UNIVERSITÉ DE MONTRÉAL

DEVELOPMENT AND QUALIFICATION OF ADVANCED COMPUTATIONAL SCHEMES FOR PRESSURIZED WATER REACTORS AND CREATION OF SPECIFIC INTERFACES TOWARDS GRS FULL-CORE TOOLS

THIBAUD REYSSET
DÉPARTEMENT DE GÉNIE PHYSIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉNERGÉTIQUE)
JUILLET 2009

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

DEVELOPMENT AND QUALIFICATION OF ADVANCED COMPUTATIONAL SCHEMES FOR PRESSURIZED WATER REACTORS AND CREATION OF SPECIFIC INTERFACES TOWARDS GRS FULL-CORE TOOLS

présenté par: REYSSET Thibaud
en vue de l'obtention du diplôme de: Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de:

M. MARLEAU Guy, Ph.D., président
M. HEBERT Alain, D.Ing., membre et directeur de recherche
M. KOCLAS Jean, Ph.D., membre et codirecteur de recherche
M. PAUTZ Andreas, Ph.D., membre

To my family, and my friends.

ACKNOWLEDGEMENTS

I wish first to sincerely thank my research director Professor Alain Hébert for accepting me in this program, giving me the opportunity to work on this subject, and providing me support all along this work. I also sincerely wish to thank Nicolas Martin for giving me all the informations I needed before starting this program, for all his advises and his help during this program, and for his friendly support during my stay of one year in Montréal. I would also like to thank Rémi Vallerent for his constant working and friendly support, from the beginning until the end of this progam. Then, I would like to thank all the members of the *Institut de Génie Nucléaire* from the *École Polytechnique de Montréal*: the professors for their support, particularly Guy Marleau for his advises, and also the students for the good atmosphere, and the support during our projects.

This project gave me the opportunity to do an internship in the GRS and I wish to sincerely thank my research director Andreas Pautz for giving me this opportunity, persevering with my research, and providing me all the support that a student needs to perform a good internship. I would also like to thank the GRS-Garching employees for their support and the good atmosphere within the company. I wish to thank particularly Yann Perrin for helping me with my work, and for his friendly support during my stay in Munich, Elisabeth Bordus, Armin Seubert, Kiril Velkov, Winfried Zwermann and Ihor Pasichnyk for their support during my research. Then, I would like to thank the other students for their friendly support, particularly Carsten Rottgen and Antonio Sureda.

Finally, I wish to sincerely thank my family, and particularly my great uncle Jean-Michel Reysset, his friend Lucie Boisseau, and her three daughters for welcoming me, providing me emotional support, and making me feel at home during my stay of one year in Montréal. Also, I would like to thank all my friends, and the people who supported me in Montréal, in Munich, and in France, for making this work such a great experience.

RÉSUMÉ

Ce projet s'intéresse au dévelopement et à la validation d'un schéma de calcul à deux niveau permettant la détermination rapide et précise de sections efficaces à l'aide du code de réseau DRAGON. Généralement, des schéma de calculs classiques à un niveau sont utilisés, employant la méthode des probabilités de collision, très longtemps privilégiée pour sa simplicité, ou la méthode des caractéristiques, pouvant traiter des configurations plus importantes (nombre d'inconnues plus élevé) ce qui mène à des résultats plus précis. Le problème est que plus la méthode utilisée implique d'inconnues, plus le calcul est long. Une solution pour obtenir à la fois rapidité et précision est ainsi de combiner ces différentes méthodes dans un schéma à deux niveaux.

Ce projet a donc pour objectif de valider un schéma de calcul avancé à deux niveaux avec l'aide du code de réseau DRAGON, en le comparant aux schémas classiques à un niveau, mais aussi en comparant les résultats obtenus avec différents autres codes.

Le schéma de calcul à deux niveaux sera ainsi validé à travers deux études comparatives sur des assemblages de réacteurs à eau légère préssurisée (REP), ce type de récteurs étant le plus répandu dans le monde. La première étude sera effectuée sur un exercice traitant d'un assemblage allemand typique 18x18 (type KONVOI). Cette étude aura pour but de valider le schéma à travers diverses comparaisons internes au code DRAGON, puis en comparant les résultats obtenus avec ceux provenant d'autres codes de calcul.

La deuxième étude portera sur un benchmark OECD traitant d'un coeur complet composé d'assemblages UO_2 et MOX. Elle sera l'occasion de la mise en place d'un schéma à deux niveaux avec DRAGON, permettant de produire des bibliothèques de sections efficaces condensées qui seront utilisées dans le système couplé QUABOX-CUBBOX/ ATHLET pour l'étude d'un transitoire sur un coeur complet, en comparaison avec des bibliothèques produites par le code HELIOS.

ABSTRACT

The goal of this project is to develop and the validate a two-level calculational scheme suitable for the fast and accurate determination of nuclear cross sections with the lattice code DRAGON. Generally, classical one-level calculation schemes are used, employing the method of collision probability, which was the preffered option due to its relative simplicity, or the method of characteristics, capable of handling larger problems (more unknowns) leading to more accurate results. The problem is that a larger number of unknowns implies a longer calculation time. One solution to achieve both speed and accuracy is to combine these different methods in a two-level scheme.

The purpose of this project is to validate an advanced two-level scheme with the use of the DRAGON lattice code, comparing it to classical one-level schemes, but also comparing the results with various other codes.

The two-level scheme will be validated through two comparative studies on Pressurized Water Reactors (PWR) assemblies, this type of reactor being the most prevalent in the world. The first study will be carried out on an exercise dealing with typical 18x18 German assembly (KONVOI type). This study will aim to validate the scheme through various internal comparisons with the DRAGON code, and then comparing the results obtained with those coming from other codes.

The second study will focus on a UO_2/MOX OECD full core benchmark. It will be an opportunity to set up a two-level scheme with DRAGON, to produce condensed cross sections libraries that will be used in the coupled code system QUABOX-CUBBOX/ ATHLET for a transient study on a complete a core, in comparison with libraries generated by the HELIOS code.

CONDENSÉ EN FRANÇAIS

De nos jours, le type de réacteur le plus répandu dans le monde est le réacteur à eau pressurisée (REP). Il est caractérisé par différents types d'assemblages de combustible, ayant des dimensions différentes (le plus souvent 16x16, 17x17, ou 18x18 barres de combustible par assemblage), et par son modérateur à l'eau légère, qui joue aussi le rôle de caloporteur. Habituellement, le carburant utilisé dans ces assemblages est constitué de dioxyde d'uranium (UO_2) ou d'un oxyde mixte (MOX). Le flux de neutrons, nécessaire pour soutenir une réaction nucléaire en chaîne dans le coeur du réacteur, domine le comportement stationnaire et transitoire du réacteur. Pour simuler les transitoires et les accidents d'un réacteur nucléaire, une connaissance approfondie, en espace et en temps, de l'évolution du flux est nécessaire. En effet, la réaction de fission dans les réacteurs, qui est la principale source d'énergie, est due à la collision entre un neutron et un noyau lourd. Il est alors compréhensible que la connaissance du flux neutronique est essentielle dans les simulations de réacteurs.

La simulation neutronique joue un rôle très important dans les analyses de sûreté des réacteurs nucléaires, qui est un élément essentiel dans une évaluation de sûreté. Ces simulations impliquent des approches déterministes ou probabilistes en vue de résoudre l'équation du transport des neutrons sur un réacteur complet. Le problème avec la simulation du comportement d'un coeur complet est qu'il est aujourd'hui impossible de l'obtenir directement. Différents niveaux de calcul doivent être effectués, en utilisant des approximations, et le flux de neutrons sur un coeur complet ne sera connu qu'après l'accomplissement de ces différents niveaux. Ici, le niveau qui nous intéresse est le calcul de réseau.

Le but du calcul de réseau est la résolution de l'équation du transport, et l'étude des valeurs caractéristiques sur une cellule unitaire représentative du coeur. Celle-ci peut être une simple cellule (crayon de combustible), une super-cellule ou un assemblage complet. L'équation du transport est alors résolue sous différentes contraintes (conditions aux frontières, diffusion isotrope ou non, état stationnaire, ...) et à différentes méthodes de calcul, déterministes (méthode des courants d'interface, probabilités de collision, ou méthode des caractéristiques) ou stochastiques. Dans notre cas, des méthodes déterministes sont utilisées, résolvant l'équation sur un découpage multigroupe. En effet, les méthodes déterministes utilisent des valeurs constantes des sections efficaces par segments d'énergie, ce qui correspond à la discrétisation multigroupe.

Ces valeurs sont données dans des bibliothèques multigroupes de sections efficaces, et sont déterminées à l'aide de divers paramètres : l'enrichissement du combustible, la composition matérielle du combustible, de la gaine et du modérateur, les températures, la concentration en poisons consommables, le burnup, etc... Le choix de la bibliothèque est ainsi la première étape du calcul. Celles-ci sont créées à l'aide de différentes évaluations créées à l'aide de mesures expérimentales, ou de modèles physiques.
La deuxième étape du calcul multigroupe est le calcul d'autoprotection des données nucléaires en fonction des caractéristiques géométriques et des concentrations isotopiques. La présence d'isotopes résonnants provoque une dépression locale dans la courbe de flux à l'endroit de la résonance. Cela affecte les taux de réaction, produit des sections efficaces et du flux. Ils sont ainsi plus faibles que si le flux était non résonnant. Cet effet s'appelle l'autoprotection. Il faut ainsi correctement évaluer les sections efficaces autoprotégées pour tenir compte de cet effet. Ce calcul peut être effectué en utilisant deux modèles différents : l'un basé sur une équivalence en dilution donnant des sections efficaces moyennées utilisées pour interpoler des intégrales de résonance, l'autre basé sur une méthode des sous-groupes utilisant des tables physiques de probabilités.

L'étape suivante est le calcul de flux. La géométrie est d'abord discrétisée et traitée à l'aide de méthodes de tracking. Puis, les informations nécessaires au calcul du flux sont générées à l'aide de la méthode des probabilités de collision ou de la méthode des caractéristiques. Mais afin de traiter une cellule ou un assemblage, un modèle de fuite est aussi requis permettant de représenter les pertes de neutrons non prises en compte par les conditions aux frontières. Après cette étape sont connus les flux et les taux de réaction en divers endroits du réseau.
Une fois ces informations connues, les sections efficaces et les coefficients de diffusion doivent être homogénéisés et condensés afin d'être utilisées dans un code de coeur. Lors de cette étape, afin de conserver les taux de réaction, une procédure d'équivalence est parfois utilisée, appelée équivalence SPH, surtout pour des géométries complexes tels des assemblages. Les taux de réaction sont ainsi corrigés à l'aide de facteurs satisfaisants à différentes conditions de normalisation.
La dernière étape du calcul est l'évolution isotopique. L'exposition d'un noyau à un flux de neutron cause sa disparition. De ce fait, les concentrations isotopiques originales vont évoluer au cours du temps. Une fois cette étape effectuée, le calcul peut reprendre à l'étape de l'autoprotection et tourner en boucle.

Tout ce calcul prend généralement beaucoup de temps, et plus la méthode utilisée est précise (supposée donner les résultats les plus précis), plus le temps de calcul est long. Mais il est possible de combiner différentes méthodes pour obtenir un schéma de calcul qui est à la fois rapide et précis : un schéma à deux niveaux. Au premier niveau, le flux est résolu avec une méthode UP1 à courant d'interface, méthode la plus rapide, puis, après condensation à un nombre de groupes bien inférieur (26 groupes au lieu de 172 à l'origine), un deuxième calcul est effectué avec une méthode des caractéristiques, la méthode supposée la plus précise. Le but de ce projet est le développement et la validation d'un tel schéma de calcul à deux niveaux.

Ce travail a été réalisé à l'aide du code déterministe de réseau DRAGON (Marleau et al., 2009), développé à l'Institut de Génie Nucléaire de l'École Polytechnique de Montréal. La version utilisée est DRAGON Version4, qui fait partie de la distribution Version4 (Hébert, 2006). Le premier objectif de ce code a été l'étude du réacteur Canada Deuterium Uranium (CANDU), mais il peut aussi être utilisé sur un grand nombre de types d'assemblages de réacteurs différents. Le code est divisé en divers modules (reprenant les diverses étapes présentées au dessus) qui échangent des données à travers des structures de données bien définies, permettant de faciliter son développement et son utilisation.

Ce projet a été réalisé dans un contexte de recherche et d'industrie à la Gesellschaft für Anlagen-und Reaktorsicherheit (GRS), qui traite de la sûreté des réacteurs nucléaires en Allemagne et à travers l'Europe et le monde. Les travaux ont été effectués sur deux exercices différents afin de valider le développement du schéma à deux niveaux.

Exercice PWR 18x18

La première partie du travail à la GRS était l'implémentation du code DRAGON sur le cluster local Linux, et de comparer la précision des résultats avec un large éventail d'autres codes sur un exercice simple. L'exercice étudié, traite d'un assemblage de type PWR 18x18 composé d'UO_2 proposé par D. Porsch (Framatome), U. Hesse and W. Zwermann (GRS), et W. Bernnat (IKE, Stuttgart university) (Porsch et al., 2006). L'assemblage est composé de 300 cellules de combustible d'UO_2 enrichi à 4% d'uranium 235, et de 24 tubes guides (dont un tube qui est aussi prévu pour accueillir un détecteur, mais qui sera considéré comme tube guide dans les calculs), le tout étant considéré aux conditions HFP (Hot Full Power). Les gaines sont composées de zyrcaloy-4.

La première étape est de choisir une modélisation qui amène à obtenir une bonne précision en un temps de calcul raisonnable. Un modèle de calcul est alors d'abord choisi, utilisant les symétries de l'assemblage et les simplifications indiquées dans l'exercice. La géométrie est ensuite discrétisée de manière à prendre en compte les effets de distribution spatiale des isotopes résonants (en divisant le combustible en quatre anneaux représentant 50%, 30%, 15%, et 5% du volume total, de l'intérieur vers l'extérieur), et le découpage du modérateur (de façon cartésienne en quatre volumes égaux) pour obtenir de bons résultats. Finalement, des regroupements de mélanges et de cellules sont étudiés. Les regroupements de mélanges ont pour but de diminuer les nombre de mélanges à traiter lors du calcul d'autoprotection et de flux, réduisant considérablement le temps de calcul. Les regroupements de cellules impliquent que certaines cellules partagent le même flux et sont là utilisés lors de l'étape de calcul d'autoprotection utilisant une méthode à courant d'interface UP1. Deux regroupements différents sont alors étudiés, le deuxième montrant de bien meilleurs résultats que le premier, dans un temps un peu supérieur que pour le premier, mais comparable. Le temps de calcul est divisé par presque 20 comparé à un calcul sans regroupement, mais avec une précision acceptable. La deuxième étape est de comparer les différentes méthodes de calcul de flux possibles dans DRAGON. Pour l'autoprotection, une méthode des sous-groupes est utilisée, et le calcul est effectué par le solveur UP1, ou le solveur MOC. Pour le flux, quatre méthodes différents sont utilisées : une méthode UP1 à courant d'interface, une méthode Pij (probabilités de collision), une méthode MOC, et un schéma à deux niveaux comme décrit précédemment (UP1 pour l'autoprotection et le niveau 1, et MOC pour le niveau 2). L'usage d'une équivalence SPH est alors étudié entre les deux niveaux, montrant de bien meilleurs résultats, sans rajouter beaucoup de temps de calcul. Au final, le calcul utilisant le solveur MOC pour le calcul d'autoprotection et de flux, et le schéma deux-niveaux sont conservés pour la comparaison finale avec les autres résultats, le premier étant supposé le plus précis, et le deuxième étant le plus rapide comparé aux autres méthodes, et donnant de très bons résultats.

Finalement, les résultats obtenus grâce à ces deux schémas sont comparés à d'autres résultats provenant de différents codes de calcul et effectués par différentes organisations. Trois codes ont été utilisés à la GRS : KENOREST (Hesse et al., 2000), HELIOS (Casal et al., 1991), et MONTEBURNS (X-5 Monte Carlo Team, 2003). Deux autres organisations ont aussi participé à cette étude : TÜV SÜD avec le code CASMO4 (Rhodes and Edenius, 2001) et l'IRSN avec APOLLO2 (Hoffman et al., 1973). Pour les calculs DRAGON, la librairie à 172 groupes basée sur l'évaluation JEFF3.1 est utilisée, correspondant à l'évaluation utilisée pour MONTEBURNS. Ce code est pris comme référence ici, car il est supposé être le plus précis, utilisant une méthode de Monte-Carlo pour effectuer les calculs. Les différents calculs sont ainsi comparés sur trois paramètres différents : le coefficient de multiplication effectif (k_{eff}), les densités isotopiques, et la puissance par crayon normalisée. Dans les trois cas, les deux schémas DRAGON montrent de très bons résultats. Cette première étude prouve ainsi la précision du code DRAGON, et valide le schéma de calcul à deux niveaux, qui donne des résultats acceptables en un temps de calcul assez court.

Benchmark Purdue

Finalement, une deuxième étude a été effectuée sur benchmark traitant de transitoires pour un coeur de réacteur de type PWR composé d'assemblages 17x17 au MOX et UO_2, proposé par T. Kozlowski et T.J. Downar (Kozlowski and Downar, 2003). Pour cette étude, seul des calculs d'assemblages seront effectués, sur deux types différents. Un premier type d'assemblage au UO_2 est composé d'un tube guide central, de 24 tubes guides qui servent aussi à l'insertion de barres de contrôle, de 160 cellules de combustible typique, et de 104 cellules (IFBA) de combustible entouré d'une fine couche de diboride de zirconium (ZrB_2) qui contrôle la réactivité sur une courte période de burnup. Le combustible est considéré pour deux enrichissements différents : 4,2 et 4,5%. Le deuxième

type d'assemblage est de type MOX composé d'un tube guide central, de 24 tubes guides (WABA) contenant un tube de Al_2-O_3-B_4C rempli d'eau et qui contrôle la réactivité à long terme, de 12 cellules de combustible MOX enrichi à 2,5% de Pu-fissile (^{239}Pu et ^{241}Pu), 76 crayons de MOX à 3,0%, et 176 crayons de MOX à 4,5% ou 5,0% suivant les conditions d'enrichissement (4,5% pour un assemblage à 4,0%, et 5,0% pour un assemblage à 4,3%). Pour chacun de ces deux assemblages, deux enrichissements différents sont donc pris en compte. Le réacteur est considéré aux conditions HFP, avec une concentration de bore de 1000 ppm dans le modérateur, et une puissance de 37,87MW/t. Les gaines sont composées de zyrcaloy-2.

Une première étude porte sur une simple cellule d'UO_2. Pour cette étude, la discrétisation est choisie, comme pour l'exercice précédent, avec un découpage du combustible en cinq anneaux de volume égal, et le même découpage pour le modérateur. Pour l'évolution isotopique, des pas de burnup ont été ajoutés à ceux du benchmark, pour mieux prendre en compte l'évolution des isotopes lourds. Des vérifications sont ensuite effectuées sur le schéma de calcul DRAGON, tout d'abord sur le calcul d'autoprotection, effectué avec un solveur UP1 et une méthode des sous-groupes (utilisée dans HELIOS), conduisant au choix d'une autoprotection à chaque pas de burnup, malgré un gain de temps considérable en n'effectuant ce calcul que pour quelques pas. Puis des vérifications sont faites sur les méthodes de calcul de flux, montrant une fois encore que le schéma de calcul à deux niveaux (avec équivalence SPH entre les 2 niveaux UP1+MOC) donne de bons résultats en comparaison avec les autres méthodes. Finalement, les résultats obtenus sont comparés à ceux donnés par HELIOS. Les différences sont assez importantes pour un calcul de cellule, mais restent acceptables. Elles montrent bien l'importance de l'ajout de pas de burnup en début d'évolution, et du choix de la bibliothèque utilisée. De ce fait, pour la suite, la librairie à 172 groupes basée sur l'évaluation ENDF/B-VII R0 est choisie, donnant les écarts les plus faibles dans ces calculs.

La deuxième étude porte sur les deux différents types d'assemblages, avec pour chacun les deux enrichissements. Encore une fois, la modélisation est étudiée et choisie. Pour la discrétisation du combustible, le choix est le même que précédemment pour tous les types de cellules. Pour les tubes guides, le modérateur dans le tube est divisé en trois anneaux de volume égal, et le modérateur autour comme pour les cellules de combustible. Comme pour l'exercice précédent, des regroupements sont utilisés pour réduire le temps de calcul. Ces regroupements distinguent cinq types de cellules : les cellules dans les coins de l'assemblage, et les bords de l'assemblage, celles qui partagent une face avec le tube central, et avec les autres tubes, et les autres cellules. Il y a ainsi 30 mélanges différents pour l'assemblage UO_2, et 40 pour l'assemblage MOX. Le temps de calcul est là encore considérablement réduit, mais les résultats montrent peu de différences. Les résultats du schéma de calcul à deux niveaux (avec équivalence SPH) sont finalement comparés aux résultats obtenus par HELIOS pour les deux enrichissements de chacun des deux assemblages. Les résultats sont acceptables, malgré de grosses différences pour l'assemblage MOX.

La dernière étape de cette étude est la création de bibliothèques de sections efficaces condensées à deux groupes et homogénéisées sur tout l'assemblage, pour l'étude d'un transitoire. Plusieurs calculs doivent être effectués avec des conditions différentes pour trois paramètres (trois conditions par parmètre) : la température du combustible, la densité du modérateur, et la concentration en bore dans le modérateur. Pour cela, un schéma de calcul est mis en place, où un premier calcul d'évolution est effectué, et les différentes densités d'atomes sont stockées pour être est ensuite utilisées pour mettre à jour les densités lors de l'évolution (mises à part celles du modérateur qui n'évolue pas) lors des calculs pour chaque ensemble de paramètres. Ce schéma est appelé calcul de reprise. Pour chaque ensemble de paramètres, les sections efficaces et autres informations importantes sont stockées dans une librairie (objet MULTICOMPO). Ces calculs sont effectués pour les deux types d'assemblages, pour leur deux enrichissements, mais aussi pour les deux assemblages UO_2 avec barres de contrôle insérées. Six objets MULTICOMPO sont

ainsi créés, qui sont ensuite convertis dans un format (NEMTAB) qui peut être lu par les codes de coeur de la GRS. Finalement, ces bibliothèques sont utilisées dans le système de codes couplés QUABOX-CUBBOX/ATHLET (Langenbuch and Velkov, 2004) de la GRS pour l'étude d'un transitoire après dilution de bore. Lors de ce transitoire, dans deux zones radiales du coeur sur trois, les concentrations de bore dans le modérateur sont diminuées en une seconde. Les résultats sont comparés à ceux obtenus avec des bibliothèques créées par HELIOS. Les résultats obtenus sont vraiment acceptables, et montrent ici que le code DRAGON, et par lui, que le schéma deux niveaux est capable de produire des bibliothèques de sections efficaces condensées et homogénéisées donnant de bons résultats dans un code de coeur.

CONCLUSION

Ces deux études ont ainsi permis de valider le schéma de calcul avancé à deux niveaux à l'aide du code DRAGON, montrant sa bonne précision et sa rapidité de temps de calcul, mais aussi sa capacité à créer des bibliothèques de sections efficaces condensées et homogénéisées pour les calculs de coeur complet, donnant de bons résultats en comparaison avec d'autres codes de réseau.

La première étude avait ainsi pour but d'effectuer différentes comparaisons, sur des calculs d'évolution, afin d'obtenir le schéma de calcul le plus performant, et de le valider. Deux niveaux de comparaison ont été abordés.
Tout d'abord, certaines vérifications ont dû être effectuées avec le code DRAGON. En effet, certains paramètres sont très sensibles, comme la discrétisation de la géométrie, le modèle choisi pour les calculs ou les méthodes utilisées dans les calculs. Afin de choisir le meilleur schéma de calcul possible, deux critères ont été utilisés : la précision et le temps de calcul. En effet, pour le même problème, plus la méthode utilisée est

précise, plus le temps de calcul est long. Ces deux critères sont très sensible au choix de la méthode utilisée pour effectuer le calcul d'auto-protection. Puis, quatre méthodes différentes ont été utilisées afin de résoudre le flux : une méthode de probabilité de collision, une méthode UP1 à courants d'interface (supposé être la plus rapide), une méthode des caractéristiques (pouvant utiliser plus d'inconnues, donc pouvant être plus précise), et un schéma à deux niveaux, qui est une combinaison de ces deux dernières méthodes. Le schéma deux-niveaux a montré de bons résultats en comparaison avec les autres calculs, et en particulier avec la méthode MOC, avec un temps de calcul très court.
Ensuite, ce schéma à deux niveaux et le calcul MOC ont été comparés à de nombreux autres calculs effectués par différentes organisations, utilisant d'autres codes. Le but ici était de prouver que le schéma deux-niveaux, et avant cela, le code DRAGON, sont en mesure de donner des résultats précis. Trois paramètres ont été utilisés pour cette validation : le coefficient effectif de multiplication, la densité des atomes, et la puissance par crayon. La comparaison des résultats sur ces paramètres a démontré que les deux schémas de calcul donnent de très bons résultats, le schéma deux-niveaux donnant la meilleure comparaison.

La deuxième étude était basée sur un benchmark, et avait pour but de démontrer la capacité de production de bibliothèques de sections efficaces condensées et homogénéisées pour les études de transitoire de coeur.
Au cours de ce benchmark, deux types d'assemblages ont été étudiés : un assemblage UO_2 et un assemblage MOX, chacun possédant deux enrichissements différents. Le problème étant complètement différent de l'exercice précédent, les mêmes vérifications ont dû être effectuées, en comparaison avec HELIOS, au niveau d'une cellule, et de l'assemblage. Les comparaisons ont montré que l'assemblage MOX est plus susceptible d'être sujet à erreur, la présence d'isotopes lourds causant plus de problèmes. Mais les résultats restent quand même acceptables, et le schéma deux-niveaux a été choisi pour produire les bibliothèques nécessaires, étant le meilleur compromis rapidité/précision.

En fait, la création de ces bibliothèques nécessite de nombreux calculs sur chacun des assemblages, avec des conditions différentes. Trois paramètres ont été étudiés, et pour chacun d'eux, trois valeurs différentes ont été étudiées. Pour cela, un schéma de calcul a été mis en place, réalisant 27 points de calcul à chacune des étapes du calcul d'évolution, et produisant la bibliothèque de sections efficaces. Après conversion dans un format compatible avec le système couplé QUABOX-CUBBOX/ATHLET, les bibliothèques ont été utilisées dans ce code pour effectuer des calculs sur un transitoire de bore. Les résultats ont finalement été comparés aux résultats obtenus avec les bibliothèques produites par HELIOS, et démontrent que DRAGON est très performant.

Ce projet a été l'occasion de montrer les avantages d'utiliser le code de réseau DRAGON. Il est en effet possible avec ce code d'utiliser un large éventail de méthodes pour résoudre l'équation du transport, mais aussi d'utiliser différents types de bibliothèques, venant de différents types d'évaluations. Finalement, un schéma de calcul industriel est habituellement développé sur plusieurs années par toute une équipe, et il est intéressant de voir que, avec DRAGON, il a été possible de développer un schéma de calcul donnant de bons résultats sur une période relativement courte.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF NOTATIONS AND SYMBOLS

ADF: Assembly Discontinuity Factor

BOC: Beginning Of Cycle

EOC: End Of Cycle

CP: Collision Probability

CPU: Central Processing Unit

$ENDF$: Evaluated Nuclear Data File

GWd/t: GigaWatt Days per Tons

GAN: Groupe d'Analyse Nucléaire

HFP: Hot Full Power

IC: Interface Current

$IFBA$: Integral Fuel Burnable Absorber

$JEFF$: Joint Evaluated Fission and Fusion

$JENDL$: Japanese Evaluated Nuclear Data Library

k_{eff}: Effective multiplication factor

k_{inf}: Infinite multiplication constant

GRS: Gesellschaft für Anlagen- und Reaktorsicherheit

MOC: Method Of Characteristics

MOX: Mixed Oxide Fuel

PWR: Pressurized Water Reactor

PWR: Pressurized Water Reactor

Q/C: QUABOX-CUBBOX

UO_2: Uranium Dioxide

$WABA$: Wet Annular Burnable Absorber

LIST OF TABLES

LIST OF APPENDICES

INTRODUCTION

Nowadays, the most prevalent type of reactors in the world is the Pressurized Water Reactor (PWR) type. It is characterized by different types of fuel assemblies, having different dimensions (mostly 16x16, 17x17, or 18x18 fuel rods per assembly), and by its very common moderator and coolant, light water. Usually, the fuel used in those assemblies is made of uranium dioxide (UO_2) or of a mixed oxide fuel (MOX). The neutron flux needed to sustain a nuclear chain reaction in the reactor core dominates the steady-state and transient behavior. To simulate transients and accidents of a nuclear reactor, a thorough knowledge, both in space and time, of the flux evolution is necessary.

Neutronic simulation plays a very important role in the safety analysis of the nuclear reactors, which is an essential element of a safety assessment. Those simulations involves deterministic or probabilistic approaches in order to solve the neutron transport equation on a whole reactor. The problem with simulating the behavior of a whole core is that it is nowadays impossible to do it directly. Different levels of calculation have to be performed, using different approximations, and the neutron flux on a full core will only be known after performing those different levels. Here, the level of interest is the lattice calculation. This step usually takes a lot of computing time, because the more accurate the method used, the longer the calculation time. But it is possible to combine different methods to achieve a calculation scheme being both fast and accurate. The purpose of this project is then the development and the validation of an advanced two-level calculation scheme.

This work was performed with the use of the deterministic lattice code DRAGON (Marleau et al., 2009), developed at the *Institut de Génie Nucléaire* in the *École Polytechnique de Montréal*. The version used is DRAGON Version4, which is part of the Version4 dis-

tribution (Hébert, 2006). The first purpose of this code was the study of the Canada Deuterium Uranium (CANDU) reactors, but can be used on a lot of different types of reactor assemblies.

This project was performed in a research, and industrial environment at the Gesellschaft für Anlagen- und Reaktorsicherheit (GRS), which deals with nuclear reactor safety. The work was there performed on two different exercises in order to validate the development of the two-level scheme.

First will be introduced the theoretical background of a lattice calculation necessary to understand the studies that have been performed.

In the second chapter, a first exercise will be studied, dealing with a 18x18 UO_2 PWR assembly. This exercise is going to be used in order to perform different comparisons on different parameters, the purpose being to develop a two-level scheme, and then to validate it, by using other results obtained with other codes.

Finally, a second study will be done on a 17x17 MOX/UO_2 PWR core transient benchmark, proposed by T. Kozlowski and T. J. Downar (Kozlowski and Downar, 2003). The two-level scheme will be used again, to produce cross section libraries for the GRS coupled code system QUABOX-CUBBOX/ATHLET (Langenbuch and Velkov, 2004). The scheme will be validated by comparing the results to the ones obtained with the HELIOS code (Casal et al., 1991).

CHAPTER 1

TRANSPORT EQUATION AND ELEMENTS OF LATTICE CALCULATION

To lay the foundations of the work we performed, the basics of a lattice calculation have to be introduced. The first step is to describe the particle flux as solution of the transport equation. Due to its complexity, this equation can be solved analytically only for some very simple cases. For realistic systems with complex geometry and detailed energy depedency, the particle flux can only be obtained using numerical methods. An introduction to elements of lattice calculation is therefore necessary. Finally, those calculations have to be performed with a lattice code : in this work, we will use the DRAGON code.

1.1 The transport equation

The transport equation describes the neutron flux distribution. This equation, and the methods used to solve it, will be presented in this section, leading to a common form used in multigroup flux calculations, self-shielding models, and equivalence calculations.

1.1.1 The particle flux

Before introducing the transport equation, the fundamental quantities describing the particle population have to be presented. An approach from statistical mechanics is used. It assumes that each particle is moving in a six-dimensional phase space : three dimensions for its position, and three dimensions for its velocity. A particle is then identified by :

- $\vec{r}$, the position

- $\vec{V_n} = \dfrac{d\vec{r}}{dt}$, the velocity, decomposed as $V_n = \|\vec{V_n}\|$ and $\hat{\Omega} = \dfrac{\vec{V}}{V}$;
- t, the time.

The population density $n(\vec{r}, V_n, \vec{\Omega}, t)$ is used to represent this population of particles, in such a way that $n(\vec{r}, V_n, \vec{\Omega}, t)\ d^3r\ dV_n\ d^2\Omega$ is the number of particles at time t, in the volume element d^3r surrounding point $\vec{r}$, in the velocity element dV_n surrounding V_n, and the solid angle element $d^2\Omega$ surrounding $\vec{\Omega}$. The fundamental quantity, usually used in reactor physics is then the particle flux ϕ defined as :

$$\phi(\vec{r}, V_n, \vec{\Omega}, t) = V_n n(\vec{r}, V_n, \vec{\Omega}, t). \tag{1.1}$$

1.1.2 Presentation of the transport equation

The transport equation is a balance of the neutron population in the volume $d^3r\ dV_n\ d^2\Omega$ surrounding $\{\vec{r}, V_n, \vec{\Omega}\}$ during Δt (Hébert, 2009) :

$$\begin{aligned} \text{Variation of the number of neutrons} = \quad & - \text{Balance of particles leaving the volume } d^3r \\ & - \text{Number of lost neutrons due to collisions} \\ & + \text{Number of new particles created} \end{aligned}$$

where :

- The variation of the number of neutron in $d^3r\ dV_n\ d^2\Omega$ during Δt is :
  $n(\vec{r}, V_n, \vec{\Omega}, t + \Delta t) - n(\vec{r}, V_n, \vec{\Omega}, t)$.
- The balance of particles leaving the volume d^3r during Δt is :
  $\vec{\nabla} \cdot \vec{\Omega}\phi(\vec{r}, V_n, \vec{\Omega}, t)\Delta t = \vec{\Omega} \cdot \vec{\nabla}\phi(\vec{r}, V_n, \vec{\Omega}, t)\Delta t$.
- The number of collisions in the volume $d^3r\ dV_n\ d^2\Omega$ during Δt is :
  $\Sigma(\vec{r}, V_n) \left[\phi(\vec{r}, V_n, \vec{\Omega}, t)\right] \Delta t$, with $\Sigma(\vec{r}, V_n)$ the macroscopic total cross section.

- The number of new particles created in this volume during Δt is : $Q(\vec{r}, V_n, \vec{\Omega}, t)\Delta t$, with $Q(\vec{r}, V_n, \vec{\Omega}, t)$ the source density.

The balance equation of the population of particles in the volume $d^3r\ dV_n\ d^2\Omega$ surrounding $\{\vec{r}, V_n, \vec{\Omega}\}$ is then :

$$\begin{aligned} \frac{n(\vec{r}, V_n, \vec{\Omega}, t+\Delta t) - n(\vec{r}, V_n, \vec{\Omega}, t)}{\Delta t} = & - \vec{\Omega} \cdot \vec{\nabla}\phi(\vec{r}, V_n, \vec{\Omega}, t) \\ & - \Sigma(\vec{r}, V_n) \left[\phi(\vec{r}, V_n, \vec{\Omega}, t)\right] \\ & + Q(\vec{r}, V_n, \vec{\Omega}, t). \end{aligned} \tag{1.2}$$

Taking the limit as $\Delta t \to 0$ leads to the differential form of the transport equation :

$$\frac{1}{V_n}\frac{\partial}{\partial t}\phi(\vec{r}, V_n, \vec{\Omega}, t) + \vec{\Omega}\cdot\vec{\nabla}\phi(\vec{r}, V_n, \vec{\Omega}, t) + \Sigma(\vec{r}, V_n)\phi(\vec{r}, V_n, \vec{\Omega}, t) = Q(\vec{r}, V_n, \vec{\Omega}, t). \tag{1.3}$$

For steady-state conditions, this equation reduces to :

$$\vec{\Omega} \cdot \vec{\nabla}\phi(\vec{r}, V_n, \vec{\Omega}) + \Sigma(\vec{r}, V_n)\phi(\vec{r}, V_n, \vec{\Omega}) = Q(\vec{r}, V_n, \vec{\Omega}). \tag{1.4}$$

Considering the energy $E = \dfrac{mV_n^2}{2}$ as independent variable instead of V_n :

$$\vec{\Omega} \cdot \vec{\nabla}\phi(\vec{r}, E, \vec{\Omega}) + \Sigma(\vec{r}, E)\phi(\vec{r}, E, \vec{\Omega}) = Q(\vec{r}, E, \vec{\Omega}). \tag{1.5}$$

Another form corresponds to an integration of $\vec{\Omega} \cdot \vec{\nabla}\phi$ over the characteristics, a straight line of direction $\vec{\Omega}$ (the particle trajectory), such that the particle position can be parametrized as $\vec{r} + s\,\vec{\Omega}$ where s is the assumed distance from the reference position $\vec{r}$ of the particle on its characteristic. The characteristic form of the equation is then :

$$\frac{d}{ds}\phi(\vec{r} + s\,\vec{\Omega}, E, \vec{\Omega}) + \Sigma(\vec{r} + s\,\vec{\Omega}, E)\phi(\vec{r} + s\,\vec{\Omega}, E, \vec{\Omega}) = Q(\vec{r} + s\,\vec{\Omega}, E, \vec{\Omega}). \tag{1.6}$$

1.1.3 Source density

In lattice calculations, the steady-state source density is used. Assuming that the fission reactions are isotropic in the laboratory frame of reference, we can write:

$$\begin{aligned} Q(\vec{r}, E, \vec{\Omega}) = & \int_{4\pi} d^2\Omega' \int_0^\infty dE'\, \Sigma_s(\vec{r}, E \leftarrow E', \vec{\Omega} \leftarrow \vec{\Omega}')\, \phi(\vec{r}, E', \vec{\Omega}') \\ & + \frac{1}{4\pi k_{eff}} Q^{fiss}(\vec{r}, E) \end{aligned} \tag{1.7}$$

where :

- $\Sigma_s(\vec{r}, E \leftarrow E', \vec{\Omega} \leftarrow \vec{\Omega}')$ is the macroscopic differential scattering cross section from energy E to energy E', and from solid angle $\vec{\Omega}$ to solid angle $\vec{\Omega}'$. This term takes into account diffusion and (n,xn) reactions.

- k_{eff} is the effective multiplication factor. To maintain a steady-state condition, the sum of absorption and leakage rates must be equal to the production rate of neutrons by fission. k_{eff} is then used to adjust the fission source, such that it matches the rate at which the neutrons are lost.

- $Q^{fiss}(\vec{r}, E)$ is the isotropic fission source. It is assumed to be independent of the energy of the incident neutron. It is written :

$$Q^{fiss}(\vec{r}, E) = \sum_{j=1}^{J^{fiss}} \chi_j(E) \int_0^\infty dE'\, \nu\Sigma_{f,j}(\vec{r}, E')\, \phi(\vec{r}, E') \tag{1.8}$$

  where : $\chi_j(E)dE$ is the probability for a neutron, emitted by the fissile nuclide j, to have an energy equal to E (within a dE interval); J^{fiss} is the total number of fissile isotopes; $\nu\Sigma_{f,j}(\vec{r}, E')$ is the number of emitted neutrons per fission times the macroscopic fission cross section of the j^{th} fissile isotope; $\phi(\vec{r}, E)$ is the integrated flux : $\phi(\vec{r}, E) = \int_{4\pi} d^2\Omega' \phi(\vec{r}, E, \vec{\Omega}')$

In isotropic media, the scattering cross section is only a function of the scattering angle, so Eq. (1.7) can be written:

$$\begin{aligned} Q(\vec{r}, E, \vec{\Omega}) = & \frac{1}{2\pi} \int_{4\pi} d^2\Omega' \int_0^\infty dE'\, \Sigma_s(\vec{r}, E \leftarrow E', \vec{\Omega} \cdot \vec{\Omega}')\, \phi(\vec{r}, E', \vec{\Omega}') \\ & + \frac{1}{4\pi k_{eff}}\, Q^{fiss}(\vec{r}, E) \end{aligned} \tag{1.9}$$

It is more convenient to expand the scattering cross section in terms of Legendre polynomials :

$$\Sigma_s(\vec{r}, E \leftarrow E', \vec{\Omega} \cdot \vec{\Omega}') = \sum_{l=0}^{L} \frac{2l+1}{2}\, \Sigma_{s,l}(\vec{r}, E \leftarrow E')\, P_l(\vec{\Omega} \cdot \vec{\Omega}') \tag{1.10}$$

where L is the maximum scattering order after which the series is troncated, and $\Sigma_{s,l}(\vec{r}, E \leftarrow E')$ are the Legendre coefficients of the scattering cross section.

It is then possible to rewrite the scattering source of Eq. (1.9) in terms of the spherical harmonic components of the flux by using the addition theorem of spherical harmonics :

$$\begin{aligned} Q(\vec{r}, E, \vec{\Omega}) = \frac{1}{4\pi} & \left[\int_0^\infty dE' \sum_{l=0}^{L} (2l+1)\, \Sigma_{s,l}(\vec{r}, E \leftarrow E') \sum_{m=-l}^{l} R_l^m(\vec{\Omega})\, \phi_l^m(\vec{r}, E') \right. \\ & \left. + \frac{1}{k_{eff}}\, Q^{fiss}(\vec{r}, E) \right] \end{aligned} \tag{1.11}$$

where

$$\phi_l^m(\vec{r}, E) = \int_{4\pi} d^2\Omega\, R_l^m(\vec{\Omega})\, \phi(\vec{r}, E, \vec{\Omega}) \tag{1.12}$$

Under the assumption of isotropic scattering in the laboratory system, the source density reduces to :

$$Q(\vec{r}, E) = \frac{1}{4\pi} \left[\int_0^\infty dE\, \Sigma_{s,0}(\vec{r}, E \leftarrow E')\, \phi(\vec{r}, E') + \frac{1}{k_{eff}}\, Q^{fiss}(\vec{r}, E) \right] \tag{1.13}$$

1.1.4 Boundary conditions

In order to solve the transport equation, boundary conditions are also needed. Considering a domain V where the particles move, it is surrounded by a boundary ∂V where boundary conditions must be imposed. To each point $\vec{r_S} \in \partial V$, its outward normal $\vec{N}(\vec{r_s})$ can be associated. Finding a solution in V requires the knowledge of the angular flux $\phi(\vec{r_s}, E, \vec{\Omega})$ for $\vec{\Omega} \cdot \vec{N}(\vec{r_s}) < 0$ (incoming flux). Many methods link the unknown incoming flux to the known outgoing one :

- A general relation is the albedo boundary condition :

$$\phi(\vec{r_s}, E, \vec{\Omega}) = \beta\, \phi(\vec{r_s}, E, \vec{\Omega'}) \quad \text{with } \vec{\Omega} \cdot \vec{N}(\vec{r_s}) < 0 \tag{1.14}$$

  where Ω' is the direction of the outgoing particle, and β can take any value between 0 and 1. $\beta = 0$ is a vacuum boundary condition, and $\beta = 1$ is a reflective one. Specular reflection corresponds to the special case where :

$$\vec{\Omega} \cdot \vec{N}(\vec{r_s}) = -\vec{\Omega'} \cdot \vec{N}(\vec{r_s}) \quad \text{and} \quad (\vec{\Omega} \times \vec{\Omega'}) \cdot \vec{N}(\vec{r_s}) = 0 \tag{1.15}$$

- The white boundary condition is a reflective condition where all particles leaving V return back in V with an isotropic angular distribution :

$$\phi(\vec{r_s}, E, \vec{\Omega}) = \frac{1}{\pi} \int_{\vec{\Omega'} \cdot \vec{N}(\vec{r_s}) > 0} d^2\Omega' \, [\, \vec{\Omega'} \cdot \vec{N}(\vec{r_s}) \,]\, \phi(\vec{r_s}, E, \vec{\Omega'}) \quad \text{with } \vec{\Omega} \cdot \vec{N}(\vec{r_s}) < 0 \tag{1.16}$$

- Finally, the periodic boundary condition represents the equality between the flux on one boundary, and the flux on another parallel boundary in a periodic lattice :

$$\phi(\vec{r_s}, E, \vec{\Omega}) = \phi(\vec{r_s} + \Delta\vec{r}, E, \vec{\Omega}) \quad \text{where } \Delta\vec{r} \text{ is the lattice pitch.} \tag{1.17}$$

1.1.5 Multigroup approach

To ensure that a numerical resolution of the transport equation can be obtained, a multigroup discretization in energy is used. It consists of dividing the energy continuum in a set of G energy groups, inside which the neutrons are assumed to behave as one-speed particles, and condensing all the energy dependent quantities over these groups. Alternatively, the lethargy variable $u = ln(E_0/E)$ can be used, such as :

$$W_g = \{u; u_{g-1} \leq u \leq u_g\} = \{E; E_g \leq E \leq E_{g-1}\}\,; \quad g = 1, G \tag{1.18}$$

where $u_g = ln(E_0/E_g)$ with E_0 the reference energy, corresponding to the maximum energy of neutrons in a reactor, and $u_0 = 0$. The energy spectrum is divided into G groups $]E_g, E_{g-1}[$, with $g \in [1, G]$, and the differential form of the transport equation in group g is written :

$$\vec{\Omega} \cdot \vec{\nabla}\phi_g(\vec{r}, \vec{\Omega}) + \Sigma_g(\vec{r})\,\phi_g(\vec{r}, \vec{\Omega}) = Q_g(\vec{r}, \vec{\Omega}). \tag{1.19}$$

$$\text{with} \quad Q_g(\vec{r}, \vec{\Omega}) = \frac{1}{4\pi} \left[\sum_{h=1}^{G} \sum_{l=0}^{L} (2l+1)\,\Sigma_{s,l,g\leftarrow h}(\vec{r}) \sum_{m=-l}^{l} R_l^m(\vec{\Omega})\,\phi_{l,h}^m(\vec{r}) + \frac{1}{k_{eff}} \sum_{j=1}^{J^{fiss}} \chi_{j,g} \sum_{h=1}^{G} \nu\Sigma_{f,j,h}(\vec{r})\,\phi_h(\vec{r}) \right] \tag{1.20}$$

The characteristic form of this equation is written :

$$\frac{d}{ds}\phi_g(\vec{r} + s\,\vec{\Omega}, \vec{\Omega}) + \Sigma_g(\vec{r} + s\,\vec{\Omega})\,\phi_g(\vec{r} + s\,\vec{\Omega}, \vec{\Omega}) = Q_g(\vec{r} + s\,\vec{\Omega}, \vec{\Omega}). \tag{1.21}$$

Finally, given the optical path $\tau_g(s) = \int_0^s ds'\Sigma_g(\vec{r} + s\,\vec{\Omega})$, the integral form of the transport equation in an infinite domain is written :

$$\phi_g(\vec{r}, \vec{\Omega}) = \int_0^\infty ds\, e^{-\tau_g(s)}\, Q_g(\vec{r} - s\,\vec{\Omega}, \vec{\Omega}). \tag{1.22}$$

1.1.6 Solution methods

The transport equation has now to be solved. Different methods can be used, divided in two different classes : the stochastic and the deterministic solution techniques. The deterministic approach is based on the application of numerical analysis techniques to the transport equation in its different forms. Four methods are prominently used : the collision probability method, based on the integral form of the transport equation (Eq. (1.22)); the method of characteristics, based on the characteristic form (Eq. (1.21)); the discrete ordinates method, and the spherical harmonics method, both based on the differential form (Eq. (1.20)). In the frame of this work, only the collision probabilities, and the characteristics methods were used. The stochastic approach is the most accurate, but also the most expensive, based on so-called Monte-Carlo methods.

1.1.6.1 The collision probability method

First, the collision probability method is a result of the spatial discretization of the integral and multigroup form of the transport equation. Such a discretization can be performed over either an infinite domain, or a finite domain surrounded by a surface with boundary conditions. We will here consider an infinite lattice of unit cells or assemblies. Integrating Eq. (1.22) over the solid angle and introducing the change of variable $\vec{r'} = \vec{r} - s\vec{\Omega}$, leads to a new form of the transport equation :

$$\begin{aligned} \phi_g(\vec{r}) = \int_{4\pi} d^2\Omega\, \phi_g(\vec{r},\vec{\Omega}) &= \frac{1}{4\pi}\int_{4\pi} d^2\Omega \int_0^{\infty} ds\, e^{-\tau_g(s)}\, Q_g(\vec{r} - s\,\vec{\Omega}) \\ &= \frac{1}{4\pi}\int_{\infty} d^3r'\, \frac{e^{-\tau_g(s)}}{s^2}\, Q_g(\vec{r'}) \end{aligned} \tag{1.23}$$

A partition of the unit cell or assembly into regions V_i is performed. The infinite set of regions V_i belonging to all the cells or assemblies in the lattice will be referred to as V_i^{∞}.

The sources of secondary neutrons is supposed to be uniform and equal to $Q_{i,j}$ for each region V_i. Multiplying Eq. (1.23) by $\Sigma_g(\vec{r})$ and integrating it over each region V_i leads to the equation :

$$\int_{V_j} d^3r\, \Sigma_g(\vec{r})\, \phi_g(\vec{r}) = \frac{1}{4\pi} \int_{V_j} d^3r\, \Sigma_g(\vec{r}) \sum_i Q_{i,g} \int_{V_i^\infty} d^3r'\, \frac{e^{-\tau_g(s)}}{s^2} \tag{1.24}$$

$$\text{with} \quad Q_{i,j} = \sum_h \Sigma_{s0,i,g\leftarrow h}\, \phi_{i,h} + \frac{1}{k_{eff}} \sum_{j=1}^{Jfiss} \chi_{j,g} \sum_{h=1}^{G} \nu\Sigma_{f,j,h}\, \phi_{i,h} \tag{1.25}$$

Finally, Eq. (1.24) can be simplified to :

$$\phi_{j,g} = \frac{1}{V_j}\, \Sigma_{j,g} \sum_i Q_{i,g}\, V_i\, P_{ij,g} \tag{1.26}$$

where :

$$\phi_{j,g} = \frac{1}{V_j} \int_{V_j} d^3r\, \phi_g(\vec{r}) \tag{1.27}$$

$$\Sigma_{j,g} = \frac{1}{V_j\, \phi_{j,g}} \int_{V_j} d^3r\, \Sigma_g(\vec{r})\, \phi_g(\vec{r}) \tag{1.28}$$

$$P_{ij,g} = \frac{1}{4\pi\, V_i} \int_{V_i^\infty} d^3r' \int_{V_j} d^3r\, \Sigma_g(\vec{r})\, \frac{e^{-\tau_g(s)}}{s^2} \tag{1.29}$$

$P_{ij,g}$ is the probability for a neutron born uniformly and isotropically in any region V_i of the lattice to undergo its first collision in the region V_j of a unit cell or assembly. It is called the collision probability (CP). In general, the total cross section is constant and equal to $\Sigma_{j,g}$ in the region V_j, so the reduced CPs can be used :

$$p_{ij,g} = \frac{P_{ij,g}}{\Sigma_{j,g}} = \frac{1}{4\pi\, V_i} \int_{V_i^\infty} d^3r' \int_{V_j} d^3r\, \frac{e^{-\tau_g(s)}}{s^2} \tag{1.30}$$

Reduced CPs have two interesting properties :

- reciprocity : $p_{ij,g} V_i = p_{ji,g} V_j$ (1.31)
- conservation : $\sum_j p_{ij,g}\, \Sigma_{j,g} = 1\ ;\ \forall i$. (1.32)

Eq. (1.26) can be simplified, using the reduced CPs and the reciprocity property :

$$\phi_{j,g} = \sum_j Q_{j,g}\, p_{ij,g} \tag{1.33}$$

The next step is the numerical evaluation of the CPs, usually performed in two steps :

- First, a tracking process is applied over the lattice, taking into account a sufficient number of neutron trajectories. The angular domain is divided in a series of tracks of direction $\vec{\Omega}_m$ and weight ω_m in order to have $\int_0^{4\pi} d^2\Omega = \sum_m \omega_m \vec{\Omega}_m = 4\pi$. For each of these directions, a normal plane is chosen, and divided in a uniform grid generating integration points $\vec{p}_{m,n}$ weighting $\Pi_{m,n}$. The tracking is then a discretization of the whole domain in integration lines and points. In order to compute the optical paths, the intersections between the tracking and the regions of the domain have to be identified.

- To compute the CPs, a numerical integration is done, using tracking information, and knowledge of the macroscopic total cross sections in each region. Eq. (1.30) can then be rewritten :

$$p_{ij,g} = \frac{1}{\Sigma_i \Sigma_j V_i} \quad \sum_m \omega_m \sum_n \Pi_{m,n} \sum_k \delta_{i,V_k} \sum_h \delta_{j,V_h} \quad [1 - e^{-\Sigma_i L_k}] e^{-\tau_{k,h}} [1 - e^{-\Sigma_j L_h}] \tag{1.34}$$

  where L_k is the distance travelled in the region V_k for a neutron born at the point $\vec{p}_{m,n}$ and traveling in direction $\vec{\Omega_m}$.

Another method exists, faster in CPU time, called the interface current (IC) method (Sanchez and McCormick, 1982). Here, for an assembly of cells, the CP matrices are computed for each uncoupled cell. The detailed flux can be rebuilt from the knowledge of the interface currents surrounding each cell. As done before, the CP matrices are :

- p_{ij} : the reduced CP for a neutron born uniformly and isotropically in the region i, to have its first collision in the region j, without leaving the cell.
- $p^{(\rho)}_{S_\alpha j}$: the reduced CP for a neutron entering from the surface S_α uniformly, with an angular distribution $\psi_\rho(\vec{\Omega}, \vec{N}^-)$ to have its first collision in the region j without leaving the cell.
- $P^{(\nu)}_{iS_\beta}$: escape probability for a neutron born uniformly and isotropically in the region i, to leave the cell by the surface S_β with an angular distribution $\psi_\nu(\vec{\Omega}, \vec{N}^+)$.
- $P^{(\rho\nu)}_{S_\alpha S_\beta}$: transmission probability for a neutron entering from the surface S_α with an angular distribution $\psi_\rho(\vec{\Omega}, \vec{N}^-)$, to leave the cell by the surface S_β with an angular distribution $\psi_\nu(\vec{\Omega}, \vec{N}^+)$.

where the base functions are chosen to satisfy the orthogonality condition :

$$\int_{\vec{\Omega}\cdot\vec{N}>0} d^2\Omega \, (\vec{\Omega}\cdot\vec{N}) \, \psi_\nu((\vec{\Omega}\cdot\vec{N})) \, \psi_\rho((\vec{\Omega}\cdot\vec{N})) = \pi\delta_{\nu\rho} \tag{1.35}$$

with $\delta_{\nu\rho}$ the delta Kronecker function.

This method leads to the same equations as Eqs. (1.25) and (1.33), but on a reduced domain and different CP (approximations to the exact CP that may suffer from the refraction effects), accelerating the calculation time.

With the CPs known, the integrated flux can be computed iteratively, using Eqs. (1.25) and (1.33). This set of equations can be written in matrix form :

$$\vec{\Phi}_g = \mathbf{W}_g \, \vec{Q}^*_g \tag{1.36}$$

where :

$$\vec{\Phi}_g = \{\phi_{i,g} \ ; \ \forall i\} \tag{1.37}$$

$$\vec{Q}^*_g = \left\{ \sum_{h \neq g} \Sigma_{s0,i,g \leftarrow h} \phi_{i,h} + \frac{1}{k_{eff}} Q^{fiss}_{i,g} \ ; \ \forall i \right\} \tag{1.38}$$

$$\mathbf{W}_g = \left[\mathbf{I} - \mathbf{P}_g \, \mathbf{S}_{s0,g \leftarrow g} \right]^{-1} \mathbf{P}_g \tag{1.39}$$

$$\mathbf{P}_g = \{p_{ij,g} \ ; \ \forall i \text{ and } j\} \tag{1.40}$$

$$\mathbf{S}_{s0,g \leftarrow g} = diag\{\Sigma_{s0,i,g \leftarrow g} \ ; \ \forall i\} \tag{1.41}$$

Two iterative processes are then imposed to this flux solution : an inner iteration process performed on the diffusion up-scattering sources until a converged multigroup thermal flux is obtained; an outer (or power) iteration process performed until convergence of the k_{eff} is achieved, using $\vec{\Phi}^{(k+1)}_g = \mathbf{W}_g \, \vec{Q}^{*(k)}_g$.

This CP method is limited by the number n of regions, because the inversion of $n \times n$ is a non-linear process, and increasing the number of regions increases considerably the CPU time and memory needed. n is usually chosen lower than 1000 to 5000 depending on the number of energy groups.

1.1.6.2 The method of characteristics

Because full assembly geometries shall be modelled in more and more detailed, the use of another method is necessary. The method of characteristics (MOC) (Askew, 1972) is a discretization of the characteristic form of the transport equation, based on an iterative calculation of the particle flux by solving this equation over tracks crossing the complete geometry. This method leads to identical results as the CP method (Wu et Roy, 2003), using the same process to generate the tracking.

We consider our domain divided in N regions. The track is represented by a set of integration lines called characteristics. The intersection between this line and the domain is a trajectory $\vec{T} = \vec{T}(\vec{\Omega}, \vec{p})$ defined by its orientation $\vec{\Omega}$, its weighting ω_T, and its starting point $\vec{p}$. This trajectory goes through K regions N_k, creating intersection points with the different region's borders : $\vec{r}_{k+1} = \vec{r}_k + L_k \vec{\Omega}$. With this notation, $\vec{r}_1$ and $\vec{r}_{K+1}$ are respectively the domain entry point, and exit point. The angular flux at each one of these points is given by :

$$\phi_g^k(\vec{T}) = \phi_g(\vec{r}_k, \vec{\Omega}) \quad \text{with } k \in [1, K]. \tag{1.42}$$

Considering a constant source ($\forall s \in [0, L_k],\ Q_g(\vec{r} + s\,\vec{\Omega}, \vec{\Omega}) = Q_g^{N_k}(\vec{\Omega})$: flat source approximation), and a constant total cross section ($\forall s \in [0, L_k],\ \Sigma_g(\vec{r} + s\,\vec{\Omega}) = \Sigma_g^{N_k}$) inside each region, and introducing the optical path $\tau_k = \Sigma_g^{N_k} L_k$, Eq. (1.21) can be integrated to obtain a relation between the incoming and outgoing angular flux :

$$\phi_g^{k+1}(\vec{T}) = \phi_g^k(\vec{T})\, e^{-\tau_k} + \frac{1 - e^{-\tau_k}}{\Sigma_g^{N_k}} Q_g^{N_k}(\vec{T}) \tag{1.43}$$

Finally, the average scalar flux is obtained by integrating Eq. (1.43) over each segment, and then over each angle :

$$\phi_{j,g} = \frac{Q_{j,g}}{\Sigma_j} - \frac{1}{\Sigma_{j,g} V_j} \sum_T \omega_T \sum_k \delta_j^{N_k} (\phi_g^{k+1} - \phi_g^k). \tag{1.44}$$

This integration is possible with the knowledge of the incoming angular flux on the boundary of the domain, for every integration line, given by the boundary conditions.

The MOC method has some advantages over the CP method. At the end, the system of equations to be solved has a dimension of $N + M$ instead of N^2 (N is the number of regions, and M the number of surfaces). This is to the expense of the system having to be solved iteratively. Large geometric domains can then be treated with this method,

while the CPs are restrained to a maximum number of regions (dependent on the memory available).

Moreover, the extension to linearly anisotropic scattering requires much less effort than in the case of the CP method. However, while the CP method only has to read the tracking information once, if the geometric and nuclear properties do not change, the MOC method has to read it for each internal iteration. Acceleration methods then have to be used, for the MOC method to be competitive (Le Tellier, 2006).

1.1.6.3 Stochastic methods

Another approach used to solve the transport equation is the stochastic approach, using the most accurate, but also the most expensive technique : the Monte-Carlo method. It is very different from the deterministic methods. Instead of solving the Boltzmann transport equation for averaged particle behavior (like it was done before), many millions of individual particles histories are simulated, using a multigroup or a continuous energy representation of the cross sections, and an accurate representation of the domain to take into account their interactions.

It is said to be stochastic because of the use of a random number generator, a function returning a random number in the interval $0 \leq x \leq 1$, used to simulate the statistical behavior of the interactions. This method is said to be exact, as long as the geometry and the interactions are correctly simulated, and the number of particule histories is sufficient. Because of this last point, the calculation times can become very long. But because this method is numerically exact, its calculations are usually used as reference calculations.

Another advantage is that it is particularly useful for some complex problems that cannot be modeled by codes using deterministic methods. It also gives the standard deviation values of the quantities computed, giving an indication of their statistical accuracy.

1.2 Elements of lattice calculation

After the presentation of the transport equation in a theoretical context, the next step is to perform a lattice calculation using a lattice code, to compute few-group cross sections for typical reactor structures (a pin cell or an assembly) as a function of the local operating conditions. It is called *lattice* code because of the calculation geometry repeating itself to infinity. These codes are based on a consistent multigroup discretization of the neutron energies. This kind of code is composed of different components, following the data flow presented in Figure (1.1), which are presented in this part.

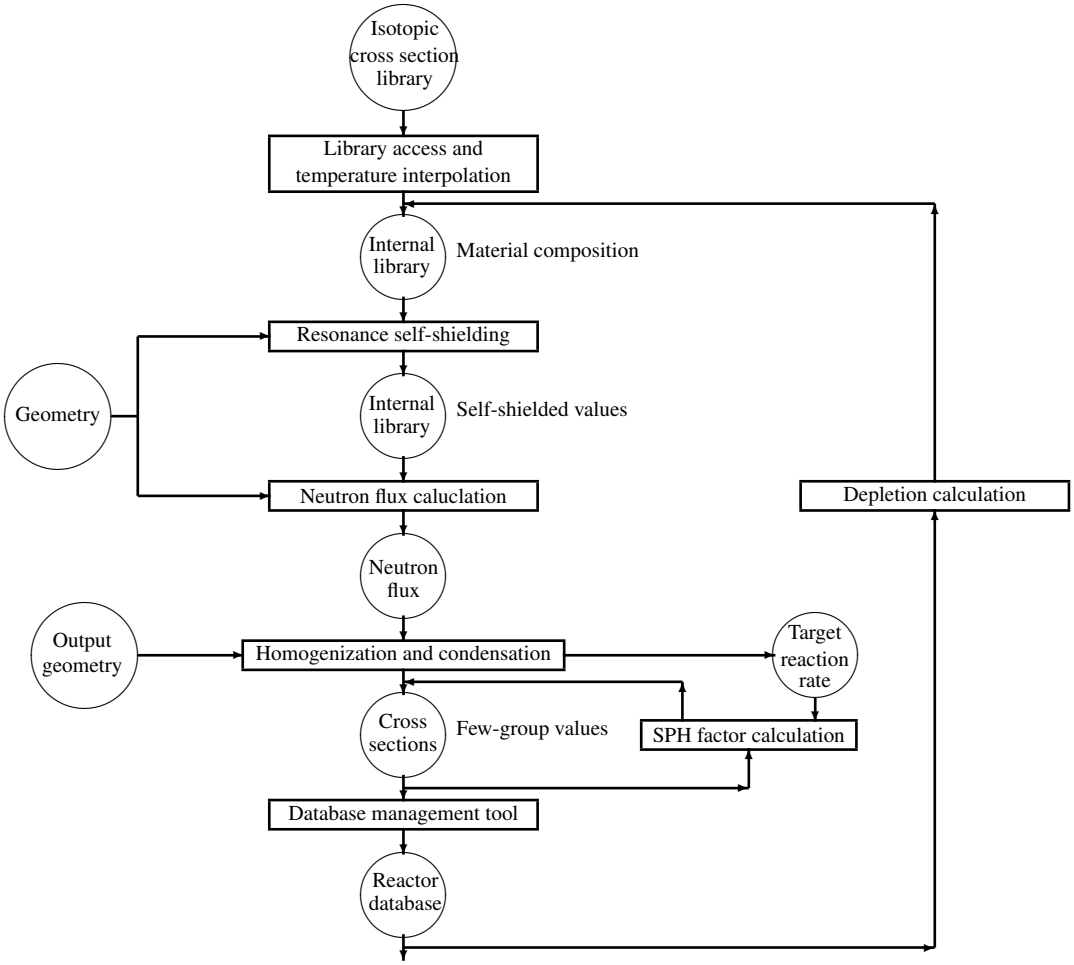


Figure 1.1 Data flow for lattice calculations

1.2.1 Nuclear data libraries

First of all, lattice calculations use nuclear data libraries as data basis, coming from the processing of the information contained in different types of evaluation files.

Experimental measurements on accelerators (ORELA facility at Oak Ridge, USA, or GELINA facility at Geel, Belgium for example) and/or estimated values from nuclear physics models are the source of information for these evaluations. The nuclear data describe properties of atomic nuclei, and the fundamental physical relationships governing their interactions (for example : cross sections, half-lives, decay modes and decay radiation properties, γ-rays from radionuclides, ...).

All this information is then written in a specific format into an evaluation file. These evaluation files can be of different format because they are produced by different evaluation working groups all around the world (ENDF for the USA, JEF for Europe, JENDL for Japan, BROND for Russia, ...). They can be of different type, containing an arbitrary number of nuclear data sets for each isotope, or only one recommended evaluation made of all the nuclear reactions for each isotope.

Finally, this data is fed to a cross section processing code, such as NJOY (MacFarlane and Muir, 2000), to produce the isotopic cross section library used by the lattice code. This process can create a multigroup or continuous library, specifically formatted for the lattice code in use (Hébert and Karthikeyan, 2005). The format used by the DRAGON code is the DRAGLIB format, but DRAGON can also use libraries in the MATXS (MacFarlane, 1984), WIMS-AECL (Donnelly, 1986), WIMS-D (Askew et al., 1966; IAEA-Nuclear Data Services, 2005), and APOLLO (Hoffman et al., 1973) formats.

The library is read by the lattice code, which recovers the isotopic data that will be used in the calculation. The nuclear data is interpolated over absolute temperature T, assuming that the cross sections vary as $\sqrt{T}$ between the given values. The code also identifies self-shielded and depleting mixtures and compute the macroscopic cross section associated with each mixture.

1.2.2 Resonance self-shielding

For lattice calculations performed in a multigroup context, all the quantities are considered to be constant in each energy group. However for some energy domains, many nuclides possesses resonances, and because the number of groups is rather small (between 50 and 300), a model is then required to average (self-shield) the complex resonance structure of cross sections over those coarse energy groups. In fact, in those resonant regions, the flux is very low. Neglecting this effect, a flux-weighted calculation using averaged cross sections leads to an overestimation of the corresponding reaction rates : this is what is called *resonance self-shielding*.

The self-shielding process consist in the calculation of estimated average reaction rates and average fluxes for each resonant isotope and for each energy group containing resonances in order to obtain self-shielded cross sections.
The purpose of self-shielding is to evaluate the microscopic self-shielded cross section $\tilde{\sigma}_{\rho,g}$ for any reaction ρ in the coarse energy group g, defined as (using the lethargy u) :

$$\tilde{\sigma}_{\rho,g} = \mu_g \frac{\int_{u_{g-1}}^{u_g} du\, \sigma_\rho(u)\, \phi(u)}{\int_{u_g-1}^{u_g} du\, \phi(u)} \tag{1.45}$$

where u_{g-1} and u_g are the lethargy limits, μ_g is the *superhomogénéisation* (SPH) factor obtained from the multigroup equivalence procedure, $\phi(u)$ is the average neutron flux where the cross section is defined, and $\sigma_\rho(u)$ is the microscopic cross section for the reaction ρ.

The problem here is that the flux is not known, its calculation requiring the self-shielded cross-sections. Some additional approximations are then required. Two different models can then be used, one based on the equivalence in dilution, the other based on a subgroup approach.

1.2.2.1 Model based on the equivalence in dilution

The first class of resonant self-shielding models is based on a two-term approximation of fuel-to-fuel CPs, leading to the calculation of averaged cross sections which are used to interpolate pre-tabulated resonance integrals. It is called the generalized Stamm'ler method (Hébert and Marleau, 1991). Only one resonant isotope of density N^* is supposed to be present in the region i. Using the CP approach, the flux in this region can be written :

$$\phi_i(u) = \sum_{j=1}^{I} p_{ij}(u) \left[R_j^+\{\phi_j(u)\} + R_j^*\{\phi_j(u)\}\right] \quad ; \quad i = 1, I \tag{1.46}$$

where :

- $R_j^+\{\phi_j(u)\} = \int_0^\infty du'\, \Sigma_{s,j}^+(u \leftarrow u')\phi_j(u')$ is the slowing down operator in region j for nuclear reactions with light isotopes.
- $R_j^*\{\phi_j(u)\} = \int_0^\infty du'\, \Sigma_{s,j}^*(u \leftarrow u')\phi_j(u')$ is the slowing down operator in region j for nuclear reactions with a single heavy isotope.

This equation is then simplified by the use of three approximations proposed by Livolant-Jeanpierre (Livolant and Jeanpierre, 1974; Stammler and Abbate, 1983) :

- First, the neutron flux is factorized into the product of a resonant fine structure function $\varphi_i(u)$ with a regular distribution in lethargy $\psi_i(u)$, called macroscopic flux, and defined as :

  $$\psi_j(u) = \frac{R_j^+\{\phi_j(u)\}}{\Sigma_j^+(u)}. \tag{1.47}$$

  This hypothesis leads to a new form of Eq. (1.45) using the resonance integral $I_{\rho,g}$ and the averaged fine-structure function $\bar{\varphi}_g$ in group g and region i :

  $$\bar{\sigma}_{\rho,i,g} = \mu_{i,g} \frac{\frac{1}{U_g}\int_{u_{g-1}}^{u_g} du\, \sigma_{\rho,i}(u)\, \phi_i(u)}{\frac{1}{\delta u_g}\int_{u_{g-1}}^{u_g} du\, \phi_i(u)} = \mu_{i,g} \frac{I_{\rho,i,g}}{\bar{\varphi}_{i,g}} \tag{1.48}$$

- Secondly, the slowing down operator for the heavy isotope is supposed to act over a short lethargy range only, resulting in :

$$R_j^+\{\phi_j(u)\} = \psi_j(u)\, R_j^+\{\varphi_j(u)\}. \tag{1.49}$$

- Finally, the distribution $\psi_i(u)$ is considered to have a flat value across the domain so that Eq. (1.46) reduces to :

$$\varphi_i(u) = \sum_{j=1}^{I} p_{ij}(u)\, [\Sigma_{s,j}^+(u) + R_j^*\{\varphi_j(u)\}] \quad ; \quad i = 1, I. \tag{1.50}$$

All the resonant regions are then merged into a single resonant region denoted as x, even if they are unconnected, giving the fuel-to-fuel collision probability :

$$p_{xx}(u) = \frac{\sum\limits_{i\in G_x} V_i \sum\limits_{j\in G_x} p_{ij}(u)\Sigma_j(u)}{\sum\limits_{i\in G_x} V_i\Sigma_i(u)} \tag{1.51}$$

where G_x is the set of indices (i, j) belonging to the resonant region x.

An approximation considered by Stamm'ler is then considered, replacing p_{xx} by a rational development of N terms represented as :

$$p_{xx}(u) = \sum_{n=1}^{N} \frac{\alpha_n(g)}{\Sigma_x^*(u) + \Sigma_{e,n}(g)} \quad ; \quad u_{g-1} \leq u \leq u_g. \tag{1.52}$$

The calculation of the coefficients $\alpha_n(g)$ and $\Sigma_{e,n}(g)$ will not be detailed here.

Additional approximations on the slowing down operator can also be considered. For example (Hébert, 2009) :

- assuming that no resonance is present in the lethargy interval $u - \epsilon < u' \leq u$

- the narrow resonance model (NR), considering that the resonances are narrow with respect to the maximum lethargy gain ϵ, and are isolated
- the wide resonance model (WR), considering that the resonances are large with respect to the maximum lethargy gain ϵ
- the statistical resonance model (ST), assuming that the resonances are narrow with respect to the maximum lethargy gain ϵ, and are numerous
- the Goldstein-Cohen approximation (Goldstein and Cohen, 1962), assuming that the resonances of the resonant isotope are intermediate and that the corresponding slowing-down operator can be represented by a linear combination of a NR and a IMNR (infinite mass narrow resonance) model.

All these models lead to an equation giving the fine-structure function, used to evaluate the resonance integral using Eq. (1.48).

The averaged fine structure in group g is finally calculated using the neutron conservation equation in a homogeneous domain :

$$\bar{\varphi}_g = 1 - \frac{1}{\bar{\Sigma}_{e,g}} \left[I_g - \sum_h \frac{U_h}{\delta u_g} I_s(g \leftarrow h) \right] \tag{1.53}$$

where I_g is the effective total resonance integral, and $I_s(g \leftarrow h)$ is the effective transfer resonance integral from group h to group g. This approach may lead to some errors, so Livolant and Jeanpierre proposed to replace the cross section $\bar{\Sigma}_{e,g}$ by the leakage function $E(\Sigma^*_{x,g}) = \dfrac{1}{p_{xx}\Sigma^*_{x,g}} - \Sigma^*_{x,g}$. This is called the Livolant-Jeanpierre normalization.

Improvements were made to the generalized Stamm'ler method to achieve better accuracy (using the Riemann integration method) and to represent distributed self-shielding effects in a fuel rod or across a fuel bundle (using the Norheim model) (Hébert, 2004).

1.2.2.2 Model based on a subgroup approach

The second class of resonance self-shielding models is based on a subgroup equation with physical probability tables, obtained by fitting tables of dilution-dependent cross sections (Hébert, 2009). It is called the statistical subgroup model, and is similar to the self-shielding model used in the WIMS-8 and HELIOS codes. Other improved models can be used (Ribon extended model) (Hébert, 2005), but are not part of this work.

The physical probability table corresponding to the total cross section in group g is computed such as to match the numerical integration results with the tabulated values for specific values of the microscopic dilution cross section σ_e to a given accuracy :

$$\bar{\sigma}(\sigma_e) = \frac{\left\langle \frac{\sigma}{\sigma+\sigma_e} \right\rangle_g}{\left\langle \frac{1}{\sigma+\sigma_e} \right\rangle_g} = \frac{\sum_{k=1}^{K} \frac{\omega_k \sigma_k}{\sigma_k+\sigma_e}}{\sum_{k=1}^{K} \frac{\omega_k}{\sigma_k+\sigma_e}} \tag{1.54}$$

with the infinite dilution microscopic cross section defined as :

$$\bar{\sigma}(\infty) = \frac{1}{\Delta u_g} \int_{u_{g-1}}^{u_g} du\, \sigma(u) = \sum_{k=1}^{K} \omega_k\, \sigma_k \tag{1.55}$$

The determination of the probability table in Eq. (1.54) is a curve fitting problem, which can be carried out as a root-mean-square (RMS) Padé approximation. The creation of these physical probability tables is outlined in (Hébert, 2005). We will here continue to proceed with a CP formalism, but the use of another solution technique of the transport equation is possible.

The probability tables are used to solve the Eq. (1.50) where the slowing down operator is given by using a ST-WR model :

$$R_i^*\{\varphi_i(u)\} = N_i^*\, r_i\{\varphi_i(u)\} = N_i^*\left[\lambda_g \langle \sigma_s^*(u)\,\varphi_i(u)\rangle_g + (1-\lambda_g)\sigma_s^*(u)\,\varphi_i(u)\right] \quad (1.56)$$

where :

$$\langle \sigma_s^*(u)\,\varphi_i(u)\rangle_g = \frac{1}{\Delta u_g}\int_{u_{g-1}}^{u_g} du\,\sigma_s^*(u)\varphi_i(u) \quad (1.57)$$

and where N_i^* is the number density of the resonant isotope in region i, and λ_g is the Goldstein-Cohen parameter of the resonant isotope in group g, set between 0 and 1. This approach is not capable of representing mutual self-shielding effects.

In the subgroup k and in the region i, Eq. (1.54) simplifies then to :

$$\varphi_i(u) = \sum_{j=1}^{I} p_{ij,k}(u)\left\{\Sigma_{s,j}^{+} + N_j^*\left[\lambda_g \sum_{l=1}^{K}\omega_l\,\sigma_{s,j,l}^*\,\varphi_{j,l} + (1-\lambda_g)\sigma_{s,j,k}^*\,\varphi_{j,k}\right]\right\} \quad (1.58)$$

where $\sigma_{s,j,l}^*$ is the microscopic P_0 scattering cross section of the resonant isotope in subgroup l, and $p_{ij,k}(u)$ is a component of the collision probability matrix, computed using the cross sections of the k-th subgroup.

Finally, the integrated flux $\langle\varphi_i\rangle_g$ and the reaction rate $\langle\sigma_{\rho,i}\varphi_i\rangle_g$ for reaction ρ in region i are computed as :

$$\langle\varphi_i\rangle_g = \sum_{k=1}^{K}\omega_k\varphi_{i,k} \quad \text{and} \quad \langle\sigma_{\rho,i}\varphi_i\rangle_g = \sum_{k=1}^{K}\omega_k\sigma_{\rho,i,k}\varphi_{i,k} \quad (1.59)$$

As for the Generalized Stamm'ler method, the self-shielded cross sections cannot be used directly in a coarse group calculation because the reaction rates are not conserved. A multigroup equivalence procedure is then performed, using SPH corrective factors $\mu_{i,g}$ for each region and coarse energy group, as presented at the beginning of this section, to determine equivalent cross sections.

1.2.3 Neutron leakage model

After self-shielding of cross sections, the main flux calculation can be performed, using one of the deterministic methods presented in the first section. But to treat elementary cells or assemblies in a finite reactor, a leakage model is required, particularly when the elementary calculation is performed in two dimensions, and/or reflective or periodic boundary conditions are used. Any leakage rate not taken into account by an explicit boundary condition must be represented by means of the leakage model. These leakage rates are computed using a homogeneous or heterogeneous B_n calculation (Petrovic and Benoist, 1996).

In a lattice calculation, the exact operating conditions and materials surrounding the unit cell or assembly are not known. The best that can be done, without further information, is to assume that all the surrounding cells or assemblies are identical to the one considered, and to adjust the neutron leakage in each group g to have $k_{eff} = 1$, using the following strategy :

1. The flux calculation inside the unit cell or assembly will be performed under closed conditions, using an infinite medium or a finite domain closed with reflective or periodic boundary conditions.

2. The condition $k_{eff} = 1$ is then enforced using a leakage model, usually performed with a fundamental mode approximation. This approximation consists in representing the neutron flux as the product of a macroscopic distribution in space $\psi(\vec{r})$ with a homogeneous or periodic fundamental flux $\varphi(\vec{r}, E, \vec{\Omega})$:

$$\phi(\vec{r}, E, \vec{\Omega}) = \psi(\vec{r})\, \varphi(\vec{r}, E, \vec{\Omega}) \tag{1.60}$$

3. In the case of a periodic lattice of unit cells or assemblies, the distribution $\psi(\vec{r})$ is assumed to be a property of the whole reactor, and to be solution of the Laplace equation :

$$\nabla^2 \psi(\vec{r}) + B^2 \psi(\vec{r}) = 0 \tag{1.61}$$

where the buckling B^2 is a real number used to obtain $k_{eff} = 1$ by adjusting the curvature of $\psi(\vec{r})$.

Without the knowledge of the complete reactor geometry, a generic solution of Eq. (1.61) is used :

$$\psi(\vec{r}) = \psi_0 \, e^{i\vec{B}\cdot\vec{r}} \tag{1.62}$$

where $\vec{B}$ is chosen to have $B^2 = \vec{B} \cdot \vec{B}$, so that $\phi(\vec{r}, E, \vec{\Omega}) = \varphi(\vec{r}, E, \vec{\Omega}) \, e^{i\vec{B}\cdot\vec{r}}$ with $\varphi(\vec{r}, E, \vec{\Omega})$ a complex quantity.

The leakage rates are first assumed to be computed in a completely homogenized unit cell or assembly (the collision rates being computed in heterogeneous representation). This allows to neglect the dependence of φ on the spatial coordinates, and to rewrite the transport equation Eq. (1.5) for the case of a finite and homogeneous geometry :

$$\begin{aligned} [\Sigma(E) + i\,\vec{B}\cdot\vec{\Omega}]\,\varphi(E,\vec{\Omega}) = & \int_{4\pi} d^2\Omega' \int_0^\infty dE'\,\Sigma_s(E \leftarrow E', \vec{\Omega} \leftarrow \vec{\Omega}')\,\varphi(E',\vec{\Omega}') \\ & + \frac{\chi(E)}{4\pi k_{eff}} \int_0^\infty dE'\,\nu\Sigma_f(E')\,\varphi(E') \end{aligned} \tag{1.63}$$

The differential scattering cross section is then expanded using zero and first order Legendre polynomials :

$$\Sigma_s(E \leftarrow E', \vec{\Omega} \leftarrow \vec{\Omega}') = \frac{1}{4\pi}\Sigma_{s,0}(E \leftarrow E') + \frac{3}{4\pi}\Sigma_{s,1}(E \leftarrow E')\,\vec{\Omega}\cdot\vec{\Omega}' \tag{1.64}$$

Integrating Eq. (1.63) in two different ways, with or without a weight factor, leads to the equation :

$$[\Sigma(E) + d(B,E)\,B^2]\,\varphi(E) = \int_0^\infty dE'\,\Sigma_{s,0}(E \leftarrow E')\,\varphi(E') + \frac{\chi(E)}{k_{eff}} \int_0^\infty dE'\,\nu\Sigma_f(E')\,\varphi(E') \quad (1.65)$$

where $d(B,E) = \dfrac{i}{B^2\,\varphi(E)} \vec{B} \cdot \int_{4\pi} d^2\Omega\,\vec{\Omega}\,\varphi(E,\vec{\Omega})$ is the leakage coefficient, dependent of $\Sigma_{s,1}(E \leftarrow E')$. These values can be easily condensed over any energy group structure.

To compute the leakage coefficient $d(B,E)$, three hypotheses can be chosen :

- Homogeneous B0 model, assuming that the scattering cross section is isotropic ($\Sigma_{s,1}(E \leftarrow E') = 0$), and that $d(B,E)$ has no spatial dependence.
- Homogeneous B1 model, assuming that the scattering cross section is anisotropic ($\Sigma_{s,1}(E \leftarrow E') \neq 0$), and that $d(B,E)$ has no spatial dependence.
- Heterogeneous B1 model, assuming that the scattering cross section is anisotropic ($\Sigma_{s,1}(E \leftarrow E') \neq 0$), and that $d(B,E)$ is space dependent.

This theory can now be introduced into the CP method. One way to do so is to replace Eq. (1.35) by this new form :

$$\vec{\Phi}_g = \mathbf{W}_g\,[\vec{Q}_g^* - d_g(B)\,B^2\,\vec{\Phi}_g] \quad (1.66)$$

where $d_g(B)\,B^2\,\vec{\Phi}_g$ is the leakage rate.

1.2.4 Condensation, homogenization, and equivalence

At the end of the flux calculation, the flux, the reaction rates, and the cross sections have been computed using a coarse energy discretization (between 50 and 300 groups). But in order to perform a full core calculation, using the exact geometry of the nuclear reactor with detailed boundary conditions, few group energy discretization must be considered (between 2 and 20). All the properties enumerated before must then be condensed over macro-regions, and homogenized over few energy groups (Hébert, 2009).

A macro region C_m is defined as a collection of regions i taken from the previous transport calculation and a coarse energy group M_k is defined as a set of elementary groups g from this calculation, so that there are finally M regions and K energy groups ($m \in [1, M]$ and $k \in [1, K]$). All the properties are then condensed and homogenized by a flux-volume homogenization method, using the following equations for :

- Volumes : $$V_m = \sum_{i \in C_m} V_i \tag{1.67}$$
- Fluxes : $$\phi_{m,k} = \frac{1}{V_m} \sum_{i \in C_m} \sum_{g \in M_k} V_i \, \phi_{i,g} \tag{1.68}$$
- Total cross sections : $$\Sigma_{m,k} = \frac{1}{V_m \, \phi_{m,k}} \sum_{i \in C_m} \sum_{g \in M_k} V_i \, \Sigma_{i,g} \, \phi_{i,g} \tag{1.69}$$
- Scattering cross sections : $$\Sigma_{s,m,k \leftarrow l} = \frac{1}{V_m \, \phi_{m,l}} \sum_{i \in C_m} \sum_{g \in M_k} \sum_{h \in M_l} V_i \, \Sigma_{s,i,g \leftarrow h} \, \phi_{i,h} \tag{1.70}$$
- Fission cross sections : $$\nu\Sigma_{f,m,k} = \frac{1}{V_m \, \phi_{m,k}} \sum_{i \in C_m} \sum_{g \in M_k} V_i \, \nu\Sigma_{f,i,g} \, \phi_{i,g} \tag{1.71}$$
- Fission spectrums : $$\chi_{m,k} = \frac{\sum_{i \in C_m} \sum_{j=1}^{Jfiss} \sum_{g \in M_k} \chi_{j,g} \sum_{h=1}^{G} V_i \nu\Sigma_{f,i,h,j} \, \phi_{i,h}}{V_m \sum_{j=1}^{Jfiss} \sum_{h=1}^{K} \nu\Sigma_{f,m,h,j} \, \phi_{m,h}} \tag{1.72}$$
- Diffusion coefficients : $$D_{m,k} = \frac{1}{V_m \, \phi_{m,k}} \sum_{g \in M_k} d_g(B) \sum_{i \in C_m} V_i \, \phi_{i,g} \tag{1.73}$$

The problem with flux/volume weighted cross sections (see Eq. (1.69) to (1.73)) is that they do not ensure reaction rates preservation, except in the case where the output geometry is homogeneous. To enforce this preservation, a *superhomogénéisation* (SPH) equivalence procedure can be performed (Hébert, 1993). Using the fluxes and reaction rates defined as :

- Integrated fluxes : $$F^*_{m,k} = \sum_{i \in C_m} \sum_{g \in M_k} V_i \, \phi_{i,g} \tag{1.74}$$
- Collision rates : $$T^*_{m,k} = \sum_{i \in C_m} \sum_{g \in M_k} V_i \, \Sigma_{i,g} \, \phi_{i,g} \tag{1.75}$$
- Leakage rates : $$L^*_{m,k} = B^2 \sum_{g \in M_k} d_g(B) \sum_{i \in C_m} V_i \, \phi_{i,g} \tag{1.76}$$
- Within-group scattering rates :
$$T^*_{w,m,k} = \sum_{i \in C_m} \sum_{g \in M_k} \sum_{h \in M_k} V_i \, \Sigma_{s0,i,g \leftarrow h} \, \phi_{i,h} - B^2 \sum_{g \in M_k} d_g(B) \sum_{i \in C_m} V_i \, \phi_{i,g} \tag{1.77}$$
- Arrival rates :
$$Q^*_{m,k \leftarrow l} = \sum_{i \in C_m} \sum_{g \in M_k} \sum_{h \in M_k} V_i \left\{ \Sigma_{s0,i,g \leftarrow h} \, \phi_{i,h} + \frac{\chi_{j,g}}{k_{eff}} \nu \Sigma_{f,i,h,j} \, \phi_{i,h} \right\} - \delta_{kl} \sum_{i \in C_m} \sum_{g \in M_k} \sum_{h \in M_k} V_i \Sigma_{s0,i,g \leftarrow h} \, \phi_{i,h} \tag{1.78}$$

SPH corrective factors are then introduced on each macro region and coarse energy group to define equivalent cross sections and diffusion coefficients by the relations :

$$\tilde{\Sigma}_{m,k} = \mu_{m,k} \, \Sigma_{m,k} = \mu_{m,k} \, \frac{T^*_{m,k}}{F^*_{m,k}} \tag{1.79}$$

$$\tilde{\Sigma}_{w,m,k} = \mu_{m,k} \, \Sigma_{w,m,k} = \mu_{m,k} \, \frac{T^*_{w,m,k}}{F^*_{m,k}} \tag{1.80}$$

and

$$\tilde{D}_{m,k} = \mu_{m,k} \, D_{m,k} = \mu_{m,k} \, \frac{L^*_{m,k}}{B^2 \, F^*_{m,k}} \tag{1.81}$$

The same SPH factor should be used to multiply every cross section belonging to a given macro region and coarse energy group to preserve the macro balance. It is also true for the flux, which is redefined as $\tilde{F}^*_{m,k} = \dfrac{1}{\mu_{m,k}} F^*_{m,k}$.

Those values are used in a macro calculation, defined as a flux calculation made over the macro region and coarse energy groups. Many types of equivalence are possible (transport-transport, transport-p_{ij}, transport-Sn, ...) but here will only be considered the transport-diffusion equivalence, where the macro calculation is a solution of the neutron diffusion equation :

$$-\vec{\nabla} \cdot \tilde{D}_k(\vec{r}) \, \vec{\nabla} \, \tilde{\phi}_k(\vec{r}) + \left[\tilde{\Sigma}_k(\vec{r}) - \tilde{\Sigma}_{w,k}(\vec{r})\right] \, \tilde{\phi}_k(\vec{r}) = Q^*_k(\vec{r}) \tag{1.82}$$

with a conservative boundary condition such as :

$$\vec{\nabla} \, \tilde{\phi}_k(\vec{r_s}) \cdot \vec{R}(\vec{r}_s) = 0 \quad \text{if } \vec{r}_s \text{ is a point of the reflective boundary.} \tag{1.83}$$

and where all the nuclear properties are constant over each macro region C_m (also for the flux with $\tilde{\phi}_{i,k} = \dfrac{1}{V_i} \displaystyle\int_{V_i} d^3r \, \tilde{\phi}_k(\vec{r})$), except for the arrival neutron source which exhibits a piecewise continuous variation given by :

$$Q^*_k(\vec{r}) = \sum_l \frac{Q^*_{m,k \leftarrow l} \, \tilde{\phi}_l(\vec{r}) \, \mu_{m,l}}{F^*_{m,k}} \quad \text{if } \vec{r} \in V_m. \tag{1.84}$$

This set of SPH factors satisfies the system of equations given by Eqs. (1.79) to (1.84), also adding a normalization condition arbitrarily chosen so as to preserve either :

- the average flux in the lattice (flux-volume normalization)
- the surface flux of the macro-geometry (Selengut normalization) (Selengut, 1960)
- information related to the last row of cells (EDF normalization) (Courau et al., 2008)

Finally, the procedure for finding this set can be carried out as a fixed-point iterative strategy, starting from an initial estimate of the factors (typically $\mu = 1$).

1.2.5 Isotopic depletion

All isotopes in a nuclear reactor may undergo an isotopic depletion, because exposing an isotope to a neutron flux produces nuclear reactions like fission or absorption, and also because some isotopes may be subject to radioactive decay. This leads to the modification of the nuclear characteristics of the original mixtures, of the macroscopic cross sections of the reactor materials, and hence of the neutron flux in the reactor. An evolution calculation must then be performed to take these effects into account, modifying the isotopic concentrations of the materials as a function of the burnup, which describes the time-integrated power (or energy) per initial mass of fissionable nuclides (expressed in MWj/t).

The depletion of K isotopes over a time stage (t_0, t_f) in each burnup mixture of the unit cell follows the following equations, also know as Bateman equations (Bateman, 1910):

$$\frac{dN_k}{dt} + \Lambda_k(t)N_k(t) = S_k(t) \quad ; \; k = 1, K \tag{1.85}$$

with
$$\Lambda_k(t) = \lambda_k + \langle \sigma_{a,k}(t)\phi(t) \rangle \; , \tag{1.86}$$

$$S_k(t) = \sum_{m=1}^{M} Y_{k,m} \langle \sigma_{f,m}(t)\phi(t) \rangle \, N_m(t) + \sum_{l=1}^{K} m_{l,k}(t) N_l(t) \; , \tag{1.87}$$

$$\langle \sigma_{x,l}(t)\phi(t) \rangle = \int_0^\infty du \, \sigma_{x,l}(u)\phi(t,u) \tag{1.88}$$

and
$$\sigma_{x,k}(t,u)\,\phi(t,u) = \sigma_{x,k}(t_0,u)\,\phi(t_0,u)$$

$$+ \; \frac{\sigma_{x,k}(t_f,u)\,\phi(t_f,u) - \sigma_{x,k}(t_0,u)\,\phi(t_0,u)}{t_f - t_0}\,(t - t_0) \tag{1.89}$$

where

K = number of depleting isotopes

L = number of fissile isotopes producing fission products

$N_k(t)$ = time dependent number density for k-th isotope

λ_k = radioactive decay constant for the k-th isotope

$\sigma_{x,k}(t,u)$ = time and lethargy dependent microscopic cross section for nuclear reaction x on k-th isotope (where $x = a$, $x = f$ and $x = \gamma$ respectively stands for absorption, fission and radiative capture cross sections). Note that the time is considered as independent variable in order to account for resonance self-shielding effects. Also, $\Sigma_{x,k} = N_k\,\sigma_{x,k}$.

$\phi(t,u)$ = time and lethargy dependent neutron flux

$Y_{k,m}$ = fission yield for production of fission product k by fissile isotope m

$m_{l,k}(t)$ = radioactive decay constant or $\langle \sigma_{x,l}(t)\phi(t) \rangle$ term for production of isotope k by isotope l.

Eqs. (1.85) form a system of coupled ordinary differential equations that can be solved by the use of different techniques such as the Laplace transform method, the Runge-Kutta family of numerical methods, or the integration factor method.

The solution is also affected by the flux normalization factors. The lattice code can perform out-of-core or in-core depletion with a choice between two normalization techniques : a constant flux depletion, where the lethargy integrated fluxes at beginning-of-cycle and end-of-cycle are set to a constant F, or a constant power depletion, where the power released per initial heavy element at beginning-of-cycle and end-of-cycle are set to a constant W. In both cases, the lattice code computes the exact burnup of the unit cell by adding an additional equation in the depletion system. This value should be used as parameter in order to tabulate the output cross sections.

1.3 The DRAGON code and its features

The work done in the context of this study was performed with the lattice code DRAGON Version4 (Marleau et al., 2009). It is part of the Version4 distribution, composed of GANLIB (which provides the common-use functionalities and ensures the normal execution of the modules), UTILIB (which provides the utility and linear algebra libraries), NJOY-99 (to produce Dragon-formatted libraries), DRAGON (presented after), TRIVAC (1D/2D/3D full-core flux solver), and DONJON (used for full-core operation studies). The work was performed with the DRAGON code, which will be presented here, with its features allowing the use of advanced calculation schemes.

1.3.1 Description of the code

The DRAGON code is the result of an effort made at *École Polytechnique de Montréal* to rationalize and unify the different models and algorithms used to solve the neutron transport equation into a single code. It is divided into many calculation modules linked together using the GAN generalized driver (Roy and Hebert, 2000), and exchanging informations via well defined data structures, in order to facilitate the development and the implementation of new calculation techniques.

The Dragon code data flow follows the same scheme as presented in Figure (1.1). The main components of this code are :

- The LIB: module, to generate or modify a DRAGON multigroup microscopic and macroscopic cross section libraries or microlib.
- Modules to analyze various geometries and to generate a tracking file for different deterministic evaluations. Three of them were used here :

  - the SYBILT: tracking module, which performs reactor assembly calculations in both rectangular and hexagonal geometries using the interface current method,
  - the NXT: module, used to generate the collision probability matrices for the cases having cluster, two-dimensional or three-dimensional mixed rectangular and cylindrical geometries,
  - The MCCGT: module, using the tracking from NXT: and performing flux integration.

- Modules for resonance self-shielding calculations. There are two of them using the two models described in Sect. (1.2.2) :
  - the SHI: module, using the generalized Stamm'ler method,
  - the USS: module, using a subgroup approach.

- The ASM: module, to prepare the group-dependent complete collision probability or the assembly matrices.

- The FLU: module, to solve the multigroup neutron transport equation using the collision probability method or an inner-iteration approach.

- The EVO: module for the isotopic depletion.

- The EDI: module, which supplies the main editing options where an equivalence method based on SPH method is available.

- The COMPO: module, which creates a reactor database to store all the nuclear data useful in reactor calculations.

It can be noted that with the use of NJOY-99, DRAGON has the ability to use its own format of libraries, but also other types of libraries, as described in Sect. (1.2.1).

1.3.2 Two-level scheme

Because of its modularity, it is possible to perform advanced calculation schemes with DRAGON. One of these schemes is dedicated to PWR assembly calculations.

A PWR assembly contains a lot of different cells (usually composed by 17x17 cells, but can also be 16x16 or 18x18). A lot of different regions have then to be taken into account in the transport calculation. Using a MOC method to solve the transport equation on this kind of assembly takes a lot of time. In order to obtain a good accuracy with a fast calculation speed, a two-level scheme was proposed in (Courau et al., 2008). The idea is to reduce the CPU time of the MOC calculation by decreasing the number of energy groups used in this calculation. The assembly calculation is then performed in two steps (a two-level calculation) :

- The first step (level 1) consist of performing a fine flux calculation using the interface current method. It is used for both self-shielding and main flux calculation, based on 172-group XMAS energy discretization microlib. The macroscopic cross sections are then condensed to a lower number of energy groups (in our case 26-groups), using a flux/volume weighting procedure, or an SPH equivalence.
- The second step (level 2) performs a detailed spatial calculation using a MOC method. The geometry can then be more discretized, but using the same number of mixtures as in the first level. The cross sections can be homogenized and condensed (full-assembly homogenized, or pin-by-pin homogenization, and condensed in typically 2 to 8 groups), to be used in a finite reactor calculation code.

The use of an SPH equivalence between these two levels has been studied during this work. It also has to be used for the final homogenization and condensation, when a pin-by-pin calculation is considered.

1.3.3 The code development

The DRAGON code is actively and continuously supported and developed at *École Polytechnique de Montréal* since 1991, under the GNU Lesser General Public License. It is simple to use, to modify, and to install on workstations that support a Fortran compiler.

Because of these features, the code is in constant evolution, using the return of experience acquired by its use in different organizations (AECL, EDF, IRSN, GRS, ...), and over different types of benchmarks. The agreements with the different organizations are then really important, in order for the code to be compared with all other existing codes, and then to be at the state-of-the-art of the lattice codes.

During this work, some improvements have been made, performing calculations on complex benchmarks, and comparing the results with a wide range of lattice codes.

CHAPTER 2

18X18 PWR EXERCISE

The first part of the work performed at the GRS was to implement the DRAGON code on the local Linux clusters, and compare its accuracy with a wide range of other codes on a simple exercise. A comparison between different methods is performed here, in order to find the best trade-off between accuracy, and CPU time. Also, a comparison on the different libraries is required, because most lattice codes use their own type of library. Finally, comparisons with many lattice code calculations are performed on the multiplication factor, the nuclide densities, and the pin power.

2.1 Description of the problem

This exercise was made to compare the results of depletion calculations, coming from different lattice codes, on some important parameters. The purpose here is to prove that the DRAGON code has a good accuracy in comparison with other lattice codes, and to show the advantages of using a two-level scheme calculation.

The calculations are performed on a 18x18 UO_2 PWR fuel assembly (Porsch et al., 2006). This exercise was proposed by D. Porsch (Framatome), U. Hesse and W. Zwermann (GRS), and W. Bernnat (IKE, Stuttgart university). The assembly is composed of 300 fuel cells of UO_2 fuel enriched at 4% of uranium 235, and 24 guide tubes (one of which is designed for a detector position, but is considered as a guide tube in the calculations), the whole assembly being considered at full power reactor.

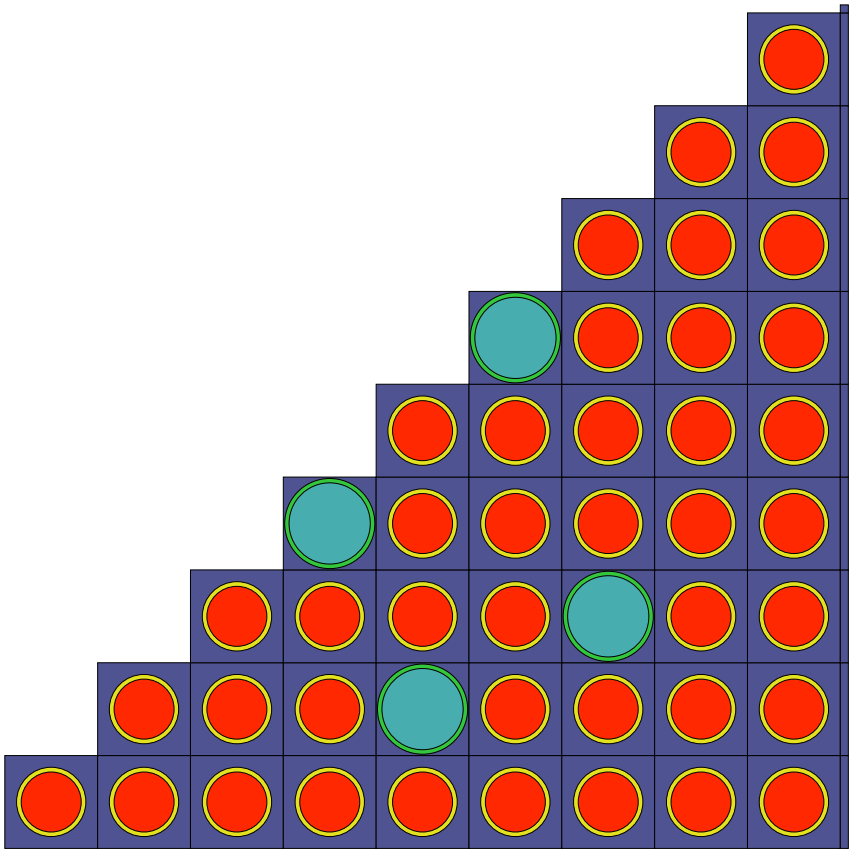


Figure 2.1 East-North-East 1/8 of the fuel assembly

The moderator is light water with a boron concentration of 500 ppm, at a temperature of 310°C and a pressure of 158 bar. The cladding of both fuel and guide tubes is made of zircaloy-4 (ZRY-4) composed of zirconium, iron, chromium, and traces of hafnium at temperature 332.8°C. The fuel is supposed to be at 500°C, in a xenon-free state at beginning of cycle. The East-North-East octant of the assembly is shown in Figure (2.1).

The data required to perform the comparisons are the multiplication factor, the isotopic global composition (for some specific actinides and fission products), the maximum fuel pin power and its associated pin position as a function of burnup, and the pin power distribution at beginning and end of depletion.

2.2 Models, discretization and regroupments

The first step was to model this assembly and to look for a configuration yielding good accuracy in a reasonable amount of CPU time. In order to do so, a calculation model was chosen, the impact of some material regroupments have been studied to decrease the CPU time, and the discretization of the domain has been refined. The calculations are performed using a one level scheme using a UP1 anisotropic interface current method for the self-shielding and the flux calculations, to perform these verifications with a fast CPU time. Here, as in the next part of this analysis, the DRAGON formatted library based on the JEFF3.1 evaluation with 172 energy groups is used. This will be explained in the section 2.4.2.

2.2.1 Model

First, the symmetries of this assembly allow this study to be built using an eighth of the assembly only. This is very important when treating such a geometry, because considering the whole assembly would take a very long time, or would even be impossible, as the number of regions considered may reach the limit for the use of a CP method.

As it is said in the description of the exercise, the fuel will extend to the cladding inner diameter (the air gap is homogenized with the fuel), and the spacers in the fuel assembly is volume homogenized with the moderator in the active region, but not with the moderator inside the guide tubes.

Concerning the cladding, even if it has the same isotopic composition for both fuel and guide tube, the distinction between the two types is made, because the temperature for each one is different.

2.2.2 Burnup evolution parameterization

The assembly presented before will undergo an evolution during a time period. Because most of the fission products acting as poison material (like ^{135}Xe or ^{149}Sm), are produced during a short period at the beginning of stage, the burnup steps have to be carefully chosen, especially for the first few steps. Here, the burnup steps given in the exercise description are used (Table (2.1)).

Index	Full power days	Burnup [GWd/t]	Index	Full power days	Burnup [GWd/t]
1	0.00	0.00	22	808.98	30.00
2	2.70	0.10	23	876.40	32.50
3	13.48	0.20	24	943.81	35.00
4	26.97	1.00	25	1011.22	37.50
5	53.95	2.00	26	1078.64	40.00
6	80.90	3.00	27	1146.06	42.50
7	107.86	4.00	28	1213.47	45.00
8	134.83	5.00	29	1280.89	47.50
9	161.80	6.00	30	1348.30	50.00
10	188.77	7.00	31	1415.72	52.50
11	215.73	8.00	32	1483.14	55.00
12	242.70	9.00	33	1550.55	57.50
13	269.67	10.00	34	1617.97	60.00
14	296.63	11.00	35	1685.39	62.50
15	337.08	12.50	36	1752.80	65.00
16	404.49	15.00	37	1820.22	67.50
17	471.91	17.50	38	1887.64	70.00
18	539.32	20.00	39	1955.05	72.50
19	606.74	22.50	40	2022.47	75.00
20	674.15	25.00	41	2089.89	77.50
21	741.56	27.50	42	2157.30	80.00

Table 2.1 Reference Burnup Times for Calculations

2.2.3 Discretization

Some calculations were performed to verify the influence of the assembly discretization. Those calculations are done using the model described before.

Here, like for the whole exercise, the fuel is discretized in different regions using an 'onion peel' model. This discretization is very important to correctly take into account the spacial distribution of the resonant absorption of the ^{238}U, but also to treat in a more realistic way the radial evolution of the fuel, as recommended in (Santamarina et al., 2004). Following these recommendations, the fuel is divided in four different rings, representing, from the inside to the outside, 50%, 30%, 15%, and 5% of the pin volume, as shown in Figure (2.2). The horizontal, or vertical discretization of the fuel will not be studied here.

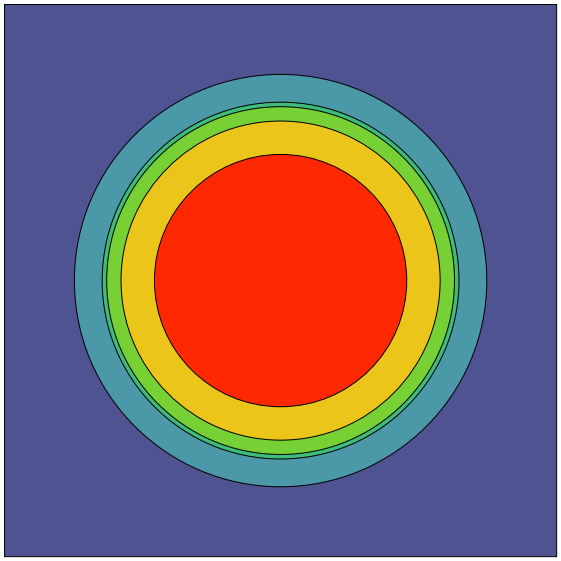


Figure 2.2 Fuel pin discretization

To confirm that this discretization is required, two calculations are performed to assess its influence : the first one, which is taken as reference, is performed with four rings, the other with only one. The results shows that for the fresh fuel state, the difference for the multiplication factor is about 150 pcm. Moreover, while performing a burnup calculation for these two cases, the k_{eff} goes from -150 pcm to 160 pcm, and the isotopic composition of the fuel shows differences of up to 2% for some fission products isotopes, and up to 1% for the ^{235}U as shown in Figure (2.3) for some important isotopes. The discretization into four rings will then be retained for the rest of this study.

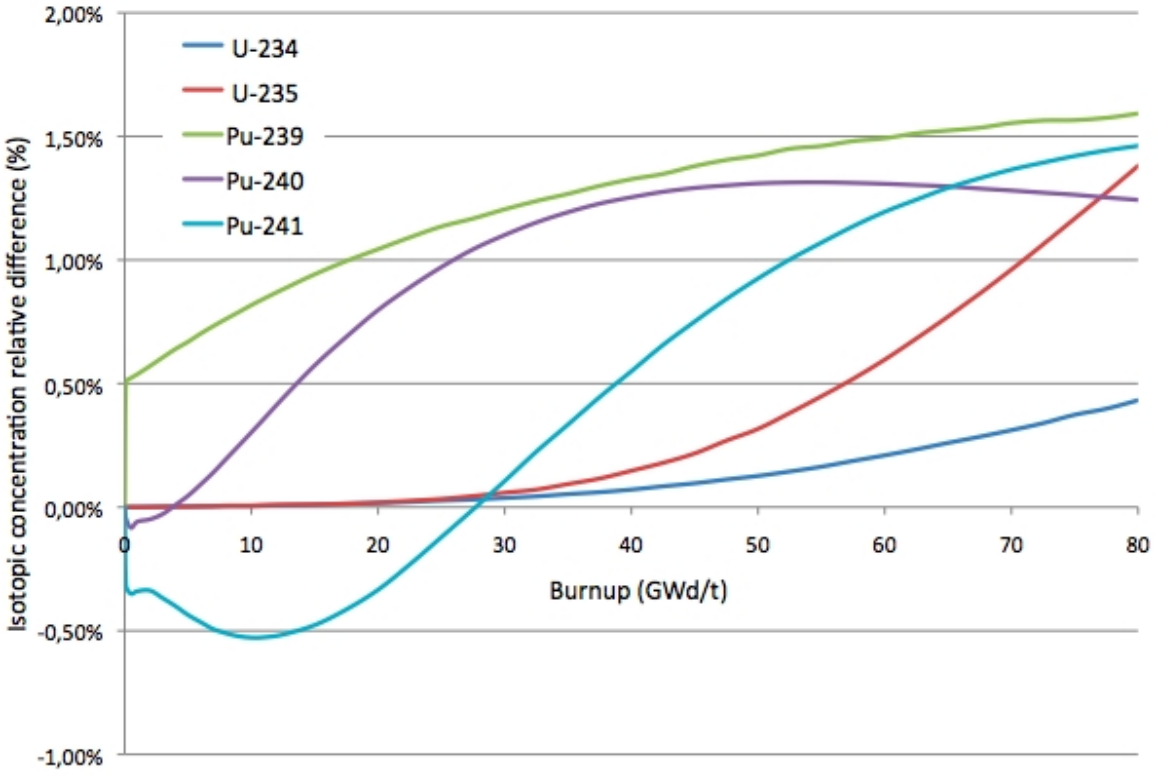


Figure 2.3 Relative differences (%) for fuel isotopic concentration

Another verification is performed on the discretization of the moderator. Two different models are used here to perform the comparison, presented on Figure (2.4) : the first one without discretization, the second one with the moderator divided in four. The results show a difference of 11 pcm between those two models, and the isotopic composition shows differences always lower than 0,2%. Because the time difference between these calculations is small (4 minutes on an 1 hour burnup calculation), the discretization of the moderator is also maintained.

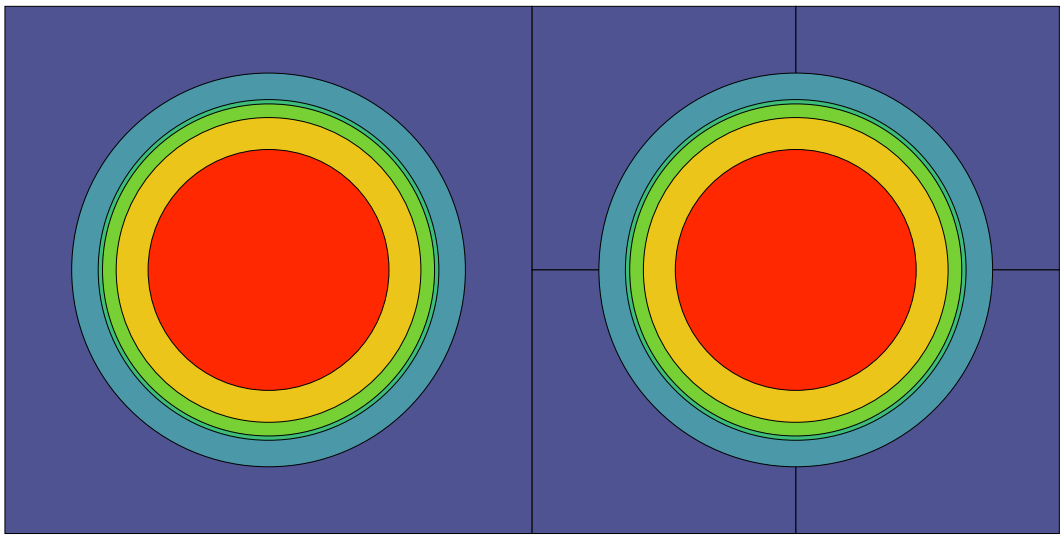


Figure 2.4 Two different discretization of the moderator

Finally, the discretization chosen for the whole assembly is shown on Figure (2.5), giving good CPU times (on this figure, regroupments are used, which are going to be explained in the next section).

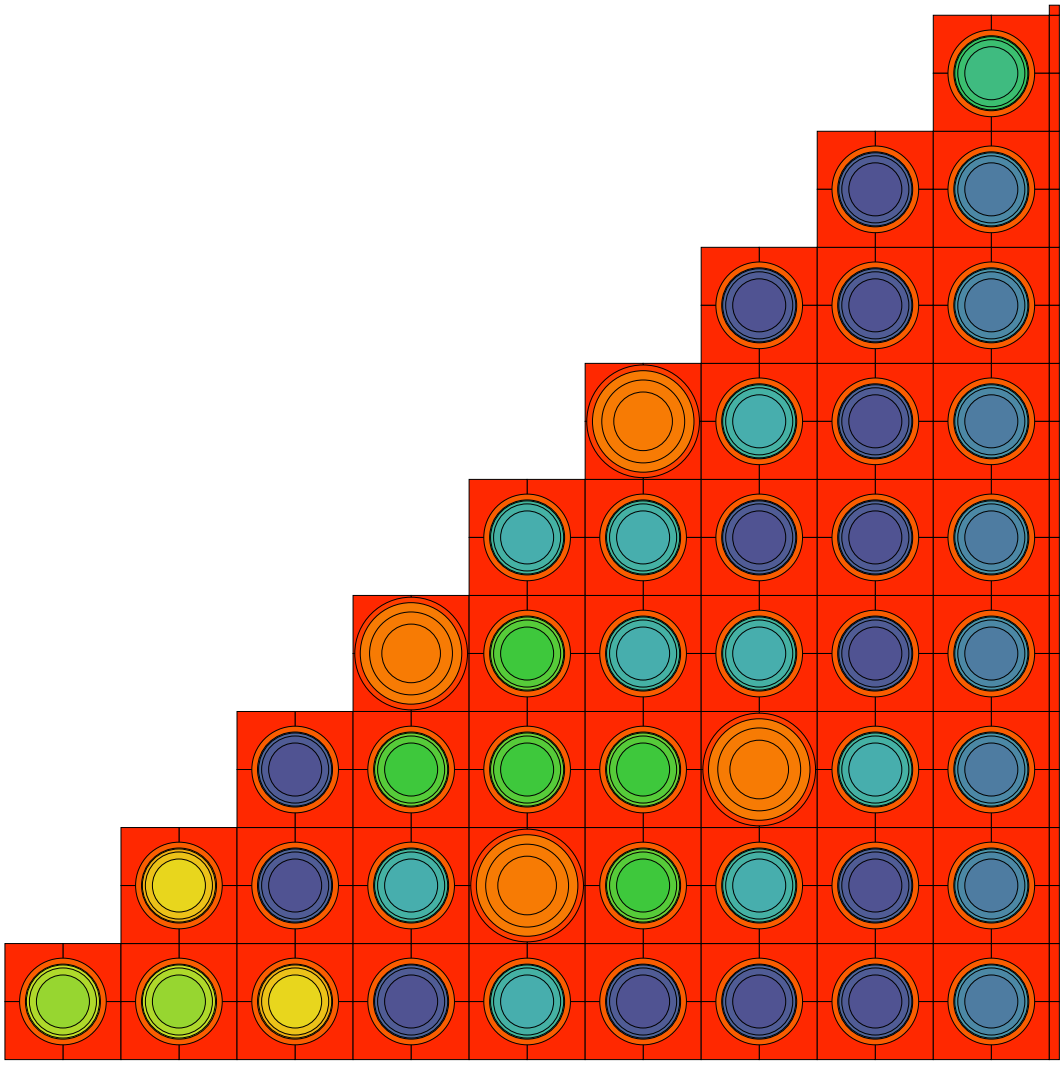


Figure 2.5 Assembly discretization

2.2.4 Regroupments

In order to reduce the number of unknowns, and to reduce the CPU time, some regroupments of fuel cells are studied. Two types of regroupments can be considered :

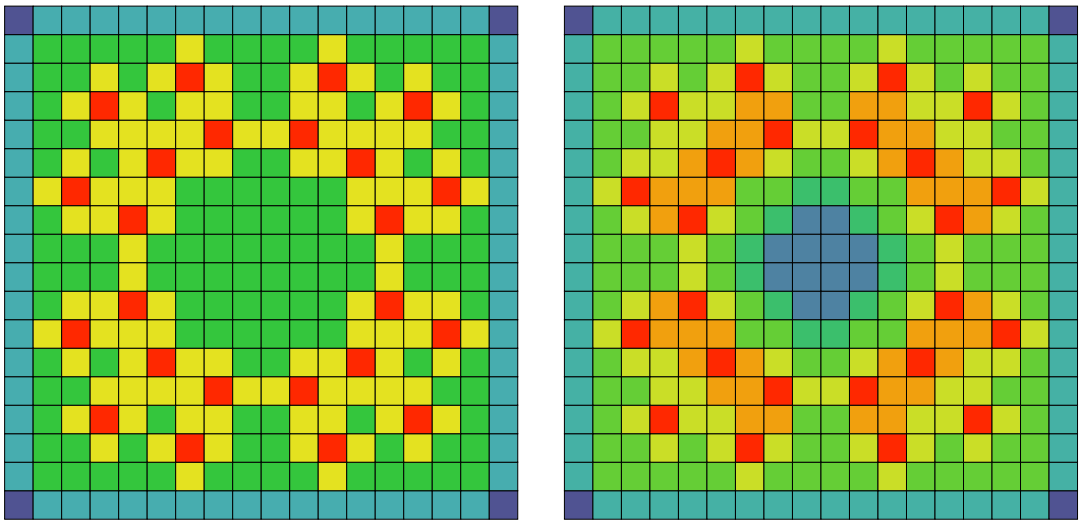
- First, the fuel cells can be regrouped, assuming that their composition will burn at the same rate. This is advised in order to differentiate the fuel mixtures depending on their position relative to the guide tubes, instrumentation tubes, corner and border of the assembly (Le Mer, 2007).
- Then, for the UP1 calculations, it is also possible to regroup some cells, considering that they are subject to identical flux conditions. The flux inside the different region of each cell that belong to the same group are assumed identical during the calculation (note that the cell orientation must be chosen). Here, the number of regroupments can only be chosen equal to, or higher than the previous one.

To reduce the CPU time, the reduction of the number of mixtures is first studied, at the same time as merging cells for the self-shielding calculation using a UP1 method. To evaluate its influence, a reference calculation is performed, where each ring of each cell evolves individually (the rings in a pin always evolve individually, the regroupments are made by fuel pins so that different pins will have the same mixture in each of their rings). Then, two regroupments are studied (see Figure (2.6)) :

- Regroupment 1, with 20 different mixtures, where are differentiated :
  - the cells in the corners of the assembly (1),
  - the cells on the borders of the assembly (2),
  - the cells sharing a face with the guide tubes (3),
  - the other cells (4).

- Regroupment 2, with 32 different mixtures is an improvement of the first one, to reduce the differences on the pin fission rates.

In our case, the number of mixtures will be reduced. But the fuel cells are also merged together using the same regroupments for the self-shielding calculation, regrouping resonant mixtures for the ^{235}U and the Zr, but not for the ^{238}U (leading to one resonant mixture for the ^{235}U, one for the Zr, and either 20 or 32 for the ^{238}U, depending on the regroupment considered).



(a) Regroupment 1 (b) Regroupment 2

Figure 2.6 Regroupments selected

The comparison is performed considering a burnup calculation of the assembly on the different parameters cited before. The results show that the differences on the multiplication factor are always lower than 70 pcm. For the global isotopic densities, the relative differences are always lower than 0,25% for regroupment 1, and always lower than 0,05% for regroupment 2, as shown in Figures (2.7) and (2.8) for some heavy nuclides.

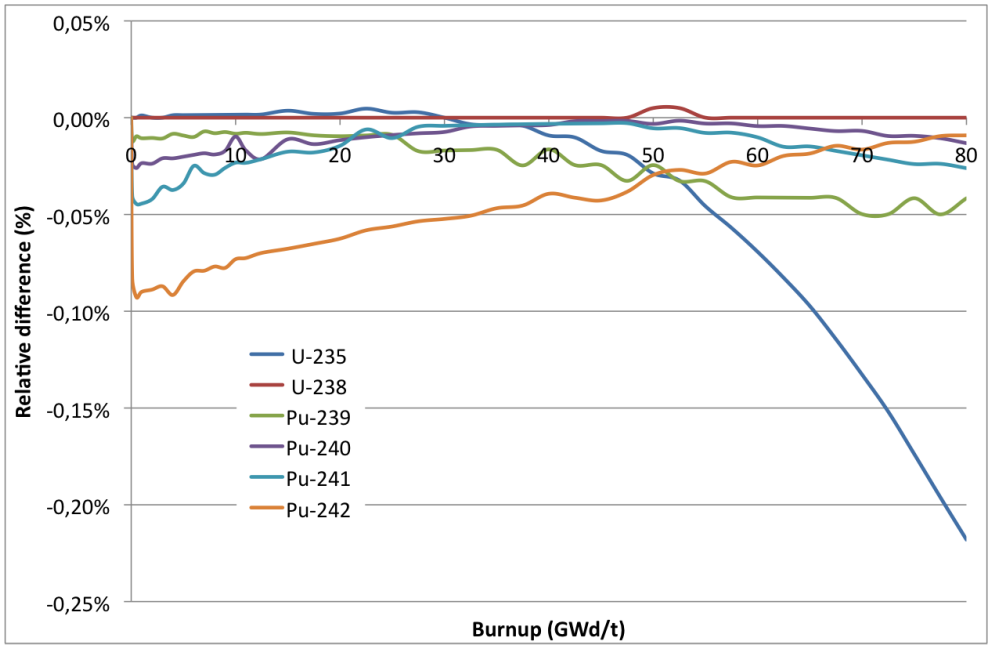


Figure 2.7 Relative differences in isotopic densities (%) between regroupment 1 and the reference calculation

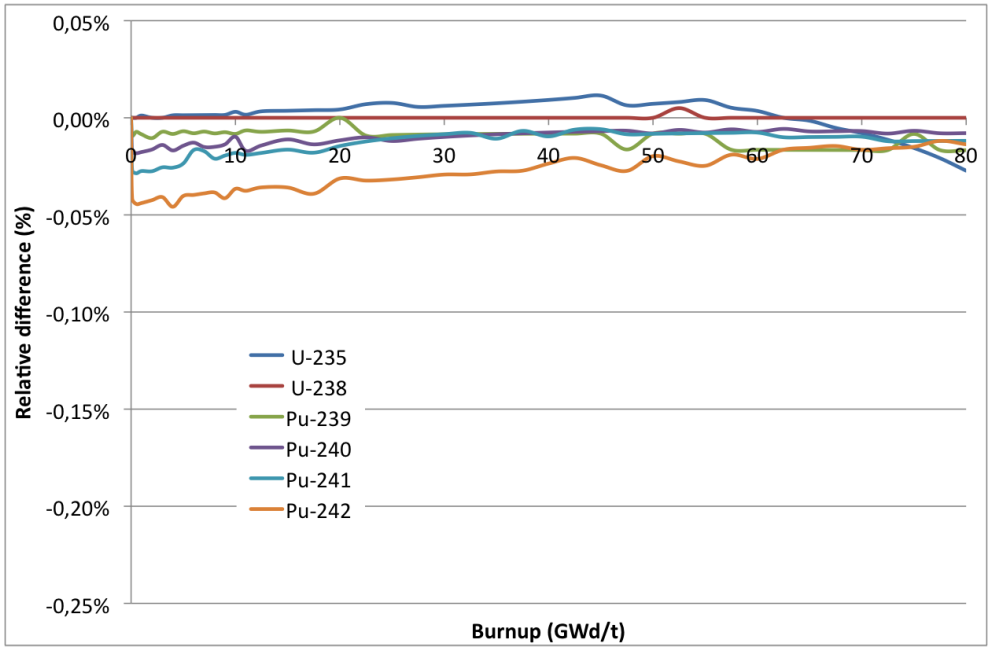
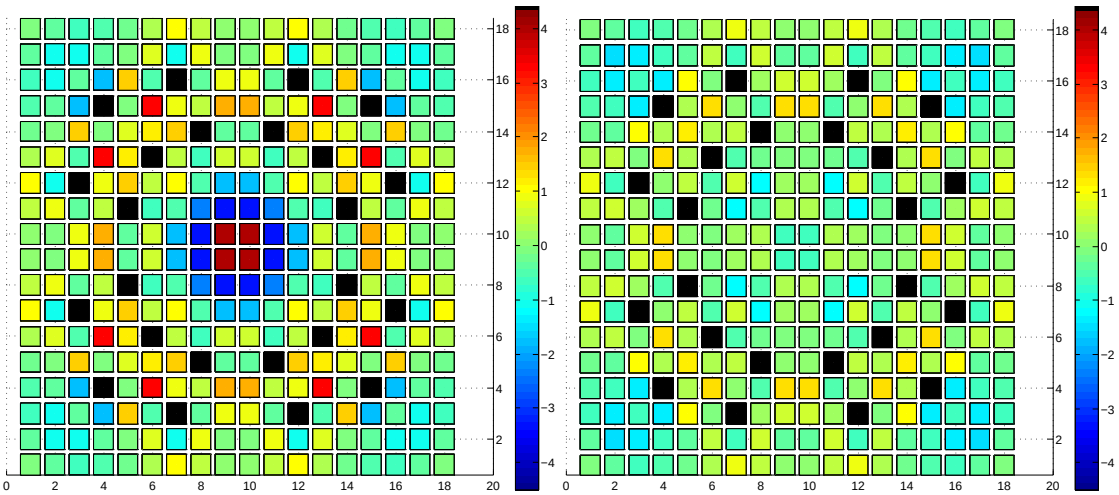


Figure 2.8 Relative differences in isotopic densities (%) between regroupment 1 and the reference calculation

For the pin fission rates, at beginning of cycle, the differences are always lower than 0.01% for both cases. But at the end of cycle, those differences are larger, with a maximum of 4% for the regroupment 1, but reduced to a maximum of 1,4% for the regroupment 2 as shown in Figure (2.9) :



(a) Between regroupment 1 and reference (b) Between regroupment 2 and reference

Figure 2.9 Relative difference (%) in the pin fission rates at end of cycle

Finally, those regroupments reduce considerably the CPU time, as shown in Table (2.2):

	CPU time (min)
Reference calculation	4641
Regroupment 2	248
Regroupment 1	224

Table 2.2 Comparison of the calculation times for the regroupments

Considering the reduction of CPU time and the acceptable results, the regroupment 2 is then kept for both reducing the number of mixtures, and merging cells during the self-shielding calculation, for the rest of this exercise. Using regroupments to merge cells in the flux calculation (of type UP1) is also possible, and will be studied in the next section.

2.3 Calculation types comparison

The model being chosen, the next step is to compare the differences between the different computational methods that can be used in DRAGON. The first type of calculation uses the collision probability method. Then, calculations are performed using the method of characteristics. Finally, a two level-scheme is used.

2.3.1 Pij calculations

The first type of calculation is performed using a collision probability method. At first, the interface current method is used, because this method is supposed to be faster. The following calculation scheme has been chosen :

- For the self-shielding, the SYBILT: module is used to perform the tracking of the geometry. The calculation is performed by the USS: module at every burnup step, using a subgroup approach with physical probability tables. Here, the linearly anisotropic (DP1) components of the inter-cell currents are used. A sufficient number of basis points for the angular integration and the spatial integration was selected.
- For the flux calculation, the SYBILT: module is also used to perform the tracking. The flux is then solved by a UP1 anisotropic interface current method, using the fixed Laplacian option (K type calculation), and without leakage model (it has almost no effect here). The same number of basis points as for the self-shielding calculation is selected.

This calculation scheme will be referred to as scheme (1).

For this scheme, a calculation where the regroupment 2 is used in the flux calculation is compared to the calculation where it is not used. It appears that it has a small influence on the multiplication factor, with a maximum difference of 15 pcm during the evolution. For the isotopic densities, the maximum difference is 0.25%, which is still acceptable. The problem comes from the pin fission rates, where the maximum difference is 2.4%. Even if the time is reduced from 248 minutes to 63 minutes, the regroupment will not be kept here, as the pin power deviation is too large.

Then, a collision probability method is used to solve the flux. The following calculation scheme has been chosen :

- The self-shielding calculation is performed using two different methods :
  - the same method as before (UP1), using the same parameters,
  - the NXT: module, with specular tracking parameters.

  Using the NXT: module for the self-shielding calculation does not change significantly the multiplication factor (10 pcm) compared with the SYBILT: module but the calculation is much longer. The UP1 method is therefore kept.

- For the flux calculation, the NXT: module is used to perform the tracking with specular conditions. The tracking parameters have been tested at burnup zero. The number of angles has a large influence on the multiplication factor, so a sufficient number (here 20) has to be taken. For the track density, it does not have a big impact (some pcm), so it can be chosen low (here 10,0 cm^{-1}).

This calculation scheme will be referred to as scheme (2) (with the UP1 method for the self-shielding).

2.3.2 MOC calculations

The next type of calculation is performed using the method of characteristics, supposed to be more accurate and longer because it can handle more unknowns. Here again, the following calculation scheme has been chosen :

- For the self-shielding calculation, as before, two methods are tested :
  - the UP1 method, using the same parameters than before,
  - the NXT: module with specular tracking parameters, followed by the MCCGT: module for the use of the characteristic method.
- For the flux calculation, the same couple of modules is used (NXT: + MCCGT:), using specular conditions. As for the Pij calculation, the tracking parameters were tested at fuel fresh state, showing the same conclusions. The same parameters are then used.

These two calculation schemes will be respectively referred to as scheme (3) (with a UP1 method for the self-shielding) and (4) (with a MOC calculation with specular conditions for the self-shielding).

2.3.3 Two level calculations

Finally, a two-level scheme calculation is performed, which should be comparable in accuracy to the MOC calculation, but with a faster calculation time. It is a combination of the two previous types of calculations :

- For the self-shielding, the SYBILT: module is used to create the tracking of the geometry. The calculation is performed by the USS: module at every burnup step.

- For the first-level flux calculation, the SYBILT: module is also used to create the tracking. The flux is then solved by the UP1 anisotropic interface current method, using the fixed Laplacian option (K type calculation), and without leakage model. Here, a simplified geometry is used, where the moderator is not discretized.

- The cross sections are then condensed to a smaller number of groups (here to 26 groups). For this condensation, the usefulness of an SPH equivalence will be studied.

- For the second-level flux calculation, the couple of modules NXT: + MCCGT: is used, with specular conditions, and the same tracking parameters as for the MOC calculation presented before. The moderator is here discretized.

This calculation scheme will then be divided in two different sets referred to as the scheme (5) (without SPH equivalence) and scheme (6) (with SPH equivalence).

Here again, a calculation where the regroupment 2 is used for the flux calculation using a UP1 method (first level) has been compared to the calculation where it is not used, for the scheme (6).
The results show that there is almost no difference between the two calculations :

- the relative difference on the multiplication factor is always lower than 5 pcm,

- for the global isotopic densities, the differences are all lower than 0,07%,

- finally, for the pin fission rates, the maximum difference at end of cycle is 0,13%.

Because the CPU time is then decreased from 655 minutes to 77 minutes, and because the differences are really acceptable, the regroupment 2 will be kept for the first-level flux calculation.

2.3.4 Comparison of the different types of calculations

A comparison between the six schemes presented before has then been performed. The scheme (4) will be taken as a reference, because the MOC calculation is supposed to be the most accurate one. A first comparison can be performed on the calculation times, as shown in table (2.3) :

	CPU time (min)
Complete MOC calculation (4)	1030
Pij calculation (2)	254
UP1 calculation (1)	248
MOC calculation (3)	195
2-level (SPH) calculation (6)	77
2-level calculation (5)	75

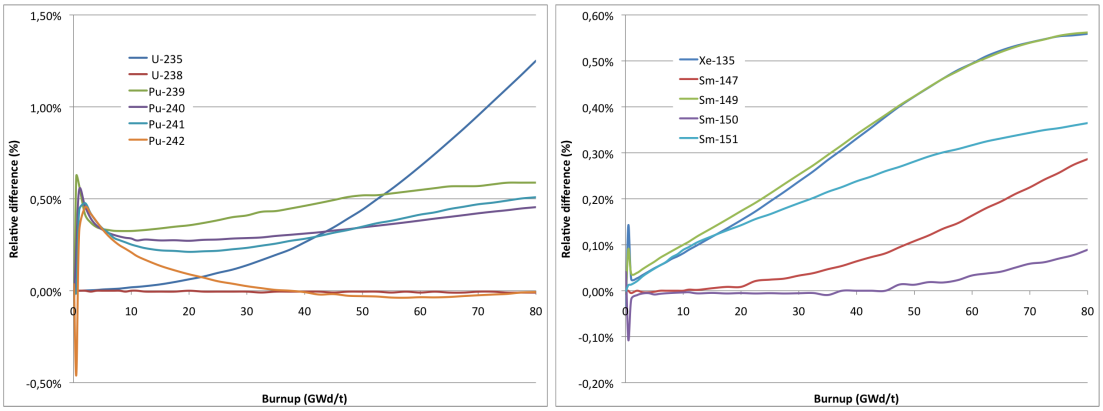
Table 2.3 Comparison of the calculation times for the different methods

The two-level scheme is clearly the fastest method. As expected, the complete MOC calculation (4) is the slowest calculation. Three different comparisons can then be performed.

2.3.4.1 Comparison of the self-shielding calculations for the MOC calculation

The first comparison that can be done is on the type of self-shielding calculation used between the two MOC schemes (3) and (4).

In fact at the fuel fresh state, there is only a difference of 44 pcm between these methods. But while performing a burnup evolution, the differences grow to a maximum of 95 pcm. For the nuclide densities, the differences are also getting larger with burnup, as shown in Figure (2.10) :



(a) Important actinides (b) Important fission products

Figure 2.10 Relative difference for some densities between schemes (3) and (4) (%)

Finally, for the pin fission rates, the differences are acceptable, with a maximum difference of 0,25%. But because of the big differencies for the k_{eff} and the densities, the two methods will be be kept separately. It can be noted here that the self-shielding calculation plays a big role, and must be carefully performed.

2.3.4.2 SPH equivalence effects

A second study can be performed on the usefulness of an SPH equivalence between the two levels of the two-level scheme. In fact, at this point of the calculation, only a condensation is performed. The problem is, an homogenization by mixture also takes place.

Here, the MOC scheme (3) will be taken as reference because it has the same self-shielding method. In comparison with this scheme, the results show a better behavior during the evolution when using the SPH equivalence. For the case with SPH equivalence, the difference on the k_{eff} with the MOC scheme is always lower than for the case without equivalence, as shown on Figure (2.11). The differences on the nuclide densities are also reduced as shown in Figure (2.12) for some important actinides.

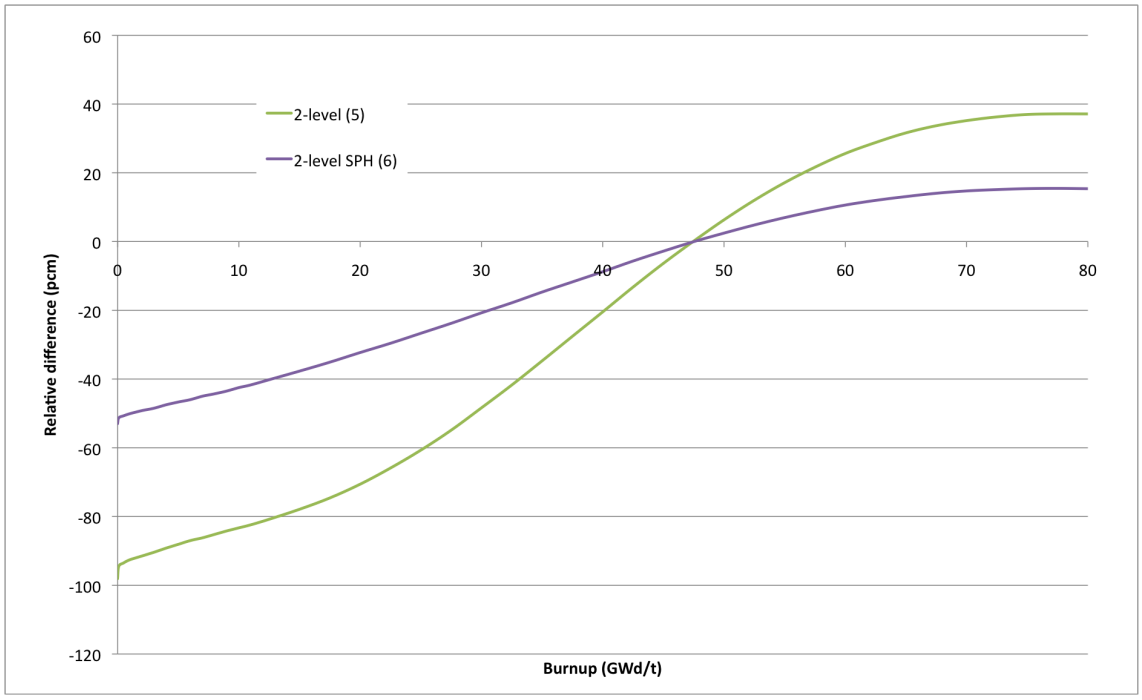
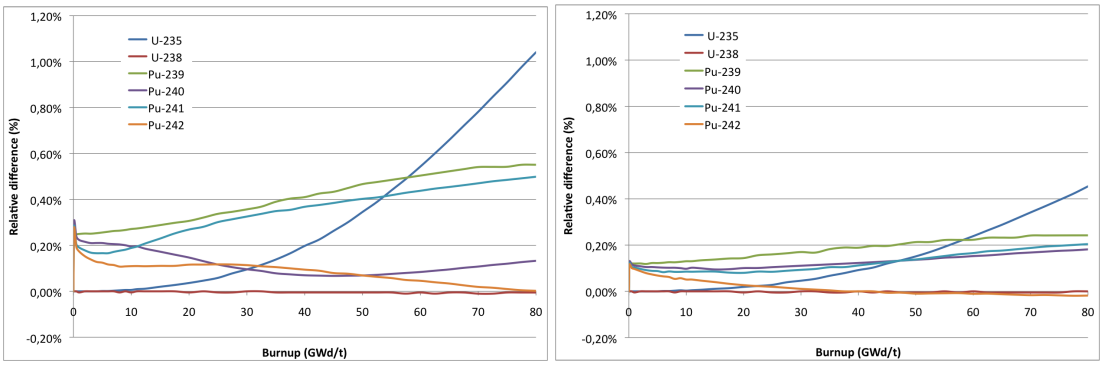


Figure 2.11 Relative difference between the two-level schemes (5) and (6), and the MOC scheme (3) (pcm)



(a) Between scheme (3) and (5)

(b) Between scheme (3) and (6)

Figure 2.12 Relative difference for some important actinides densities (%)

Because the SPH equivalence does not add a lot of time to the calculations, it will be kept in the two-level scheme for the rest of the study.

2.3.4.3 Comparison of the flux calculation methods

Then, a final comparison can be performed between the different methods used for the flux calculation, using the same self-shielding model (UP1 method). This comparison concerns the calculation schemes (1), (2), (3), and (6). The MOC scheme (3) will be here taken as reference.

Concerning the multiplication factor, the differences are acceptable with a maximum of 65 pcm for the Pij calculation scheme (2). The two-level scheme (6) shows a behavior in between the two others schemes as shown in Figure (2.13).

For the isotopic densities, the differences are rather small, with a maximum value of 0,63% for the ^{242m}Am, at end of cycle for the Pij scheme (2). The root mean square deviation was also used to compare the differences at end of cycle for all the isotopes studied in this exercise, as shown in Table (2.4). For the ^{235}U, the maximum difference is of 0,5% for the same scheme (2).

Finally, the two-level scheme has the lowest differences for the pin fission rates, as shown in Table (2.5). This can be explained by the use of the MOC method at the second level of the calculation, leading to the best behavior compared to the MOC scheme (3).

As a conclusion of these last comparison, the MOC scheme (4) and the two-level scheme (6) are kept for the final comparison with the other codes, the first one supposedly being the most accurate, and the second one being the best trade-off between accuracy and CPU time.

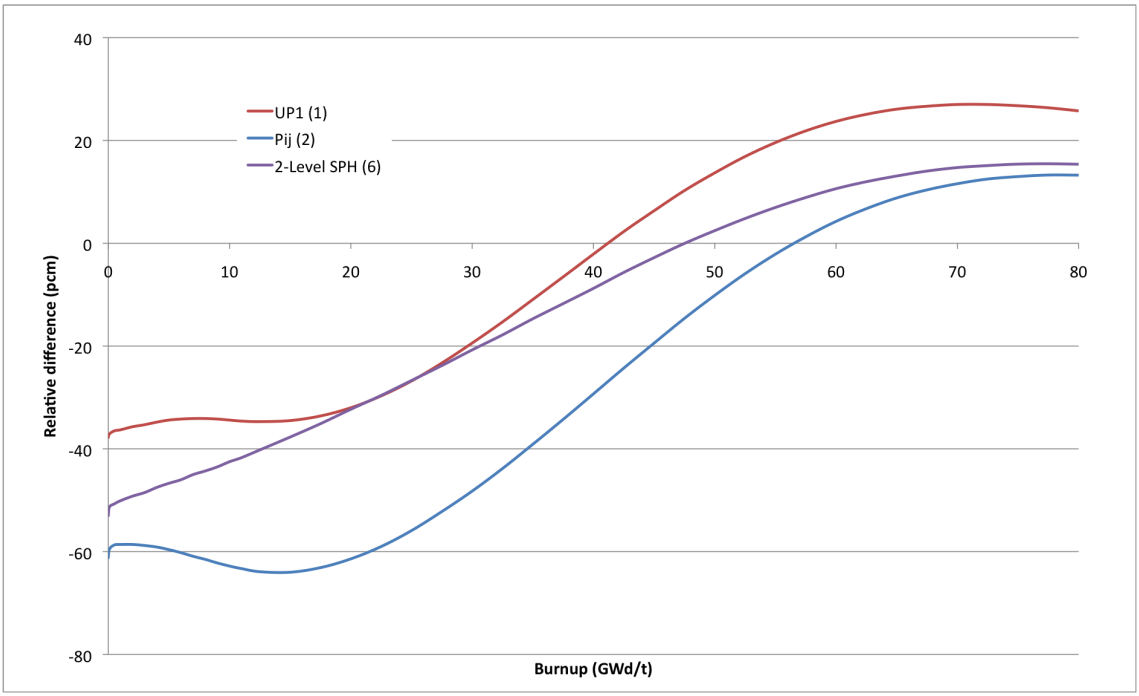


Figure 2.13 Relative difference between schemes (1), (2), (6), and the MOC scheme (3) (pcm)

	UP1	Pij	2-level (SPH)
Maximum absolute deviation (%)	0,43	0,63	0,45
Root mean square deviation (%)	0,16	0,24	0,18

Table 2.4 Differences between the schemes (1), (2), (6), compared with the MOC scheme (3) for isotopic densities at end of cycle

	UP1		Pij		2-level (SPH)	
	BOC	EOC	BOC	EOC	BOC	EOC
Maximum absolute deviation (%)	0,69	0,59	0,43	0,52	0,06	0,16
Root mean square deviation (%)	0,25	0,30	0,14	0,16	0,02	0,08

Table 2.5 Differences between the schemes (1), (2), (6), compared with the MOC scheme (3) for the pin fission rates at beginning of cycle (BOC) and end of cycle (EOC)

2.4 Comparison with others calculations

Now that all the previous verifications have been done, the final step is to compare the results from the DRAGON calculations with the results obtained using other codes. This final comparison is done using three different parameters : the multiplication factor, the isotopic densities, and the pin power.

2.4.1 Codes used for comparison

For this exercise, different organization participated, performing the calculations with different codes. All of these codes are based on different evaluations, which makes the comparison a little complicated.

Three codes were used by the GRS :

- KENOREST (Hesse et al., 2000) which is based on the code HAMMER (Suich and Honeck, 1967) for lattice calculations, the depletion code ORIGEN (Bell, 1973) and the Monte-Carlo code KENO (Petrie and Landers, 1983). The cross section libraries of KENO and of OREST are based on the JEF-2.2 evaluation.
- HELIOS (Casal et al., 1991), a 2D transport-theory code for neutronic and gamma-dose calculation in fuel assemblies and similar structures. This code uses an adjusted ENDF/B-VI based library.
- MONTEBURNS (Poston and Trellue, 1999) which links the Monte Carlo Neutral-Particle transport code MCNP (X-5 Monte Carlo Team, 2003) with the isotope generation and depletion code ORIGEN2 (Croff, 1980). It uses libraries based on a JEFF-3.1 evaluation.

Two other organizations participated to this project :

- TÜV SÜD with the code CASMO4 (Rhodes and Edenius, 2001) using libraries based on JEF-2.2 and ENDF/B-VI evaluations.
- IRSN with the code APOLLO2 (Hoffman et al., 1973) using APOLIB based on JEF-2.2 evaluation.

To facilitate the comparison, two parameters are used : the average and twice the relative standard deviation defined as :

$$\text{The average :} \qquad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.1}$$

$$\text{The standard deviation :} \qquad \sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{2.2}$$

$$\text{Twice the relative standard deviation :} \qquad 2\sigma_{rel} = \frac{2\sigma}{\bar{x}} \tag{2.3}$$

where x_i is the parameter of interest calculated by the participant i, and n is the total number of participants.

This parameters will help to compare the DRAGON calculations (complete MOC scheme (4) and two-level scheme (6)) with the other calculations, on the three parameters cited above.

2.4.2 Reference library

This exercise was made to compare results coming from different codes performing depletion calculations. These codes use different types of libraries, based on different types of evaluation. Because the DRAGON code is able to perform calculations using different types of libraries, and also DRAGON formatted libraries (draglib) based on different evaluations, a comparison is made to see the influence of those libraries on the results of the calculations.

For this exercise, only 172-groups (X-MAS 172) draglibs are used, because the purpose was to compare DRAGON to the other codes (so only DRAGON format libraries were selected), and because the small number of groups allows to reduce the calculation time. The comparison will be then limited to the type of evaluation used, and will only be performed on the multiplication factor.

As shown in Figure (2.14), the results are really sensitive to the type of evaluation used while comparing DRAGON to MONTEBURNS. The calculation performed with the library based on the JEFF-3.1 evaluation shows a better behavior at the beginning of cycle, but the difference grows afterwards. The difference may come from the fact that the libraries are not exactly the same. These results shows that it is very important to compare calculations performed with libraries that are the most similar.

The reference library chosen to be used for the next calculations was the DRAGON formatted library based on the JEFF-3.1 evaluation. This library is chosen because it is based on the same evaluation as the library used by the code MCNP for the MONTEBURNS calculation, which is supposed to be the most accurate calculation.

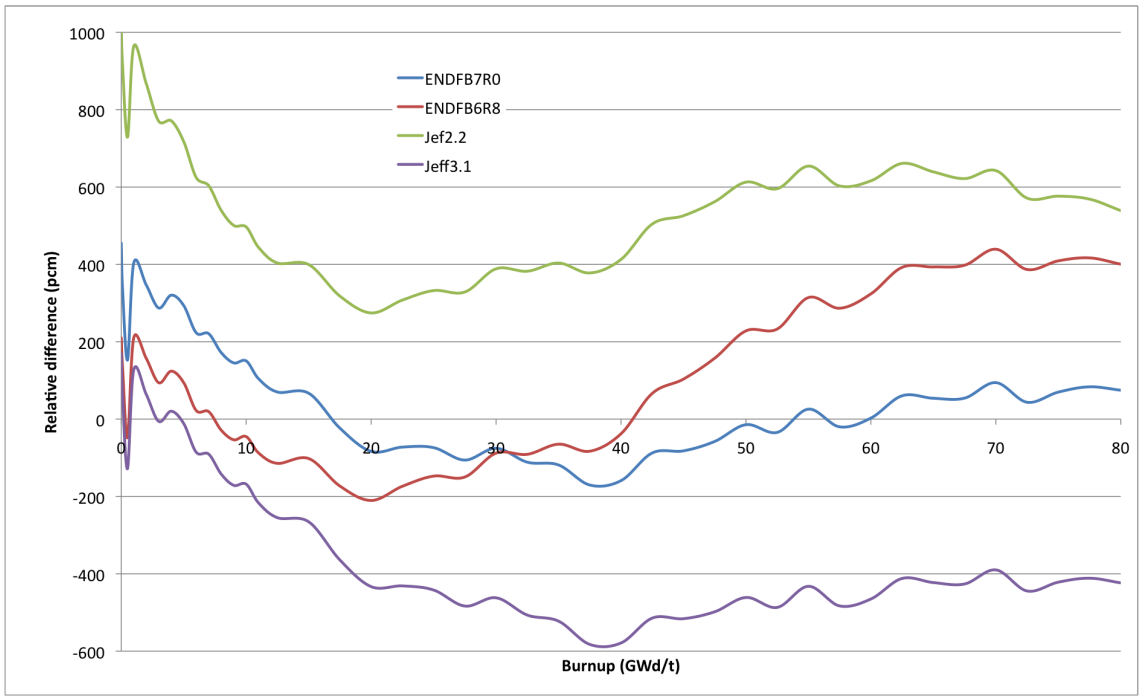


Figure 2.14 Relative difference between DRAGON and MCNP

2.4.3 Difference in k_{eff}

The first comparison is made on the effective multiplication factor. A first simple comparison is performed between all the codes, taking the MONTEBURNS calculation as reference as shown in Figure (2.15).

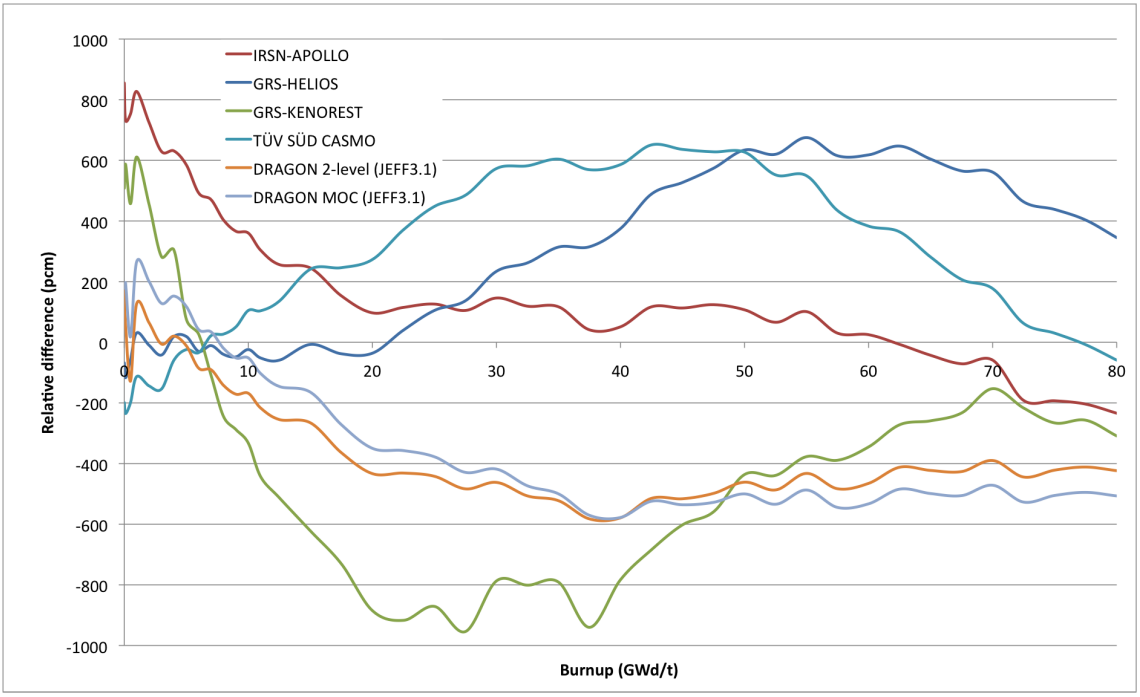


Figure 2.15 Relative difference between the different codes used, and the MONTEBURNS calculation (pcm)

We can see here that the spread is really wide (2000 pcm). The DRAGON calculations have a good behavior at the beginning, but at the end of cycle, the difference is larger, but still acceptable. The two-level scheme gives better results at the end of cycle, but the results from those two calculations are still comparable.

Now, comparing the DRAGON calculations to the average and the relative standard deviation, the two schemes show good results, lower than the average, but inside the spread of results, as shown in Figure (2.16).

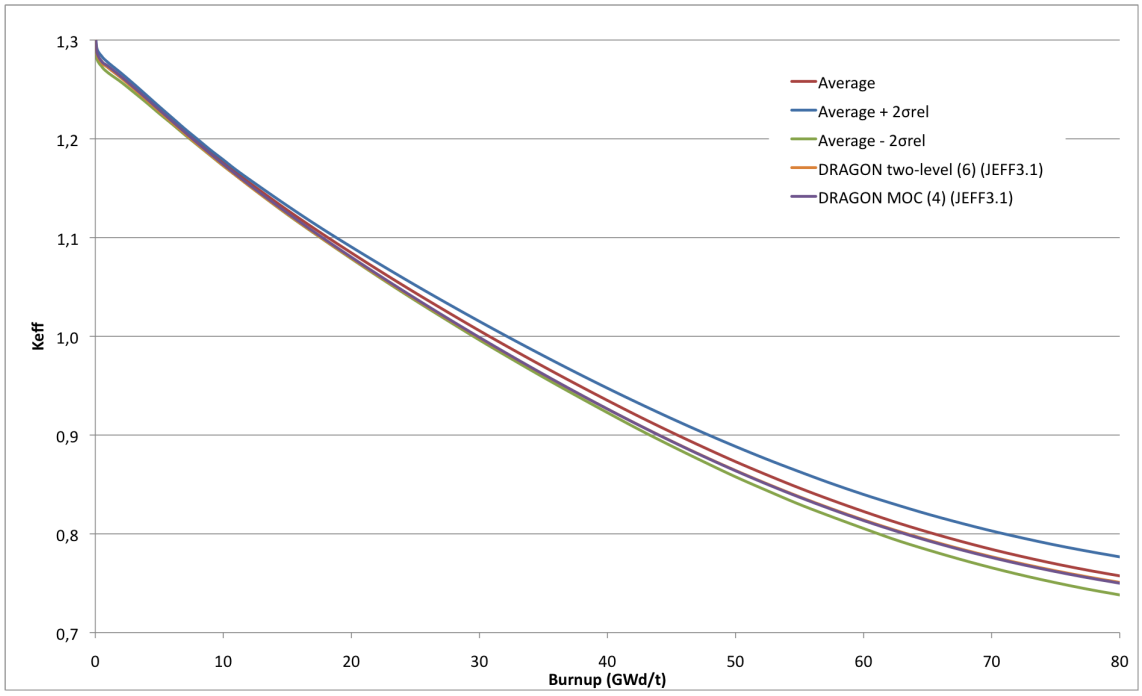
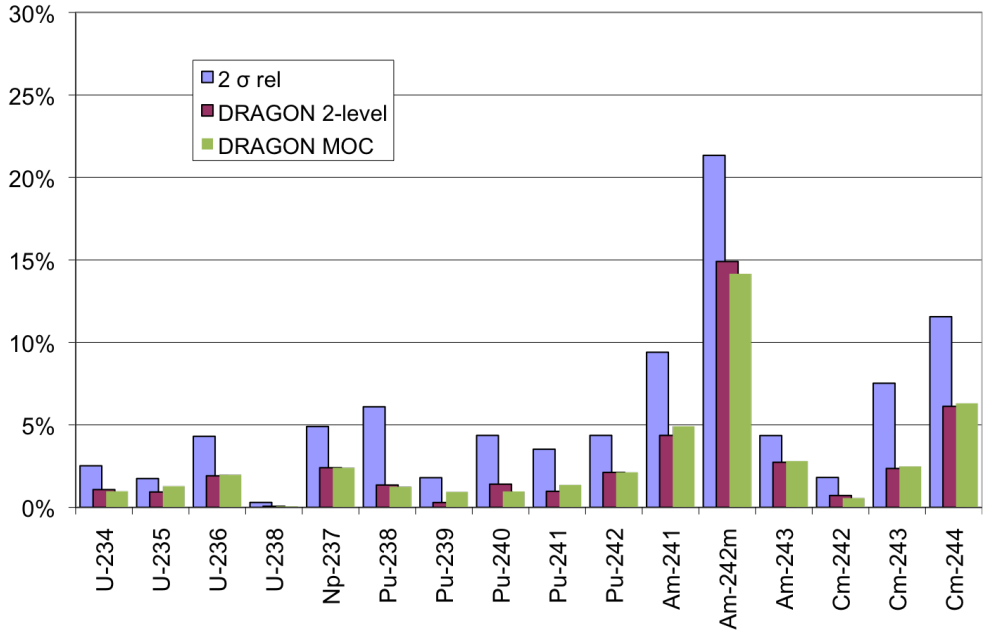


Figure 2.16 Spread in k_{eff} : comparison between DRAGON and the average

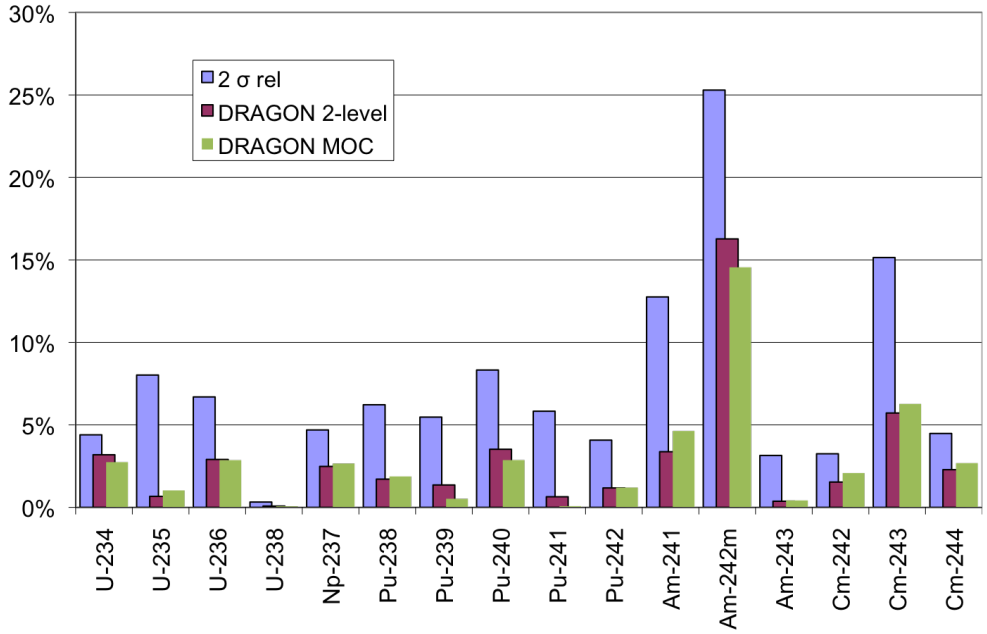
For this parameter, the DRAGON calculations show a good behavior, first in comparison with the MONTEBURNS calculation, even if the differences at end of cycle are a little high but still acceptable, and secondly compared to the spread of all the other calculations. Also, the spread between the two DRAGON schemes is small compared to the overall spread.

2.4.4 Isotopic densities differences

Another way to compare the accuracy of the DRAGON code is to take a look at the nuclide densities. One can then compare the depletion and the formation of the different isotopes. Here again, the DRAGON calculations are in good agreement with the other calculations.

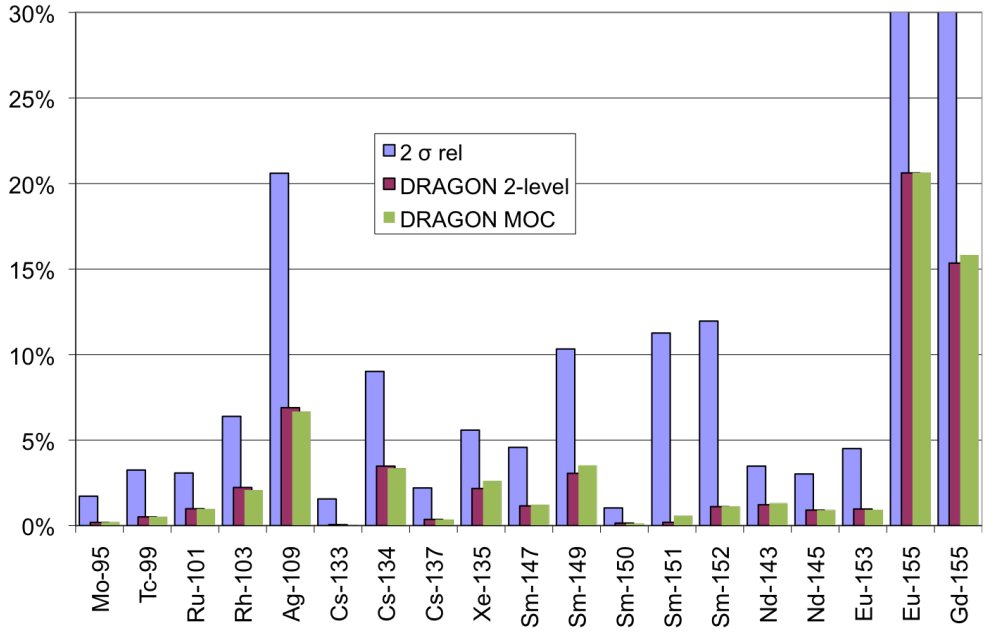


(a) At middle of cycle

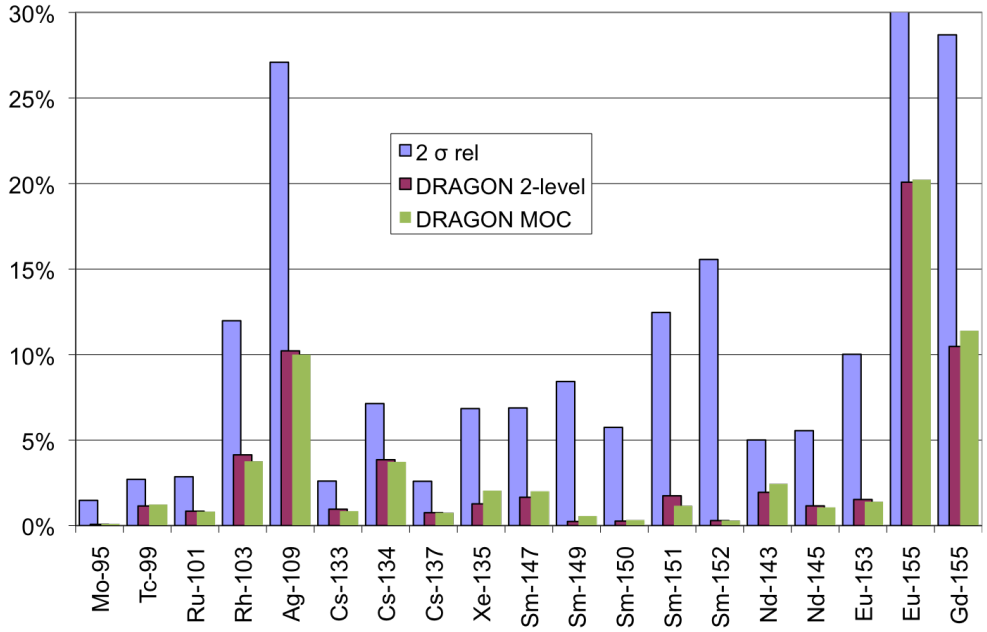


(b) At end of cycle

Figure 2.17 Spread in actinides : comparison between DRAGON and the average (%)



(a) At middle of cycle



(b) At end of cycle

Figure 2.18 Spread in fission products : comparison between DRAGON and the average (%)

Figure (2.17) shows the relative difference between each one of the two DRAGON calculations and the average mean value of the nuclide densities for some important actinides at two different burnup steps : middle of cycle (40 GWd/t), and end of cycle (80 GWd/t). The isotope having the biggest difference is the ^{242m}Am. Not taking this isotope into account, the differences are always lower than 7%. The DRAGON schemes are always inside the spread of all results.

Figure (2.18) shows the same relative differences except for some important fission products. In this figure, one sees big differences for the ^{155}Eu and the ^{155}Gd. This effect is mainly due to the CASMO-4 results that differ considerably from the average value obtained by all the other codes. As a result the average value is distorted towards CASMO-4 making all the other codes look bad. If the CASMO-4 results are removed from the average, the standard deviation $2\sigma_{rel}$ decreases to 10 % for those two isotopes at middle of cycle, and at 10% for the ^{155}Eu and 25% for the ^{155}Gd at end of cycle. Changing this, the DRAGON calculations still remains inside the spread.

For this second comparison, the DRAGON code still shows good results in comparison with the other codes. The two-level scheme gives results comparable to the complete MOC scheme, which prove its accuracy.

2.4.5 Pin power differences

Finally, the last comparison concerns the pin power distribution. The comparison is made on the normalized peaking factor, and its position in the assembly.

The peak position in the assembly is the same for all the codes (from the left top corner, seventh pin to the right, and fifth pin down). Figure (2.19) shows the normalized peaking factor at beginning of cycle, and end of cycle. The DRAGON code agrees well with the other codes, the two-level scheme and the MOC scheme showing a comparable behavior.

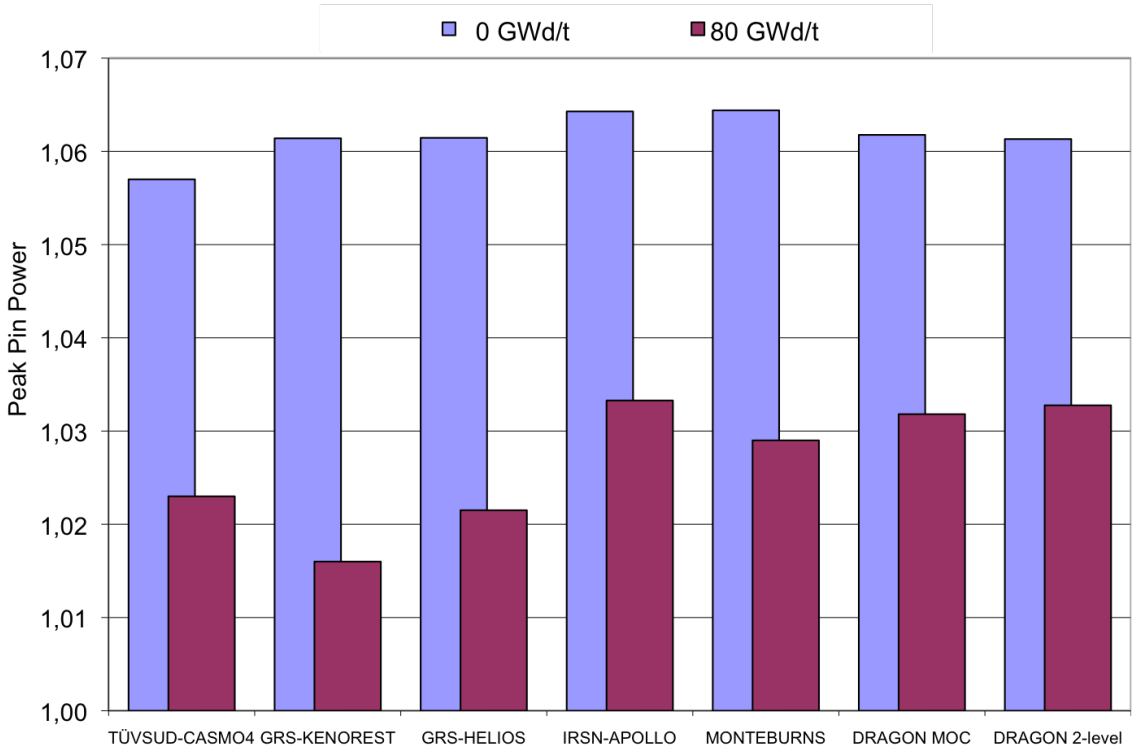


Figure 2.19 Comparison between all the codes on the normalized peaking factor

All these comparisons show that the DRAGON code has a good behavior in comparison with the other codes used in this exercise. Even if for some parameters the differences can be high, they are still acceptable. Also, the two-level scheme appears to be a good trade-off between accuracy and CPU time, giving good results in a very short CPU time. But this exercise also points out that one has to carefully chose the library to use in the calculations, in order to perform the best comparison possible.

CHAPTER 3

OECD/NEA AND U.S. NRC PWR MOX/UO_2 CORE TRANSIENT BENCHMARK

The last part of the work focuses on a PWR MOX/UO_2 core transient benchmark (Kozlowski and Downar, 2003). First, calculations are made for a single fuel cell, to perform some verifications of DRAGON in comparison with GRS calculations. Then, fuel assembly calculations are performed, and compared with the HELIOS results. Finally, parameterized cross section library are generated by DRAGON for use with the GRS coupled code system QUABOX-CUBBOX/ATHLET (Langenbuch and Velkov, 2004).

3.1 Description of the benchmark

The benchmark was designed to evaluate the ability of modern reactor kinetic codes to predict the transient response of a core partially loaded with MOX fuel. The purpose of the work here is to produce cross section libraries for whole-core calculations.

This benchmark was proposed by T. Kozlowski and T. J. Downar from Purdue University. It deals with 17x17 PWR MOX and UO_2 assemblies, to perform steady-state and core transient calculations : control rod ejection and boron dilution transients, the latter being a proposal of GRS (Velkov et al., 2009). The core configuration is shown on Figure (3.1), composed of different types of fuel assemblies, with different enrichments for UO_2 and different Pu_{fiss} contents in the MOX asemblies, and at various burnup state.

In this framework, only fuel assembly calculations are studied. The core is composed of two different types of 17x17 PWR assemblies. The UO_2 assembly layout is presented in

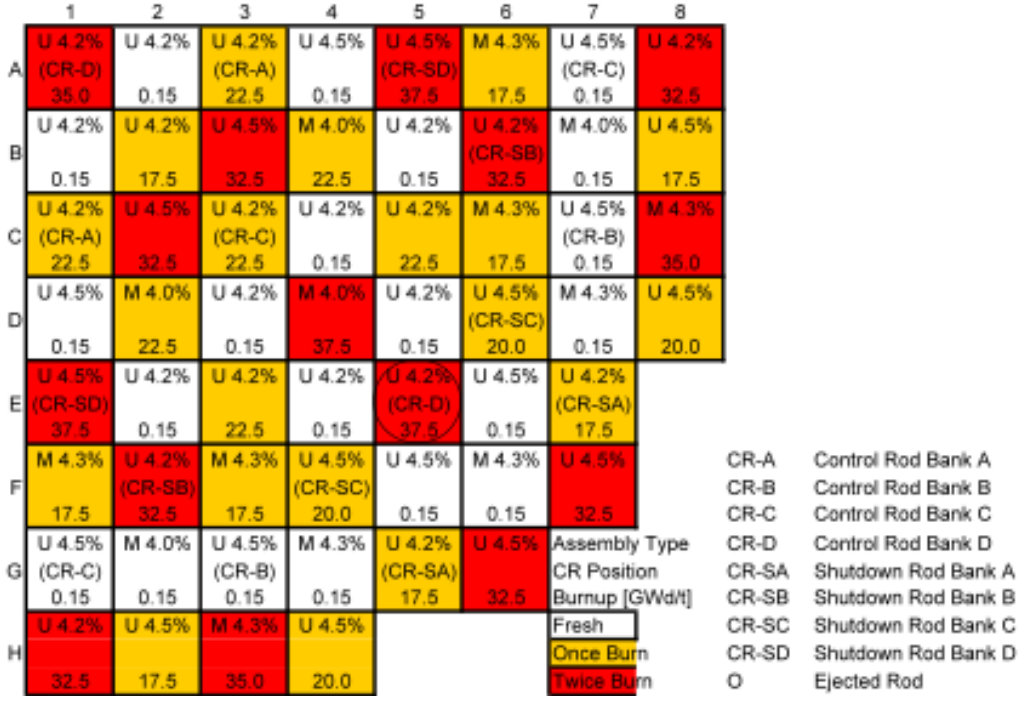


Figure 3.1 Core configuration (1/4 core)

Figure (3.2) and the MOX assembly layout in Figure (3.3) :

- The UO_2 assembly is composed of a central guide tube, 24 guide tubes also designed for control rods, 104 Integral Fuel Burnable Absorber (IFBA) pins, and 160 fuel pins. The IFBA pins compensate excess reactivity of fresh UO_2 fuel. They have the same design as the fuel pins, but the fuel coated with zirconium diboride (ZrB_2), as shown in Figure (3.4). This assembly is considered at two different enrichments : 4,2% and 4,5% of ^{235}U. Both fuel assemblies are present in the core in both controlled and uncontrolled state, i.e. four different UO_2 fuel assemblies need to be studied. The control rod cell is presented in Figure (3.4).

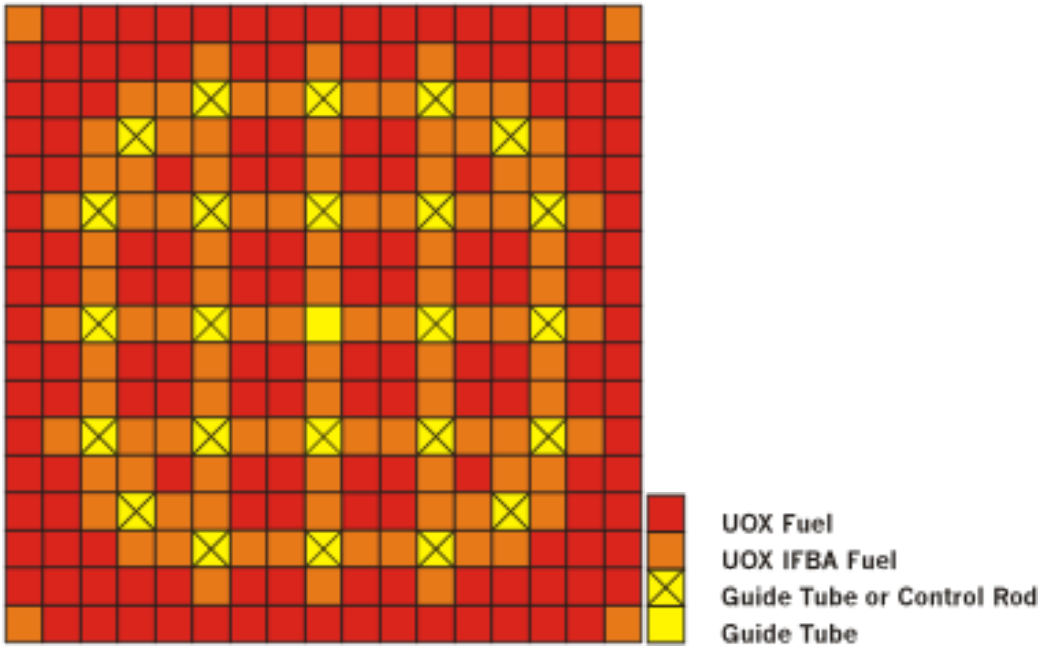


Figure 3.2 UO_2 fuel assembly

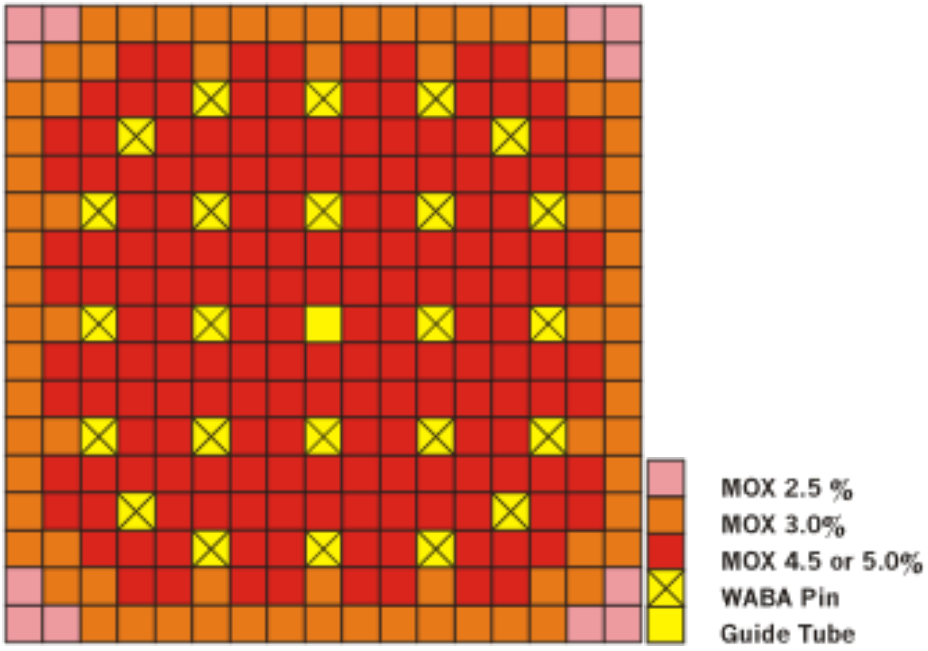
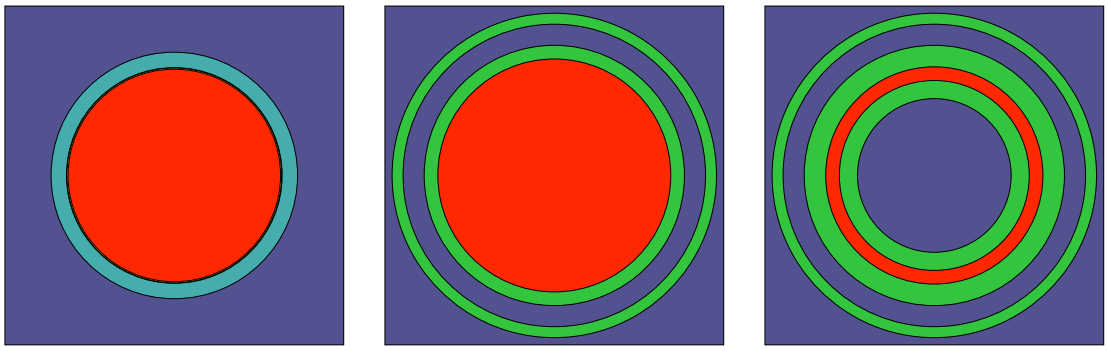


Figure 3.3 MOX fuel assembly



(a) IFBA cell (b) Control rod cell (c) WABA cell

Figure 3.4 Special cells design

- The MOX assembly is composed of one central guide tube, 24 Wet Annular Burnable Absorber (WABA) pins, 12 MOX pins with 2,5% of Pu-fissile (^{239}Pu and of ^{241}Pu), 76 MOX pins with 3,0% contents, and 176 MOX pins with either 4,5% or 5,0%, depending on the enrichment conditions. In fact, this assembly has a global enrichment of : 4,0% (with 176 4,5% MOX pins), and 4,3% (with 176 5,0% MOX pins). The WABA pins are composed of Al_2-O_3-B_4C with wet (water-filled) central region and Zircaloy cladding, as shown in Figure (3.4), and are used for long term reactivity control of the MOX assemblies.

For the first fuel assembly calculations, each assembly is considered at hot full power state. The moderator is light water with a boron concentration of 1000 ppm, at a temperature of 580°K and a pressure of $15,5$ MPa. The cladding of both fuel and guide tubes is made of zircaloy-2 (ZRY-2) composed of zirconium, iron, chromium, nitrogen and tin at a temperature of $600^{\circ}K$. The fuel is supposed to be at 900°K, in a xenon-free state at beginning of cycle with a fuel power of $37,87$MW/t. Later, some parameters will be modified in order to create the reactor database.

3.2 Single UO_2 fuel cell calculations

The first study made for this benchmark is based on a single UO_2 fuel cell (without IFBA coating). The purpose is to verify the DRAGON results by comparing them with the previously generated GRS HELIOS calculations, and perform some verifications and improvements on the calculation scheme before starting the calculations on a whole assembly, and further, generating cross sections for 3D-calculations.

3.2.1 Discretization and model

The study focusses first on a UO_2 fuel cell. In order to perform the best comparison with HELIOS, exactly the same discretization of the fuel and the moderator is used for the calculations. The fuel is divided in five rings with equal volumes, having different physical mixtures to take into account the depletion effects, and the moderator is divided in four zones, as shown in Figure (3.5).

For the burnup, the steps given in the benchmark and listed in Table (3.1) are used at first. Some steps have been added afterwards to improve the isotopic evolution, as important poison materials are produced during a short period at beginning of stage. This will be explained in Section 3.2.3.1.

The comparisons for this first part will be done on the infinite multiplication factor, the calculations being performed in DRAGON's fundamental mode. Some improvements have then to be done in order to obtain a fast calculation scheme and accurate results for the assembly calculations.

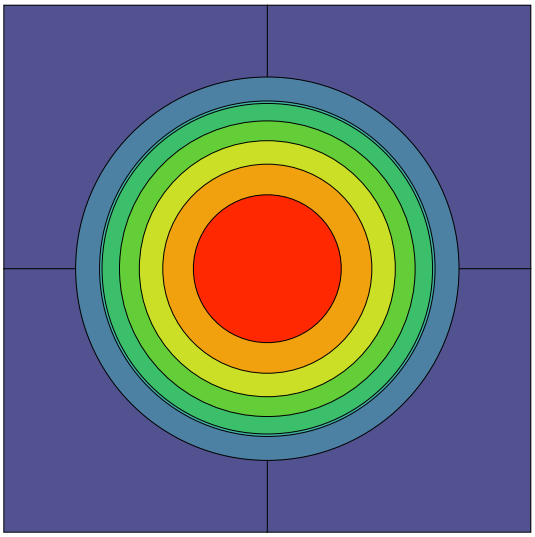


Figure 3.5 UO_2 fuel cell spatial discretization

Index	Burnup [GWd/t]	Index	Burnup [GWd/t]
1	0.00	18	20.00
2	0.15	19	22.50
3	0.50	20	25.00
4	1.00	21	27.50
5	2.00	22	30.00
6	3.00	23	32.50
7	4.00	24	35.00
8	5.00	25	37.50
9	6.00	26	40.00
10	7.00	27	42.50
11	8.00	28	45.00
12	9.00	29	47.50
13	10.00	30	50.00
14	11.00	31	52.50
15	12.50	32	55.00
16	15.00	33	57.50
17	17.50	34	60.00

Table 3.1 Reference burnup steps for calculations

3.2.2 Verification of the calculation scheme

As it was said before, the purpose of this benchmark is to create a cross section database to be used in a core simulator. The generation of cross sections must be optimized in terms of CPU time, without loosing significant accuracy. The main performance parameter is then the calculation time, because many evolution calculations will have to be performed on the same assembly with different parameters. In order to do so, and before making comparisons with the HELIOS calculations, some verifications are performed in order to reduce the calculation time, while still giving a good accuracy.

3.2.2.1 Self-shielding

The self-shielding calculation is the part of the scheme which takes the most CPU time. It was shown in the previous exercise that using a MOC method on an assembly calculation costs a lot of time, without improving too much the accuracy. So for this benchmark, it is performed using a UP1 anisotropic interface current method. The SYBILT: module is used to perform the tracking of the geometry. Again, the number of points for the angular and spatial integration was optimized. The calculation is then performed by the USS: module at every burnup step, using a subgroup approach with physical probability tables as it is done in HELIOS.

To reduce the calculation time, one possibility is to perform the self-shielding calculation only for some burnup steps (punctual self-shielding), and not every step (continuous self-shielding), as recommended in (Santamarina et al., 2004). In fact, while performing this calculation only every four burnup step, the calculation time is divided by two in our case. The problem is the k_∞ values oscillate about the values obtained with a self-shielding calculation performed at every step as shown in Figure (3.6).

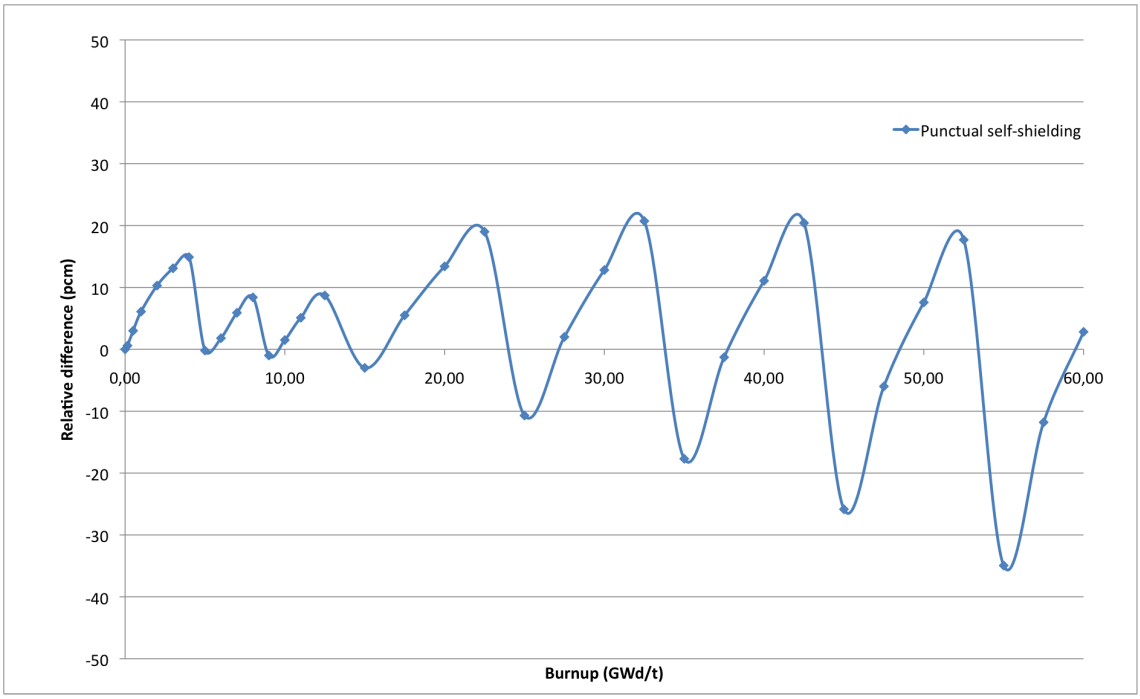


Figure 3.6 Relative difference in k_∞ between continuous and punctual self-shielding

Note that the HELIOS calculations are performed with a continuous self-shielding. So, even if the difference on the k_∞ is small, and the gain in time is large, the punctual self-shielding was not kept, in order to follow more closely the HELIOS calculations.

3.2.2.2 Flux calculation

As for the previous exercise, different models can be used for the flux calculation. Three are studied in this part, all of them solving the flux by using the fixed effective multiplication factor option (B type calculation) with a B1 leakage model :

- a UP1 anisotropic interface current method, using the SYBILT: module.

- a MOC calculation using the NXT: module with specular tracking parameters, followed by the MCCGT: module for the use of the characteristic method.

- a two-level scheme, with a UP1 calculation at the first level, and a MOC calculation for the second one. The use of an SPH equivalence has also been studied, leading to the same conclusions as in Section 2.3.4.2 : the SPH equivalence gives better results compared with the MOC calculation, as shown in Figure (3.7).

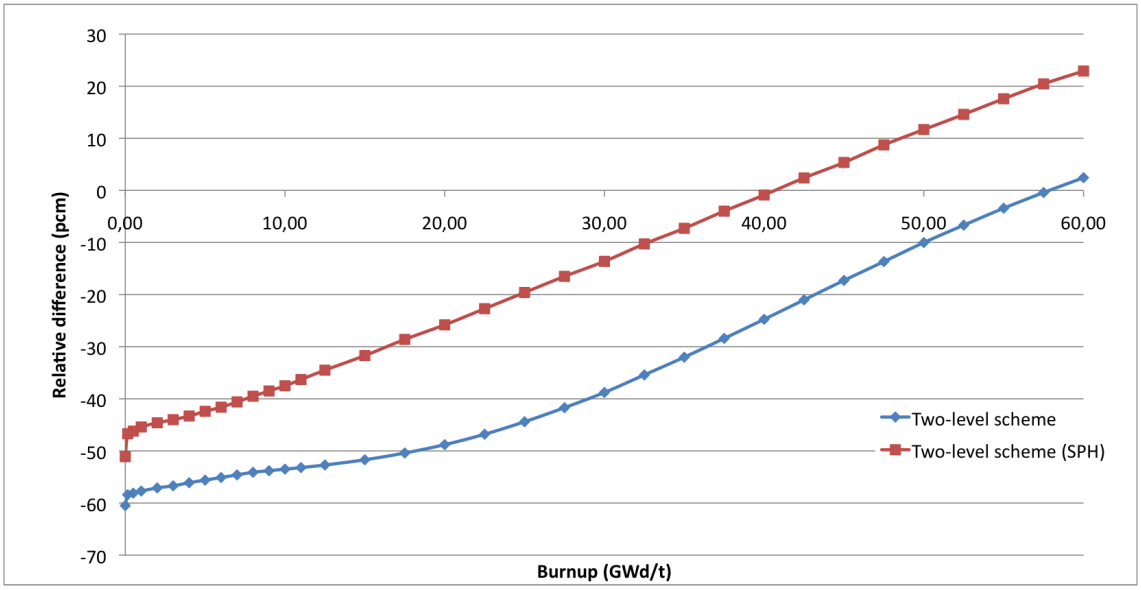


Figure 3.7 Relative difference between the two-level schemes and the MOC calculation

Here, for a one-cell calculation, the difference in CPU time between those models is very small (1 or 2 minutes over a 25 minutes calculation). And so is the relative difference on the k_∞, taking the MOC calculation as reference (with a maximum of 60 pcm for the two-level scheme without SPH equivalence). All those calculations giving almost the same results, only the two-level scheme is kept for the final comparison with HELIOS, this scheme being the most interesting.

3.2.3 Comparison with Helios Calculations

A comparison between the results obtained with HELIOS at the GRS, and the results from DRAGON on this cell can then be performed. Only the k_∞ have been compared here. This first comparison showed a problem to be solved. Then, different libraries have been used, in order to find the one giving the best results.

3.2.3.1 Comparing the infinite multiplication factors

The HELIOS calculations have been performed using three different libraries based on an ENDF/B-VI evaluation : 47 groups adjusted/unadjusted, and 190 groups adjusted.
The difference arises from the number of groups used, and modifications made in the 'adjusted' libraries : the capture integral of the ^{238}U has been modified to fit the results to operation data of LWR.
For the DRAGON calculation, a library based on the ENDF/B-VII Release 0 evaluation was used first. The comparison between this calculation and the results from HELIOS is shown in Figure (3.8).

The differences are quite large for a single-cell calculation, but so are the differences between the HELIOS calculations. The DRAGON calculation can be considered acceptable. The only problem is at beginning of cycle, where an oscillation occurs.

In order to get rid of this oscillation, some burnup steps have been added before the first burnup step. At first, three burnup steps are added, as recommended in (Santamarina et al., 2004) : 37, 75 and 112.5 MWd/t. Finally, only adding one of these burnup steps (75 MWd/t) permits to obtain a good behavior, as shown in Figure (3.9) in comparison with the HELIOS calculation using the 190 groups adjusted library (the effect being the same with the others). This step will be added for the rest of this benchmark.

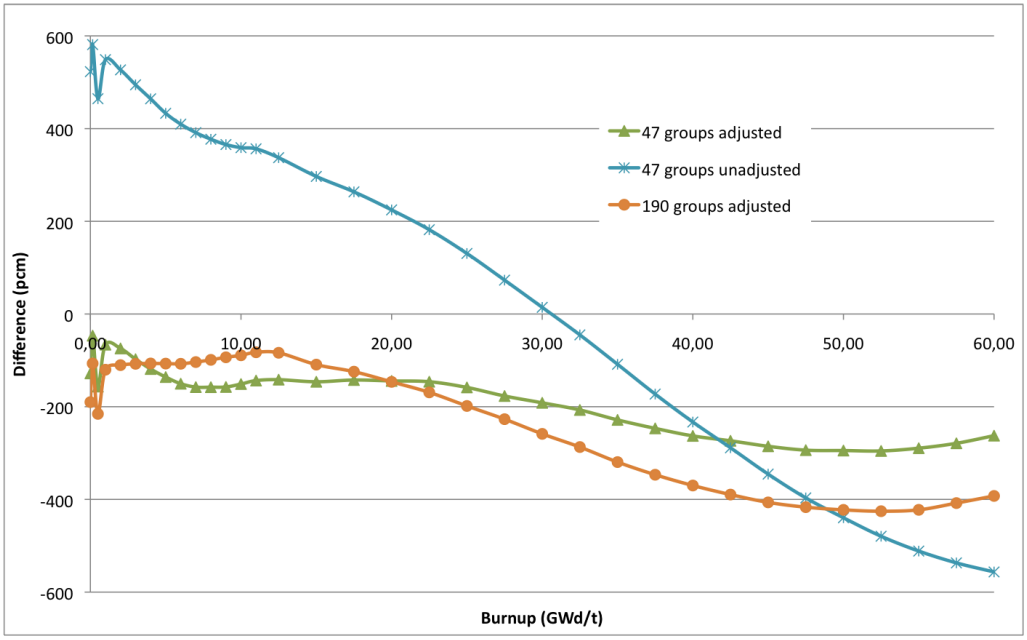


Figure 3.8 Relative difference between HELIOS calculations, and DRAGON

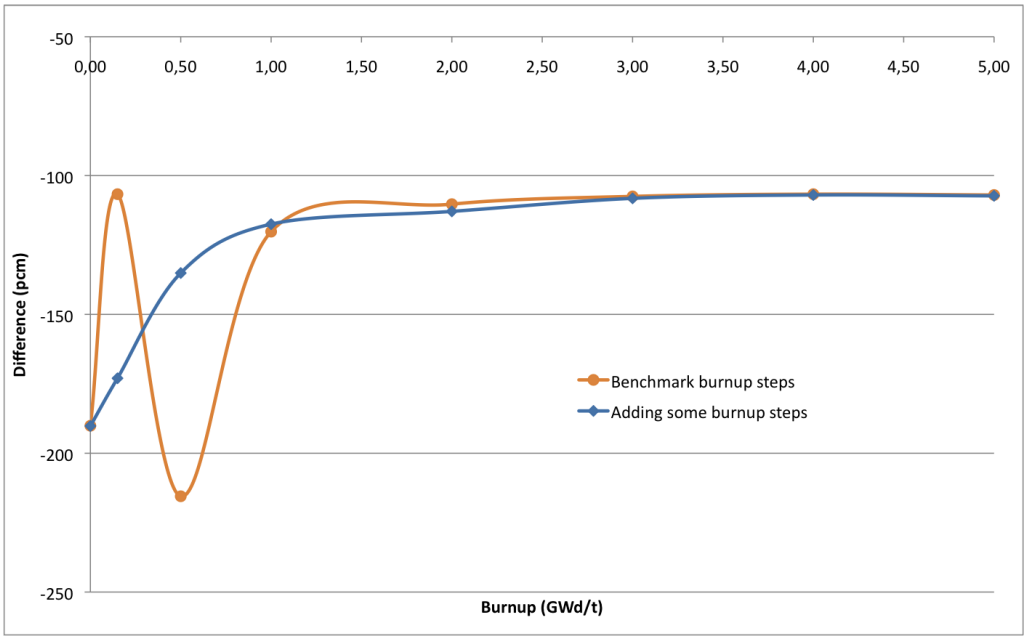


Figure 3.9 Relative difference between DRAGON and HELIOS 190 groups adjusted

3.2.3.2 Different evaluation based libraries

Finally, a comparison is made between the different evaluations used to produce the DRAGON libraries. In fact, HELIOS uses libraries based on a ENDF/B-VI evaluation, but some are adjusted. So, in order to have a good comparison, two DRAGON calculations are first performed using libraries based on ENDF/B-VI Release 8 and ENDF/B-VII Release 0.

While comparing the HELIOS calculations to the DRAGON ones, it appears that the HELIOS adjusted libraries are closer to the DRAGON library using a ENDF/B-VII Release 0 evaluation, and the unadjusted library shows a better comparison with the DRAGLIB using a ENDF/B-VI Release 8 evaluation, as shown in Figures (3.11) and (3.12).

For the assembly calculations, an adjusted library is used in HELIOS. Also, for the multiparameter reactor database, the highest burnup point used is 37.5 GWd/t. So, for the rest of the calculations, the DRAGON library based on a ENDF/B-VII Release 0 evaluation will be used, showing the best comparison, as shown in Figure (3.10).

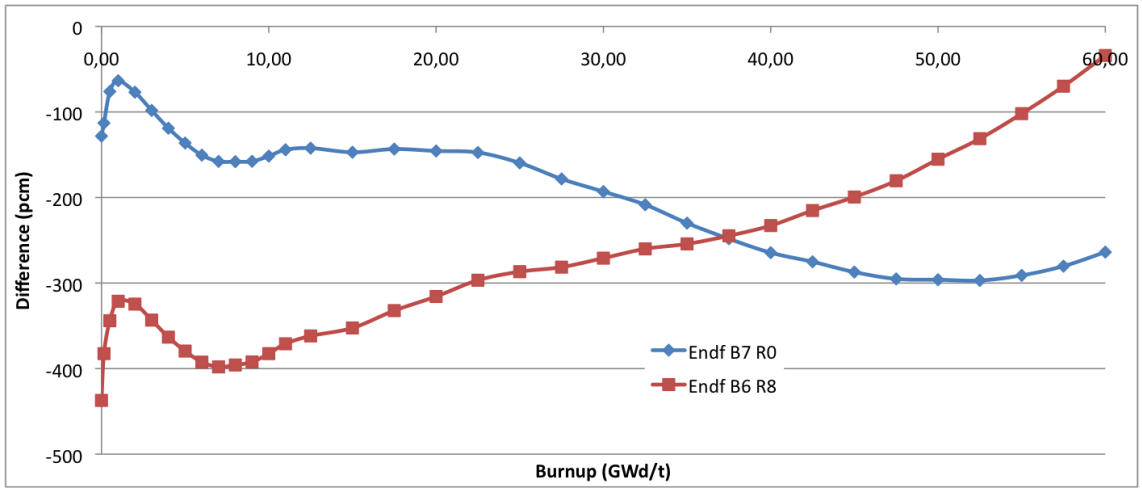


Figure 3.10 Relative difference between DRAGON and HELIOS (190 groups adjusted)

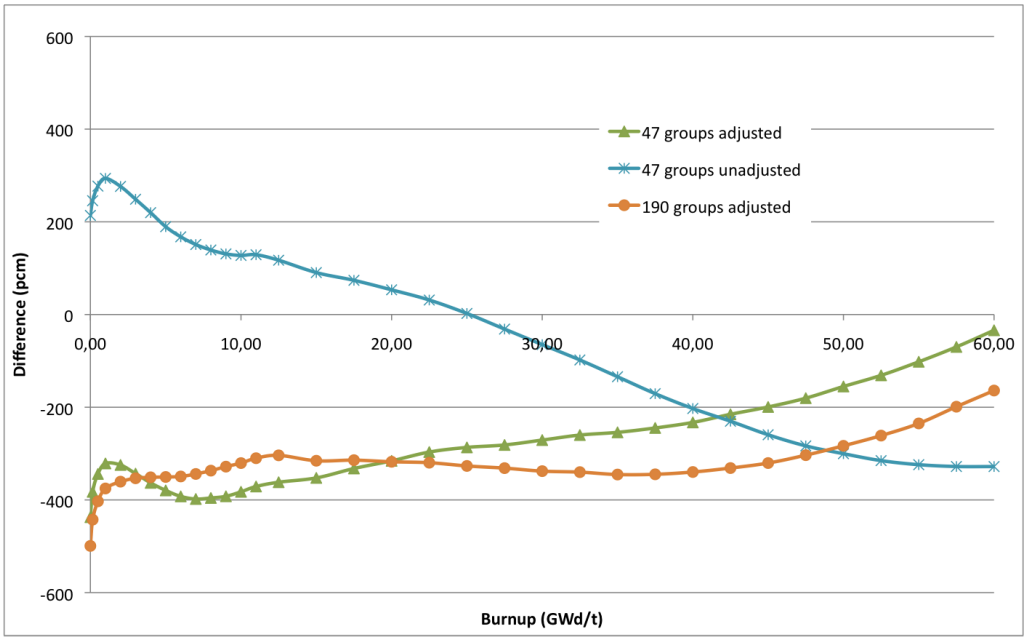


Figure 3.11 Relative difference between HELIOS and DRAGON (ENDF/B-VI R8)

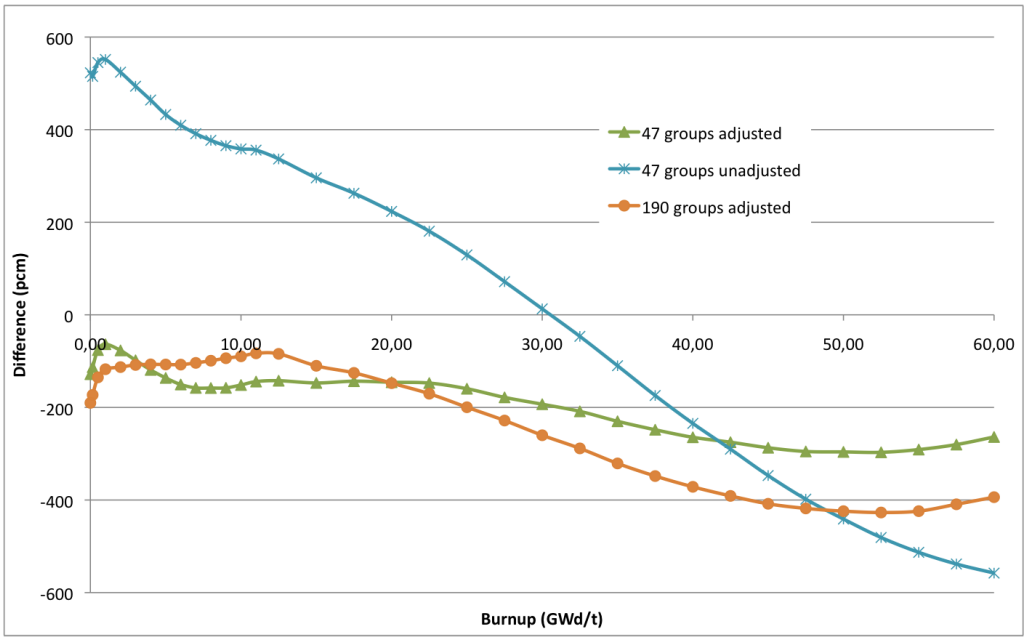


Figure 3.12 Relative difference between HELIOS and DRAGON (ENDF/B-VII R0)

3.3 Fuel assembly calculations

After dealing with only single cell, the next step is to perform fuel assembly calculations. Four different assembly calculations have to be performed (two different assemblies, each one with two different enrichments, the control rodded assemblies will not be studied in this part). Here again, the purpose is to make some comparisons with the HELIOS calculations before the creation of the multiparameter database.

3.3.1 Model, discretization and regroupments

For this benchmark, two different types of assembly are studied, presented before in Figures (3.2) and (3.3). As for the previous study, the symmetries in an infinite lattice allow to only deal with an octant of the assembly. This is once again very important, in order to reduce considerably the CPU time.

As for the single-cell calculation, the same discretization as for the HELIOS calculation is used. The fuel is divided into five radial regions of equal volume, the moderator outside the tubes is divided into four, and the moderator inside the guide tubes is divided into three regions of equal volumes. The two discretized assemblies are presented in Figures (3.13) and (3.14).

For these two assemblies, regroupments are considered, as it was done in the previous study. They are used to reduce the number of mixtures, and, at the same time, to merge cells for the self-shielding calculation using a UP1 method. Only one regroupment by assembly is studied, following the recommendations from (Le Mer, 2007). Those regroupments are presented in Figure (3.15), and distinguish :

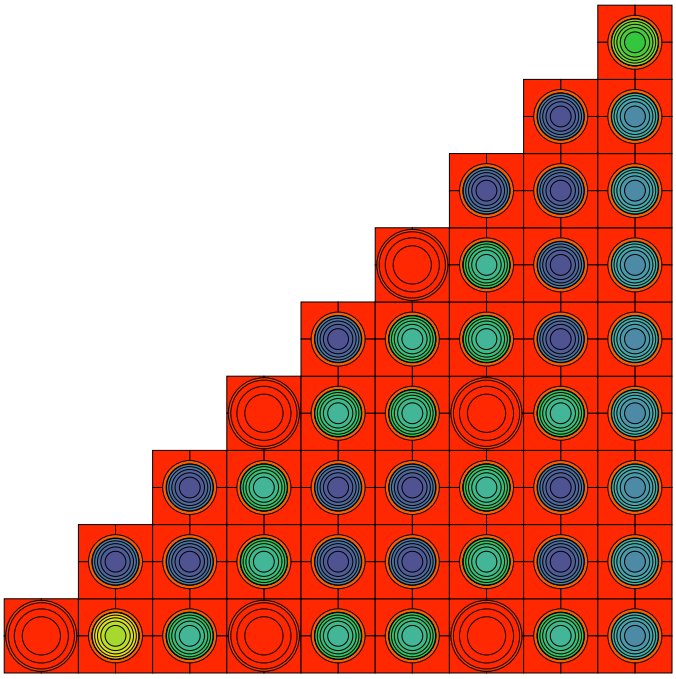


Figure 3.13 Discretization of the UO_2 fuel assembly

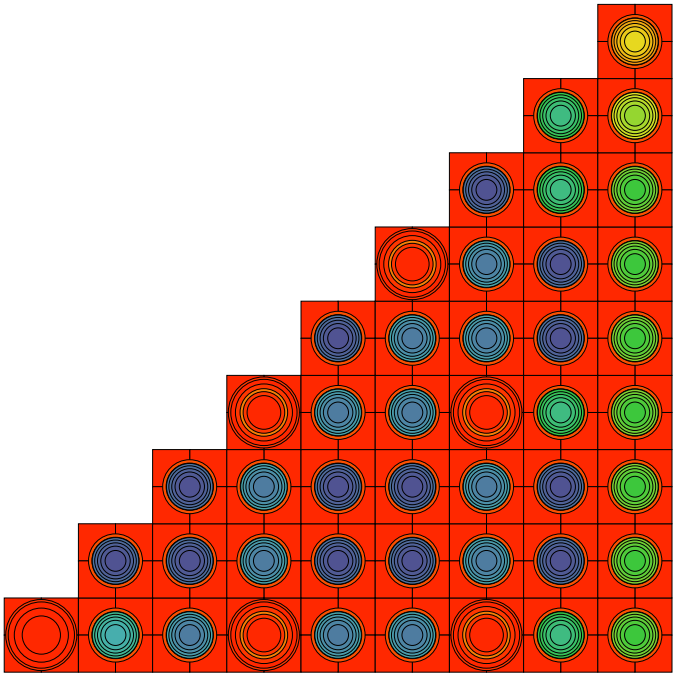
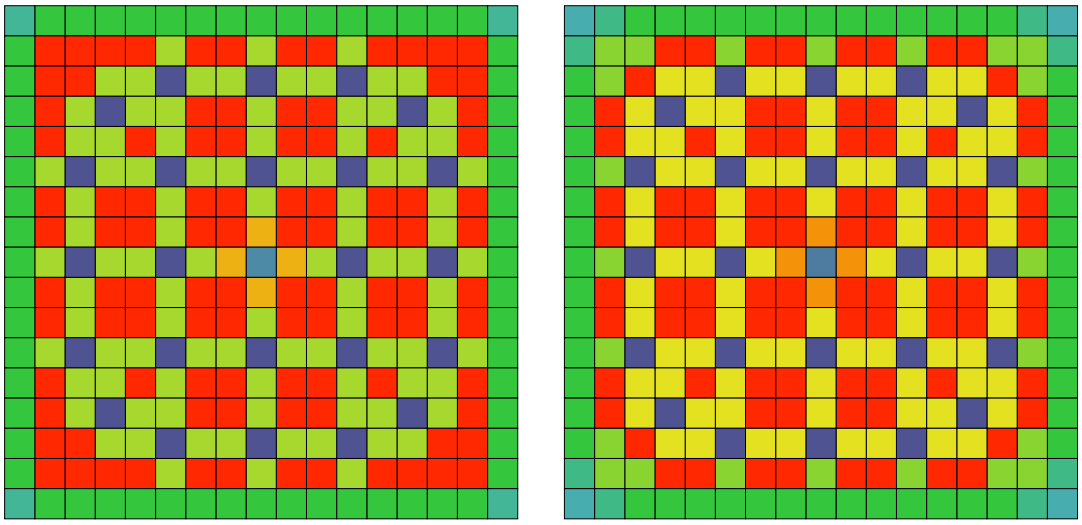


Figure 3.14 Discretization of the MOX fuel assembly

- the cells in the corner of the assembly (1),
- the cells on the border of the assembly (2),
- the cells sharing a face with the central guide tubes (3),
- the cells sharing a face with the other guide tubes (4),
- the other cells (5).



(a) For the UO_2 assembly (b) For the MOX assembly

Figure 3.15 Regroupments used in the calculations

There will then be 30 different mixtures (25 fuel mixtures) for the UO_2 assembly, and 40 different mixtures (35 fuel mixtures) for the MOX assembly.

Here again, a reference calculation is performed for the two types of assemblies, where the burnup in each ring of each cell is treated individually. The comparison is made considering k_∞ evolution of the assemblies The results show that the differences on the infinite multiplication factor are always lower than 50 pcm in both cases. The reduction of time being considerable, therefore the two regroupments shown in Figure (3.15) will be used for the next steps of this study.

3.3.2 Calculation scheme

The self-shielding calculations are performed using the same interface current method as for the previous exercise. The SYBILT: module is then used to perform the tracking of the geometry and a sufficient number of points for the angular and spatial integration has been chosen. The calculation is performed by the USS: module at every burnup step, using a subgroup approach with physical probability tables which is the same method as the one used in HELIOS. Then, the flux calculation is performed using the same two-level scheme as for the last study, but with different options :

- First-level calculation : the flux is solved by the UP1 interface current method, using the fixed effective multiplication factor option (B type option), and with a B1 leakage model. A simplified geometry is used, where the moderator is not discretized, and where the regroupments are used to merge cells together.
- The cross sections are then condensed to a smaller number of groups (26 groups). During this condensation, an SPH equivalence is performed.
- Second-level calculation : the flux is solved by a MOC method, using a fixed buckling option (K type) where the buckling calculated at the first level is imposed, and with a B1 leakage model. It is faster than using the B type option, and give almost the same results (less than 1 pcm difference). The moderator is here discretized.

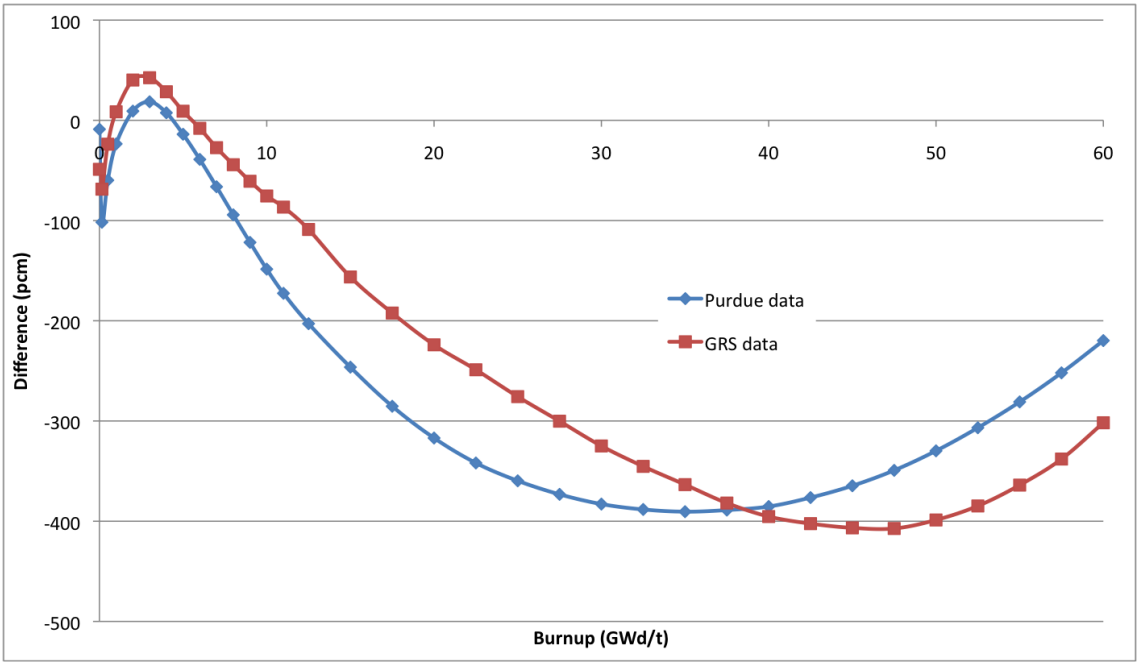
The previous study proved the accuracy and the fast calculation time of such a scheme, but a simple verification is needed to confirm this choice. Thus, for each assembly, two calculations are performed, each one using a different method to solve the flux : a UP1 method, and a MOC method. The results show differences on the k_{inf} always lower than 60 pcm between the two-level scheme and those schemes. Also the calculation time is two times shorter compared to the MOC calculation. The two-level scheme seems here again to be the best choice.

3.3.3 Comparison with HELIOS calculations

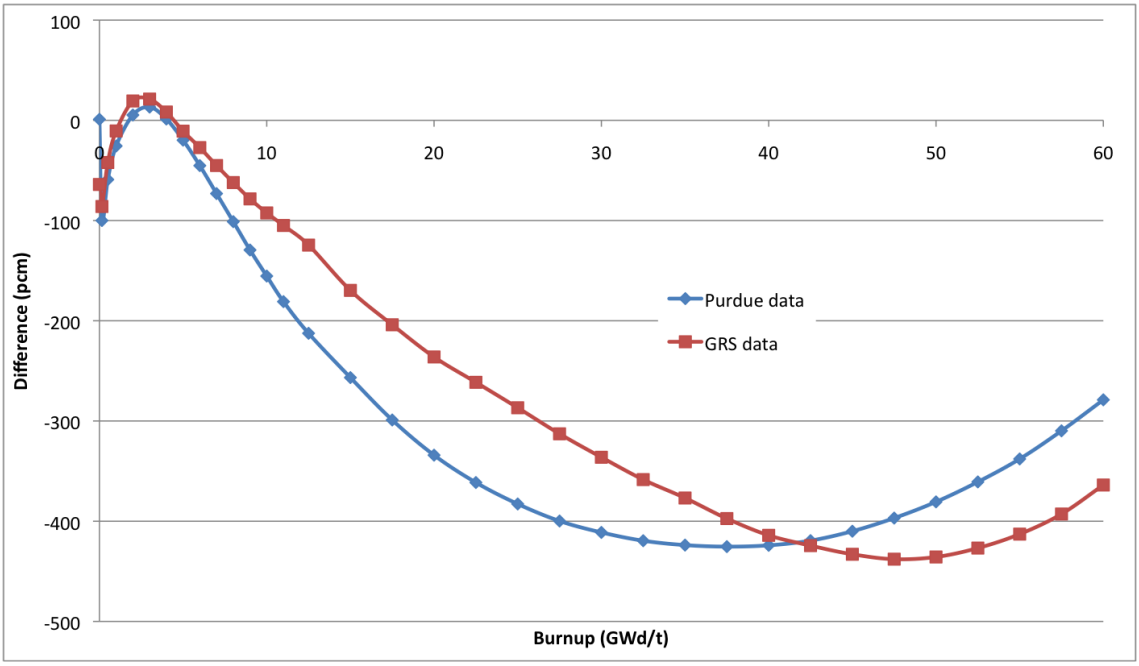
For this part, two different sets of results are used : the original results from Purdue University, and the results from GRS, both calculated by HELIOS. The comparison is performed only on the infinite multiplication constant, which gives a good overview of the accuracy of the calculation. The two types of assembly are studied, and for each assembly, the two different enrichments or Pu_{fiss} contents are used. For those four comparisons, the DRAGON calculation is taken as reference, and is compared with the two sets of data :

- For the UO_2 assembly, the comparisons are shown on Figure (3.19). The results show good agreement with the two sets of results. The relative difference between DRAGON and the two HELIOS calculations is always lower than 450 pcm. It can be pointed here that the two HELIOS calculations give comparable results.
- For the MOX assembly, the comparisons are shown on Figure (3.17). The results do not agree as well as for UO_2 assemblies. The relative difference reaches a maximum of almost 900 pcm for the comparison with the results from GRS, and a maximum of 700 pcm compared to the results from Purdue University. Also, the two HELIOS calculations give different results, with a maximum difference of 450 pcm at the beginning.

The two types of assemblies show different behavior. It seems here that the MOX assembly is more sensitive than the UO_2 one. In fact, the presence of heavy resonant isotopes in the fuel makes the self-shielding calculation more sensitive, because of the resonant cross sections, and also because the number of groups used in the libraries is different. But even with this difference, the results from DRAGON are still acceptable, and this scheme will be used to produce cross section libraries for core simulations.

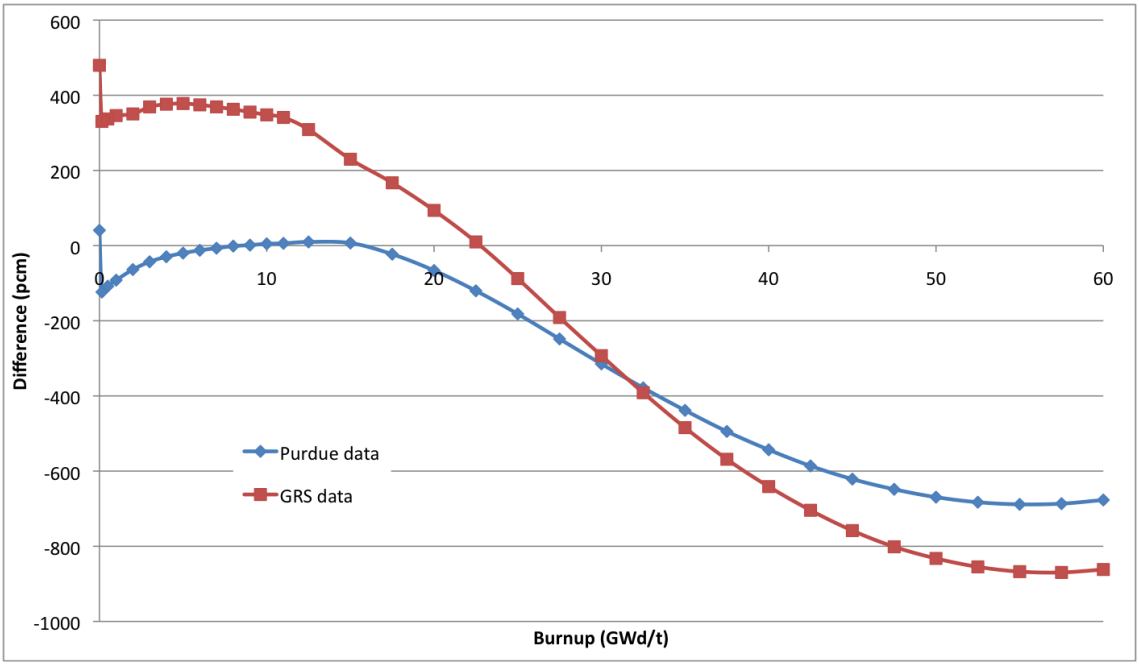


(a) With an enrichment of 4.2%

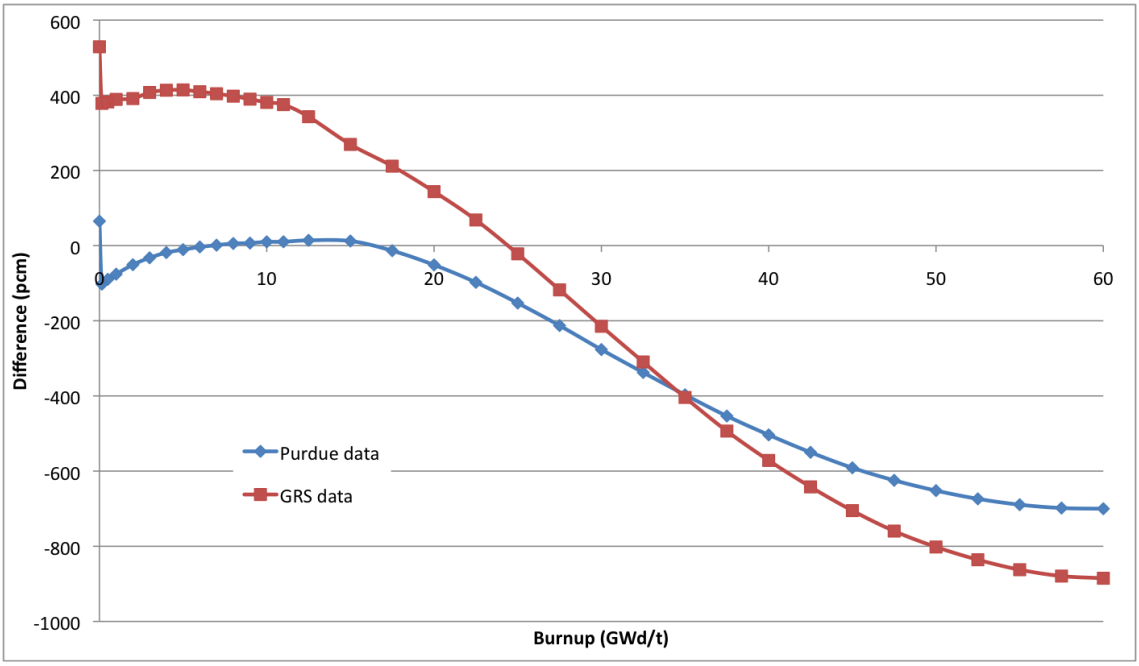


(b) With an enrichment of 4.5%

Figure 3.16 Relative difference between DRAGON and HELIOS for the UO_2 assembly



(a) With an enrichment of 4.0%



(b) With an enrichment of 4.3%

Figure 3.17 Relative difference between DRAGON and HELIOS for the MOX assembly

3.4 Multiparameter database creation

This benchmark is intended as an heterogeneous or homogenized assembly benchmark. Here, it will only be treated as an homogeneous assembly benchmark where 2-groups assembly homogenized cross sections are required. For this purpose, different burnup calculations will be performed with DRAGON on the two types of assemblies, using different sets of thermal-hydraulics parameters, and the results will be saved in a database. Then, this file will be converted in a NEMTAB-like format to provide the GRS core simulator with cross sections. Finally, calculations will be performed with this code to compare the libraries produced by DRAGON to the HELIOS ones.

3.4.1 Creation scheme

In order to create the cross section database, a calculation scheme has to be set up, where different calculations will have to be performed on the same assembly but with different parameters. Here we will first describe the branch conditions for these calculations.

3.4.1.1 Branching conditions

The cross sections are computed for three different fuel temperatures, moderator densities, and boron concentrations. The values chosen are supposed to cover the expected range of core operating conditions. The values of these parameters used as branch conditions are presented in Figure (3.18). The pressure is supposed to be constant in the reactor (15.5 MPa), so that the moderator temperature effect is treated implicitly in the moderator density. The central point at hot full power (HFP) gives the conditions for the reference burnup calculation.

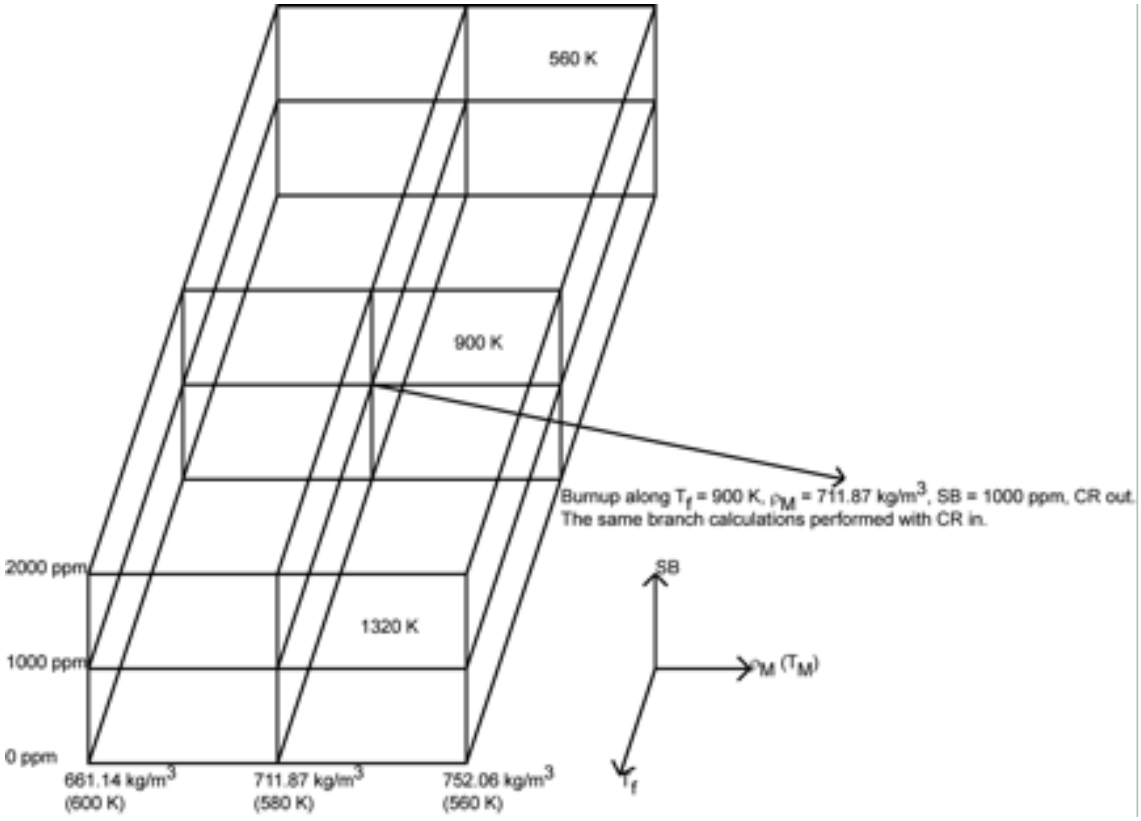


Figure 3.18 Cross section branch model

For the energy group structure, the seperation energy is 0.625 eV. The up-scattering has to be removed, but instead of using the formula given in the benchmark, this is done directly in the DRAGON calculation.

The branching calculations will be performed on the two types of assemblies, each one using two different enrichments or Pu_{fiss} content. For the UO_2 assembly, two additional calculations will have to be performed with the control rods inserted inside the guide tubes These calculations will be integrated in the unrodded assembly calculations, as explained in the next part.

3.4.1.2 Calculation scheme

For this kind of calculation, a particular scheme must be put into place, in order to compute the cross sections for all the sets of parameters, without having to perform too many calculations, and gain a lot of CPU time.

In fact, not all the burnup steps are used inside the required library. Only seven of them are important : 0.15, 17.5, 20.0, 22.5, 32.5, 35.0, and 37.5 GWd/t. Also, a branching calculation is done to study the local effects modifying some parameters in transients. So only one burnup calculation, with a fine mesh for the burnup steps, is needed in this case. The calculation will then be performed as follows :

- First, the database (a MULTICOMPO object) is initialized to take into account four parameters : the burnup, the fuel temperature, the moderator density, and the boron concentration.
- A first burnup calculation is performed, as it was done before for the assembly comparison, at nominal conditions (HFP conditions). During this calculation, a BURN object is created where the isotopic densities of each mixture at every burnup step are stored.
- Then the parameter loop starts. For each set of the three parameters, the calculation is performed as follow for a whole burnup :
    1. A new library, taking into account the parameter changes, is created.
    2. For the required burnup steps (and only those burnup steps), the densities of the depleting mixtures are updated in that library, by the use of the BURN object. Here, the moderator is not updated, because this mixture does not evolve in the calculations. A self-shielding calculation is then performed.

3. This updated library is used in the two-level flux calculation. The two-level scheme used is the same as the one used in the previous assembly calculations.
4. The flux is normalized to the power, and the cross sections are condensed in two groups, and homogenized over the whole assembly. A transport-diffusion equivalence is used during this process, which is comparable to an Assembly Discontinuity Factor (ADF) correction, by using a Selengut normalization (Selengut, 1960).
5. Finally, the resulting cross sections are stored in the MULTICOMPO object for this set of parameters.

- The parameters are then modified, and the loop is restarted. When all the sets of parameters have been used, the MULTICOMPO object is saved as an ASCII file for further use.

An important point here is the creation of the database for the UO_2 control rodded assembly. In fact, the calculations for this assembly have to be performed at the same time as the unrodded one. The control rod mixture is then present in the first library of the UO_2 calculation. This mixture is not used in the first burnup calculation, so it is not depleted. It is only used in the parameters loop at steps 3 and 4 of the previous scheme to perform the flux calculation for the rodded assembly, in parallel of the unrodded one. This calculation corresponds to an insertion of the control rods at each required burnup step.

In the end, six different MULTICOMPO objects are created. The problem is that the core simulators in the GRS cannot read the DRAGON format. A conversion process is then needed.

3.4.2 Format conversion

In order to transform the MULTICOMPO into NEMTAB-like format libraries, a conversion program had to be written. The format was used for the OECD MSLB benchmark (Ivanov et al., 1999) and is described in Appendix II. The program was written in FORTRAN (presented in Appendix III), to convert the MULTICOMPO objects containing 2-group assembly homogenized cross sections only.

The problem is that some information is missing in the MULTICOMPO object, and have to be calculated during the conversion process :

- The absorption cross sections $\Sigma_{a,i}$: they can be calculated for the group i using the total cross section Σ_i , and the scattering cross sections $\Sigma_{s,i\leftarrow i}$ and $\Sigma_{s,i\leftarrow j}$, with the equation :

$$\Sigma_{a,i} = \Sigma_i - \Sigma_{s,i\leftarrow i} - \Sigma_{s,i\leftarrow j} \tag{3.1}$$

- The delayed neutron fraction β_l for the group of precursor l : they can be calculated using $\nu\Sigma_{f,i,l}$, the product of $\Sigma_{f,i,l}$, the fission cross section with $\nu_{i,l}^D$, the averaged number of fission-emitted delayed neutron produced in the precursor group l , $\nu\Sigma_{f,i}$, the product of $\Sigma_{f,i}$, the fission cross section with $\nu_{i,1}^{ss}$, the steady-state number of neutron per fission, and the multigroup weighted neutron flux spectrum $\phi_{w,i}$, using the formula :

$$\beta_l = \frac{\nu\Sigma_{f,1,l}\phi_{w,1} + \nu\Sigma_{f,2,l}\phi_{w,2}}{\nu\Sigma_{f,1}\phi_{w,1} + \nu\Sigma_{f,2}\phi_{w,2}} \tag{3.2}$$

- Some values had to be averaged over the 27 parameters calculations : the fission spectrum χ_i , the inverse velocity $< 1/v >_i$, the delay neutron decay constant λ_i, and the delay neutron fraction β_i .

3.4.3 Comparison with GRS results

The converted libraries have been used in the GRS coupled code system QUABOX-CUBBOX/ATHLET (Q/C) (Langenbuch and Velkov, 2004) for the study of a PWR core transient. The results obtained using these libraries will be compared to the results obtained with libraries produced by the HELIOS code.

3.4.3.1 The coupled system QUABOX-CUBBOX/ATHLET

The coupled code system QUABOX-CUBBOX/ATHLET is based on the neutronic core code QUABOX-CUBBOX (Langenbuch et al., 1977) and the thermal-hydraulic system code ATHLET (Lerchl and Austregesilo, 2003).

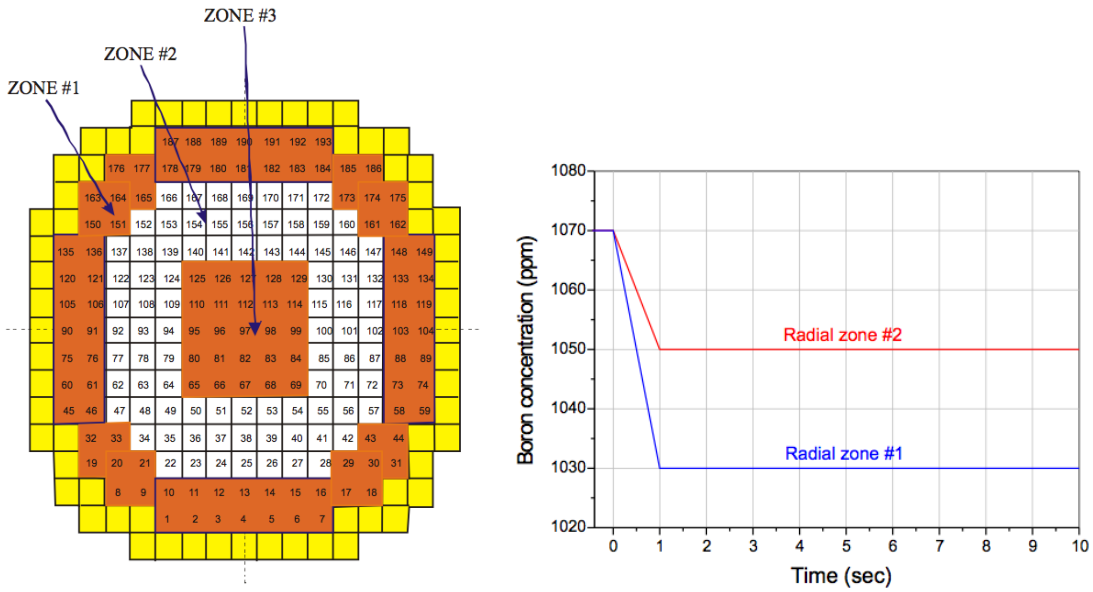
The ATHLET code is a thermal-fluid dynamic system code developed at GRS for a wide range of applications. Different models of fluid dynamics can be chosen, in order to treat problems such as anticipated and abnormal plant transients, small and intermediate leaks, or large breaks in PWR and BWRs. This code has a highly modular structure allowing an easy implementation of different physical models.

The QUABOX-CUBBOX code is a 3D neutronic core model solving the neutron diffusion equation with two prompt neutron groups and six groups of delayed neutron precursors. It is based on a coarse mesh method with a polynomial expansion of the neutron fluxes in each energy group. The dependence on thermal-hydraulic parameters of the homogenized cross sections allows the code to take into account the reactivity feedback.

The core model used in the coupled system is a full core representation. The radial calculation mesh corresponds to a single node per assembly. The active core height is discretized in axial meshes with dimensions equal to the fuel assembly pitch. For the calculations, the cross section data are applied to the Q/C code.

3.4.3.2 A radially asymmetric boron transient

The core configuration is based on the Westinghouse PWR core specified in the Purdue benchmark documentation and presented in Figure (3.1). For this transient, the core is divided in three radial zones, as presented in Figure (3.19(a)), where the yellow squares represent the reflector assemblies :



(a) Definition of the 3 radial zones

(b) Linear changes of the boron concentration

Figure 3.19 Definition of some parameters for the transient

The conditions chosen at the beginning of this transient correspond to the HFP state conditions with a boron concentration of 1070 ppm. Then, different amounts of coolant with different boron concentrations are injected at the core inlet within one second, corresponding to a linear decrease of the boron concentration within one second from 1070 ppm to 1030 ppm in ZONE #1 and from 1070 ppm to 1050 ppm in ZONE #2. Those linear changes are shown in Figure (3.19(b)). For this analysis, only the first 10 seconds are of interest.

3.4.3.3 Comparison with HELIOS

The cross section libraries produced by the DRAGON code are used in the coupled system QUABOX-CUBBOX/ATHLET to study the transient described in section 3.4.3.2. The results are compared to thoses obtained by using HELIOS produced libraries.

The HELIOS cross sections are corrected by ADFs inside the coupled code. The cross sections of the DRAGON libraries are already corrected by an equivalence procedure during the condensation and homogenization of the cross sections. Because the data for the reflector have not been produced by DRAGON, they have been taken from the HELIOS calculations and corrected by ADFs. The evolution normalized to the power at $t = 0$ is illustrated in Figure (3.20) for the transient, showing a very good agreement :

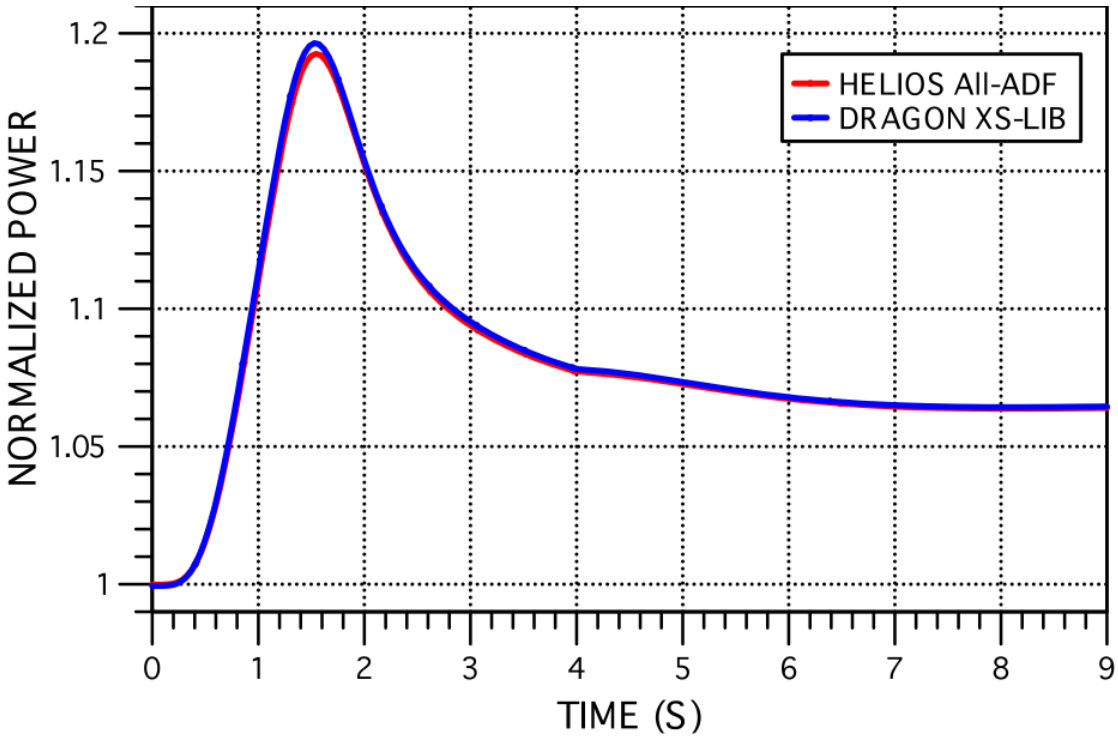


Figure 3.20 Boron transient (3 zones)

Then, comparisons of local quantities have been performed for some assemblies for the axial power, moderator temperatures, and moderator densities at different times :

1. Beginning of transient : $t = 6sec$;
2. At maximum power : $t = 7,57sec$;
3. End of transient : $t = 20sec$.

Because there are a large number of different curves, those data are shown in Appendix IV. They show that the diferencies remain very small. The assemblies giving the largest deviations are the MOX, and the UO_2 rodded assemblies. It can be seen here that these errors have an impact on the axial power. What is surprising is the difference for the control rodded assembly. In fact, a comparison of k_∞ for the DRAGON and the HELIOS libraries has been done for HFP conditions, the results being shown in Figures (3.21) and (3.22).

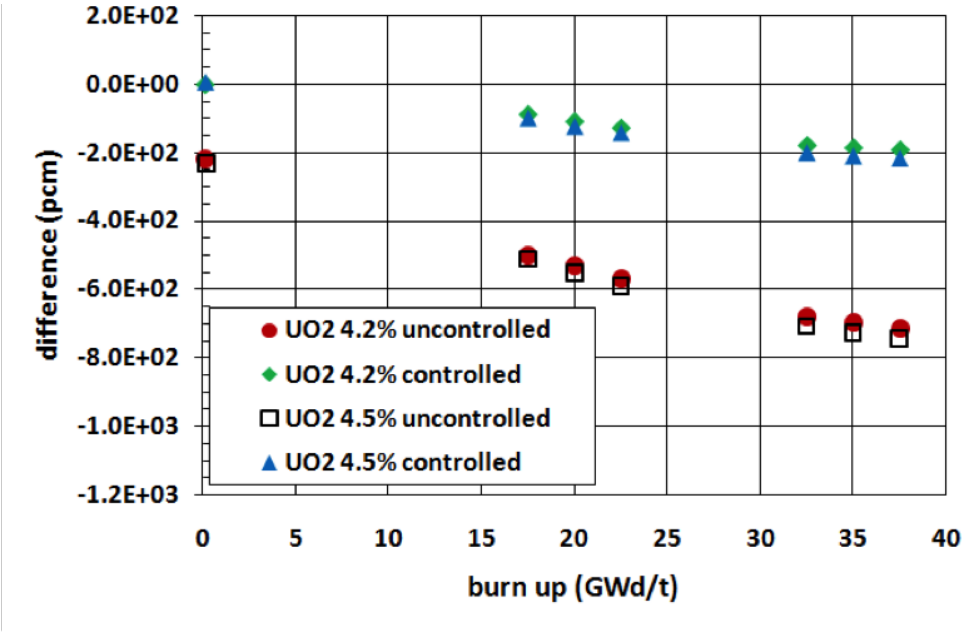


Figure 3.21 k_∞ comparison of HELIOS and DRAGON for the UO_2 assemblies

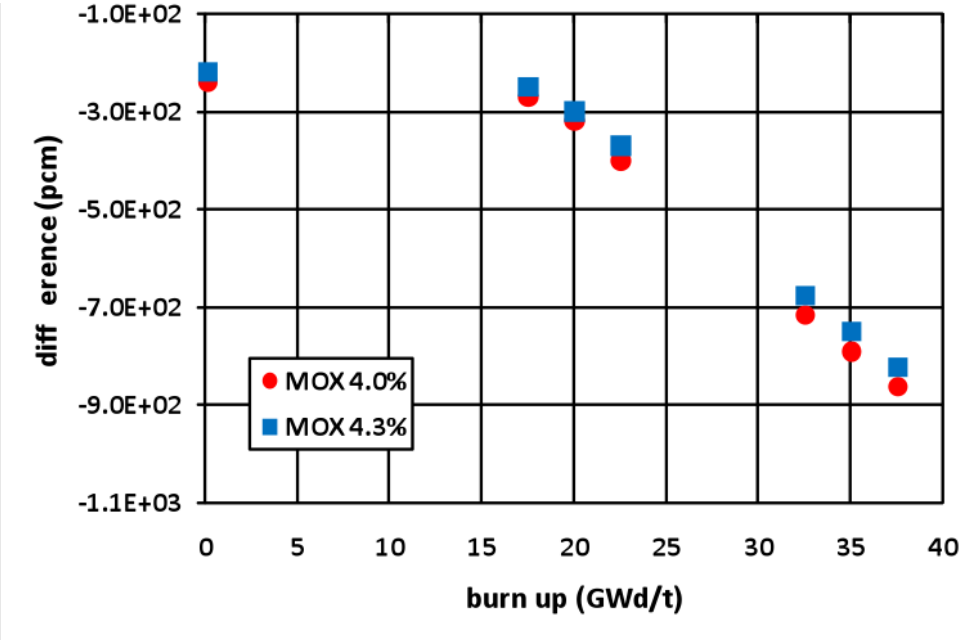


Figure 3.22 k_∞ comparison of HELIOS and DRAGON for the MOX assemblies

The differences for the MOX and the UO_2 assemblies are larger than the one found during the assembly comparisons. For the control rodded assemblies, the comparison shows that they have the lowest difference. Maybe, the cross sections are different, but some compensation effects occurs during this calculation.

The main point here is that the DRAGON code is able to create condensed and homogenized libraries that give comparable results to calculations using libraries coming from HELIOS. The two-level scheme is then able to provide acceptable cross sections for core calculation studies.

CONCLUSION

In this project, two different studies, based on different PWR reactor assemblies, have been carried out to validate a two-level calculation scheme with the use of the DRAGON code, showing a good accuracy and fast calculation time. It also shows that DRAGON is able to create condensed and homogenized cross section libraries for whole core calculations, giving good results in comparison with other lattice codes.

The first study was intended to perform comparisons on depletion calculations to obtain the best calculation scheme, and validate it. Two types of comparisons were performed. First, some verifications had to be performed with the DRAGON code. Indeed, the results are very sensitive to some calculation parameters including the discretization of the geometry, the model chosen for the calculations, or the method used in the calculations. In order to chose the optimum scheme, two criteria were combined : the accuracy and the CPU time. In fact, the more accurate the method used, the longer the calculation time. Those two criteria appeared to be very sensitive to the type of method used for the self-shielding calculation. Then, four different methods were used to solve the flux : a collision probability method, a UP1 interface current method (supposedly the fastest), a method of characteristics (supposedly more accurate), and a two-level scheme, combination of the two last methods. The two-level scheme showed good results compared to the other calculations, particularly with the MOC method, with a short calculation time. This two-level scheme, and the MOC method, were compared to calculations performed by different organizations, using different codes. The purpose was to prove that the two-level scheme, and particularly the DRAGON code, are able to give accurate results. Three different parameters were used for validation : the effective multiplication factor, the nuclide densities, and the pin power. Comparing the results on these parameters, the two schemes showed very good agreement, the two-level scheme showing the best.

The second study was based on an OECD/NEA PWR benchmark and was intended to demonstrate the production of condensed and homogenized cross section libraries for core transient studies.

In this benchmark, two different types of PWR assemblies were used : a UO_2 and a MOX assembly, each one with two different enrichments or Pu_{fiss} contents. Although the problem is completely different from the previous study, the same verifications had to be performed, in comparison with HELIOS, at the level of a pin cell, and the fuel assembly. The comparisons showed that the MOX assembly is more challenging because of the presence of heavy isotopes. In summary, the results were acceptable, and the two-level scheme was proved applicable to produce reliable cross section libraries for use with neutronics core codes.

In fact, the creation of these parameterized libraries required repeated calculations for each assembly with different thermal-hydraulic conditions. Three parameters were studied, each one with three different values. For this purpose, a calculation scheme has been developped, that performs the 27 calculation points at each one of the required burnup steps, and produces a parameterized cross section library. After conversion into a format that can be read by the coupled code system QUABOX-CUBBOX/ATHLET, the libraries were used to calculate a boron transient with this coupled code system. The results were finally compared to a calculation using HELIOS produced libraries, demonstrating the quality of the DRAGON generated cross sections.

This project gave also the opportunity of showing the advantages of using the DRAGON code. It is in fact possible with this code to use a wide range of methods to solve the transport equation, but also to use different types of libraries, coming from different types of evaluations. Finally, it usually takes many years to develop a very good calculation scheme, and it is interesting to see that with DRAGON, such a scheme giving good results was developed in a very short time.

REFERENCES

ASKEW, J., 1972, *A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries*, Report AEEW-M 1108, United Kingdom Atomic Energy Establishment, Winfrith.

ASKEW, J., FAYERS, F. et KEMSHELL, P., 1966, "A General Description of the Lattice Code WIMS", *J. Brit. Nucl. EnergySoc.*, 5, pp. 564.

BATEMAN, H., 1910, "Solution of a System of Differential Equations Occuring in the Theory of Radioactiv Transformations", *Proc. Cambridge Philos. Soc.*, 15, 423427.

BELL, M. J.., 1973, *ORIGEN - The ORNL-Isotope Generation and Depletion Code*, Report ORNL/TM-4628, Oak Ridge National Laboratory, Oak Ridge.

CASAL, J. J., STAMM'LER, R. J. J., VILLARINO, E. A., FERRI, A. A., 1991, "HELIOS: Geometric Capabilities of a New Fuel-Assembly Program", *Intl. Topical Meeting on Advances in Mathematics, Computations, and Reactor Physics*, Pittsburgh, USA, April 1991.

COURAU, T., COMETTO, M., GIRARDI, E., COUYRAS, D., SCHWARTZ, N., 2008, "Elements of Validation of Pin-by-Pin Calculations with the Future EDF Calculation Scheme Based on APOLLO2 and COCAGNE Codes", *Proceedings of ICAPP08 Anaheim*, CA USA, June 812, 2008.

CROFF, A. G., 2003, *A Users Manual for the ORIGEN2 Computer Code*, Report ORNL/TM-7175, Oak Ridge National Laboratory, Oak Ridge.

DONNELLY, J., 1986, *WIMS-CRNL, A Users Manual for the Chalk River Version of WIMS*, Atomic Energy of Canada Limited, AECL-8955.

GOLDSTEIN, R., COHEN, E., 1962, “Theory of Resonance Absorption of Neutrons“, *Nucl. Sci. Eng.*, 13, 132.

HÉBERT, A., 1993, “A Consistent Technique for the Pin-by-Pin Homogenization of a Pressurized Water Reactor Assembly”, *Nucl. Sci. Eng.*, 113, 227.

HÉBERT, A., 2004, “Revisiting the Stammler Self-shielding Model”, *Proc. of the 25th Annual Conf. of the Canadian Nuclear Society*, Toronto, Canada.

HÉBERT, A., 2005, “The Ribon Extended Self-Shielding Model”, *Nucl. Sci. Eng.*, 151, 124.

HÉBERT, A., 2006, “Towards Version4”, *PHYSOR-2006 workshop*, Vancouver, Canada.

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

HÉBERT, A., KARTHIKEYAN, R., 2005, “Interfacing NJOY with Advanced Lattice Codes”, *Workshop on NJOY-2005 and User Group Meeting*, May 2, Issy les Moulineaux, France, 2005.

HÉBERT, A., MARLEAU, G., 1991, “Generalization of the Stammler Method for the Self-Shielding of Resonant Isotopes in Arbitrary Geometries”, *Nucl. Sci. Eng.*, 108, 230.

HESSE, U., GMAL, B., VOGGENBERGER, T., BALEANU, M., HUMMELSHEIM, K., HEINICKE, W., ZWERMANN, W., LANGENBUCH, S., 2000, “KENOREST, A New Three Dimensional Fuel Assembly Code System Combining KENO-Va and OREST for Reactivity and Inventory Calculations”, *ANS International Topical Meeting PHYSOR 2000*, Pittsburgh, Pennsylvania.

HOFFMAN, A., JEANPIERRE, F., KAVENOKY, A., LIVOLANT, M., LORAIN, H., 1973, *APOLLO : Code multigroupe de résolution de l'équation du transport pour les neutrons thermiques et rapides*, Note CEA-N-1610, Commissariat à l'Énergie Atomique, France.

IAEA - NUCLEAR DATA SERVICES, 2005, *WIMS Library Update*, http://www-nds.iaea.org/wimsd/.

IVANOV, K. N., BEAM, T. M., BARATTA, A. J., 1999, *PWR Main Steam Line Break (MSLB) Benchmark, Volume I: Final Specifications*, NEA/NSC/DOC(99)8, http://www.nea.fr/html/science/docs/1999/.

KOZLOWSKI, T., DOWNAR, T. J., 2003, *OECD/NEA and U.S. NRC PWR MOX/UO_2 Core Transient Benchmark*, Purdue University, West Lafayette, Indiana, U.S.A..

LANGENBUCH, S., MAURER, W., WERNER, W., 1977, "Coarse Mesh Flux Expansion Method for the Analysis of Space-time Effects in Large LWR Cores", *Nucl. Sci. Eng.*, 63, 437.

LANGENBUCH, S., VELKOV, K., 2004, "The Capability of the Coupled Code System ATHLET-QUABOX/CUBBOX - Experience and Development", *ANS Annual Meeting*, Pittsburgh, Pennsylvania.

LE MER, J., 2007, *Simulations du comportement neutronique des REP en utilisant les codes DRAGON et DONJON*, Mémoire MScA, École Polytechnique de Montréal.

LE TELLIER, R., 2006, *Développement de la méthode des caractéristiques pour le calcul de réseau*, Thèse PhD, École Polytechnique de Montréal.

LERCHL, G., AUSTREGESILO, H., 2003, *ATHLET Mod2 Cycle A, User's Manual*, GRS 2003.

LIVOLANT, M., JEANPIERRE, F., 1974, *Autoprotection des résonances dans les réacteurs nucléaires. Application aux isotopes lourds*, Note CEA-R-4533, Commissariat à l'Énergie Atomique.

MACFARLANE, R., 1984, *TRANSX-CTR: A Code for Interfacing MATXS Cross-Section Libraries to Nuclear Transport Codes for Fusion Systems Analysis*, Report LA-9863-MS, Los Alamos National Laboratory, Los Alamos.

MACFARLANE, R., MUIR, D., 2000, *NJOY99.0 Code System For Producing Pointwise and Multigroup Neutron and Photon Cross Sections from ENDF/B Data*, Report PSR-480/NJOY99.0, Los Alamos National Laboratory, Los Alamos.

MARLEAU, G., HÉBERT, A., ROY, R., 1991, *A User Guide For DRAGON Version4*, Report IGE-294, Institut de génie nucléaire, École Polytechnique de Montréal.

PETRIE, L. M., LANDERS, N. F., 1999, *KENO V.a., An Improved Monte Carlo Criticality Program with Supergrouping*, Report NUREG/CR-0200, Volume 2, Section F1, Oak Ridge National Laboratory, Oak Ridge.

PETROVIC, I., BENOIST, P., 1996, "B_N Theory: Advances and New Models for Neutron Leakage Calculation", *Advances in Nuclear Science and Technology*, Vol. 24.

PORSCH, D., HESSE, U., ZWERMANN, W., BERNNAT, W., 2006, *Specification of a PWR Fuel Assembly, UO2 (4 w/o U-235) 18 x 18 24, for Comparative Depletion Calculations*, June 2006.

POSTEN, D. I., TRELLUE, H. R., 1999, *Users Manual Version 2.0 for MONTEBURNS, Version 1.0*, Report LA-UR-99-4999, Studsvik, Sweden.

RHODES, J., EDENIUS, M., 2001, *CASMO-4, A Fuel Assembly Burn-up Code*, Report SSP-01/400, Studsvik, Sweden.

ROY, R., HÉBERT, A., 2000, *The GAN Generalized Driver*, Report IGE-158, Institut de génie nucléaire, École Polytechnique de Montréal.

SANCHEZ, R., McCORMICK, N., 1982, "A Review of Neutron Transport Approximations", *Nucl. Sci. Eng.*, 80:481-535.

SANTAMARINA, A., COLLIGNON, C., GARAT, C., 2004, "French Calculation Schemes for Light Water Reactor Analysis", *Proc of Int. Mtg. on the Physics of Fuel Cycles and Advanced Nuclear Systems PHYSOR 2004*, Chicago, ANS.

SELENGUT, D. S., 1960, "Diffusion Coefficients for Heterogeneous Systems", *Trans. Am. Nucl. Soc.*, 3, 398.

SUICH, J. E., HONECK, H. C., 1967, *The HAMMER-System - Heterogeneous Analysis by Multigroup Methods of Exponentials and Reactors*, Report TID-4500.

STAMMLER, R., ABBATE, M., 1983, "Methods of Steady-State Reactor Physics in Nuclear Design", *Academic Press*, London.

VELKOV, K., SEUBERT, A., PASICHNYK, I., PAUTZ, A., 2009 "Boron Transient Analysis by the Coupled Code Systems ATHLET-QUABOX/CUBBOX and ATHLET-TORD-TD", *ICONE-17*, Brussels, Belgium.

WU, G. J., ROY, R., 2003 "Acceleration Techniques for Trajectory-based Deterministic 3D Transport Solvers", *Annals of Nuclear Energy*, 30:116.

X-5 Monte Carlo Team, 2003, *MCNP A General Monte Carlo N-Particle Transport Code, Version 5*, Report LA-UR-03-1987, Los Alamos National Laboratory, Los Alamos.

APPENDIX I

SAMPLE OF 2-GROUP CROSS SECTIONS NEMTAB-LIKE FORMAT

```
*      Mod Dens      Boron ppm      Fuel Temp       Mod Temp
             3              3              3              0
         661.14         711.87         752.06
           0.00        1000.00        2000.00
         560.00         900.00        1320.00
*
* ------------------------------------------------------------
* BURNUP   0.15
* ------------------------------------------------------------
*
* Transport XSEC Table
*
* GROUP       1
   XS(D1,B1,F1)   XS(D2,B1,F1)   XS(D3,B1,F1)
   XS(D1,B2,F1)   XS(D2,B2,F1)   XS(D3,B2,F1)
   XS(D1,B3,F1)   XS(D2,B3,F1)   XS(D3,B3,F1)
   XS(D1,B1,F2)   XS(D2,B1,F2)   XS(D3,B1,F2)
   XS(D1,B2,F2)   XS(D2,B2,F2)   XS(D3,B2,F2)
   XS(D1,B3,F2)   XS(D2,B3,F2)   XS(D3,B3,F2)
   XS(D1,B1,F3)   XS(D2,B1,F3)   XS(D3,B1,F3)
   XS(D1,B2,F3)   XS(D2,B2,F3)   XS(D3,B2,F3)
   XS(D1,B3,F3)   XS(D2,B3,F3)   XS(D3,B3,F3)
* GROUP       2
    ...
*
* Absorption XSEC Table
    ...
*
* Nu-Fission XSEC Table
    ...
*
* Kappa-Fission XSEC Table
    ...
*
* Scattering XSEC Table
*
* GROUP       1 ->           2
    ...
* GROUP       2 ->           1
    ...
*
* ADF Table
    ...
*
* Fission Spectrum
*
* GROUP       1              2
       CHI(G1)        CHI(G2)
*
* Inverse Velocity
*
* GROUP       1              2
      IVEL(G1)       IVEL(G2)
*
* Delay Neutron Decay Constant (Lambda)
*
* GROUP       1              2              3              4              5              6
    LAMBDA(G1)     LAMBDA(G2)     LAMBDA(G3)     LAMBDA(G4)     LAMBDA(G5)     LAMBDA(G6)
*
* Delay Neutron Fraction (Beta)
*
* GROUP       1              2              3              4              5              6
      BETA(G1)       BETA(G2)       BETA(G3)       BETA(G4)       BETA(G5)       BETA(G6)
*
* ------------------------------------------------------------
* BURNUP  17.50
* ------------------------------------------------------------
...
```

APPENDIX II

MULTICOMPO TO NEMTAB-LIKE FORMAT CONVERSION PROGRAM

```
      program conversion

c ---------------------------------------------------------------------
c --- Declare the Variables :
c ---------------------------------------------------------------------

      real, dimension(3) :: Mod_Dens
      real, dimension(3) :: Boron_ppm
      real, dimension(3) :: Fuel_Temp
      real, dimension(26) :: Burnup
      real, dimension(7) :: Burnup_t
      real, dimension(7) :: Burnup_it
      real, dimension(7) :: Burnup_i
      real, dimension(7,27,2) :: NWT0
      real, dimension(7,27,2) :: NTOT0
      real, dimension(7,27,2) :: NUSIGF
      real, dimension(7,27,2) :: NFTOT
      real, dimension(7,27,2) :: H_FACTOR
      real, dimension(7,27,2) :: OVERV
      real, dimension(7,2) :: INVEL
      real, dimension(7,27,2) :: NUSIGF01
      real, dimension(7,27,2) :: NUSIGF02
      real, dimension(7,27,2) :: NUSIGF03
      real, dimension(7,27,2) :: NUSIGF04
      real, dimension(7,27,2) :: NUSIGF05
      real, dimension(7,27,2) :: NUSIGF06
      real, dimension(7,27,2) :: CHI
      real, dimension(7,2) :: AVCHI
      real, dimension(7,27,2) :: STRD
      real, dimension(7,27,4) :: SCAT00
      real, dimension(7,27,6) :: LAMBDA_D
```

```
      real, dimension(7,6) :: LAMBDA
      real, dimension(7,27,6) :: BETA_D
      real, dimension(7,6) :: BETA
      real, dimension(7,27,2) :: SIGABS
      character*80 :: line
      integer :: i, j, k, ind, bu_step, param, group

      ind = 0
      bu_step = 0
      param = 1

c ---------------------------------------------------------------------
c --- Open the Dragon Database :
c ---------------------------------------------------------------------

      open(10,file='DbM43',form='formatted')

      open(20,file='DbM43.txt',form='formatted')

c ---------------------------------------------------------------------
c --- Recover all the Datas :
c ---------------------------------------------------------------------

c --- Recover the Parameters Datas
c -------------------------------

100   continue

      read(10,'(A80)',end=200) line

c --- Read Moderator Density values : Mod_Dens

      if(line(1:12).eq.'pval00000001') then
        read(10,*) Mod_Dens(1),
     &             Mod_Dens(2),
     &             Mod_Dens(3)

c --- Read Boron Concentration values : Boron_ppm

      elseif(line(1:12).eq.'pval00000002') then
        read(10,*) Boron_ppm(1),
     &             Boron_ppm(2),
     &             Boron_ppm(3)

c --- Read Fuel Temperature values : Fuel_Temp
```

```
       elseif(line(1:12).eq.'pval00000003') then
         read(10,*) Fuel_Temp(1),
     &              Fuel_Temp(2),
     &              Fuel_Temp(3)

c --- Read Burnup values : Burnup

       elseif(line(1:12).eq.'pval00000004') then
         read(10,*) Burnup(1),
     &              Burnup(2),
     &              Burnup(3),
     &              Burnup(4),
     &              Burnup(5)
         read(10,*) Burnup(6),
     &              Burnup(7),
     &              Burnup(8),
     &              Burnup(9),
     &              Burnup(10)
         read(10,*) Burnup(11),
     &              Burnup(12),
     &              Burnup(13),
     &              Burnup(14),
     &              Burnup(15)
         read(10,*) Burnup(16),
     &              Burnup(17),
     &              Burnup(18),
     &              Burnup(19),
     &              Burnup(20)
         read(10,*) Burnup(21),
     &              Burnup(22),
     &              Burnup(23),
     &              Burnup(24),
     &              Burnup(25)
         read(10,*) Burnup(26)
       goto 101

       endif

       goto 100

c --- Count the number of burnup steps
c -----------------------------------

101    continue
       read(10,'(A80)',end=200) line
```

```
      if(line(1:34).eq.
     &   '->       5       0       0      -1') then
        ind = ind + 1
      goto 102

      else
        goto 101

      endif

c --- Only keep the good burnup steps datas
c ---------------------------------------

102   continue
      read(10,'(A80)',end=200) line

      if (ind.eq.3.or.ind.eq.18.or.ind.eq.19.or.ind.eq.20
     &    .or.ind.eq.24.or.ind.eq.25.or.ind.eq.26) then
        bu_step = bu_step + 1
        if (param.eq.1) then
          Burnup_t(bu_step) = Burnup(ind) /10
          Burnup_it(bu_step) = nint(Burnup_t(bu_step))
          Burnup_i(bu_step) = Burnup_it(bu_step) / 100
        endif
        goto 103
      else
        goto 101
      endif

c --- Look for all the information
c ------------------------------

103   continue
      read(10,'(A80)',end=200) line

c --- Read Average Flux (Beta Calculation) : NWT0

      if (line(1:4).eq.'NWT0') then
        read(10,*) NWT0(bu_step,param,1),
     &             NWT0(bu_step,param,2)
      goto 103

c --- Read Total Cross-Sections : NTOT0

      elseif (line(1:5).eq.'NTOT0') then
        read(10,*) NTOT0(bu_step,param,1),
```

```
     &             NTOT0(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections : NUSIGF

      elseif (line(1:8).eq.'NUSIGF  ') then
        read(10,*) NUSIGF(bu_step,param,1),
     &             NUSIGF(bu_step,param,2)
      goto 103


c --- Read Fission Cross-Sections : NFTOT

      elseif (line(1:5).eq.'NFTOT') then
        read(10,*) NFTOT(bu_step,param,1),
     &             NFTOT(bu_step,param,2)
      goto 103

c --- Read H_FACTOR

      elseif (line(1:8).eq.'H-FACTOR') then
        read(10,*) H_FACTOR(bu_step,param,1),
     &             H_FACTOR(bu_step,param,2)
      goto 103

c --- Read Inverse of the Neutron Velocity : OVERV

      elseif (line(1:5).eq.'OVERV') then
        read(10,*) OVERV(bu_step,param,1),
     &             OVERV(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections for precusor group 1
c     (Beta Calculation) : NUSIGF01

      elseif (line(1:8).eq.'NUSIGF01') then
        read(10,*) NUSIGF01(bu_step,param,1),
     &             NUSIGF01(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections for precusor group 2
c     (Beta Calculation) : NUSIGF02
```

```
      elseif (line(1:8).eq.'NUSIGF02') then
        read(10,*) NUSIGF02(bu_step,param,1),
     &             NUSIGF02(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections for precusor group 3
c     (Beta Calculation) : NUSIGF03

      elseif (line(1:8).eq.'NUSIGF03') then
        read(10,*) NUSIGF03(bu_step,param,1),
     &             NUSIGF03(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections for precusor group 4
c     (Beta Calculation) : NUSIGF04

      elseif (line(1:8).eq.'NUSIGF04') then
        read(10,*) NUSIGF04(bu_step,param,1),
     &             NUSIGF04(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections for precusor group 5
c     (Beta Calculation) : NUSIGF05

      elseif (line(1:8).eq.'NUSIGF05') then
        read(10,*) NUSIGF05(bu_step,param,1),
     &             NUSIGF05(bu_step,param,2)
      goto 103


c --- Read Nu*Sigmaf Cross-Sections for precusor group 6
c     (Beta Calculation) : NUSIGF06

      elseif (line(1:8).eq.'NUSIGF06') then
        read(10,*) NUSIGF06(bu_step,param,1),
     &             NUSIGF06(bu_step,param,2)
      goto 103


c --- Read Fission Spectrum : CHI

      elseif (line(1:3).eq.'CHI') then
        read(10,*) CHI(bu_step,param,1),
```

```
     &             CHI(bu_step,param,2)
      goto 103


c --- Read Transport Cross-Sections : STRD

      elseif (line(1:4).eq.'STRD') then
        read(10,*) STRD(bu_step,param,1),
     &             STRD(bu_step,param,2)
      goto 103

c --- Read Scattering Cross-Sections : SCAT00

      elseif (line(1:6).eq.'SCAT00') then
        read(10,*) SCAT00(bu_step,param,1),
     &             SCAT00(bu_step,param,2),
     &             SCAT00(bu_step,param,3)
        SCAT00(bu_step,param,4) = 0.00000
      goto 103


c --- Read Delayed Neutron radioactive decay constant : LAMBDA_D

      elseif (line(1:8).eq.'LAMBDA-D') then
        read(10,*) LAMBDA_D(bu_step,param,1),
     &             LAMBDA_D(bu_step,param,2),
     &             LAMBDA_D(bu_step,param,3),
     &             LAMBDA_D(bu_step,param,4),
     &             LAMBDA_D(bu_step,param,5),
     &             LAMBDA_D(bu_step,param,6)

        if (ind.eq.26) then
          ind = 0
          bu_step = 0
          param = param + 1
        endif

        goto 101

      else
        goto 103
      endif

c ---------------------------------------------------------------------
c --- Close the Database
c ---------------------------------------------------------------------
```

```
 200  continue
      close(10)

c ---------------------------------------------------------------------
c --- Calculate the missing values
c ---------------------------------------------------------------------

c --- Calculate the Absorption Cross-Sections :

      DO i=1, 7
        DO j=1, 27
          SIGABS(i,j,1) = NTOT0(i,j,1) - SCAT00(i,j,1) - SCAT00 (i,j,3)
          SIGABS(i,j,2) = NTOT0(i,j,2) - SCAT00(i,j,2)
        ENDDO
      ENDDO

c --- Average the Fission Spectrum :

c      DO i=1, 7
c        DO k=1, 2
          AVCHI = sum(CHI,dim=2)/27
c        ENDDO
c      ENDDO

c --- Average the Inverse Velocities :

c      DO i=1, 7
c        DO k=1, 2
          INVEL = sum(OVERV,dim=2)/27
c        ENDDO
c      ENDDO

c --- Average the Lambda :

c      DO i=1, 7
c        DO k=1, 6
          LAMBDA = sum(LAMBDA_D,dim=2)/27
c        ENDDO
c      ENDDO

c --- Calculate the Beta :

      DO i=1, 7
        DO j=1, 27
          BETA_D(i,j,1) = (( NUSIGF01(i,j,1) * NWT0(i,j,1) ) +
```

```
     &                     ( NUSIGF01(i,j,2) * NWT0(i,j,2) )) /
     &                    (( NUSIGF(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF(i,j,2) * NWT0(i,j,2) ))
          BETA_D(i,j,2) = (( NUSIGF02(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF02(i,j,2) * NWT0(i,j,2) )) /
     &                    (( NUSIGF(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF(i,j,2) * NWT0(i,j,2) ))
          BETA_D(i,j,3) = (( NUSIGF03(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF03(i,j,2) * NWT0(i,j,2) )) /
     &                    (( NUSIGF(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF(i,j,2) * NWT0(i,j,2) ))
          BETA_D(i,j,4) = (( NUSIGF04(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF04(i,j,2) * NWT0(i,j,2) )) /
     &                    (( NUSIGF(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF(i,j,2) * NWT0(i,j,2) ))
          BETA_D(i,j,5) = (( NUSIGF05(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF05(i,j,2) * NWT0(i,j,2) )) /
     &                    (( NUSIGF(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF(i,j,2) * NWT0(i,j,2) ))
          BETA_D(i,j,6) = (( NUSIGF06(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF06(i,j,2) * NWT0(i,j,2) )) /
     &                    (( NUSIGF(i,j,1) * NWT0(i,j,1) ) +
     &                     ( NUSIGF(i,j,2) * NWT0(i,j,2) ))
        ENDDO
       ENDDO

c --- Average the Beta :

c      DO i=1, 7
c        DO k=1, 6
          BETA = sum(BETA_D,dim=2)/27
c        ENDDO
c      ENDDO

c ---------------------------------------------------------------------
c --- Write XS in the nemtap format
c ---------------------------------------------------------------------

c --- Write the parameters datas

       write(20,'(a)') '*      Mod Dens      Boron ppm
     &      Fuel Temp       Mod Temp'
       write(20,'(a)') '              3              3
     &              3             0'
       write(20,'(3F15.2)') (Mod_Dens(i), i=1,3)
       write(20,'(3F15.2)') (Boron_ppm(i), i=1,3)
```

```
      write(20,'(3F15.2)') (Fuel_Temp(i), i=1,3)

c --- Write each Burnup Step Information

      DO i=1, 7
        write(20,'(a)') '*'
        write(20,'(a)') '* ---------------------------------------------
     &-------------'
        write(20,'(A11,F7.2)') '* BURNUP   ', Burnup_i(i)
        write(20,'(a)') '* ---------------------------------------------
     &-------------'
        write(20,'(a)') '*'
        write(20,'(a)') '* Transport XSEC Table'
        write(20,'(a)') '*'
        write(20,'(a)') '* GROUP       1'
        write(20,'(3ES15.5)') (STRD(i,j,1), j=1,27)
        write(20,'(a)') '* GROUP       2'
        write(20,'(3ES15.5)') (STRD(i,j,2), j=1,27)
        write(20,'(a)') '*'
        write(20,'(a)') '* Absorption XSEC Table'
        write(20,'(a)') '*'
        write(20,'(a)') '* GROUP       1'
        write(20,'(3ES15.5)') (SIGABS(i,j,1), j=1,27)
        write(20,'(a)') '* GROUP       2'
        write(20,'(3ES15.5)') (SIGABS(i,j,2), j=1,27)
        write(20,'(a)') '*'
        write(20,'(a)') '* Nu-Fission XSEC Table'
        write(20,'(a)') '*'
        write(20,'(a)') '* GROUP       1'
        write(20,'(3ES15.5)') (NUSIGF(i,j,1), j=1,27)
        write(20,'(a)') '* GROUP       2'
        write(20,'(3ES15.5)') (NUSIGF(i,j,2), j=1,27)
        write(20,'(a)') '*'
        write(20,'(a)') '* Kappa-Fission XSEC Table'
        write(20,'(a)') '*'
        write(20,'(a)') '* GROUP       1'
        write(20,'(3ES15.5)') (H_FACTOR(i,j,1), j=1,27)
        write(20,'(a)') '* GROUP       2'
        write(20,'(3ES15.5)') (H_FACTOR(i,j,2), j=1,27)
        write(20,'(a)') '*'
        write(20,'(a)') '* Scattering XSEC Table'
        write(20,'(a)') '*'
        write(20,'(a)') '* GROUP       1 ->           1'
        write(20,'(3ES15.5)') (SCAT00(i,j,1), j=1,27)
        write(20,'(a)') '* GROUP       1 ->           2'
        write(20,'(3ES15.5)') (SCAT00(i,j,3), j=1,27)
```

```
         write(20,'(a)') '* GROUP       2 ->           1'
         write(20,'(3ES15.5)') (SCAT00(i,j,4), j=1,27)
         write(20,'(a)') '* GROUP       2 ->           2'
         write(20,'(3ES15.5)') (SCAT00(i,j,2), j=1,27)
         write(20,'(a)') '*'
         write(20,'(a)') '* ADF Table'
         write(20,'(a)') '*'
         write(20,'(a)') '* GROUP       1'
         write(20,'(3F15.5)') (1.0, j=1,27)
         write(20,'(a)') '* GROUP       2'
         write(20,'(3F15.5)') (1.0, j=1,27)
         write(20,'(a)') '*'
         write(20,'(a)') '* Fission Spectrum'
         write(20,'(a)') '*'
         write(20,'(a)') '* GROUP       1              2'
         write(20,'(2ES15.5)') (AVCHI(i,k), k=1,2)
         write(20,'(a)') '*'
         write(20,'(a)') '* Inverse Velocity'
         write(20,'(a)') '*'
         write(20,'(a)') '* GROUP       1              2'
         write(20,'(2ES15.5)') (INVEL(i,k), k=1,2)
         write(20,'(a)') '*'
         write(20,'(a)') '* Delay Neutron Decay Constant (Lambda)'
         write(20,'(a)') '*'
         write(20,'(a)') '* GROUP       1              2              3
     &              4              5              6'
         write(20,'(6ES15.5)') (LAMBDA(i,k), k=1,6)
         write(20,'(a)') '*'
         write(20,'(a)') '* Delay Neutron Fraction (Beta)'
         write(20,'(a)') '*'
         write(20,'(a)') '* GROUP       1              2              3
     &              4              5              6'
         write(20,'(6ES15.5)') (BETA(i,k), k=1,6)
       ENDDO
       write(20,'(a)') 'END'

c ---------------------------------------------------------------------
c --- End of the Program
c ---------------------------------------------------------------------
      end program
```

APPENDIX III

ASSEMBLIES AXIAL DATA

III.1 Assemblies axial power

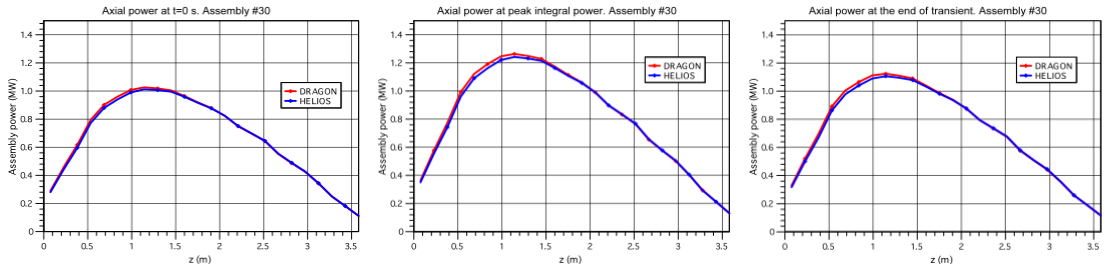


Figure III.1 MOX 4.3%, burnup 0.15 GWd/t

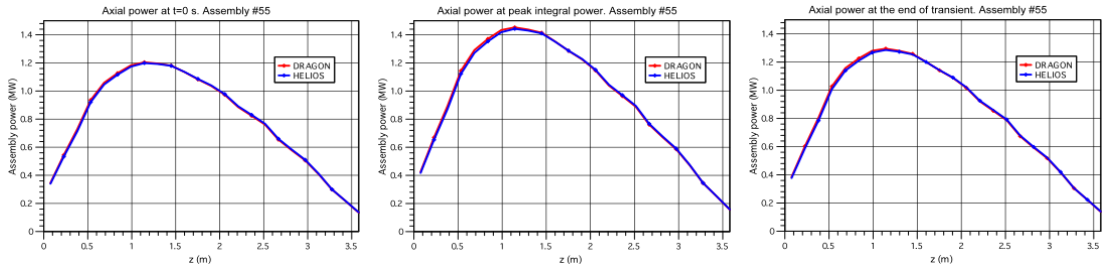


Figure III.2 MOX 4.0%, burnup 37.5 GWd/t

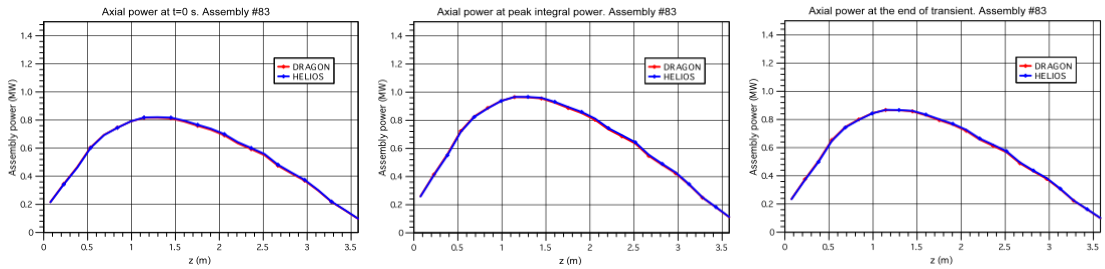


Figure III.3 UO_2 4.2% uncontrolled, burnup 17.5 GWd/t

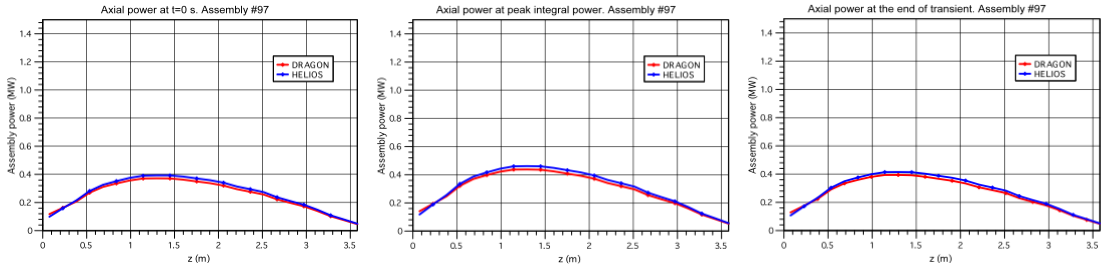


Figure III.4 UO_2 4.2% controlled, burnup 35 GWd/t

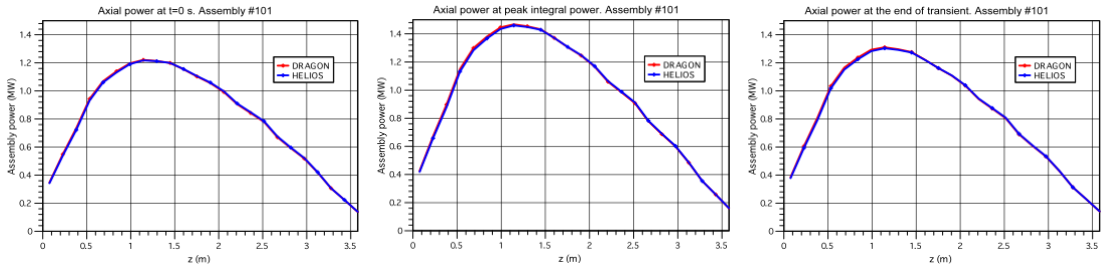


Figure III.5 UO_2 4.5% uncontrolled, burnup 37.5 GWd/t

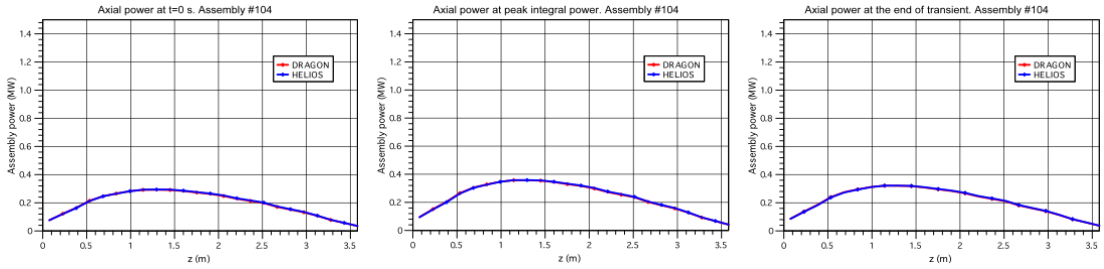


Figure III.6 UO_2 4.2% uncontrolled, burnup 37.5 GWd/t

III.2 Assemblies moderator temperature

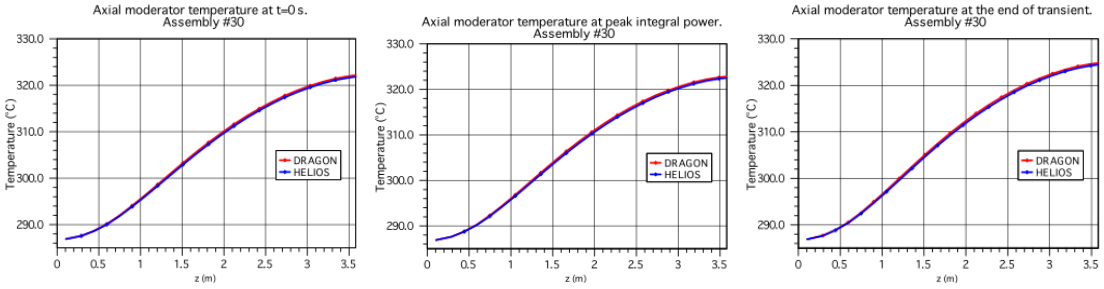


Figure III.7 MOX 4.3%, burnup 0.15 GWd/t

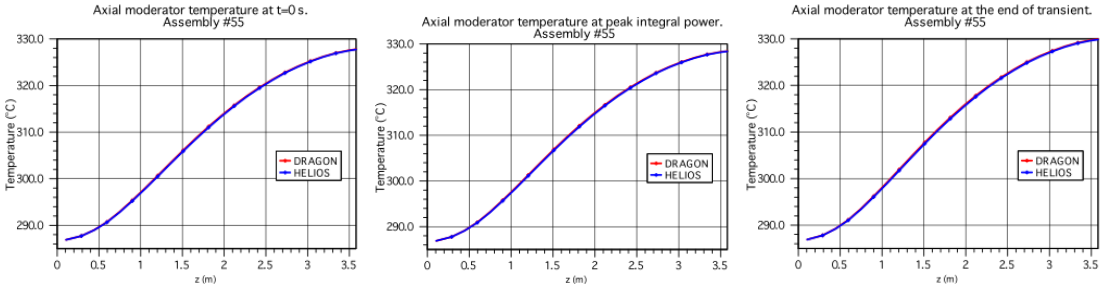


Figure III.8 MOX 4.0%, burnup 37.5 GWd/t

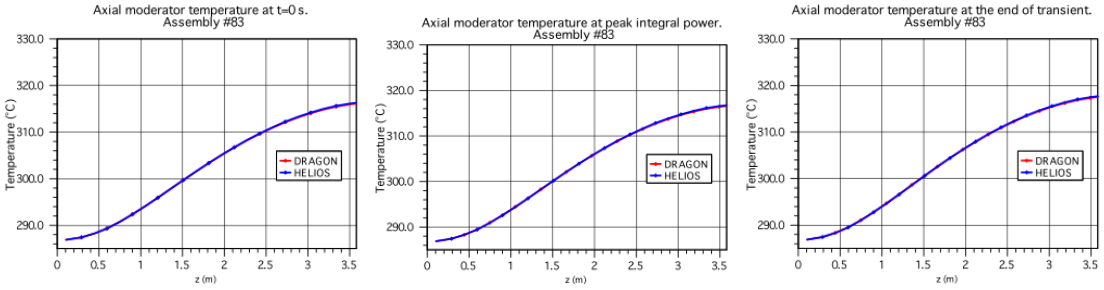


Figure III.9 UO_2 4.2% uncontrolled, burnup 17.5 GWd/t

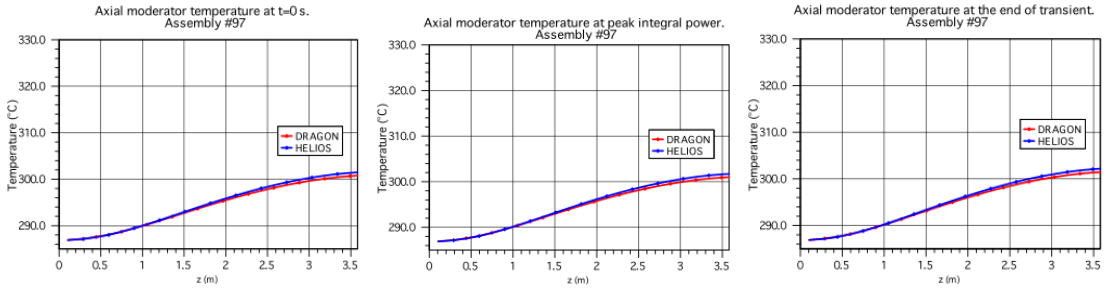


Figure III.10 UO_2 4.2% controlled, burnup 35 GWd/t

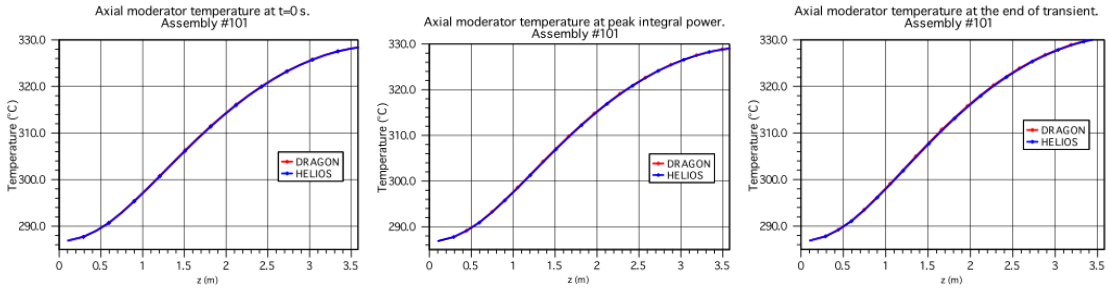


Figure III.11 UO_2 4.5% uncontrolled, burnup 37.5 GWd/t

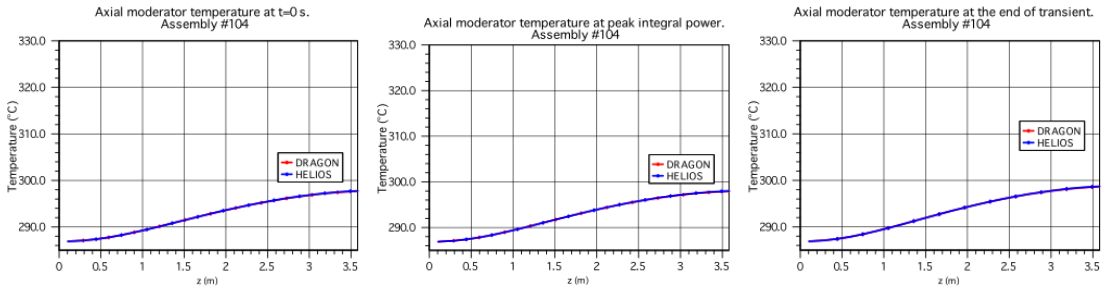


Figure III.12 UO_2 4.2% uncontrolled, burnup 37.5 GWd/t

III.3 Assemblies moderator densities

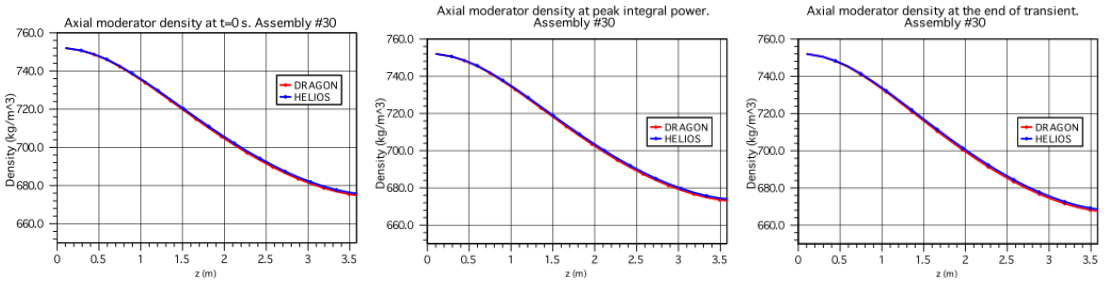


Figure III.13 MOX 4.3%, burnup 0.15 GWd/t

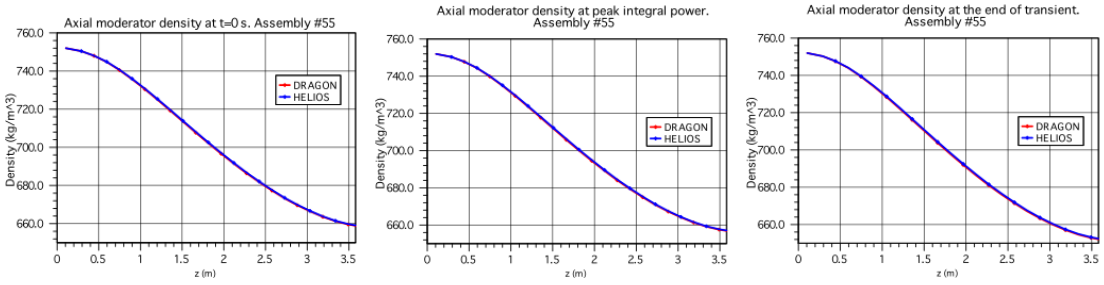


Figure III.14 MOX 4.0%, burnup 37.5 GWd/t

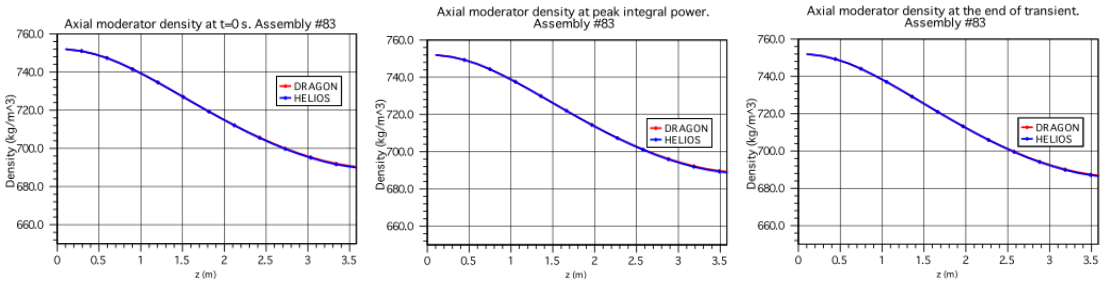


Figure III.15 UO_2 4.2% uncontrolled, burnup 17.5 GWd/t

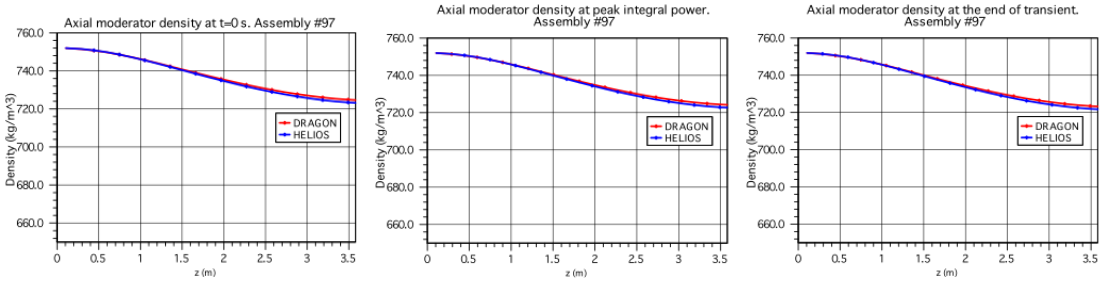


Figure III.16 UO_2 4.2% controlled, burnup 35 GWd/t

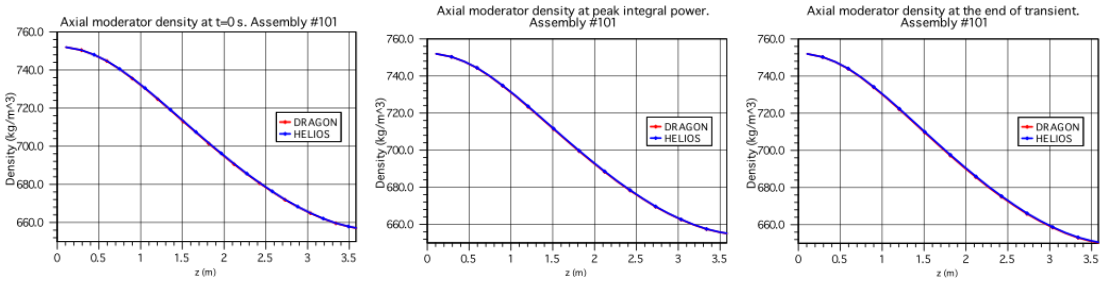


Figure III.17 UO_2 4.5% uncontrolled, burnup 37.5 GWd/t

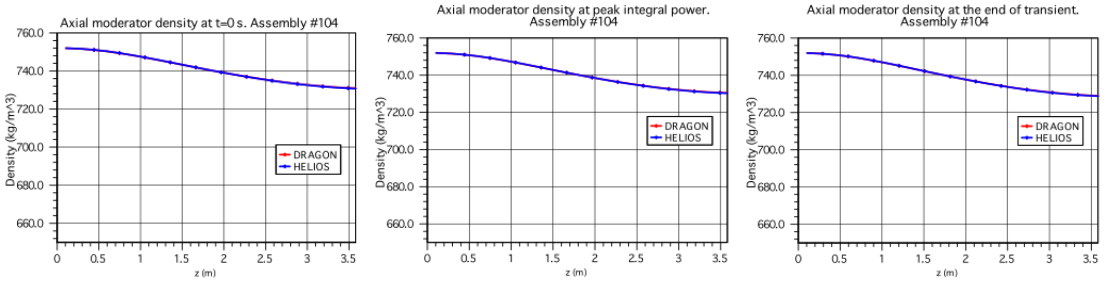


Figure III.18 UO_2 4.2% uncontrolled, burnup 37.5 GWd/t