduce*) ensuite ces composantes afin d’obtenir le flux correctif global qui sera ensuite partagé (*broadcasted*) à tous les processus.

Une telle opération de communication, bloquante sur tous les processus participants, sera

potentiellement pénalisante au niveau du temps de calcul, d'autant plus que la résolution du système correctif utilise elle aussi un processus itératif. Compte tenu de la performance de ACA, on est toutefois en droit de s'attendre à ce que la performance s'en trouve tout de même améliorée par rapport à un système non-préconditionné. Il faut toutefois mentionner que comme la méthode ACA repose elle-même sur une condensation au niveau du *tracking*, elle imposera une contrainte au niveau de la répartition des charges.

5.5.4 Parallélisation sur les lignes extrudées

Toutes les approches décrites dans les sections précédentes sont par ailleurs compatibles avec un algorithme de résolution MoC classique (en 2 ou en 3 dimensions), et peuvent être appliquées à un niveau supérieur à l'algorithme de résolution prismatique présenté à la section 4. Celui-ci nous offre par contre une nouvelle possibilité de parallélisation qui était inaccessible par les solveurs classiques, et qui pourrait permettre de retrouver le niveau de parallélisation disponible dans les solveurs explicitement 3-D.

En effet, au niveau de l'extrusion des lignes, présenté à l'origine à la figure 4.6 et reproduit à la figure 5.1, on peut envisager un nouvel axe de parallélisme complètement nouveau. En se basant sur un algorithme simplifié d'ordonancement rouge-noir typique (Saad.Y., 2003, Sec. 12) (ici, rouge-vert), on peut ainsi traiter un tel cas. Dans un premier temps, il importe de réaliser que deux lignes contigües ne peuvent être traitées parallèlement, en raison des conflits d'écriture qui surviendront au niveau de l'écriture des flux angulaires. Pour cette raison, on sépare l'ensemble des lignes extrudées en deux groupes, rouge et vert sur la figure, puis on sépare chacun de ces deux groupes en un ensemble de paquets. L'interdiction sur les lignes contigües nous empêche donc de traiter un paquet rouge en même temps qu'un paquet vert, mais nous laisse cependant traiter l'ensemble des paquets verts en parallèle, puis l'ensemble des paquets rouges. Ce processus est décrit à l'algorithme 2, où les bouches en *gras italique* sont parallélisables.

Un des éléments déterminants à considérer dans le cas qui nous concerne est la taille de paquets. En effet, il importe de la choisir de manière appropriée afin de s'assurer, dans un premier temps, qu'elle soit assez importante afin que deux lignes de deux paquets de la même couleur ne contribuent pas à la même région spatiale, ce qui reviendrait à un nouveau conflit en écriture. Dans un deuxième temps, il faut aussi qu'ils soient suffisamment petits afin de permettre une efficacité parallèle appropriée.

Cette option de parallélisation sur les lignes extrudées est la plus novatrice de celles décrites dans cette section, et repose en particulier sur un aspect distinctif de la méthode de projection

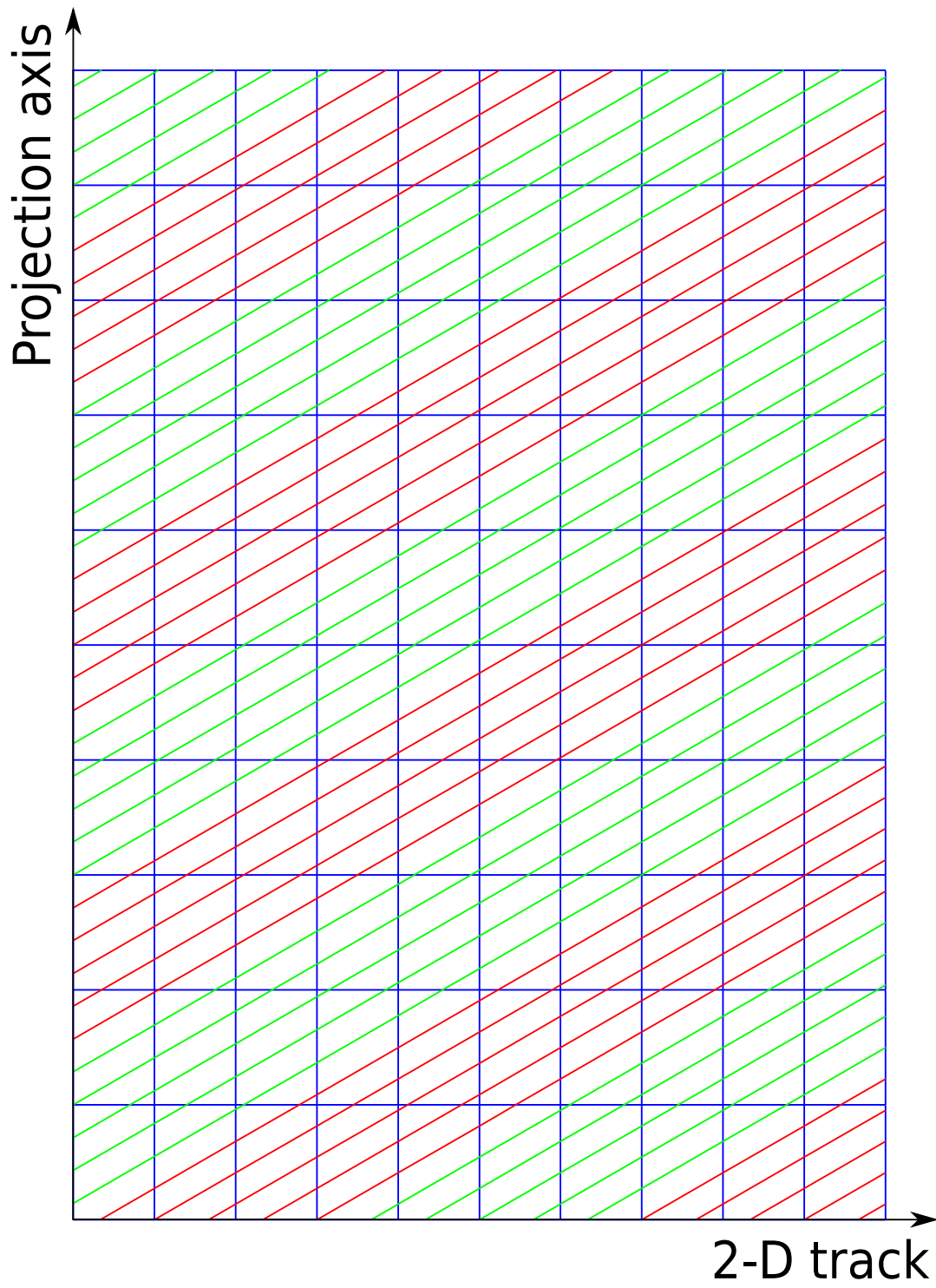


Figure 5.1 Traitement parallèle des paquets en extrusion

```

Initialisation;
for Toutes les lignes 2-D do
  Lire la ligne 2-D;
  for Tous les angles polaires  $[0, \pi/2]$  do
    Calculer le poids;
    Calculer le nombre de paquets;
    for Tous les paquets rouges do
      for Toutes les lignes d'un paquet do
        for Angle polaire positif et négatif do
          Extruder la ligne et concaténer;
          Normaliser les longueurs;
          for Tous les groupes d'énergie non-convergés do
            Résoudre selon les relations de propagation;
          end
        end
      end
    end
  end
  for Tous les paquets verts do
    for Toutes les lignes d'un paquet do
      for Angle polaire positif et négatif do
        Extruder la ligne et concaténer;
        Normaliser les longueurs;
        for Tous les groupes d'énergie non-convergés do
          Résoudre selon les relations de propagation;
        end
      end
    end
  end
  for Toutes les lignes restantes do
    for Angle polaire positif et négatif do
      Extruder la ligne et concaténer;
      Normaliser les longueurs;
      for Tous les groupes d'énergie non-convergés do
        Résoudre selon les relations de propagation;
      end
    end
  end
end

```

Algorithme 2 : Algorithme d'extrusion parallèle

prismatique généralisée décrite dans le chapitre 4. Les autres option ayant par ailleurs déjà été implantées sous une forme ou une autre tant dans DRAGON que dans d'autres codes. C'est donc cette avenue qui a été choisie dans ce projet.

Dans ce cadre, la solution envisagée est de déterminer la grosseur des paquets en fonction de la largeur apparente de la plus grande région spatiale pour un angle d'extrusion donné. L'implantation actuelle demande à l'utilisateur de choisir lui-même la taille des paquets, mais des développements futurs pourront choisir de se baser sur la première extrusion qui est effectuée afin de calculer les volumes numériques (détaillés à la section 4.4.5) afin d'en déterminer la taille minimale en fonction de la section maximale. Cette taille serait stockée dans le fichier de traçage, pour ensuite être utilisée dans le solveur parallèle.

CHAPITRE 6 RÉSULTATS

Ce chapitre expose les résultats obtenus grâce aux développements présentés précédemment, et ce, en deux temps : L’algorithme prismatique, puis sa parallélisation.

Dans un premier temps, nous présentons la validation de la technique d’extrusion prismatique présentée dans le chapitre 4 à travers l’analyse de deux cas de figure. Dans la section 6.1, on met d’abord de l’avant une géométrie simple et avec un nombre de régions réduit, qui pourra être analysée complètement, et dont les résultats pourront être comparés simplement avec les solveurs de référence 3-D. Dans le second cas, présenté dans la section 6.2, un assemblage de REP simplifié, mais déformé, illustre plus clairement les avantages de la méthode en permettant d’analyser un cas de très grandes dimensions avec des besoins en mémoire et en temps de calcul fort réduits par rapport aux méthodes 3-D traditionnelles.

Dans un second temps, nous présentons une validation de la parallélisation de l’extrusion. Un premier cas permet de valider l’aspect parallèle du calcul et les techniques présentées dans le chapitre 5, sur un cas utilisant une géométrie prismatique homogène (section 6.3.1). Finalement, la parallélisation est testée sur un dernier cas-test inhomogène, l’assemblage présenté dans la validation prismatique (section 6.3.2) afin de valider les deux aspects (parallèle et prismatique) du développement en même temps.

6.1 Simulation d’un embout de crayon

6.1.1 Description du problème

Le premier problème simulé afin de vérifier et de valider les algorithmes de projection et d’extrusion concerne le bout d’un crayon d’uranium enrichi, entouré de modérateur / caloporteur (eau légère). La composition isotopique des mélanges est donnée au tableau 6.1, où on a utilisé les sections efficaces correspondantes de l’évaluation ENDF/B-VI à 172 groupes (`endfb6gx` en format WIMS-D4). Dans le contexte de ce premier cas-test, on n’utilise que le modérateur et le combustible.

On utilise donc une géométrie semblable à celle présentée à la figure 4.3, comportant deux cellules carrées superposées de 5 cm de côté, donc celle inférieure comporte un crayon de combustible de 3 cm de diamètre. Dans ce cas, la résultante de la projection donne la combinaison présentée aux figures 6.1 et 6.2. Dans ce cas, on a aussi utilisé une discrétisation

Tableau 6.1 Composition isotopique des mélanges

Description du mélange	Isotope	Concentration ($10^{24}cm^{-3}$)
Modérateur / Caloporteur 600°	1H	4.42326E-2
	^{16}O	2.21162E-2
	^{10}B	0.51066E-5
	^{11}B	2.55549E-5
Gaine 600°	Zr (Nat)	4.21838E-2
Combustible 600°	^{16}O	4.60093E-2
	^{234}U	7.31651E-6
	^{235}U	9.10661E-4
	^{238}U	2.21490E-2

selon l'axe de projection de 3 (SPLITZ 3) à la fois dans la cellule contenant le crayon ainsi que dans le modérateur qui lui est superposé. On peut donc déjà observer la construction de la matrice de projection, dont le résultat est présenté au tableau 6.2. La numérotation présentée à la figure 6.2 correspond à celle du tableau, mais afin d'alléger la notation de la figure, seulement les régions du premier plan de chaque cellule y sont identifiées (z_1 et z_4).

Ce cas présente l'avantage d'être simplissime, en comportant très peu de régions spatiales, permettant une convergence rapide et un temps de calcul réduit. On peut dès lors effectuer des comparaisons directes avec les solveurs 3-D classiques déjà implantés dans DRAGON, pouvant eux aussi bien entendu simuler une géométrie non-homogène selon z . De plus, ce cas à petite échelle contient suffisamment peu de régions spatiales pour nous permettre d'effectuer une vérification directe et complète du fonctionnement de l'algorithme de projection, ainsi que de la matrice de correspondance qui en résulte. Cette vérification est d'autant simplifiée par le fait que la géométrie projetée n'est composée que de deux cellules superposées, pour un total de six plans de discrétisation.

De plus, lors du passage dans le module MCCGT :, la première extrusion qui permet de calculer les facteurs de normalisation et les volumes numériques, nous donne les erreurs obtenues dans le calcul de ces volumes et surfaces. On peut dès lors analyser ces erreurs à la fois en rapport à la géométrie projetée et à la géométrie extrudée reconstituée.

En ce qui concerne les paramètres de traçage, on a utilisé une densité de lignes 3-D de 100 lignes/cm², équivalente à une densité planaire de 10 lignes/cm doublée d'une densité en extrusion d'encore 10 lignes/cm. On a aussi utilisé une quadrature angulaire produisant dans les deux cas 96 directions par octant.

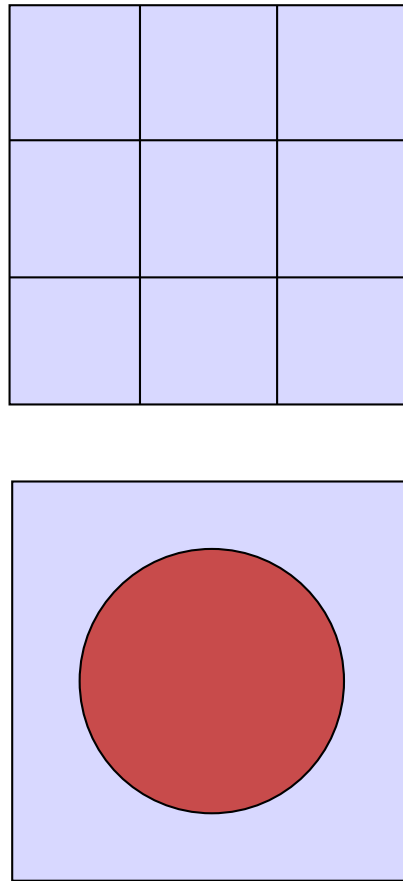


Figure 6.1 Section de crayon

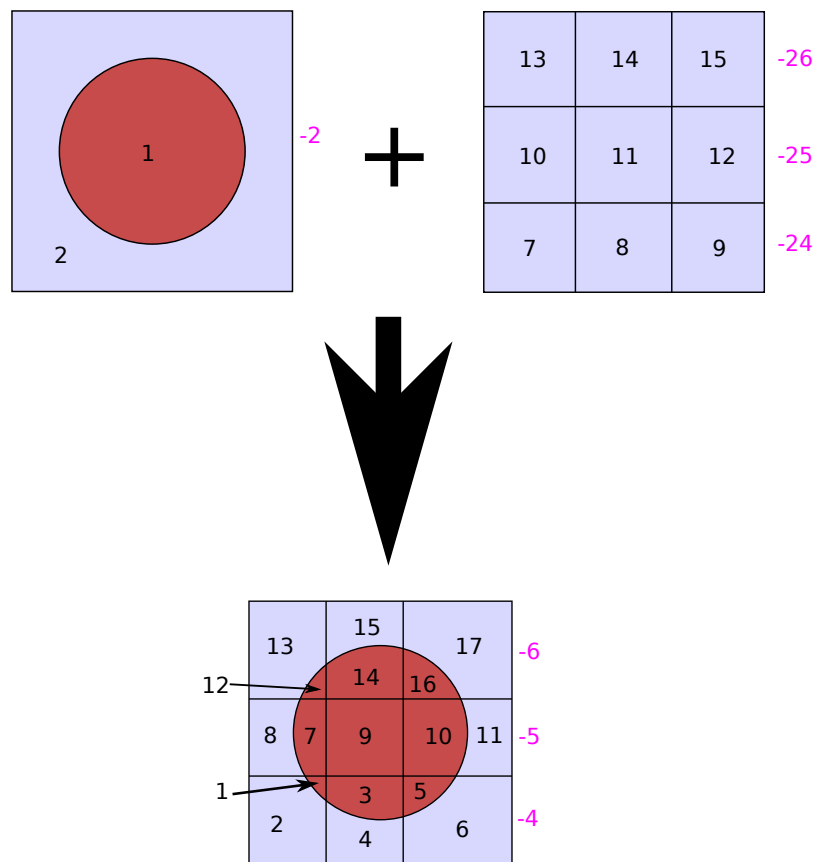


Figure 6.2 Combinaison des plans de projection et numérotation partielle

Tableau 6.2 Table de correspondance 2-D \rightarrow 3-D pour l'embout d'un crayon

Région 2-D	Surface 3-D inférieure	Région ou surface 3-D						Surface 3-D supérieure
		z_1	z_2	z_3	z_4	z_5	z_6	
-12	0	-10	-11	-12	-48	-49	-50	0
-11	0	-10	-11	-12	-45	-46	-47	0
-10	0	-10	-11	-12	-42	-43	-44	0
-9	0	-7	-8	-9	-39	-40	-41	0
-8	0	-7	-8	-9	-36	-37	-38	0
-7	0	-7	-8	-9	-33	-34	-35	0
-6	0	-4	-5	-6	-26	-29	-32	0
-5	0	-4	-5	-6	-25	-28	-31	0
-4	0	-4	-5	-6	-24	-27	-30	0
-3	0	-1	-2	-3	-17	-20	-23	0
-2	0	-1	-2	-3	-16	-19	-22	0
-1	0	-1	-2	-3	-15	-18	-21	0
0	0	0	0	0	0	0	0	0
1	-13	1	3	5	7	16	25	-51
2	-14	2	4	6	7	16	25	-51
3	-13	1	3	5	8	17	26	-52
4	-14	2	4	6	8	17	26	-52
5	-13	1	3	5	9	18	27	-53
6	-14	2	4	6	9	18	27	-53
7	-13	1	3	5	10	19	28	-54
8	-14	2	4	6	10	19	28	-54
9	-13	1	3	5	11	20	29	-55
10	-13	1	3	5	12	21	30	-56
11	-14	2	4	6	12	21	30	-56
12	-13	1	3	5	13	22	31	-57
13	-14	2	4	6	13	22	31	-57
14	-13	1	3	5	14	23	32	-58
15	-14	2	4	6	14	23	32	-58
16	-13	1	3	5	15	24	33	-59
17	-14	2	4	6	15	24	33	-59

Dans tous les cas utilisant la représentation prismatique de la géométrie, on a utilisé une quadrature planaire trapézoïdale (EQW) doublée d’une quadrature polaire gaussienne (GAUS) pour le processus d’extrusion. Pour l’équivalence dans la représentation classique 3-D de la géométrie, on a utilisé une quadrature Santamarina-Sanchez (SMS – Sanchez *et al.* (2002)) qui convolve une quadrature de Gauss-Legendre et trapézoïdale, en plus d’effectuer une rotation suivant les trois axes (x, y et z) afin de conserver le caractère « level-symmetric » de la quadrature. Cette rotation n’est bien entendu pas appliquée dans le cas prismatique comme l’axe de projection est, dans le cas général, unique.

6.1.2 Résultats et discussion

Pour commencer, nous présentons au tableau 6.3, les erreurs sur les volumes discutées à la fin de la dernière section. Ces volumes sont moyennés sur l’ensemble des angles (à la fois en 2-D [projection] et en 3-D [extrusion]), mais on peut aussi vérifier que les erreurs angulaires sont toutes inférieures à 0,67% lors de l’extrusion à l’aide de l’erreur maximale.

De plus, on peut vérifier que le processus de traçage 2-D est effectué normalement en comparant les volumes 2-D numériques qui sont aussi calculés lors du traçage, et en s’assurant que leur valeur s’approche suffisamment de celle des volumes calculés analytiquement. Dans ce cas, on trouve une erreur maximale de 3,25%. Cette erreur, bien qu’acceptable, reste supérieure à l’erreur en extrusion.

Cette observation provient à notre avis des plus petites surfaces (et régions) dans la géométrie 2-D en raison des frontières virtuelles créées lors de la projection, et on peut le vérifier en raffinant le traçage uniquement dans le plan projeté, ce qui permet de réduire cette erreur. Une observation supplémentaire permet par ailleurs d’appuyer cette affirmation : Le plus petit volume dans la géométrie 3-D originale représente 2% du volume total, alors que la plus petite surface (virtuelle) dans la géométrie projetée représente moins de 0,4% de la surface totale, un fait qui indique qu’une densité plus importante est requise dans ce type de situation.

Tableau 6.3 Erreurs sur les surfaces et les volumes calculés

	Géométrie projetée		Géométrie Extrudée	
	Volumes (surfaces)	Surfaces (arêtes)	Volumes	Surfaces
Erreur RMS	0.050 %	1.1 %	0.15 %	0.44 %
Erreur Maximale	0.13 %	3.3 %	0.25 %	0.67 %
Erreur moyenne	0.0035 %	-0.40 %	-0.051 %	-0.40 %

Les résultats détaillés pour ce premier cas représentant le bout d'un crayon sont présentés au tableau 6.4. On a ici effectué un simple calcul de flux (TYPE K), qui a convergé avec une précision établie par défaut à 5 pcm, ce qui est supérieur à la différence observée dans les calculs. La rotation selon les trois axes mentionnée plus haut explique vraisemblablement l'absence de correspondance absolument exacte entre les deux méthodes de résolution.

Comme ces résultats l'illustrent bien, le méthode est fiable et robuste, en plus de ne pas affecter le processus itératif, comme les modifications effectuées n'affectent que les itérations internes du calcul MoC, et que le solveur au final ne voit que les lignes d'intégration 3-D recombinaées, similaires aux lignes originales du solveur traditionnel. On observe donc que le traçage est lu le même nombre de fois dans les deux cas, en plus d'effectuer le même nombre d'itérations internes à chaque itération externe.

De plus, il est intéressant d'effectuer la vérification que l'algorithme de traçage en projection fonctionne tel qu'il était prévu sur la géométrie projetée. Outre la vérification effectuée à l'aide de la matrice qui vient d'être présentée, on peut, à l'aide du module TLM: s'assurer que les lignes sont bien tracées, et que les régions virtuelles créées lors de la projection sont bel et bien de dimensions et aux endroits adéquats.

On présente donc à la figure 6.3 un exemple d'une cellule toujours du même type (avec toutefois une quadrature plus grossière) : le bout d'un crayon, constituée de deux sections cubiques superposées, dont l'une, autrement uniforme, contient un cylindre (dans lequel on observe l'effet de la discrétisation radiale due à SPLITR 2), et l'autre une discrétisation cartésienne 2×2 . On observe donc que le résultat du traçage 2-D effectué sur la géométrie projetée, présente à la fois les caractéristiques combinées des deux cellules : le traçage cartésien 2×2 et le crayon, le tout superposé.

Finalement, il est aussi très intéressant de noter les améliorations apportées en termes de temps de calcul et en besoins mémoire de la méthode. La taille du fichier de traçage passe de plus de 350Mo à moins de 75ko, une amélioration considérable. De plus, le temps de calcul est fortement diminué, passant de plus de 13 minutes à environ deux minutes (dont moins d'une seconde à projeter la géométrie, et à effectuer le traçage 2-D). On retrouve par ailleurs le même niveau de précision qu'avec la méthode traditionnelle.

6.1.3 Validation

On effectue par la suite une validation plus approfondie, en trois étapes sur ce bout de crayon, en faisant varier la géométrie utilisée. On commence par valider les résultats d'une supercellule

Tableau 6.4 Comparaison des solveurs MoC 3-D et prismatique

Type de solveur utilisé	MoC Régulier	MoC Prismatique	Diff (pcm)
Temps CPU	791	134	
Tracking lu	44 fois	44 fois	
k_∞	1.13663	1.13664	-1
Nombre de calculs de flux	81	81	
Groupe 1			
Flux Moyenné	1.51404E-01	1.51403E-01	1
Flux Intégré	3.78510E+01	3.78507E+01	1
Taux de collision	2.00447E+01	2.00443E+01	2
Taux d'absorption	3.64668E-01	3.64693E-01	-7
Taux de fission ($\nu\Sigma_f$)	2.85107E-01	2.85127E-01	-7
Taux de production	1.13663E+00	1.13664E+00	-1
Diffusion (1 → 1)	1.90175E+01	1.90171E+01	2
Diffusion (1 → 2)	6.62549E-01	6.62522E-01	4
Groupe 2			
Flux Moyenné	4.69658E-02	4.69614E-02	9
Flux Intégré	1.17414E+01	1.17404E+01	9
Taux de collision	1.80953E+01	1.80935E+01	10
Taux d'absorption	6.35332E-01	6.35307E-01	4
Taux de fission ($\nu\Sigma_f$)	8.51519E-01	8.51508E-01	1
Taux de production	0.00000E+00	0.00000E+00	–
Diffusion (2 → 2)	1.74328E+01	1.74310E+01	10
Diffusion (2 → 1)	2.72163E-02	2.72137E-02	10

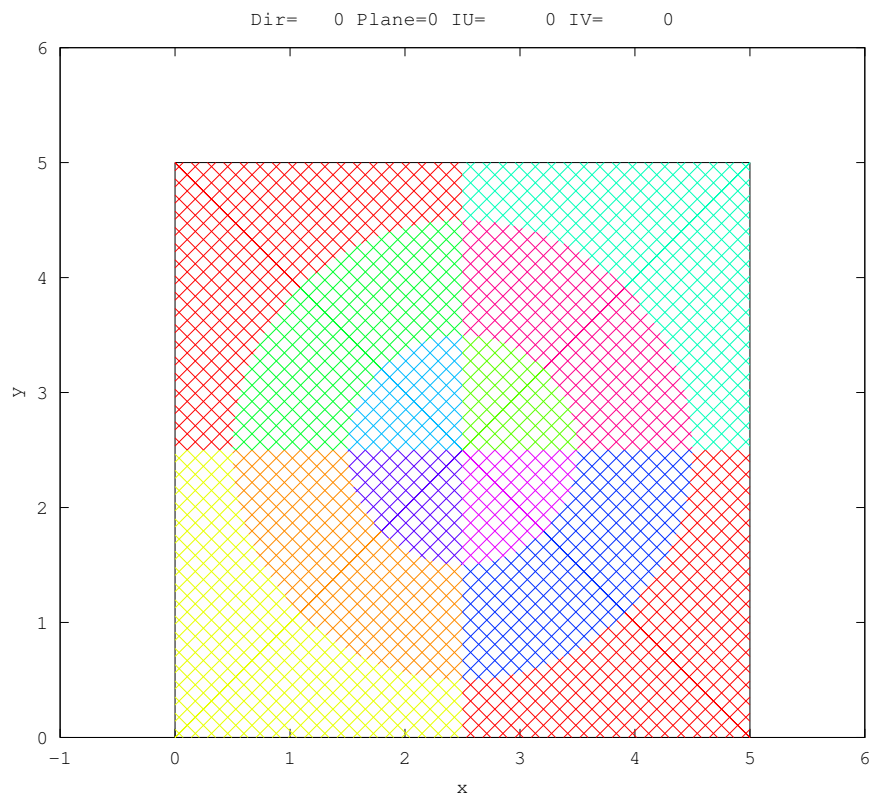


Figure 6.3 Traçage sur la géométrie projetée

uniforme selon l'axe d'extrusion avec toutes les méthodes applicables et disponibles dans Dragon : Calcul MoC 2-D, Calcul MoC 3-D, Calcul probabilités de collision (CP) 3-D, Calcul MoC prismatique uniforme et Calcul MoC prismatique non-uniforme. On trouve ces résultats au tableau 6.5.

On conserve ensuite la même discrétisation géométrique sur la hauteur de la géométrie, mais on remplace le combustible du crayon supérieur par du modérateur (en conservant le SPLITR). On peut ici utiliser les méthodes suivantes : Calcul MoC 3-D, Calcul CP 3-D, Calcul MoC prismatique uniforme et Calcul MoC prismatique non-uniforme. On trouve ces résultats au tableau 6.6

Finalement, on remplace la discrétisation de la cellule supérieure par une discrétisation purement cartésienne, toujours entièrement remplie de modérateur. On pourra ici comparer les cas de : Calcul MoC 3-D, Calcul CP 3-D et Calcul MoC prismatique non-uniforme. On trouve ces résultats au tableau 6.7

La première chose qu'on peut remarquer à observer est l'adéquation entre les résultats des calculs MoC entre les trois méthodes utilisées, ne comportant que de minimes différences (de l'ordre du pcm) pouvant être reliées à la précision numérique dans les cas 3-D et prismatiques

Ensuite, on peut observer que les propriétés numériques de l'algorithme sont conservées, en termes de diminution des capacités mémoire et de diminution du temps de calcul. Les valeurs de k_{eff} ne sont pas non plus en parfaite adéquation avec celles présentées dans la section 6.1.2 en raison de variations dans la géométrie utilisée.

On a aussi retiré les discrétisations cartésiennes pour ne conserver que trois ou quatre régions spatiales (selon le cas) pour ne pas surcharger les tableaux de résultats lors de l'inclusion des taux de réaction par région – ceux-ci sont disponibles à l'annexe II. On y trouve que les comparaisons avec la méthode des caractéristiques 3-D sont acceptables, généralement similaires à celles obtenues par la méthode uniforme lorsqu'applicable, et inférieure à la différence avec la méthode des CP.

Tableau 6.5 Validation – Composition uniforme

Méthode	K_{eff}	Temps NXT: + ASM: + FLU: (s)	Taille
MoC 2-D	1.18887	1	56ko
CP 3-D	1.18805	86	363Mo
MoC 3-D	1.18811	222	363Mo
Prism Uniforme	1.18819	38	82ko
Prism Non-uniforme	1.18819	36	82ko

Tableau 6.6 Validation – Géométrie uniforme

Méthode	K_{eff}	Temps NXT: + ASM: + FLU: (s)	Taille
CP 3-D	1.13545	101	363Mo
MoC 3-D	1.13590	837	363Mo
Prism Uniforme	1.13591	89	82ko
Prism Non-uniforme	1.13591	110	82ko

Tableau 6.7 Validation – Géométrie non-uniforme

Méthode	K_{eff}	Temps NXT: + ASM: + FLU: (s)	Taille
CP 3-D	1.13557	75	335Mo
MoC 3-D	1.13615	737	335Mo
Prism Non-uniforme	1.13616	94	82ko

Enfin, mentionnons dans le premier cas à la composition uniforme, la déviation un peu plus importante de la valeur du k_{eff} en 2-D vis-à-vis des valeurs 3-D (à la fois régulier et prismatique).

Cette déviation est due à deux facteurs. Premièrement, la discrétisation en z imposée par les cellules superposées dans les cas 3-D (la frontière entre les deux cellules) affecte le traçage des lignes en séparant des régions qui seraient normalement jointes. Dans un second temps, le traitement des conditions frontières (mentionnées dans la section 3.3.6). Dans le cas qui nous intéresse, des conditions frontières de réflexion (mot-clé REFL) sont utilisées pour les directions $z+$ et $z-$, et associées à un traçage isotrope (mot-clé TISO). Ceci correspond donc à des conditions de réflexion isotropes alors que le modèle 2-D correspond à des conditions de réflexion spéculaires.

6.1.4 Convergence Spatiale

Il est par ailleurs intéressant d'étudier les propriétés de convergence spatiale de l'algorithme de projection et d'extrusion, et notamment la dépendance à l'ordre de la quadrature angulaire choisie, notamment en fonction de l'angle polaire qui est particulièrement déterminant pour une méthode prismatique. Dans ce cas, la géométrie reste la même, ainsi que la densité de lignes de traçage – on utilise ici les mêmes densités que précédemment, soit 10×10 et 100 – on varie simplement l'ordre de la quadrature angulaire choisie. Les résultats sont présentés au tableau 6.8 et à la figure 6.4.

Dans cette étude de convergence spatiale, on observe donc que le raffinement de la quadrature permet effectivement de converger sur une valeur du k_{eff} , sans nécessairement permettre de

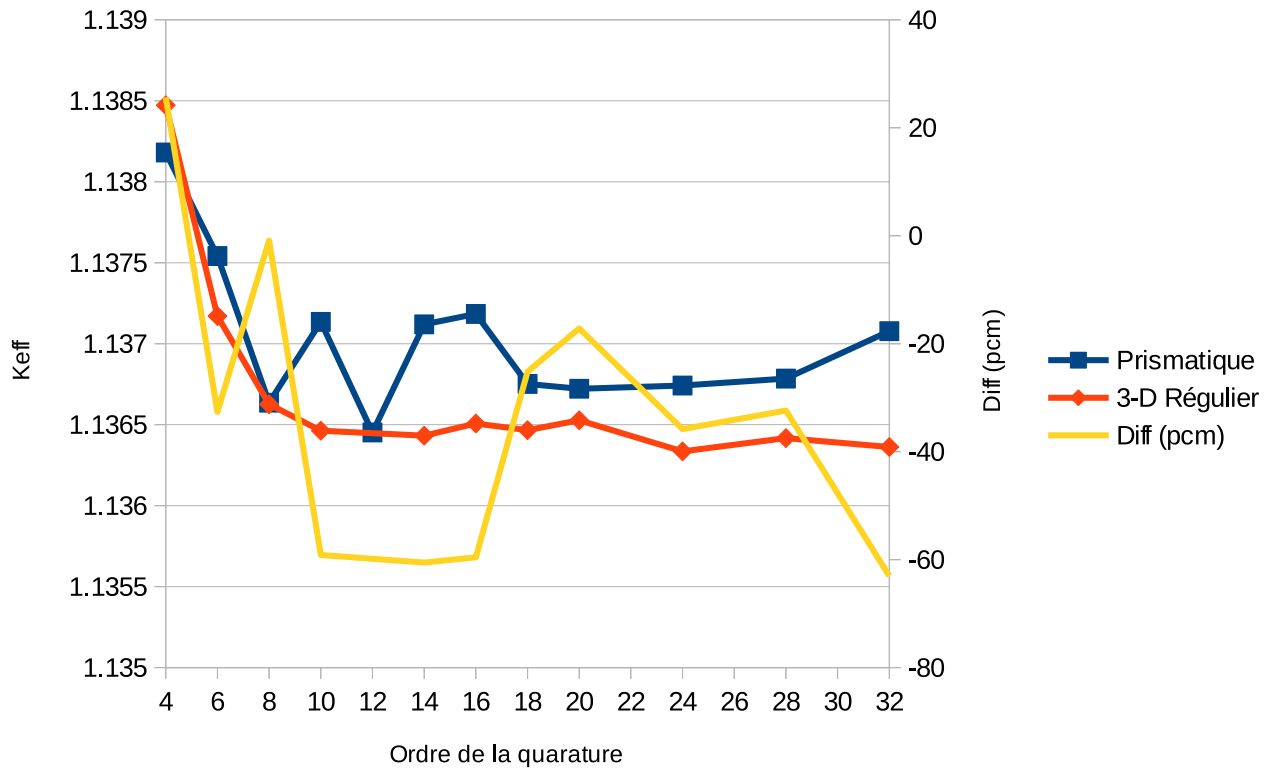


Figure 6.4 Convergence Spatiale - résultats

Tableau 6.8 Convergence spatiale - résultats

Quadrature	Prismatique	3-D Régulier	Diff (pcm)
(4 × 6)	1.13818	1.13847	26
(6 × 9)	1.13754	1.13717	-33
(8 × 12)	1.13664	1.13663	-1
(10 × 15)	1.13714	1.13646	-59
(14 × 21)	1.13712	1.13643	-61
(16 × 24)	1.13718	1.13651	-60
(18 × 27)	1.13675	1.13647	-25
(20 × 30)	1.13672	1.13653	-17
(24 × 36)	1.13674	1.13634	-36
(28 × 42)	1.13679	1.13642	-32
(32 × 48)	1.13708	1.13636	-63

réduire l'écart par rapport à la valeur de référence (méthode 3-D). L'explication la plus plausible à ce niveau reste que comme les régions spatiales sont déjà de grande taille, le processus est déjà pratiquement convergé dans le cas construit à l'aide de l'ordre de la quadrature angulaire la plus faible. (4×6 – planaire $\varphi \times$ polaire θ). Notons par ailleurs que la densité de lignes utilisée dans le cas 3-D est en quelque sorte équivalente à une densité effective environ du triple en raison de la rotation mentionnée dans la section 6.1.

Dans le meilleur des cas (8×12), on observe une correspondance quasi-exacte entre les deux méthodes, et on peut en conclure qu'il s'agit d'un cas particulier où les deux quadratures correspondent particulièrement bien, et peuvent donc être considérées presque équivalentes, ou alors que les différences introduites par la rotation décrite au paragraphe précédent sont compensées au fur et à mesure de la progression du calcul.

Dans tous les cas, on observe des différences généralement faibles, de l'ordre de quelques dizaines de pcm, ce qui reste en deçà de la marge qui serait jugée acceptable en raison du traitement numérique lui-même, entre autres en raison de l'ordre des opérations, de la précision machine ou du processus de convergence, de même que les différences entre les quadratures mentionnées précédemment. On peut donc conclure que la solution implantée dans ce cas fonctionne de manière adéquate, et présente les avantages escomptés à la fois pour l'amélioration de la vitesse d'exécution et la réduction des besoins mémoire.

6.2 Simulation d'un assemblage flambé

6.2.1 Description du problème

Dans cette section, nous décrivons une version simplifiée d'un assemblage REP, composé de 3×3 crayons ayant été déformés sous l'effet de la poussée due au flux d'eau dans la cuve, combinée au flux de neutrons et autres gradients thermiques présents dans le coeur du réacteur. Une modélisation simplifiée d'un tel assemblage a un profil semblable à celui présentée à la figure 6.5. Dans ce cas, nous avons utilisé les mêmes sections efficaces que pour le bout de crayon, présentées au tableau 6.1, toujours avec les évaluations ENDF/B-VI à 172 groupes. La lame d'eau est modélisée en utilisant la même composition isotopique que le modérateur, en restant cependant traitée séparément dans le cadre du calcul. L'assemblage est donc composé de 9 crayons de combustible de 0,419 cm de rayon, recouverts d'une gaine de 0,57 mm d'épaisseur. Ces crayons sont séparés par un pas de réseau de 1,26 cm, et l'ensemble est entouré d'une lame d'eau de 0,42 mm d'épaisseur.

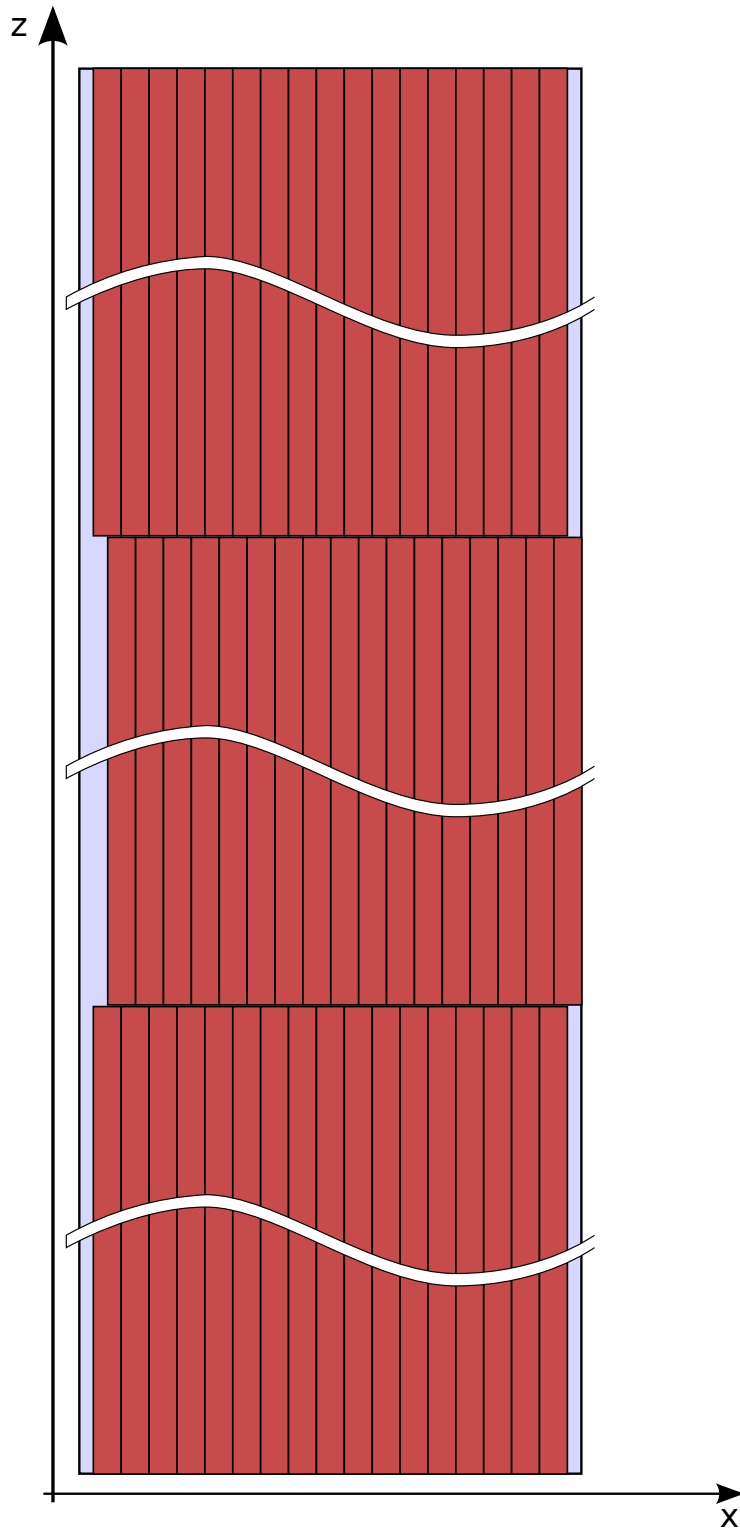


Figure 6.5 Assemblage flambé – vue de côté

Cependant, en raison d'une incompatibilité de la numérotation de la nouvelle géométrie projetée avec les modules de traçage 2-D tels que disponibles à l'heure actuelle dans DRAGON, l'implémentation développée ne permet pas le traitement de la superposition directe de crayons cylindriques (tels que présentés à la figure 6.8), ainsi que précédemment illustré à la figure 4.2. On aura donc recours à une méthode alternative, où un processus de *super-homogénéisation* (Hébert et Mathonnière, 1993) est réalisé pour effectuer une équivalence transport-transport sur les géométries 2-D pour chaque cellule contenant des crayons cylindriques, et une cellule dite « rectangularisée », où le volume matériel est conservé, et les sections efficaces utilisées sont issues du module SPH:.

Les dimensions globales (pas de réseau, lame d'eau) décrites ci-haut ne sont donc pas modifiées, mais les nouveaux crayons rectangularisés utilisés pour le calcul de flux ont maintenant 0,742 cm de côté, et sont recouverts d'une gaine de 0,51 mm d'épaisseur. L'équivalence est dans tous les cas effectuée avec un pas traçage de 20 lignes/cm et une quadrature de 10 directions à poids égaux. Les sections efficaces équivalentes sont calculées à 172 groupes et condensées à deux groupes (avec une limite à 0,625eV), et ce sont ces dernières qui sont utilisées pour la suite du calcul.

Dans ce cadre, six correspondances en tout ont été effectuées, trois chacune pour les sections d'assemblage droites puis déformées. Cette procédure est symbolisée par la figure 6.9 pour les cellules de coin, mais aura aussi été effectuée pour les cellules de côté et centrales, où la même discrétisation a été utilisée que pour le calcul de flux principal. Comme les propriétés physiques des 4 cellules de coin (voir figure 6.6) de chaque assemblage sont équivalentes en termes de propriétés neutroniques, ne différant que par une rotation de $\pi/2$, on n'effectue qu'une seule telle équivalence pour les 4, en réutilisant les nouvelles sections efficaces. La même logique s'applique aux 4 cellules de côté de l'assemblage.

Cette rectangularisation des crayons de combustible nous permet d'effectuer subséquemment la superposition des cellules contenant ces crayons décalés, la différence n'étant représentée entre les deux que grâce à une superposition de grilles cartésiennes non-régulières. La section équivalente de l'assemblage global rectangularisé est donc présentée à la figure 6.10. Les détails dans le fichier d'entrée sont disponibles à l'annexe I.

Dans ce cas précis, nous avons utilisé une quadrature équivalente dans les deux cas (comme dans la section précédente), soit un couplage d'une quadrature azimuthale trapézoïdale combinée avec une quadrature polaire de Legendre proposée par Sanchez *et al.* (2002), pour fournir au final le même nombre de directions dans le cas 3-D et prismatique. Finalement, la densité des lignes a été encore une fois établie à 100 lignes/cm en 3-D, et à 10 lignes/cm sur

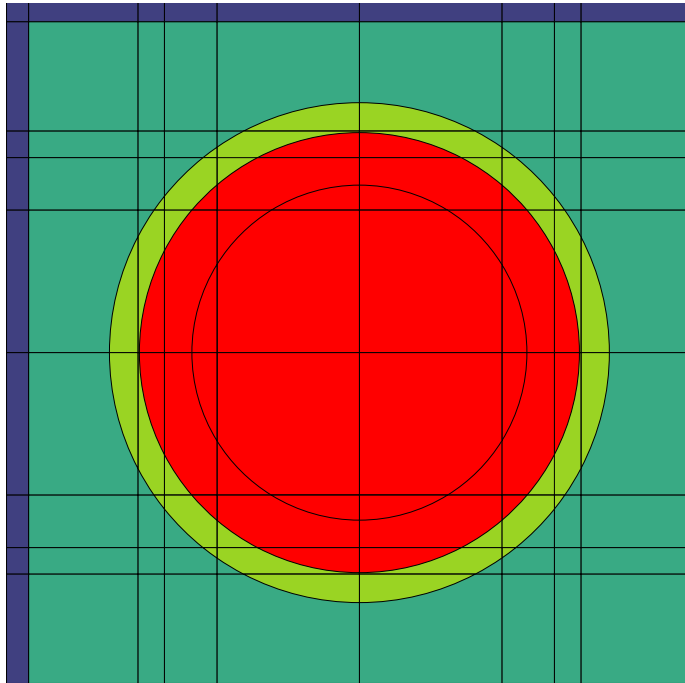


Figure 6.6 Crayon cylindrique non-déformé

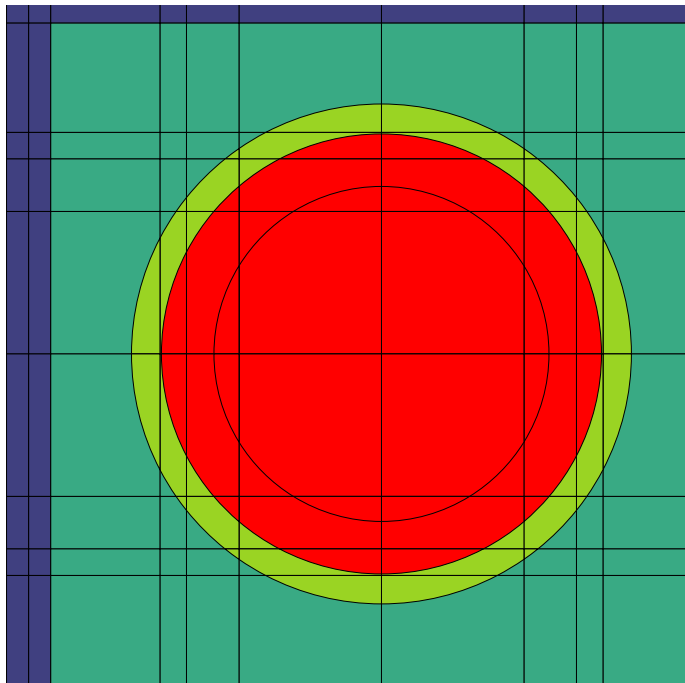


Figure 6.7 Crayon cylindrique déformé

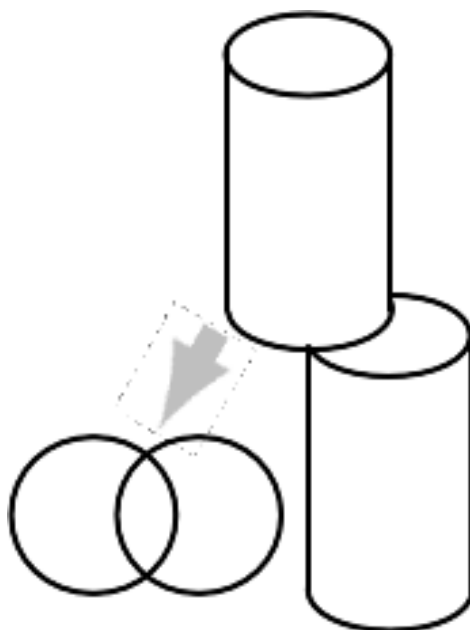


Figure 6.8 Superposition de crayons cylindriques

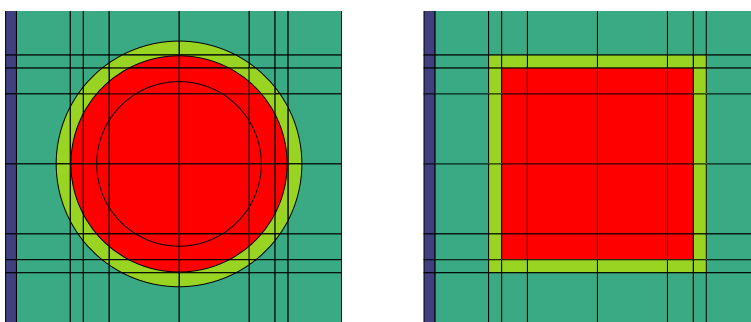


Figure 6.9 Équivalence entre les cellules régulières et rectangularisées

la géométrie projetée doublée de 10 lignes/cm en extrusion, pour la même densité dans les deux cas.

Finalement, on peut voir le résultat de la projection de l'assemblage complet à la figure 6.11. Remarquons sur cette dernière que l'on peut observer très clairement les régions de très petite taille, virtuelles, notamment près de la gaine du combustible. Ces dernières auront été créées par le processus de projection, et sont principalement dues à la faible différence entre la déviation que nous avons appliquée à la section centrale de l'assemblage et l'épaisseur de la gaine.

Les résultats en termes de k_{eff} sont présentés au tableau 6.9. Comme on peut l'observer, toutes les différences, incluant l'effet du fluage, sont bien en deçà de la tolérance du processus de calcul.

Comme il était attendu, la taille du fichier contenant les lignes de traçage est de beaucoup inférieure à celle du fichier produit par la procédure 3-D équivalente. Ainsi, la quantité de mémoire nécessaire pour la méthode traditionnelle 3-D se situe au delà de 3Go, alors que la géométrie projetée ne requiert que 345ko, incluant l'espace de stockage additionnel nécessaire pour la géométrie 2-D avec ses régions virtuelles, et la matrice de projection associée.

Finalement, comme dans le cas du bout de crayon étudié dans la section précédente, des améliorations considérables au niveau du temps de calcul sont observées lors de l'emploi de la méthode prismatique. Ainsi, on passe d'environ 200 000 secondes (56 heures) pour la méthode traditionnelle 3-D à moins de 35 000 secondes (10 heures) – les temps de résolution étant à toutes fins pratiques égaux entre les géométries droites et déformées.

Cependant, comme on peut le voir au tableau 6.9, les résultats du calcul de flux et de k_{eff} des assemblages droit et flué sont très similaires, à environ 50 pcm d'écart, soit moins que la différence entre les deux méthodes (3-D prismatique et 3-D traditionnelle).

Afin de tenter d'exagérer la différence entre les deux méthodes, nous avons modifié la géo-

Tableau 6.9 Résultats assemblage déformé

	Prismatique	Non-Prismatique	Diff (pcm)
Droit	1.28452	1.28547	-74
Déformé	1.28449	1.28517	-53
Diff (pcm)	2	24	
2-D	1.28483		

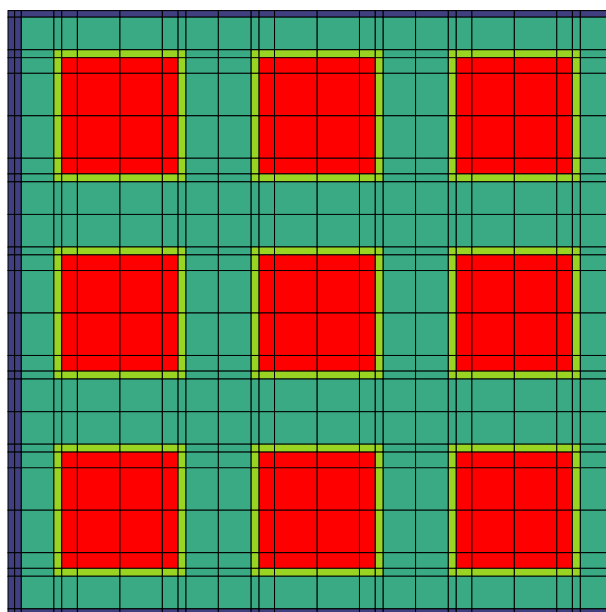
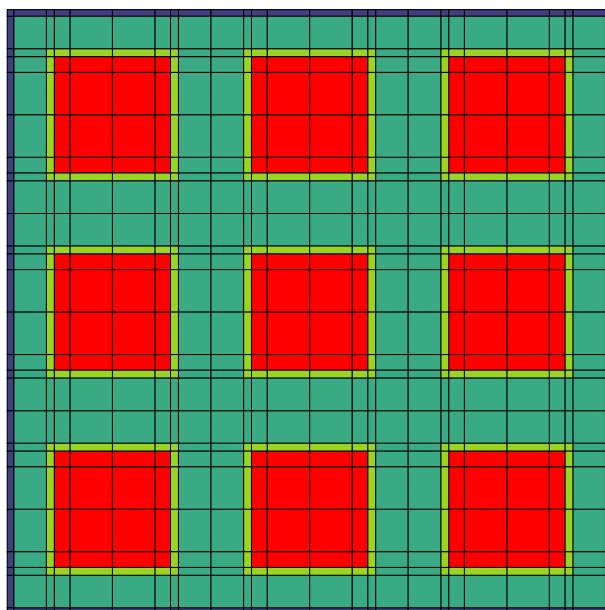


Figure 6.10 Cellules cartésiennes normales et déformées

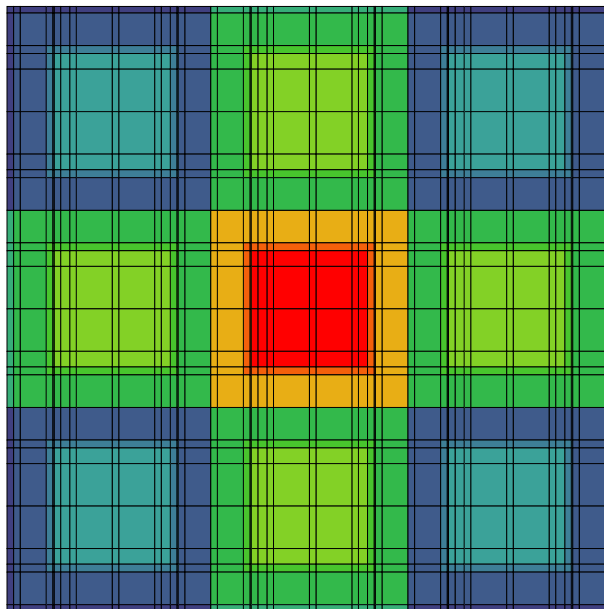


Figure 6.11 Cellules cartésiennes projetées

métrie quelque peu en remplaçant l'un des crayons de combustible du côté de l'assemblage par un trou d'eau, ainsi qu'exagéré le flambage pour le crayon central de la section du milieu de manière à ce qu'il touche la cellule suivante. La région centrale de l'assemblage est donc représentée par une section semblable à celle présentée à la figure 6.12.

Le calcul de flux sur cette nouvelle section présente toujours les mêmes caractéristiques que l'assemblage plus légèrement flambé, c'est-à-dire que le fichier de traçage reste réduit à une taille plusieurs ordres de grandeur inférieure à celle du fichier normalement généré, et le calcul s'effectue environ dix fois plus rapidement. Toutefois, la différence entre les assemblages flambé et droit (qui inclut toujours le trou d'eau) reste inférieure à celle entre le calcul de flux normal et le calcul prismatique, tel que démontré au tableau 6.10.

6.2.2 Discussion

Des efforts précédents de Le Tellier *et al.* (2008) ont montré une amélioration du temps de calcul lors de l'utilisation d'un algorithme prismatique de calcul, en ne permettant toutefois pas les variations géométriques le long de l'axe de projection. En plus de réduire le temps de calcul nécessaire pour le module de traçage (incluant l'algorithme de projection), la quantité de mémoire a été dramatiquement réduite, d'un facteur de l'ordre de 10 000.

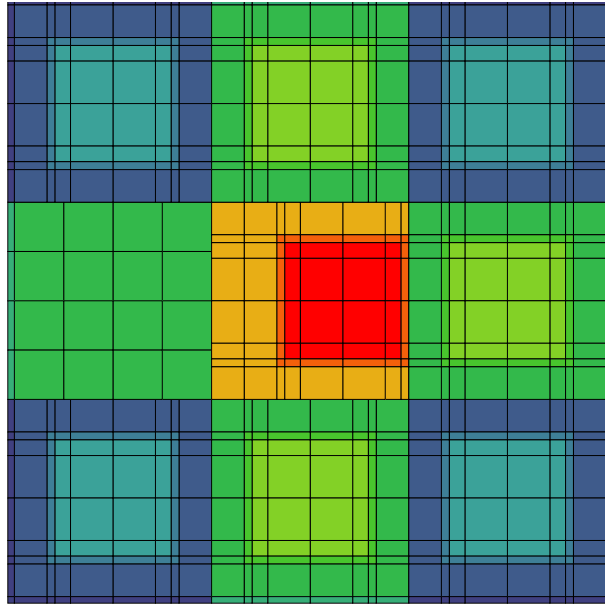


Figure 6.12 Flambage exagéré

Tableau 6.10 Résultats assemblage déformé-vide

	Prismatique	Non-Prismatique	Diff (pcm)	2-D
Droit	1.29452	1.29522	70	1.29440
Déformé	1.29402	1.29473	70	1.29293
Diff (pcm)	49	49	–	146

Dans le cas de cette géométrie en particulier, on peut supposer que comme les propriétés géométriques en font une tour de très grande hauteur par rapport à sa base (10:1), le coût de l'extrusion pour les directions ayant de très petit cosinus polaire peut mener à certaines instabilités numériques, en plus d'un certain surcoût d'extrusion et de recombinaison plus important que dans le cas du crayon étudié précédemment. De plus, les accélérations de préconditionnement ne sont plus compatibles avec le mode de recombinaison des segments tel qu'implanté.

Finalement, malgré ce contretemps, à terme, la multiplication des architectures parallèles fait généralement en sorte que de plus en plus les applications sont limitées par les accès en lecture et en écriture sur les disques, et un tel compromis entre le temps de calcul (qui s'en trouve tout de même réduit) et la lecture mémoire aura tendance à être bénéfique à plus long terme.

6.2.3 Comparaison et performance

La méthode reste cependant à perfectionner. L'aspect le plus important à cet égard reste les problèmes qui surviennent lors de la superposition de crayons cylindriques non-alignés. Ainsi, il a été envisagé de modifier le module de traçage afin d'en calculer les longueurs, ce qui aurait été possible dans le cas présent. Cependant, cette option laissait intact le problème de la numérotation des régions ainsi créées, qui devient rapidement très complexe dans le cas général à n crayons.

Une méthode a été mise de l'avant que nous croyons être en mesure de régler ce problème impliquant un nouveau schéma de numérotation selon chaque ligne d'intégration en fonction de sa hauteur. Celle-ci aurait cependant nécessité une réécriture d'une bonne partie de l'algorithme de projection et des routines de traçage, alors qu'un des avantages principaux de la méthode décrite ici est de reposer en grande partie sur les routines déjà disponibles dans les codes existants. Il a donc été décidé de retenir l'approche décrite plus tôt (chapitre 4), en réservant l'option de traiter les crayons de combustible autrement lors de développements subséquents.

Dans le cas qui nous concerne, ce problème est réglé en utilisant plutôt l'équivalence transport-transport décrite plus tôt, qui permet le traitement de la géométrie sans recourir à la superposition des sections de crayon cylindriques désalignées.

De plus, certains problèmes additionnels surviennent dus à la particularité des problèmes d'intérêt pour les géométries prismatiques. En particulier, des géométries qui varient très peu

dans l'axe de projection mèneront à la création de régions virtuelles de très petites tailles, qui – répétons-le – n'ont pas de signification physique directe. Elles viennent toutefois poser des problèmes de convergence numérique en raison de leur petite taille, et, incidemment, de la petite taille des segments qui les traversent.

Dans le cas de l'assemblage déformé qui est considéré ici, on observe ce phénomène de manière très claire, car l'épaisseur de la lame d'eau du déplacement est très près de l'épaisseur de la gaine du combustible, et on produit donc des régions dont l'épaisseur équivaut à la différence entre les deux, tel qu'illustré à la figure 6.11.

Un autre aspect à considérer dans les géométries prismatiques généralement considérées est que l'axe de projection sera généralement plus étendu spatialement que les autres – on n'a qu'à penser à une série de grappes de combustibles dans un réacteur CANDU, ou encore à des barres de contrôle dans les REP.

En particulier, pour notre assemblage déformé, ce rapport des dimensions de la base par rapport à la hauteur d'environ 10:1 se retrouve généralement aussi dans les dimensions des sous-régions spatiales considérées afin de ne pas surcharger le nombre de régions. Cette particularité a pour effet de produire un nombre important de régions de très petites tailles sur les surfaces supérieure et inférieure de la géométrie, qui seront difficilement atteignables par les rayons extrudés en raison des instabilités numériques dues aux relatives petites tailles des segments projetés, et requerront possiblement un ordre de quadrature polaire plus élevé.

6.3 Calcul parallèle

Finalement, nous développons ici le second volet de ce projet, soit la parallélisation du solveur MoC. Comme il a été mentionné dans le chapitre 5, cette parallélisation recoupe deux aspects, soit une parallélisation sur les lignes d'extrusion dans un premier temps, et une boucle sur les groupes. Cette boucle sur les groupes présente par ailleurs l'avantage de s'appliquer aux autres méthodes de résolution déjà implantées dans le solveur MCGT :. Les tests initiaux de performance de cette parallélisation sur les groupes d'énergie sont prometteurs, et il s'agit sans aucun doute d'une méthode qui devrait être mieux exploitée dans de futurs développements.

Cependant, la parallélisation sur les groupes reste instable dans son implantation actuelle, et les résultats restent variables, sans toujours apporter un avantage direct. Nous avons préféré nous pencher davantage sur les résultats de la méthode de parallélisation sur les lignes d'extrusion en géométrie prismatique, car il s'agit ici d'une parallélisation toute particulière et propre à cette méthode de résolution – quoiqu'elle s'applique aussi avec le solveur prismatique

uniforme de LeTellier. La parallélisation implantée dans le cadre de ce projet s'applique donc à la fois aux deux méthodes prismatiques, uniforme ou non, la méthode uniforme ne se présentant que comme un cas particulier du solveur général.

6.3.1 Cas uniforme

Le premier cas-test pour le solveur parallèle s'applique à une géométrie homogène axialement, afin de tester directement la parallélisation et d'écarter les déviations qui auraient pu être dues à la non-uniformité de la géométrie dans l'axe de projection. On utilise donc dans un premier temps une géométrie relativement pathologique pour le traitement parallèle - deux cellules superposées, l'une contenant du modérateur et l'autre du combustible, formant un assemblage de $10 \times 10 \times 40$ cm lui-même discrétisé en $10 \times 10 \times 200$ régions.

L'architecture matérielle utilisée ici reste très simple, consistant en deux CPU physiques (partageant le même *socket*), multithreadés à deux fils chacun. Les résultats de ce calcul sont présentés au tableau 6.11 et à la figure 6.13

Une observation avant d'entrer dans l'analyse des résultats numériques en tant que tels. On observe ici dans un premier temps la précision de la méthode : tous les K_{eff} sont à toutes fins pratiques égaux, les seules différences, de l'ordre du pcm, peuvent être expliquées par des changements dans l'ordre des opérations. On peut notamment penser à deux fils d'exécution qui terminent les opérations qui leurs sont assignées dans un ordre différent en fonction du nombre total de fils. En effet, en changeant le nombre de fils d'exécution, l'utilisateur a pour effet de séparer les opérations à effectuer en un nombre différent d'ensembles, qui seront répartis différemment entre les processeurs, et ne se termineront du coup pas nécessairement dans le même ordre.

Tableau 6.11 *Speedup* et efficacité pour une géométrie uniforme

	Temps FLU: (s)	K_{eff}	Taille fichier traçage	Speedup	Efficacité
Prismatique séquentiel	854	1.05767	294 Ko	1.000	1.000
Parallèle 1 <i>thread</i>	855	1.05767	294 Ko	0.999	0.999
Parallèle 2 <i>threads</i>	471	1.05767	294 Ko	1.813	0.907
Parallèle 3 <i>threads</i>	415	1.05767	294 Ko	2.058	0.686
Parallèle 4 <i>threads</i>	329	1.05767	294 Ko	2.596	0.649
Parallèle 5 <i>threads</i>	419	1.05767	294 Ko	2.038	0.408

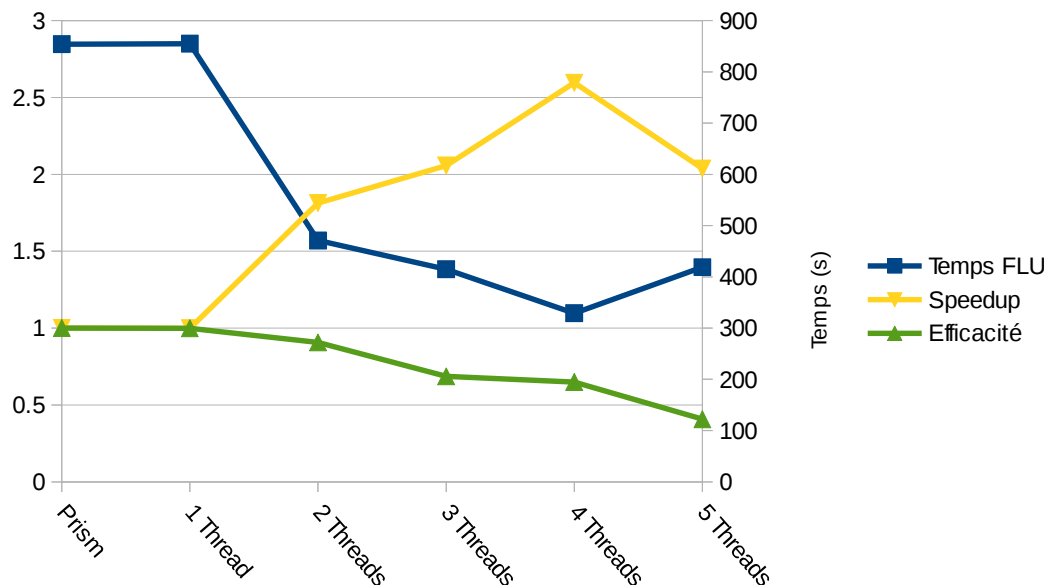


Figure 6.13 *Speedup* et efficacité pour une géométrie uniforme

Du point de vue de la performance parallèle, on observe que la parallélisation des boucles telle qu'implantée (et présentée à la section 5.5.4) permet une réduction du temps de calcul tel qu'attendu. Il est par ailleurs normal que l'accélération et l'efficacité soient réduites en fonction du nombre de *threads* utilisés, comme le prévoit la loi d'Amdhal présentée à la section 5.3.2. En raison notamment des sections qui restent intrinsèquement non-parallélisables (par exemple, le dernier paquet dans l'algorithme noir-rouge), mais aussi du fait que les boucles parallélisées sont imbriquées dans plusieurs autres boucles qui sont toujours pour l'instant non-parallélisées (par exemple, les boucles sur les angles 2-D et les boucles sur les angles d'extrusion). Néanmoins, l'utilisation de 4 *threads* en parallèle sur ce problème nous permet de réduire de plus de 60% le temps de calcul nécessaire pour la résolution de l'équation de transport, ce qui représente un avantage non-négligeable.

6.3.2 Assemblage

Finalement, le solveur parallèle est appliqué à l'assemblage déformé présenté à la figure 6.5 et qui est l'objet de la section 6.2. Dans ce cas, on considère déjà l'avantage qui est acquis en raison de l'utilisation de l'algorithme de projection, auquel on conjugue la performance de la parallélisation sur le décalage des lignes d'extrusion. Mentionnons que l'algorithme parallèle rouge-noir proposé et décrit dans la section 5.5.4 se doit de fonctionner sur une géométrie

dont la discrétisation selon l’axe de projection est suffisante.

En effet, comme la taille minimale des paquets doit correspondre à la section maximale des régions reconstruites selon tous les angles d’extrusion, le cas décrit plus tôt dans la section 6.2 n’obtiendra aucun avantage de parallélisation, comme les trois sections d’assemblage superposées sont de dimensions trop étendues spatialement selon l’axe d’extrusion. Autrement dit, la taille minimale des paquets requise pour paralléliser cet assemblage est trop grande, et requerrait que toutes les lignes soient traitées dans la dernière condition de l’algorithme 2. Cela revient à un algorithme séquentiel dont la performance finale sera pire que celle de la version séquentielle en raison du surcoût de la parallélisation qui reste inutilisée.

Afin d’établir la validité de cette méthode de parallélisation, cette même géométrie a été reprise, en discrétisant longitudinalement chaque section en 10 parts égales, pour 30 sections en tout. Cette discrétisation supplémentaire, couplée aux faibles dimensions des régions dans le plan de projection, permet une ségrégation adéquate des paquets, et donc une parallélisation plus efficace. Les résultats en utilisant 4 *threads* sont présentés au tableau 6.12.

On remarque tout d’abord une amélioration du temps de calcul non-négligeable, passant d’un peu moins d’environ 1h47 pour l’algorithme séquentiel prismatique tel qu’implanté actuellement, à un peu moins d’1h20. En raison des particularités de la géométrie considérée (discrétisation grossière selon l’axe de projection, faible étendue spatiale), nous attendons des gains plus importants si la méthode est étendue à des géométries de plus grandes dimensions spatiales.

L’efficacité reste cependant assez faible. Plusieurs facteurs peuvent être considérés pour expliquer cette performance parallèle décevante. En premier lieu, on se doit de considérer que la boucle parallélisée se situe au cœur de l’algorithme de résolution MoC – et est imbriquée dans plusieurs autres boucles séquentielles. Ces dernières viennent imposer un surcoût à chaque itération. Ce facteur est et reste intrinsèquement lié à la performance globale du module FLU:, par la loi d’Amdahl (équation (5.4)).

De plus, dans l’exemple précédent concernant le *speedup* sur le cas uniforme (section 6.3.1),

Tableau 6.12 *Speedup* et efficacité pour un assemblage déformé

	Temps FLU: (s)	K_{eff}	Taille fichier traçage	Speedup	Efficacité
Prismatique régulier	6423	1.284462	810 Ko	–	–
Prismatique parallèle	4772	1.284455	810 Ko	1.346	0.336

on n'a considéré que la première itération de puissance afin d'en évaluer la performance. Comme le cas ne représentait pas un cas physiquement représentatif, la convergence globale de l'algorithme devenait moins pertinente. Dans ce cas d'assemblage déformé et non-uniforme, on a laissé l'algorithme converger jusqu'à la précision désirée – dans ce cas, 10^{-5} – ce qui explique les temps de calcul plus importants.

Finalement, mentionnons aussi par ailleurs que la discrétisation supplémentaire mentionnée en début de section permet d'expliquer la différence dans la valeur du K_{eff} et des flux moyennés observés entre cette simulation et celle décrite à la section 6.2.

6.4 Sommaire

Ce chapitre aura réussi à démontrer la validité de la méthode développée dans le cadre de cette thèse, à l'aide de résultats numériques. Dans un premier temps, l'algorithme de projection a été testé et validé sur un cas de faibles dimensions – le bout d'un crayon de combustible – puis sur une géométrie de dimensions plus étendues – un assemblage de REP ayant subi un flambage.

La parallélisation de la méthode a aussi été vérifiée, et a été prouvée robuste, à l'aide de deux cas-test. Dans un premier cas, une géométrie prismatique mais uniforme (traitée aussi avec l'algorithme classique de DRAGON) a permis de valider les résultats numériques, et de prouver une accélération significative du temps nécessaire pour la résolution du problème. Dans un second temps, l'assemblage flambé utilisé pour la validation prismatique a été simulé.

En tout et partout, l'algorithme MoC de résolution prismatique parallèle permet une réduction très significative des besoins en stockage mémoire, ce qui ouvre la porte à des algorithmes à grande échelle conservant le fichier de traçage en mémoire vive. De plus, la diminution du temps de calcul nécessaire à la résolution du flux est aussi très significative, due à la fois à la parallélisation et à la projection et l'extrusion. L'algorithme permet notamment de conserver la même précision numérique et les capacités de simulation de variations axiales dans la géométrie à toutes fins pratiques équivalentes à un algorithme 3-D classique.

La conjugaison de ces deux avantages ouvre la porte à des calculs en transport 3-D à très grande échelle, notamment si la méthode développée est mariée à des méthodes de parallélisation plus classiques (par angle, par groupe d'énergie, etc.) Par ailleurs, si l'augmentation continue de la puissance des calculateurs observée au cours des dernières années se poursuit, il est permis de supposer que la simulation d'un réacteur complet en transport 3-D non-structuré pourrait être possible dans un avenir rapproché.

CHAPITRE 7 CONCLUSION

Ce projet de doctorat avait pour but ultime de développer une méthode plus efficace de résolution de l'équation de transport déterministe en géométries non-structurées, en tirant parti des propriétés prismatiques des réacteurs et d'une parallélisation de l'algorithme. Une représentation prismatique généralisée de la géométrie est proposée, qui ne présente pas d'approximation supplémentaire par rapport à un schéma 3-D traditionnel - moyennant cependant certaines contraintes sur les géométries admissibles. Par exemple, les réacteurs CANDU-6 devront être exclus en raison des dispositifs de réactivité cylindriques orthogonaux aux canaux de combustible.

La méthode des caractéristiques a été retenue afin de résoudre l'équation de transport, car elle a été prouvée robuste, fiable, efficace et surtout applicable au traitement de géométries à grande étendue spatiale. De plus, MoC permet de traiter efficacement les problèmes liés à l'anisotropie de la diffusion, en plus de fournir les flux angulaires. On doit cependant utiliser des techniques d'accélération afin d'améliorer la convergence du processus itératif intrinsèque à cette méthode.

Comme cette méthode repose à la base sur plusieurs opérations fondamentalement indépendantes les unes des autres, une parallélisation du solveur a été appliquée. Celle-ci se base sur les travaux effectués dans les solveurs MoC déjà implantés dans le code DRAGON. Plusieurs axes de parallélisation sont envisagés, afin de trouver la solution la plus performante, pour ensuite envisager un calcul à l'échelle d'un réacteur complet.

À chacune des étapes d'implantation d'une nouvelle procédure, le schéma est qualifié par rapport à des résultats provenant de codes reposant sur des méthodes différentes, pour y trouver une précision équivalente, tout en requérant moins de ressources, tant en termes de mémoire qu'en temps de calcul.

La méthode prismatique généralisée proposée permet de réduire le temps de calcul nécessaire pour résoudre l'équation de transport d'un facteur 10, de même que de réduire les besoins en mémoire de 4 ordres de grandeur, tout en conservant la même précision.

De plus, le couplage de cette méthode avec une parallélisation nouvelle et novatrice sur les lignes d'extrusion permet d'obtenir un gain de *speedup* supplémentaire pouvant atteindre 2,5 en utilisant 4 *threads*, ce qui convient à la plupart des architectures grand public de nos jours.

7.1 Avenues futures

Dans le cadre du développement de ce projet, de nombreuses avenues ont été examinées dont certaines ont été rejetées et d'autres ont simplement été laissées de côté par manque de temps, ou parce qu'elles n'amenaient pas directement au résultat escompté, tout en présentant des avantages intéressants. Nous soulignons ici certaines de ces avenues parallèles.

Dans un premier temps, il importe de traduire la version développée du code dans un standard moderne, c'est-à-dire convertir les fichiers source du FORTRAN 77 à Fortran 2003, qui supporte notamment intrinsèquement l'allocation dynamique de la mémoire sans devoir recourir à des `malloc` externes. En particulier, le passage de DRAGON4 à DRAGON5 permettra par surcroît de bénéficier des avancées effectuées dans cette nouvelle version du code.

Puis, nous devons de mentionner l'aspect du calcul parallèle. Dans l'optique d'un calcul de coeur complet en transport déterministe hétérogène, il s'agit de l'avenue la plus évidente permettant d'augmenter la taille du problème simulé sans avoir recours à des approximations supplémentaires. L'aspect exploré dans le cadre de cette thèse – la parallélisation sur les lignes d'extrusion – ne fait qu'effleurer la surface de ce qui est possible dans ce contexte. Différents niveaux de parallélisme imbriqués pourraient assurément tirer profit des architectures des supercalculateurs modernes, et d'autant plus réduire le temps nécessaire au calcul de problèmes de très grandes taille.

Dans cette optique, l'implantation d'un modèle de performances afin de monitorer et d'arriver à projeter les performances du code et de l'algorithme devra assurément être effectué. En particulier, si la taille des problèmes qu'on désire simuler continue à croître, la mise en place d'un tel modèle sera plus qu'utile pour établir les besoins et concentrer les ressources aux étapes de calcul présentant les plus grands besoins.

La superposition de sections cylindriques non-concentriques est une avenue intéressante dans la mesure où un niveau de détail et de précision supplémentaire est requis dans les simulations – et à un certain point, cette précision supplémentaire viendra nécessairement d'une représentation numérique plus fidèle au système physique sous-jacent.

Finalement, il sera intéressant de conjuguer les différentes évolutions des dernières années dans les méthodes numériques – tant dans le développement des méthodes d'accélération que des méthodes de traçage avancées – mais surtout, comment elles pourront être intégrées et jumelées aux avancements qui ont été présentés ici.

RÉFÉRENCES

- AMDAHL, G. M. (1967). Validity of the single processor approach to achieving large scale computing capabilities. *In Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*, AFIPS '67 (Spring), pages 483–485, New York, NY, USA. ACM.
- ASKEW, J. R. (1972). A characteristics formulation of the neutron transport equation in complicated geometries. Technical Report AEEW-M 1108, Atomic Energy Establishment, Winfrith, United Kingdom, Winfrith.
- BOYD, W., SHANER, S., LI, L., FORGET, B. et SMITH, K. (2014). The openmoc method of characteristics neutral particle transport code. *Annals of Nuclear Energy*, 68:43–52.
- BOYD III, W. R. D. (2014). *Massively parallel algorithms for method of characteristics neutral particle transport on shared memory computer architectures*. Thèse de doctorat, Massachusetts Institute of Technology.
- DAGUM, L. et MENON, R. (1998). OpenMP : an industry standard API for shared-memory programming. *Computational Science & Engineering, IEEE*, 5(1):46–55.
- DAHMANI, M. (2006). *Modélisation du calcul distribué pour des problèmes de transport de neutrons à grande échelle*. Thèse de doctorat, École Polytechnique de Montréal.
- DAHMANI, M., LE TELLIER, R. et MARLEAU, G. (2008). Modeling reactivity devices for Advanced CANDU Reactors using the code DRAGON. *Ann. Nucl. Energy*, 35:804–812.
- DAHMANI, M., LE TELLIER, R., ROY, R. et HÉBERT, A. (2005). An efficient preconditioning technique using Krylov subspace methods for 3D characteristics solvers. *Annals of Nuclear Energy*, 32(8):876 – 896.
- DAHMANI, M. et ROY, R. (2005). Parallel solver based on three-dimensional characteristics method : Design and performance analysis. *Nucl. Sci. Eng.*, 150:155–169.
- DAHMANI, M. et ROY, R. (2006). Scalability modeling for deterministic particle transport solvers. *International Journal of High Performance Computing and Applications*, pages 541–556.
- DAHMANI, M., ROY, R. et KOCLAS, J. (2004). New computational methodology for large 3D neutron transport problems. *In PHYSOR-2004 International Meeting on the Physics of Fuel Cycles and Advanced Nuclear Systems*, Chicago, IL. (Proceedings available on CD-Rom).
- FÉVOTTE, F. et LATHUILLÈRE, B. (2013). Micado : Parallel implementation of a 2D-1D iterative algorithm for the 3D neutron transport problem in prismatic geometries. Rapport

technique, American Nuclear Society, 555 North Kensington Avenue, La Grange Park, IL 60526 (United States).

G. GUNOW, S. Shaner, B. F. et SMITH, K. (2016). “reducing 3d moc storage requirements with axial on-the-fly ray tracing”. In *PHYSOR, Sun Valley, Idaho*.

GABRIEL, E., FAGG, G. E., BOSILCA, G., ANGSKUN, T., DONGARRA, J. J., SQUYRES, J. M., SAHAY, V., KAMBADUR, P., BARRETT, B., LUMSDAINE, A., CASTAIN, R. H., DANIEL, D. J., GRAHAM, R. L. et WOODALL, T. S. (2004). Open MPI : Goals, concept, and design of a next generation MPI implementation. In *Proceedings, 11th European PVM/MPI Users’ Group Meeting*, pages 97–104, Budapest, Hungary.

GEIST, A., BEGUELIN, A., DONGARRA, J., JIANG, W., MANCHEK, R. et SUNDERAM, V. (1994). *PVM : Parallel Virtual Machine : A Users’ Guide and Tutorial for Networked Parallel Computing*. MIT Press, Cambridge, MA, USA.

GLASSTONE, S. et SESONSKE, A. (1994). *Nuclear Reactor Engineering*. Chapman & Hall, New York, 4^e édition.

GROPP, W., LUSK, E. L., SKJELLUM, A. et THAKUR, R. (1999). *Using MPI : Portable Parallel Programming with the Message-Passing Interface (Scientific and Engineering Computation) (Volume 1)*. The MIT Press, seconde édition.

GUSTAFSON, J. L. (1988). Reevaluating Amdahl’s law. *Communications of the ACM*, 31:532–533.

HALSALL, M. J. (1980). Cactus : A characteristics solution to the neutron transport equation in complicated geometries. Rapport technique AEEW-R 1291, United Kingdom Atomic Energy Establishment.

HÉBERT, A. (2009). *Applied Reactor Physics*. Presses Internationales Polytechnique, Montréal.

HÉBERT, A. et MARLEAU, G. (1991). Generalization of the Stamm’ler method for the self-shielding of resonant isotopes in arbitrary geometries. *Nucl. Sci. Eng.*, 108:230–239.

HÉBERT, A. et MATHONNIÈRE, A. (1993). Development of a third-generation *Superhomogénéisation* method for the homogenization of a pressurized water reactor assembly. *Nucl. Sci. Eng.*, 115:129–141.

HÉBERT, A. (2011). High-order diamond differencing along finite characteristics. *Nuclear Science and Engineering*, 169(1):81–97.

HÉBERT, A. (2016). High-order linear discontinuous and diamond differencing schemes along cyclic characteristics. *Nuclear Science and Engineering*, 184(4):591–603.

- LAJOIE, M.-A., FÉVOTTE, F. et MARLEAU, G. (2014). A generalization of 3d prismatic characteristics along a nonuniform projection mesh. *In SNA+ MC 2013-Joint International Conference on Supercomputing in Nuclear Applications+ Monte Carlo*, page 02203. EDP Sciences.
- LE TELLIER, R. (2006). *Développement de la méthode des caractéristiques pour le calcul de réseau*. Thèse de doctorat, École Polytechnique de Montréal, Montréal.
- LE TELLIER, R. et HÉBERT, A. (2007). An improved algebraic collapsing acceleration with general boundary conditions for the characteristics method. *Nuclear science and engineering*, 156(2):121–138.
- LE TELLIER, R., HÉBERT, A. et MARLEAU, G. (2006). The implementation of a 3D characteristics solver for the generation of incremental cross sections for reactivity devices in a CANDU reactor. *In PHYSOR-2006, American Nuclear Society's Topical Meeting on Reactor Physics*, Vancouver, Canada. (Proceedings available on CD-Rom).
- LE TELLIER, R. et HÉBERT, A. (2006). On the integration scheme along a trajectory for the characteristics method. *Annals of Nuclear Energy*, 33(14–15):1260–1269.
- LE TELLIER, R., MARLEAU, G., DAHMANI, M. et HÉBERT, A. (2008). Improvements of the reactivity devices modeling for the Advanced CANDU Reactor. *Ann. Nucl. Energy*, 35:868–876.
- LEVESQUE, J. et WAGENBRETH, G. (2010). *High Performance Computing : Programming and Applications*. Chapman & Hall/CRC Computational Science. CRC Press.
- LEWIS, E. E. et MILLER JR., W. F. (1993). *Computational Methods of Neutron Transport, 2nd edition*. John Wiley & Sons, Inc., New York.
- MARLEAU, G. (2001). Dragon theory manual part 1 : Collision probability calculations. Rapport technique IGE-236 Rev. 1, École Polytechnique de Montréal.
- MARLEAU, G., HÉBERT, A. et ROY, R. (2006). A user guide for DRAGON Version4. Technical Report IGE-294, Institut de Génie Nucléaire, École Polytechnique de Montréal, Montréal. Available freely at <http://www.polymtl.ca/merlin>.
- PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A. et VETTERLING, W. T. (1994). *Numerical Recipes, Second Edition (FORTRAN Version)*. Cambridge University Press, Cambridge, MA.
- QADDOURI, A., ROY, R., MAYRAND, M. et GOULARD, B. (1996). Collision probability calculation and multigroup flux solvers using PVM. *Nuclear science and engineering*, 123(3): 392–402.
- ROY, R. (1998). The cyclic characteristics method. *In International Topical Meeting on the Physics of Nuclear Science and Technology*, pages 407–414, Long Island, NY.

- ROY, R. et HÉBERT, A. (2000). The GAN Generalized Driver. Rapport technique IGE-158, École Polytechnique de Montréal.
- SAAD, Y. (2003). *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd édition.
- SANCHEZ, R. et CHETAINE, A. (2000). A synthetic acceleration for a two-dimensional characteristic method in unstructured meshes. *Nuclear Science and Engineering*, 136(1): 122–139.
- SANCHEZ, R., MAO, L. et SANTANDREA, S. (2002). Treatment of Boundary Conditions in Trajectory-Based Deterministic Transport Methods. *Nucl. Sci. Eng.*, 140:23–50.
- SANTANDREA, S. et SANCHEZ, R. (2002a). Acceleration techniques for the characteristics method on unstructured meshes. *Ann. Nucl. Energy*, 29:323–352.
- SANTANDREA, S. et SANCHEZ, R. (2002b). Positive linear and nonlinear surface characteristic schemes for the neutron transport equation in unstructured geometries. *In Proceedings of Physor-2002*, Seoul, Korea.
- SCIANNANDRONE, D. (2015). *Acceleration and higher order schemes of a characteristic solver for the solution of the neutron transport equation in 3D axial geometries*. Thèse de doctorat, Paris 11. Thèse de doctorat, Université Paris-XI, Paris.
- SIEGEL, S. F., SIEGEL, A. R. et RABITI, C. (2008). Unic code : Algorithmic specification of the method of long characteristics. Rapport technique ANL/MCS-TM-301, Mathematics and Computer Science Division, Argonne National Laboratory.
- SUSLOV, I. R. (1993). Solution of transport equation in 2–and 3–dimensional irregular geometry by the method of characteristics. *In Joint Int. Conf. on Mathematical Methods and Supercomputing in Nuclear Applications, Karlsruhe, Germany*.
- WU, G. et ROY, R. (1999). Self-collision rebalancing techniques for the MCI characteristics solver. *In Proceedings of the 20th Annual Canadian Nuclear Society Conference*, Toronto.
- WU, G. et ROY, R. (2001). Parallelization of characteristics solvers for 3D neutron transport. *In Recent Advances in Parallel Virtual Machine and Message Passing Interface*, volume 2131 de *Lecture Notes in Computer Science*, pages 344–351. Springer Berlin / Heidelberg.
- WU, G. J. et ROY, R. (2003). Acceleration techniques for trajectory-based deterministic 3D transport solvers. *Ann. Nucl. Energy*, 30:567–583.
- YOUNG, D. (1954). Iterative methods for solving partial difference equations of elliptic type. *Transactions of the American Mathematical Society*, 76(1):92–111.

ANNEXE I - FICHIERS D'ENTRÉE

Équivalence transport-transport et superhomogénéisation

```
*DECK Equivalence2DPins
```

```
*-----
```

```
* Nom          : Equivalence2DPins.x2m
*              de Assemblage2D.x2m
* Usage        : Generation des bibliotheques
*              multigroupes pour flambage
*              avec equivalence SPH
*              Trois types de cellules
*              1) coin
*              2) cote
*              3) centre
* Auteur       : M-A Lajoie   (2013-03-23)
*              : G. Marleau   (2013-07-08)
```

```
*-----
```

```
* Modules
```

```
MODULE      GEO: LIB: NXT: PSP: SHI: ASM: FLU: EDI:
            UTL: DELETE: END: SPH: ;
```

```
* Data structures
```

```
LINKED_LIST CoinP CoteP CentreP CoinC CoteC CentreC
            TrackingP TrackingC Pij Flux Biblio
            Edit172 Edit2 Macrolib Temp ;
SEQ_BINARY  IntlinesP IntlinesC ;
SEQ_ASCII   CoinP.ps CoteP.ps CentreP.ps
            CoinC.ps CoteC.ps CentreC.ps
            MacCoin172 MacCoin2
            MacCote172 MacCote2
            MacCentre172 MacCentre2 ;
```

```
* Variables
```

```
REAL RESEAU ESPACE RAYONP RAYONC HEIGHT PEAU :=
    1.26  0.042  0.419  0.476  50.  0.1 ;
```

```

REAL  SIDECEL SIDECELD DESPACE COTECLAD COTEPIN
      CLAD_G   CLAD_D   PIN_G   PIN_D   PEAU_G   PEAU_D
      CLAD_LG  CLAD_LD  PIN_LG  PIN_LD  PEAU_LG  PEAU_LD
      CLADD_G  CLADD_D  PIND_G  PIND_D  PEAUD_G  PEAUD_D
      CLADD_LG CLADD_LD PIND_LG PIND_LD PEAUD_LG PEAUD_LD ;
REAL  RPEAU POSPACE MOSPACE DPOSPACE DMOSPACE ;

```

```
*-----
```

```
* Evaluate meaningful quantities
```

```
*-----
```

```

EVALUATE RPEAU      := RAYONP PEAU - ;
EVALUATE POSPACE   := ESPACE 0.5 * ;
EVALUATE MOSPACE   := POSPACE -1.0 * ;
EVALUATE SIDECEL   := RESEAU ESPACE + ;
EVALUATE SIDECELD := SIDECEL ESPACE + ;
EVALUATE DESPACE   := ESPACE 2.E0 * ;
EVALUATE DPOSPACE  := POSPACE 2.0 * ;
EVALUATE DMOSPACE  := MOSPACE 2.0 * ;
EVALUATE COTECLAD := $Pi_R SQRT RAYONC * ;
EVALUATE COTEPIN  := $Pi_R SQRT RAYONP * ;
EVALUATE CLAD_G    := RESEAU COTECLAD - 0.5 * ;
EVALUATE CLAD_D    := RESEAU COTECLAD + 0.5 * ;
EVALUATE PIN_G     := RESEAU COTEPIN - 0.5 * ;
EVALUATE PIN_D     := RESEAU COTEPIN + 0.5 * ;
EVALUATE PEAU_G    := PIN_G PEAU + ;
EVALUATE PEAU_D    := PIN_D PEAU - ;
EVALUATE CLAD_LG   := CLAD_G ESPACE + ;
EVALUATE CLAD_LD   := CLAD_D ESPACE + ;
EVALUATE PIN_LG    := PIN_G ESPACE + ;
EVALUATE PIN_LD    := PIN_D ESPACE + ;
EVALUATE PEAU_LG   := PIN_LG PEAU + ;
EVALUATE PEAU_LD   := PIN_LD PEAU - ;
EVALUATE CLADD_G   := ESPACE CLAD_G + ;
EVALUATE CLADD_D   := ESPACE CLAD_D + ;
EVALUATE PIND_G    := ESPACE PIN_G + ;
EVALUATE PIND_D    := ESPACE PIN_D + ;
EVALUATE PEAUD_G   := PIND_G PEAU + ;

```

```
EVALUATE PEAUD_D := PIND_D PEAU - ;
EVALUATE CLADD_LG := ESPACE CLAD_LG + ;
EVALUATE CLADD_LD := ESPACE CLAD_LD + ;
EVALUATE PIND_LG := ESPACE PIN_LG + ;
EVALUATE PIND_LD := ESPACE PIN_LD + ;
EVALUATE PEAUD_LG := PIND_LG PEAU + ;
EVALUATE PEAUD_LD := PIND_LD PEAU - ;
```

```
ECHO "SIDECEL " SIDECEL ;
ECHO "SIDECELD" SIDECELD ;
ECHO "DESPACE " DESPACE ;
ECHO "COTECLAD" COTECLAD ;
ECHO "COTEPIN " COTEPIN ;
ECHO "CLAD_G " CLAD_G ;
ECHO "CLAD_D " CLAD_D ;
ECHO "PIN_G " PIN_G ;
ECHO "PIN_D " PIN_D ;
ECHO "PEAU_G " PEAU_G ;
ECHO "PEAU_D " PEAU_D ;
ECHO "CLAD_LG " CLAD_LG ;
ECHO "CLAD_LD " CLAD_LD ;
ECHO "PIN_LG " PIN_LG ;
ECHO "PIN_LD " PIN_LD ;
ECHO "PEAU_LG " PEAU_LG ;
ECHO "PEAU_LD " PEAU_LD ;
ECHO "CLADD_G " CLADD_G ;
ECHO "CLADD_D " CLADD_D ;
ECHO "PIND_G " PIND_G ;
ECHO "PIND_D " PIND_D ;
ECHO "PEAUD_G " PEAUD_G ;
ECHO "PEAUD_G " PEAUD_G ;
ECHO "CLADD_LG" CLADD_LG ;
ECHO "CLADD_LD" CLADD_LD ;
ECHO "PIND_LG " PIND_LG ;
ECHO "PIND_LD " PIND_LD ;
ECHO "PEAUD_LG" PEAUD_LG ;
```



```

;
CoinC := GEO: :: CAR2D 8 8
  EDIT 100 !<<EDILEV>>
  X- REFL X+ REFL
  Y- REFL Y+ REFL
  MESHX 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
           <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
  SPLITX 1 1 1 1 2 1 1 1
  MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
           <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>
  SPLITY 1 1 1 2 1 1 1 1
  MIX 1 2 2 2 2 2 2 2 ! MIX 1 = LAME D'EAU
      1 2 3 3 3 3 3 2 ! MIX 2 = MODERATEUR
      1 2 3 4 4 4 3 2 ! MIX 3 = CLADDING
      1 2 3 4 4 4 3 2 ! MIX 4 = FUEL
      1 2 3 4 4 4 3 2
      1 2 3 3 3 3 3 2
      1 2 2 2 2 2 2 2
      1 1 1 1 1 1 1 1

```

```

;
CoteC := GEO: :: CAR2D 7 8
  EDIT 100 !<<EDILEV>>
  X- REFL X+ REFL
  Y- REFL Y+ REFL
  MESHX 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
           <<PIN_D>> <<CLAD_D>> <<RESEAU>>
  SPLITX 1 1 1 2 1 1 1
  MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
           <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>
  SPLITY 1 1 1 2 1 1 1 1
  MIX 2 2 2 2 2 2 2
      2 3 3 3 3 3 2
      2 3 4 4 4 3 2
      2 3 4 4 4 3 2
      2 3 4 4 4 3 2
      2 3 3 3 3 3 2

```

```

      2 2 2 2 2 2 2
      1 1 1 1 1 1 1
;
CentreC := GEO: :: CAR2D 7 7
  EDIT 100 !<<EDILEV>>
  X- REFL X+ REFL
  Y- REFL Y+ REFL
  MESHX 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
           <<PIN_D>> <<CLAD_D>> <<RESEAU>>
  SPLITX 1 1 1 2 1 1 1
  MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
           <<PIN_D>> <<CLAD_D>> <<RESEAU>>
  SPLITY 1 1 1 2 1 1 1
  MIX 2 2 2 2 2 2 2
      2 3 3 3 3 3 2
      2 3 4 4 4 3 2
      2 3 4 4 4 3 2
      2 3 4 4 4 3 2
      2 3 3 3 3 3 2
      2 2 2 2 2 2 2
;
Biblio := LIB: ::
  NMIX 4 CTRA WIMS
  MIXS LIB: WIMSD4 FIL: endfb6gx
  MIX 1 600.0
      H1      = '3001'  4.42326E-2
      O16     = '6016'  2.21162E-2
      B10     = '1010'  0.51066E-5
      B11     = '11'    2.55549E-5
  MIX 2 600.0
      H1      = '3001'  4.42326E-2
      O16     = '6016'  2.21162E-2
      B10     = '1010'  0.51066E-5
      B11     = '11'    2.55549E-5
  MIX 3 600.0
      Zr      = '91'    4.21838E-2

```

```

MIX 4 600.0
      016 = '6016' 4.60093E-2
      U234 = '234' 7.31651E-6 1
      U235 = '2235' 9.10661E-4 1
      U238 = '8238' 2.21490E-2 1
;
*
* Generate Editing with SPH, 2 and 172 groups
* Cellule de coin
*
TrackingP IntlinesP := NXT: CoinP :: EDIT 2 TISO 10 20.0 ;
TrackingC IntlinesC := NXT: CoinC :: EDIT 2 TISO 10 20.0 ;
CoinP.ps := PSP: TrackingP :: TYPE MIXTURE ;
CoinC.ps := PSP: TrackingC :: TYPE MIXTURE ;
Biblio := SHI: Biblio TrackingP IntlinesP ;
Pij := ASM: Biblio TrackingP IntlinesP ;
Flux := FLU: Pij Biblio TrackingP ::
      TYPE K ;
Edit172 := EDI: Flux Biblio TrackingP TrackingC :: !IntlinesC ::
      MERG MIX 1 2 3 4
      SAVE ON CoinP ; !SPH MTRK ;
Edit2 := EDI: Flux Biblio TrackingP TrackingC :: !IntlinesC ::
      COND 0.625 MERG MIX 1 2 3 4
      SAVE ON CoinP ; !SPH MTRK ;
Pij Flux := DELETE: Pij Flux ;
Temp := SPH: Edit172 TrackingC IntlinesC ::
      STEP UP CoinP
      MACRO ON STD ;
MacCoin172 := Temp ;
Temp := DELETE: Temp ;
Temp := SPH: Edit2 TrackingC IntlinesC ::
      STEP UP CoinP
      MACRO ON STD ;
MacCoin2 := Temp ;
Edit2 Edit172 Temp := DELETE: Edit2 Edit172 Temp ;
*

```

```

Macrolib := MacCoin172 ;
Pij := ASM: Macrolib TrackingC IntlinesC ;
Flux := FLU: Pij Macrolib TrackingC ::
    TYPE K ;
Pij Flux Macrolib := DELETE: Pij Flux Macrolib ;
Macrolib := MacCoin2 ;
Pij := ASM: Macrolib TrackingC IntlinesC ;
Flux := FLU: Pij Macrolib TrackingC ::
    TYPE K ;
Pij Flux Macrolib := DELETE: Pij Flux Macrolib ;
TrackingP IntlinesP := DELETE: TrackingP IntlinesP ;
TrackingC IntlinesC := DELETE: TrackingC IntlinesC ;
*
* Generate Editing with SPH, 2 and 172 groups
* Cellule de cote
*
TrackingP IntlinesP := NXT: CoteP :: EDIT 2 TISO 10 20.0 ;
TrackingC IntlinesC := NXT: CoteC :: EDIT 2 TISO 10 20.0 ;
CoteP.ps := PSP: TrackingP :: TYPE MIXTURE ;
CoteC.ps := PSP: TrackingC :: TYPE MIXTURE ;
Biblio := SHI: Biblio TrackingP IntlinesP ;
Pij := ASM: Biblio TrackingP IntlinesP ;
Flux := FLU: Pij Biblio TrackingP ::
    TYPE K ;
Edit172 := EDI: Flux Biblio TrackingP TrackingC :: !IntlinesC ::
    MERG MIX 1 2 3 4
    SAVE ON CoinP ; !SPH MTRK ;
Edit2 := EDI: Flux Biblio TrackingP TrackingC :: !IntlinesC ::
    COND 0.625 MERG MIX 1 2 3 4
    SAVE ON CoinP ; !SPH MTRK ;
Pij Flux := DELETE: Pij Flux ;
Temp := SPH: Edit172 TrackingC IntlinesC ::
    STEP UP CoinP
    MACRO ON STD ;
MacCote172 := Temp ;
Temp := DELETE: Temp ;

```

```

Temp := SPH: Edit2 TrackingC IntlinesC ::
    STEP UP CoinP
    MACRO ON STD ;
MacCote2 := Temp ;
Edit2 Edit172 Temp := DELETE: Edit2 Edit172 Temp ;
!Edit172 := UTL: Edit172 :: STEP UP CoinP STEP UP MACROLIB ;
!MacCote172 := Edit172 ;
!!Edit172 := UTL: Edit172 :: STEP DOWN ; !STEP DOWN ;
!Edit2 := UTL: Edit2 :: STEP UP CoinP STEP UP MACROLIB ;
!MacCote2 := Edit2 ;
!!Edit2 := UTL: Edit2 :: STEP DOWN ; !STEP DOWN ;
!Edit172 Edit2 := DELETE: Edit172 Edit2 ;
*
Macrolib := MacCote172 ;
Pij := ASM: Macrolib TrackingC IntlinesC ;
Flux := FLU: Pij Macrolib TrackingC ::
    TYPE K ;
Pij Flux Macrolib := DELETE: Pij Flux Macrolib ;
Macrolib := MacCote2 ;
Pij := ASM: Macrolib TrackingC IntlinesC ;
Flux := FLU: Pij Macrolib TrackingC ::
    TYPE K ;
Pij Flux Macrolib := DELETE: Pij Flux Macrolib ;
TrackingP IntlinesP := DELETE: TrackingP IntlinesP ;
TrackingC IntlinesC := DELETE: TrackingC IntlinesC ;
*
* Generate Editing with SPH, 2 and 172 groups
* Cellule du centre
*
TrackingP IntlinesP := NXT: CentreP :: EDIT 2 TISO 10 20.0 ;
TrackingC IntlinesC := NXT: CentreC :: EDIT 2 TISO 10 20.0 ;
CentreP.ps := PSP: TrackingP :: TYPE MIXTURE ;
CentreC.ps := PSP: TrackingC :: TYPE MIXTURE ;
Biblio := SHI: Biblio TrackingP IntlinesP ;
Pij := ASM: Biblio TrackingP IntlinesP ;
Flux := FLU: Pij Biblio TrackingP ::

```



```

        TYPE K ;
Edit172 := EDI: Flux Biblio TrackingP TrackingC :: !IntlinesC ::
    MERG MIX 1 2 3 4
    SAVE ON CoinP ; !SPH MTRK ;
Edit2 := EDI: Flux Biblio TrackingP TrackingC :: !IntlinesC ::
    COND 0.625 MERG MIX 1 2 3 4
    SAVE ON CoinP ; !SPH MTRK ;
Pij Flux Biblio := DELETE: Pij Flux Biblio ;
Temp := SPH: Edit172 TrackingC IntlinesC ::
    STEP UP CoinP
    MACRO ON STD ;
MacCentre172 := Temp ;
Temp := DELETE: Temp ;
Temp := SPH: Edit2 TrackingC IntlinesC ::
    STEP UP CoinP
    MACRO ON STD ;
MacCentre2 := Temp ;
Edit2 Edit172 Temp := DELETE: Edit2 Edit172 Temp ;
!Edit172 := UTL: Edit172 :: STEP UP CoinP STEP UP MACROLIB ;
!MacCentre172 := Edit172 ;
!!Edit172 := UTL: Edit172 :: STEP DOWN ; !STEP DOWN ;
!Edit2 := UTL: Edit2 :: STEP UP CoinP STEP UP MACROLIB ;
!MacCentre2 := Edit2 ;
!!Edit2 := UTL: Edit2 :: STEP DOWN ; !STEP DOWN ;
!Edit172 Edit2 := DELETE: Edit172 Edit2 ;
*
Macrolib := MacCentre172 ;
Pij := ASM: Macrolib TrackingC IntlinesC ;
Flux := FLU: Pij Macrolib TrackingC ::
    TYPE K ;
Pij Flux Macrolib := DELETE: Pij Flux Macrolib ;
Macrolib := MacCentre2 ;
Pij := ASM: Macrolib TrackingC IntlinesC ;
Flux := FLU: Pij Macrolib TrackingC ::
    TYPE K ;
Pij Flux Macrolib := DELETE: Pij Flux Macrolib ;

```

```
TrackingP IntlinesP := DELETE: TrackingP IntlinesP ;  
TrackingC IntlinesC := DELETE: TrackingC IntlinesC ;
```

```
END: ;
```

Assemblage 3-D Cartésien prismatique flambé

*DECK Flambage

*-----

```
* Nom          : Flambage.x2m
* Type         : Fichier de donnees DRAGON
* Usage        : Tests des geometries prismatiques sur
                 un flambage 4x4
* Auteur       : M-A Lajoie
* Date         : 2013-03-23
```

*-----

* Modules

MODULE GEO: NXT: END: PSP: ASM: DELETE: FLU: MCCGT: EDI: ;

* Data structures

LINKED_LIST TRACKING GEOM ASSBLY FLU Macrolib EDINAM ;

SEQ_BINARY TRKBIN ;

SEQ_ASCII TRKASCII MacComp12 PSPFIL FLUX ;

* Variables

INTEGER

EDILEV := 0 ;

REAL

RESEAU ESPACE RAYONP RAYONC HEIGHT HPEAU PEAU :=

1.26 0.042 0.419 0.476 10. 1.0 0.1 ;

REAL

SIDECEL SIDECELD DESPACE COTECLAD COTEPIN

CLAD_G CLAD_D PIN_G PIN_D PEAU_G PEAU_D

CLAD_LG CLAD_LD PIN_LG PIN_LD PEAU_LG PEAU_LD

CLADD_G CLADD_D PIND_G PIND_D PEAUD_G PEAUD_D

CLADD_LG CLADD_LD PIND_LG PIND_LD PEAUD_LG PEAUD_LD ;

REAL PSZ1 PSZ2 PSZ3 PDZ1 PDZ2 PDZ3 PDZ4 :=

0.0

HEIGHT HPEAU -

HEIGHT

0.0

HPEAU

HEIGHT HPEAU -

```

      HEIGHT ;
INTEGER iszpeau := 1 ;
*-----
* Evaluate meaningful quantities
*-----
EVALUATE SIDECEL := RESEAU ESPACE + ;
EVALUATE SIDECELD := RESEAU ESPACE 2.E0 * + ;
EVALUATE DESPACE := ESPACE 2.E0 * ;
EVALUATE COTECLAD := $Pi_R SQRT RAYONC * ;
EVALUATE COTEPIN := $Pi_R SQRT RAYONP * ;
EVALUATE CLAD_G := RESEAU COTECLAD - 0.5 * ;
EVALUATE CLAD_D := RESEAU COTECLAD + 0.5 * ;
EVALUATE PIN_G := RESEAU COTEPIN - 0.5 * ;
EVALUATE PIN_D := RESEAU COTEPIN + 0.5 * ;
EVALUATE PEAU_G := PIN_G PEAU + ;
EVALUATE PEAU_D := PIN_D PEAU - ;
EVALUATE CLAD_LG := CLAD_G ESPACE + ;
EVALUATE CLAD_LD := CLAD_D ESPACE + ;
EVALUATE PIN_LG := PIN_G ESPACE + ;
EVALUATE PIN_LD := PIN_D ESPACE + ;
EVALUATE PEAU_LG := PIN_LG PEAU + ;
EVALUATE PEAU_LD := PIN_LD PEAU - ;
EVALUATE CLADD_G := ESPACE CLAD_G + ;
EVALUATE CLADD_D := ESPACE CLAD_D + ;
EVALUATE PIND_G := ESPACE PIN_G + ;
EVALUATE PIND_D := ESPACE PIN_D + ;
EVALUATE PEAUD_G := PIND_G PEAU + ;
EVALUATE PEAUD_D := PIND_D PEAU - ;
EVALUATE CLADD_LG := ESPACE CLAD_LG + ;
EVALUATE CLADD_LD := ESPACE CLAD_LD + ;
EVALUATE PIND_LG := ESPACE PIN_LG + ;
EVALUATE PIND_LD := ESPACE PIN_LD + ;
EVALUATE PEAUD_LG := PIND_LG PEAU + ;
EVALUATE PEAUD_LD := PIND_LD PEAU - ;

```

```

ECHO "SIDECEL " SIDECEL ;

```

```

ECHO "SIDECELD" SIDECELD ;
ECHO "DESPACE " DESPACE ;
ECHO "COTECLAD" COTECLAD ;
ECHO "COTEPIN " COTEPIN ;
ECHO "CLAD_G " CLAD_G ;
ECHO "CLAD_D " CLAD_D ;
ECHO "PIN_G " PIN_G ;
ECHO "PIN_D " PIN_D ;
ECHO "PEAU_G " PEAU_G ;
ECHO "PEAU_D " PEAU_D ;
ECHO "CLAD_LG " CLAD_LG ;
ECHO "CLAD_LD " CLAD_LD ;
ECHO "PIN_LG " PIN_LG ;
ECHO "PIN_LD " PIN_LD ;
ECHO "PEAU_LG " PEAU_LG ;
ECHO "PEAU_LD " PEAU_LD ;
ECHO "CLADD_G " CLADD_G ;
ECHO "CLADD_D " CLADD_D ;
ECHO "PIND_G " PIND_G ;
ECHO "PIND_D " PIND_D ;
ECHO "PEAUD_G " PEAUD_G ;
ECHO "PEAUD_G " PEAUD_G ;
ECHO "CLADD_LG" CLADD_LG ;
ECHO "CLADD_LD" CLADD_LD ;
ECHO "PIND_LG " PIND_LG ;
ECHO "PIND_LD " PIND_LD ;
ECHO "PEAUD_LG" PEAUD_LG ;
ECHO "PEAUD_LG" PEAUD_LG ;

```

*-----

* Description of the Geometry

*-----

```

GEOM := GEO: :: CAR3D 3 3 3
  EDIT <<EDILEV>>
  X- REFL X+ REFL
  Y- REFL Y+ REFL

```

Z- REFL Z+ REFL

CELL CEL-LGB CEL-LB CEL-LDB
 CEL-LG CEL CEL-LD
 CEL-LGH CEL-LH CEL-LDH

CELF-LGB CELF-LB CELF-LDB
 CELF-LG CELF CELF-LD
 CELF-LGH CELF-LH CELF-LDH

CELH-LGB CELH-LB CELH-LDB
 CELH-LG CELH CELH-LD
 CELH-LGH CELH-LH CELH-LDH

::: CEL-LGH := GEO: CAR3D 8 8 2

MESHX 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
 <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>

SPLITX 1 1 1 1 2 1 1 1

MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
 <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>

SPLITY 1 1 1 2 1 1 1 1

MESHZ <<PSZ1>> <<PSZ2>> <<PSZ3>>

SPLITZ 1 <<iszpeau>>

MIX 1 2 2 2 2 2 2 2 ! MIX 1 = LAME D'EAU

1 2 3 3 3 3 3 2 ! MIX 2 = MODERATEUR

1 2 3 4 4 4 3 2 ! MIX 3 = CLADDING

1 2 3 4 4 4 3 2 ! MIX 4 = FUEL

1 2 3 4 4 4 3 2

1 2 3 3 3 3 3 2

1 2 2 2 2 2 2 2

1 1 1 1 1 1 1 1

1 2 2 2 2 2 2 2

1 2 3 3 3 3 3 2

1 2 3 4 4 4 3 2

1 2 3 4 4 4 3 2

1 2 3 4 4 4 3 2

```

1 2 3 3 3 3 3 2
1 2 2 2 2 2 2 2
1 1 1 1 1 1 1 1
;
::: CEL-LH := GEO: CAR3D 7 8 2
MESHX 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
        <<PIN_D>> <<CLAD_D>> <<RESEAU>>
SPLITX 1 1 1 2 1 1 1
MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
        <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>
SPLITY 1 1 1 2 1 1 1 1
MESHZ <<PSZ1>> <<PSZ2>> <<PSZ3>>
SPLITZ      1      <<iszpeau>>
MIX 6 6 6 6 6 6 6
      6 7 7 7 7 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 7 7 7 7 6
      6 6 6 6 6 6 6
      5 5 5 5 5 5 5

      6 6 6 6 6 6 6
      6 7 7 7 7 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 7 7 7 7 6
      6 6 6 6 6 6 6
      5 5 5 5 5 5 5
;
::: CEL-LDH := GEO: CEL-LGH
MESHX 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
        <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>
SPLITX 1 1 1 2 1 1 1 1
MIX 2 2 2 2 2 2 2 1

```

```

2 3 3 3 3 3 2 1
2 3 4 4 4 3 2 1
2 3 4 4 4 3 2 1
2 3 4 4 4 3 2 1
2 3 3 3 3 3 2 1
2 2 2 2 2 2 2 1
1 1 1 1 1 1 1 1

2 2 2 2 2 2 2 1
2 3 3 3 3 3 2 1
2 3 4 4 4 3 2 1
2 3 4 4 4 3 2 1
2 3 4 4 4 3 2 1
2 3 3 3 3 3 2 1
2 2 2 2 2 2 2 1
1 1 1 1 1 1 1 1

;
::: CEL-LG := GEO: CAR3D 8 7 2
MESHX 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
      <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
SPLITX 1 1 1 1 2 1 1 1
MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
      <<PIN_D>> <<CLAD_D>> <<RESEAU>>
SPLITY 1 1 1 2 1 1 1
MESHZ <<PSZ1>> <<PSZ2>> <<PSZ3>>
SPLITZ      1      <<iszpeau>>
MIX 5 6 6 6 6 6 6 6
      5 6 7 7 7 7 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 7 7 7 7 6
      5 6 6 6 6 6 6 6

      5 6 6 6 6 6 6 6
      5 6 7 7 7 7 7 6

```



```

5 6 7 8 8 8 7 6
5 6 7 8 8 8 7 6
5 6 7 8 8 8 7 6
5 6 7 7 7 7 7 6
5 6 6 6 6 6 6 6
;
::: CEL := GEO: CAR3D 7 7 2
MESHX 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
      <<PIN_D>> <<CLAD_D>> <<RESEAU>>
SPLITX 1 1 1 2 1 1 1
MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
      <<PIN_D>> <<CLAD_D>> <<RESEAU>>
SPLITY 1 1 1 2 1 1 1
MESHZ <<PSZ1>> <<PSZ2>> <<PSZ3>>
SPLITZ      1      <<iszpeau>>
MIX 10 10 10 10 10 10 10 !2 2 2 2 2 2 2
    10 11 11 11 11 11 10 !2 3 3 3 3 3 2
    10 11 12 12 12 11 10 !2 3 4 4 4 3 2
    10 11 12 12 12 11 10 !2 3 4 4 4 3 2
    10 11 12 12 12 11 10 !2 3 4 4 4 3 2
    10 11 11 11 11 11 10 !2 3 3 3 3 3 2
    10 10 10 10 10 10 10 !2 2 2 2 2 2 2

    10 10 10 10 10 10 10
    10 11 11 11 11 11 10
    10 11 12 12 12 11 10
    10 11 12 12 12 11 10
    10 11 12 12 12 11 10
    10 11 11 11 11 11 10
    10 10 10 10 10 10 10
;
::: CEL-LD := GEO: CEL-LG
MESHX 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
      <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>
SPLITX 1 1 1 2 1 1 1 1
MIX 6 6 6 6 6 6 6 5

```

```

6 7 7 7 7 7 6 5
6 7 8 8 8 7 6 5
6 7 8 8 8 7 6 5
6 7 8 8 8 7 6 5
6 7 7 7 7 7 6 5
6 6 6 6 6 6 6 5

6 6 6 6 6 6 6 5
6 7 7 7 7 7 6 5
6 7 8 8 8 7 6 5
6 7 8 8 8 7 6 5
6 7 8 8 8 7 6 5
6 7 7 7 7 7 6 5
6 6 6 6 6 6 6 5

;
::: CEL-LGB := GEO: CEL-LGH
MESHY 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
      <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
SPLITY 1 1 1 1 2 1 1 1
MIX 1 1 1 1 1 1 1 1
    1 2 2 2 2 2 2 2
    1 2 3 3 3 3 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 3 3 3 3 2
    1 2 2 2 2 2 2 2

    1 1 1 1 1 1 1 1
    1 2 2 2 2 2 2 2
    1 2 3 3 3 3 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 3 3 3 3 2
    1 2 2 2 2 2 2 2

```

```

;
::: CEL-LB := GEO: CEL-LH
  MESHY 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
        <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
  SPLITY 1 1 1 1 2 1 1 1
  MIX 5 5 5 5 5 5 5
      6 6 6 6 6 6 6
      6 7 7 7 7 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 7 7 7 7 6
      6 6 6 6 6 6 6

      5 5 5 5 5 5 5
      6 6 6 6 6 6 6
      6 7 7 7 7 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 8 8 8 7 6
      6 7 7 7 7 7 6
      6 6 6 6 6 6 6

;
::: CEL-LDB := GEO: CEL-LDH
  MESHY 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
        <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
  SPLITY 1 1 1 1 2 1 1 1
  MIX 1 1 1 1 1 1 1 1
      2 2 2 2 2 2 2 1
      2 3 3 3 3 3 2 1
      2 3 4 4 4 3 2 1
      2 3 4 4 4 3 2 1
      2 3 4 4 4 3 2 1
      2 3 3 3 3 3 2 1
      2 2 2 2 2 2 2 1

```

```

1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 1
2 3 3 3 3 3 2 1
2 3 4 4 4 3 2 1
2 3 4 4 4 3 2 1
2 3 4 4 4 3 2 1
2 3 3 3 3 3 2 1
2 2 2 2 2 2 2 1
;
::: CELF-LGH := GEO: CAR3D 8 8 3
MESHX 0.0 <<DESPACE>> <<CLADD_LG>> <<PIND_LG>> <<PEAUD_LG>>
        <<PEAUD_LD>> <<PIND_LD>> <<CLADD_LD>> <<SIDECEL>>
SPLITX 1 1 1 1 2 1 1 1
MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
        <<PIN_D>> <<CLAD_D>> <<RESEAU>> <<SIDECEL>>
SPLITY 1 1 1 2 1 1 1 1
MESHZ <<PDZ1>> <<PDZ2>> <<PDZ3>> <<PDZ4>>
SPLITZ <<iszpeau>> 1 <<iszpeau>>
MIX 1 2 2 2 2 2 2 2
    1 2 3 3 3 3 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 3 3 3 3 2
    1 2 2 2 2 2 2 2
    1 1 1 1 1 1 1 1

    1 2 2 2 2 2 2 2
    1 2 3 3 3 3 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 3 3 3 3 2
    1 2 2 2 2 2 2 2
    1 1 1 1 1 1 1 1

```

```

1 2 2 2 2 2 2 2
1 2 3 3 3 3 3 2
1 2 3 4 4 4 3 2
1 2 3 4 4 4 3 2
1 2 3 4 4 4 3 2
1 2 3 3 3 3 3 2
1 2 2 2 2 2 2 2
1 1 1 1 1 1 1 1
;
::: CELF-LH := GEO: CELF-LGH
MESHX 0.0 <<ESPACE>> <<CLADD_G>> <<PIND_G>> <<PEAUD_G>>
      <<PEAUD_D>> <<PIND_D>> <<CLADD_D>> <<RESEAU>>
SPLITX 1 1 1 1 2 1 1 1
MIX 6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 7 7 7 7 6
    6 6 6 6 6 6 6 6
    5 5 5 5 5 5 5 5

    6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 7 7 7 7 6
    6 6 6 6 6 6 6 6
    5 5 5 5 5 5 5 5

    6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6

```

```

        6 6 7 7 7 7 7 6
        6 6 6 6 6 6 6 6
        5 5 5 5 5 5 5 5
;
::: CELF-LDH := GEO: CELF-LH
MESHX 0.0 <<ESPACE>> <<CLADD_G>> <<PIND_G>> <<PEAUD_G>>
        <<PEAUD_D>> <<PIND_D>> <<CLADD_D>> <<SIDECEL>>
SPLITX 1 1 1 1 2 1 1 1
MIX 2 2 2 2 2 2 2 2
      2 2 3 3 3 3 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 3 3 3 3 2
      2 2 2 2 2 2 2 2
      1 1 1 1 1 1 1 1

      2 2 2 2 2 2 2 2
      2 2 3 3 3 3 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 3 3 3 3 2
      2 2 2 2 2 2 2 2
      1 1 1 1 1 1 1 1

      2 2 2 2 2 2 2 2
      2 2 3 3 3 3 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 3 3 3 3 2
      2 2 2 2 2 2 2 2
      1 1 1 1 1 1 1 1
;
::: CELF-LG := GEO: CAR3D 8 7 3

```

```

MESHX 0.0 <<DESPACE>> <<CLADD_LG>> <<PIND_LG>> <<PEAUD_LG>>
        <<PEAUD_LD>> <<PIND_LD>> <<CLADD_LD>> <<SIDECEL>>
SPLITX 1 1 1 1 2 1 1 1
MESHY 0.0 <<CLAD_G>> <<PIN_G>> <<PEAU_G>> <<PEAU_D>>
        <<PIN_D>> <<CLAD_D>> <<RESEAU>>
SPLITY 1 1 1 2 1 1 1
MESHZ <<PDZ1>> <<PDZ2>> <<PDZ3>> <<PDZ4>>
SPLITZ      <<iszpeau>>      1      <<iszpeau>>
MIX 5 6 6 6 6 6 6 6
      5 6 7 7 7 7 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 7 7 7 7 6
      5 6 6 6 6 6 6 6

      5 6 6 6 6 6 6 6
      5 6 7 7 7 7 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 7 7 7 7 6
      5 6 6 6 6 6 6 6

      5 6 6 6 6 6 6 6
      5 6 7 7 7 7 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 8 8 8 7 6
      5 6 7 7 7 7 7 6
      5 6 6 6 6 6 6 6
;
::: CELF := GEO: CELF-LG
MESHX 0.0 <<ESPACE>> <<CLADD_G>> <<PIND_G>> <<PEAUD_G>>
        <<PEAUD_D>> <<PIND_D>> <<CLADD_D>> <<RESEAU>>
SPLITX 1 1 1 1 2 1 1 1

```

```

MIX 10 10 10 10 10 10 10 10
    10 10 11 11 11 11 11 10
    10 10 11 12 12 12 11 10
    10 10 11 12 12 12 11 10
    10 10 11 12 12 12 11 10
    10 10 11 11 11 11 11 10
    10 10 10 10 10 10 10 10

    10 10 10 10 10 10 10 10
    10 10 11 11 11 11 11 10
    10 10 11 12 12 12 11 10
    10 10 11 12 12 12 11 10
    10 10 11 12 12 12 11 10
    10 10 11 11 11 11 11 10
    10 10 10 10 10 10 10 10

    10 10 10 10 10 10 10 10
    10 10 11 11 11 11 11 10
    10 10 11 12 12 12 11 10
    10 10 11 12 12 12 11 10
    10 10 11 12 12 12 11 10
    10 10 11 11 11 11 11 10
    10 10 10 10 10 10 10 10
;
::: CELF-LD := GEO: CELF
MESHX 0.0 <<ESPACE>> <<CLADD_G>> <<PIND_G>> <<PEAUD_G>>
        <<PEAUD_D>> <<PIND_D>> <<CLADD_D>> <<SIDECEL>>
SPLITX 1 1 1 1 2 1 1 1
MIX 6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 7 7 7 7 6
    6 6 6 6 6 6 6 6

```



```

6 6 6 6 6 6 6 6
6 6 7 7 7 7 7 6
6 6 7 8 8 8 7 6
6 6 7 8 8 8 7 6
6 6 7 8 8 8 7 6
6 6 7 7 7 7 7 6
6 6 6 6 6 6 6 6

6 6 6 6 6 6 6 6
6 6 7 7 7 7 7 6
6 6 7 8 8 8 7 6
6 6 7 8 8 8 7 6
6 6 7 8 8 8 7 6
6 6 7 7 7 7 7 6
6 6 6 6 6 6 6 6

;
::: CELF-LGB := GEO: CELF-LGH
MESHY 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
        <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
SPLITTY 1 1 1 1 2 1 1 1
MIX 1 1 1 1 1 1 1 1
    1 2 2 2 2 2 2 2
    1 2 3 3 3 3 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 3 3 3 3 2
    1 2 2 2 2 2 2 2

    1 1 1 1 1 1 1 1
    1 2 2 2 2 2 2 2
    1 2 3 3 3 3 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 4 4 4 3 2
    1 2 3 3 3 3 3 2

```

```

1 2 2 2 2 2 2 2
;
1 1 1 1 1 1 1 1
1 2 2 2 2 2 2 2
1 2 3 3 3 3 3 2
1 2 3 4 4 4 3 2
1 2 3 4 4 4 3 2
1 2 3 4 4 4 3 2
1 2 3 3 3 3 3 2
1 2 2 2 2 2 2 2
;
::: CELF-LB := GEO: CELF-LH
MESHY 0.0 <<ESPACE>> <<CLAD_LG>> <<PIN_LG>> <<PEAU_LG>>
      <<PEAU_LD>> <<PIN_LD>> <<CLAD_LD>> <<SIDECEL>>
SPLITY 1 1 1 1 2 1 1 1
MIX 5 5 5 5 5 5 5 5
    6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 7 7 7 7 6
    6 6 6 6 6 6 6 6

    5 5 5 5 5 5 5 5
    6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 8 8 8 7 6
    6 6 7 7 7 7 7 6
    6 6 6 6 6 6 6 6

    5 5 5 5 5 5 5 5
    6 6 6 6 6 6 6 6
    6 6 7 7 7 7 7 6

```

```

6 6 7 8 8 8 7 6
6 6 7 8 8 8 7 6
6 6 7 8 8 8 7 6
6 6 7 7 7 7 7 6
6 6 6 6 6 6 6 6
;
::: CELF-LDB := GEO: CELF-LB
MESHX 0.0 <<ESPACE>> <<CLADD_G>> <<PIND_G>> <<PEAUD_G>>
        <<PEAUD_D>> <<PIND_D>> <<CLADD_D>> <<SIDECEL>>
SPLITX 1 1 1 1 2 1 1 1
MIX 1 1 1 1 1 1 1 1
      2 2 2 2 2 2 2 2
      2 2 3 3 3 3 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 3 3 3 3 2
      2 2 2 2 2 2 2 2

      1 1 1 1 1 1 1 1
      2 2 2 2 2 2 2 2
      2 2 3 3 3 3 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 3 3 3 3 2
      2 2 2 2 2 2 2 2

      1 1 1 1 1 1 1 1
      2 2 2 2 2 2 2 2
      2 2 3 3 3 3 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 4 4 4 3 2
      2 2 3 3 3 3 3 2
      2 2 2 2 2 2 2 2

```

```

;
::: CELH-LGB := GEO: CEL-LGB
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LB := GEO: CEL-LB
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LDB := GEO: CEL-LDB
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LG := GEO: CEL-LG
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH := GEO: CEL
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LD := GEO: CEL-LD
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LGH := GEO: CEL-LGH
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LH := GEO: CEL-LH
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1
;
::: CELH-LDH := GEO: CEL-LDH
  MESHZ <<PSZ1>> <<HPEAU>> <<PSZ3>>
  SPLITZ <<iszpeau>> 1

```

```

;
;
TRACKING TRKBIN := NXT: GEOM ::
! EDIT 1000
  PRIZ 20.0
  GAUS 8
  TISO EQW 12 20.0
! NOTR
;
!TRKASCI := TRACKING ;
PSPFIL := PSP: TRACKING ::
  TYPE MIXTURE
;
!END: ;
Macrolib := MacComp12 ;
TRACKING := MCCGT: TRACKING TRKBIN GEOM ::
  EDIT 10 !<<EDILEV>>
! DIFC ILUO TMT
! LCMD 6
! MAXI 1
! EPSI 1E-5
! MCU 400000
! AAC 150 NONE
! DIFC NONE
  NONE
  AAC 0
! TMT
  KRYL 0
  SCR 0
! HDD 10.0
  STIS 0
! LEXF
! MAXI 1
;
ASSBLY := ASM: Macrolib TRACKING TRKBIN ::
  EDIT 0
```

```
    ARM SKIP
;
FLU := FLU: ASSBLY Macrolib TRACKING TRKBIN ::
!  EDIT 10
    ACCE 3 0
    REBA OFF
    TYPE K
    EXTE 1E-3
    THER 1E-3
;
FLUX := FLU ;
EDINAM := EDI: Macrolib TRACKING FLU ::
    EDIT 1000  MERG COMP
;
END: ;
QUIT "LIST" .
```

ANNEXE II - TAUX DE RÉACTION

Premier cas - composition uniforme

Région	Groupe		CP 3-D	Prism Uniforme	Prism Général	MOC 3-D
1	1	Flux moyen	1.4483E-01	1.4482E-01	1.4482E-01	1.4482E-01
		Flux intégré	9.0998E+00	9.0993E+00	9.0993E+00	9.0991E+00
		Taux de collision	3.4521E+00	3.4519E+00	3.4519E+00	3.4519E+00
		Taux d'absorption	2.7912E-01	2.7910E-01	2.7910E-01	2.7910E-01
		Taux de fission	2.2652E-01	2.2651E-01	2.2651E-01	2.2650E-01
		Taux de production	5.9403E-01	5.9410E-01	5.9410E-01	5.9406E-01
		Diffusion intra	3.1655E+00	3.1653E+00	3.1653E+00	3.1653E+00
		Diffusion entre	7.4937E-03	7.4933E-03	7.4933E-03	7.4932E-03
	2	Flux moyen	1.0375E-02	1.0378E-02	1.0378E-02	1.0376E-02
		Flux intégré	6.5188E-01	6.5204E-01	6.5204E-01	6.5197E-01
		Taux de collision	4.5572E-01	4.5583E-01	4.5583E-01	4.5577E-01
		Taux d'absorption	1.9700E-01	1.9705E-01	1.9705E-01	1.9703E-01
		Taux de fission	3.6751E-01	3.6760E-01	3.6760E-01	3.6755E-01
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
2	1	Flux moyen	1.3785E-01	1.3786E-01	1.3786E-01	1.3786E-01
		Flux intégré	8.5697E+00	8.5702E+00	8.5702E+00	8.5704E+00
		Taux de collision	5.1053E+00	5.1056E+00	5.1056E+00	5.1057E+00
		Taux d'absorption	4.3521E-03	4.3521E-03	4.3521E-03	4.3521E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	4.8881E+00	4.8883E+00	4.8883E+00	4.8885E+00
		Diffusion entre	2.1287E-01	2.1288E-01	2.1288E-01	2.1289E-01
	2	Flux moyen	1.7883E-02	1.7858E-02	1.7858E-02	1.7877E-02
		Flux intégré	1.1118E+00	1.1102E+00	1.1102E+00	1.1114E+00
		Taux de collision	1.8566E+00	1.8540E+00	1.8540E+00	1.8560E+00
		Taux d'absorption	1.9531E-02	1.9504E-02	1.9504E-02	1.9524E-02

Région	Groupe		CP 3-D	Prism Uniforme	Prism Général	MOC 3-D
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	1.8345E+00	1.8319E+00	1.8319E+00	1.8338E+00
		Diffusion entre	2.6242E-03	2.6205E-03	2.6205E-03	2.6233E-03
3	1	Flux moyen	1.4483E-01	1.4482E-01	1.4482E-01	1.4482E-01
		Flux intégré	9.0998E+00	9.0993E+00	9.0993E+00	9.0991E+00
		Taux de collision	3.4521E+00	3.4519E+00	3.4519E+00	3.4519E+00
		Taux d'absorption	2.7912E-01	2.7910E-01	2.7910E-01	2.7910E-01
		Taux de fission	2.2652E-01	2.2651E-01	2.2651E-01	2.2650E-01
		Taux de production	5.9403E-01	5.9408E-01	5.9408E-01	5.9406E-01
		Diffusion intra	3.1655E+00	3.1653E+00	3.1653E+00	3.1653E+00
		Diffusion entre	7.4937E-03	7.4933E-03	7.4933E-03	7.4932E-03
	2	Flux moyen	1.0375E-02	1.0377E-02	1.0377E-02	1.0376E-02
		Flux intégré	6.5188E-01	6.5201E-01	6.5201E-01	6.5197E-01
		Taux de collision	4.5572E-01	4.5580E-01	4.5580E-01	4.5577E-01
		Taux d'absorption	1.9700E-01	1.9704E-01	1.9704E-01	1.9703E-01
		Taux de fission	3.6751E-01	3.6758E-01	3.6758E-01	3.6755E-01
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	2.5751E-01	2.5756E-01	2.5756E-01	2.5754E-01
		Diffusion entre	1.2068E-03	1.2070E-03	1.2070E-03	1.2070E-03
4	1	Flux moyen	1.3785E-01	1.3785E-01	1.3785E-01	1.3786E-01
		Flux intégré	8.5697E+00	8.5701E+00	8.5701E+00	8.5704E+00
		Taux de collision	5.1053E+00	5.1055E+00	5.1055E+00	5.1057E+00
		Taux d'absorption	4.3516E-03	4.3521E-03	4.3521E-03	4.3521E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	4.8881E+00	4.8883E+00	4.8883E+00	4.8885E+00
		Diffusion entre	2.1287E-01	2.1288E-01	2.1288E-01	2.1289E-01
	2	Flux moyen	1.7883E-02	1.7857E-02	1.7857E-02	1.7877E-02
		Flux intégré	1.1118E+00	1.1101E+00	1.1101E+00	1.1114E+00
		Taux de collision	1.8566E+00	1.8539E+00	1.8539E+00	1.8560E+00
		Taux d'absorption	1.9531E-02	1.9502E-02	1.9502E-02	1.9524E-02
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00

Région	Groupe	CP 3-D	Prism Uniforme	Prism Général	MOC 3-D
	Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	Diffusion intra	1.8345E+00	1.8317E+00	1.8317E+00	1.8338E+00
	Diffusion entre	2.6242E-03	2.6203E-03	2.6203E-03	2.6233E-03

Second cas – Composition variable

Region	Groupe		CP 3-D	Prism Uniforme	Prism Général	MOC 3-D
1	1	Flux moyen	1.8199E-01	1.8195E-01	1.8195E-01	1.8194E-01
		Flux intégré	1.1435E+01	1.1432E+01	1.1432E+01	1.1431E+01
		Taux de collision	4.3380E+00	4.3370E+00	4.3370E+00	4.3367E+00
		Taux d'absorption	3.5074E-01	3.5066E-01	3.5066E-01	3.5064E-01
		Taux de fission	2.8465E-01	2.8458E-01	2.8458E-01	2.8456E-01
		Taux de production	1.1355E+00	1.1359E+00	1.1359E+00	1.1359E+00
		Diffusion intra	3.9779E+00	3.9769E+00	3.9769E+00	3.9766E+00
		Diffusion entre	9.4168E-03	9.4145E-03	9.4145E-03	9.4138E-03
	2	Flux moyen	2.4019E-02	2.4034E-02	2.4034E-02	2.4034E-02
		Flux intégré	1.5092E+00	1.5101E+00	1.5101E+00	1.5101E+00
		Taux de collision	1.0550E+00	1.0557E+00	1.0557E+00	1.0557E+00
		Taux d'absorption	4.5607E-01	4.5636E-01	4.5636E-01	4.5636E-01
		Taux de fission	8.5080E-01	8.5133E-01	8.5133E-01	8.5134E-01
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	5.9615E-01	5.9652E-01	5.9652E-01	5.9653E-01
		Diffusion entre	2.7938E-03	2.7955E-03	2.7955E-03	2.7956E-03
2	1	Flux moyen	1.6811E-01	1.6809E-01	1.6809E-01	1.6810E-01
		Flux intégré	1.0451E+01	1.0450E+01	1.0450E+01	1.0450E+01
		Taux de collision	6.2261E+00	6.2253E+00	6.2253E+00	6.2256E+00
		Taux d'absorption	5.3072E-03	5.3067E-03	5.3067E-03	5.3067E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	5.9612E+00	5.9604E+00	5.9604E+00	5.9607E+00
		Diffusion entre	2.5961E-01	2.5957E-01	2.5957E-01	2.5958E-01
	2	Flux moyen	3.6126E-02	3.6124E-02	3.6124E-02	3.6149E-02
		Flux intégré	2.2459E+00	2.2457E+00	2.2457E+00	2.2473E+00
		Taux de collision	3.7506E+00	3.7503E+00	3.7503E+00	3.7529E+00
		Taux d'absorption	3.9455E-02	3.9452E-02	3.9452E-02	3.9480E-02
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	3.7058E+00	3.7055E+00	3.7055E+00	3.7081E+00

Region	Groupe		CP 3-D	Prism Uniforme	Prism Général	MOC 3-D
		Diffusion entre	5.3012E-03	5.3008E-03	5.3008E-03	5.3045E-03
3	1	Flux moyen	1.2725E-01	1.2728E-01	1.2728E-01	1.2729E-01
		Flux intégré	7.9956E+00	7.9973E+00	7.9973E+00	7.9978E+00
		Taux de collision	4.7633E+00	4.7643E+00	4.7643E+00	4.7646E+00
		Taux d'absorption	4.0603E-03	4.0612E-03	4.0612E-03	4.0617E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	4.5606E+00	4.5616E+00	4.5616E+00	4.5618E+00
		Diffusion entre	1.9861E-01	1.9865E-01	1.9865E-01	1.9866E-01
	2	Flux moyen	6.3486E-02	6.3404E-02	6.3404E-02	6.3391E-02
		Flux intégré	3.9889E+00	3.9838E+00	3.9838E+00	3.9830E+00
		Taux de collision	6.6613E+00	6.6528E+00	6.6528E+00	6.6515E+00
		Taux d'absorption	7.0076E-02	6.9986E-02	6.9986E-02	6.9972E-02
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	6.5818E+00	6.5734E+00	6.5734E+00	6.5721E+00
		Diffusion entre	9.4154E-03	9.4034E-03	9.4034E-03	9.4015E-03
4	1	Flux moyen	1.2642E-01	1.2645E-01	1.2645E-01	1.2645E-01
		Flux intégré	7.8592E+00	7.8613E+00	7.8613E+00	7.8614E+00
		Taux de collision	4.6820E+00	4.6833E+00	4.6833E+00	4.6834E+00
		Taux d'absorption	3.9911E-03	3.9921E-03	3.9921E-03	3.9926E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	4.4828E+00	4.4840E+00	4.4840E+00	4.4841E+00
		Diffusion entre	1.9522E-01	1.9527E-01	1.9527E-01	1.9528E-01
	2	Flux moyen	6.4364E-02	6.4263E-02	6.4263E-02	6.4268E-02
		Flux intégré	4.0014E+00	3.9951E+00	3.9951E+00	3.9954E+00
		Taux de collision	6.6822E+00	6.6717E+00	6.6717E+00	6.6722E+00
		Taux d'absorption	7.0295E-02	7.0184E-02	7.0184E-02	7.0190E-02
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	6.6024E+00	6.5920E+00	6.5920E+00	6.5926E+00
		Diffusion entre	9.4449E-03	9.4300E-03	9.4300E-03	9.4308E-03

Troisième cas – Géométrie variable

Region	Groupe		CP 3-D	Prism Général	MOC 3-D
1	1	Flux moyen	1.8199E-01	1.8195E-01	1.8194E-01
		Flux intégré	1.1435E+01	1.1432E+01	1.1431E+01
		Taux de collision	4.3378E+00	4.3369E+00	4.3366E+00
		Taux d'absorption	3.5073E-01	3.5066E-01	3.5063E-01
		Taux de fission	2.8464E-01	2.8458E-01	2.8456E-01
		Taux de production	1.1356E+00	1.1362E+00	1.1362E+00
		Diffusion intra	3.9777E+00	3.9769E+00	3.9766E+00
		Diffusion entre	9.4164E-03	9.4144E-03	9.4137E-03
	2	Flux moyen	2.4023E-02	2.4041E-02	2.4041E-02
		Flux intégré	1.5094E+00	1.5105E+00	1.5106E+00
		Taux de collision	1.0552E+00	1.0560E+00	1.0560E+00
		Taux d'absorption	4.5614E-01	4.5649E-01	4.5650E-01
		Taux de fission	8.5093E-01	8.5158E-01	8.5159E-01
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	5.9624E-01	5.9669E-01	5.9670E-01
		Diffusion entre	2.7942E-03	2.7964E-03	2.7964E-03
2	1	Flux moyen	1.6811E-01	1.6809E-01	1.6810E-01
		Flux intégré	1.0451E+01	1.0450E+01	1.0450E+01
		Taux de collision	6.2261E+00	6.2254E+00	6.2257E+00
		Taux d'absorption	5.3077E-03	5.3067E-03	5.3072E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	5.9612E+00	5.9605E+00	5.9608E+00
		Diffusion entre	2.5960E-01	2.5957E-01	2.5959E-01
	2	Flux moyen	3.6101E-02	3.6109E-02	3.6134E-02
		Flux intégré	2.2443E+00	2.2448E+00	2.2464E+00
		Taux de collision	3.7479E+00	3.7488E+00	3.7514E+00
		Taux d'absorption	3.9427E-02	3.9437E-02	3.9464E-02
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	3.7032E+00	3.7041E+00	3.7066E+00
		Diffusion entre	5.2975E-03	5.2987E-03	5.3024E-03

Region	Groupe		CP 3-D	Prism Général	MOC 3-D
3	1	Flux moyen	1.2684E-01	1.2686E-01	1.2687E-01
		Flux intégré	1.5855E+01	1.5858E+01	1.5859E+01
		Taux de collision	9.4456E+00	9.4472E+00	9.4475E+00
		Taux d'absorption	8.0519E-03	8.0538E-03	8.0538E-03
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	9.0437E+00	9.0452E+00	9.0455E+00
		Diffusion entre	3.9384E-01	3.9391E-01	3.9392E-01
	2	Flux moyen	6.3909E-02	6.3780E-02	6.3775E-02
		Flux intégré	7.9886E+00	7.9725E+00	7.9719E+00
		Taux de collision	1.3341E+01	1.3314E+01	1.3313E+01
		Taux d'absorption	1.4034E-01	1.4006E-01	1.4005E-01
		Taux de fission	0.0000E+00	0.0000E+00	0.0000E+00
		Taux de production	0.0000E+00	0.0000E+00	0.0000E+00
		Diffusion intra	1.3182E+01	1.3155E+01	1.3154E+01
		Diffusion entre	1.8856E-02	1.8818E-02	1.8817E-02