



Titre: Développement d'un code de thermohydraulique pour le couplage
neutronique / thermohydraulique

Auteur: Majid Fassi Fehri
Author:

Date: 2008

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Fassi Fehri, M. (2008). Développement d'un code de thermohydraulique pour le
couplage neutronique / thermohydraulique [Mémoire de maîtrise, École
Citation: Polytechnique de Montréal]. PolyPublie. <https://publications.polymtl.ca/8249/>

 Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/8249/>
PolyPublie URL:

**Directeurs de
recherche:** Alberto Teyssedou
Advisors:

Programme: Non spécifié
Program:

UNIVERSITÉ DE MONTRÉAL

DÉVELOPPEMENT D'UN CODE DE THERMOHYDRAULIQUE
POUR LE COUPLAGE NEUTRONIQUE / THERMOHYDRAULIQUE

MAJID FASSI FEHRI
DÉPARTEMENT DE GÉNIE PHYSIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

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

AVRIL 2008



Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 978-0-494-41555-9

Our file Notre référence

ISBN: 978-0-494-41555-9

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

DÉVELOPPEMENT D'UN CODE DE THERMOHYDRAULIQUE
POUR LE COUPLAGE NEUTRONIQUE / THERMOHYDRAULIQUE

présenté par : FASSI FEHRI Majid

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. TEYSSEDOU Alberto, Ph.D., membre et directeur de recherche

M. GIRARD René, Ph.D., membre

DÉDICACE

À mes parents,
mes sœurs et
ma famille.

REMERCIEMENTS

Je tiens à remercier très sincèrement M. Alberto Teyssedou pour son aide et son appui inestimables sans lesquelles ce mémoire n'aurait pas vu le jour. Je voudrais également remercier Richard Chambon pour sa grande contribution à la réalisation de ce mémoire. Également, je souhaiterais remercier Alexi Popov pour son aide notamment en ce qui concerne la rédaction en anglais d'une partie de ce mémoire. Aussi, je souhaiterais remercier Mme. Élisabeth Varin, M. Guy Marleau, M. André Olekhnovitch, Mme. Mounia Berdaï et M. Altan Tapucu pour le soutien qu'ils m'ont apporté.

Par ailleurs, je voudrais remercier l'ensemble des membres de l'IGN et plus spécialement mes collègues, pour l'ambiance agréable et stimulante qui m'a permis de réaliser ce mémoire avec beaucoup de plaisir.

Finalement, je remercie très chaleureusement mes parents et mes sœurs pour tout le support et les encouragements indispensables qu'ils m'ont apporté pour l'accomplissement de ce travail.

RÉSUMÉ

Les écoulements diphasiques et le transfert de chaleur dans le domaine de l'énergie nucléaire sont des sujets cruciaux tant pour le fonctionnement que la sécurité des centrales. L'objectif de ce mémoire est de développer un code de thermohydraulique, nommé ARTHUR (Advanced Routines of Thermal-Hydraulic for Unsteady-states Reactors), pour la simulation des écoulements diphasiques dans les canaux des réacteurs nucléaires CANDU. Ce code sera couplé avec des codes de neutronique pour la simulation complète des réacteurs durant les transitoires. Nous avons considéré une géométrie simplifiée des grappes CANDU pour alléger les calculs de thermohydraulique et rendre les temps de calculs couplés transitoires raisonnables. Ainsi, le crayon central de la grappe et l'écoulement autour de celui-ci est considéré comme représentatif de la grappe.

Dans un premier temps, nous avons développé les équations de conservations de la masse, de la quantité de mouvement et de l'énergie selon le modèle d'écoulements diphasique à écart de vitesses. Afin d'avoir un système d'équations fermé, nous avons utilisé des corrélations donnant le taux de vide et le titre de l'écoulement pour l'ébullition sous-refroidie, ainsi que rapport de vitesses entre les phases de l'écoulement. Ces équations ont été discrétisées afin de les programmer en Fortran 90 pour effectuer des simulations numériques. Le schéma de discrétisation choisi ici est de second ordre dans l'espace et de premier ordre, implicite, dans le temps. Pour le calcul des propriétés thermodynamiques et d'échange de l'eau et de la vapeur, nous avons utilisé le code IAPWS-97IF fourni par American Society of Mechanical Engineers en Fortran 77. Nous avons validé la partie écoulements du code de thermohydraulique en comparant les résultats fournis par ce dernier avec des valeurs expérimentales. Par ailleurs, nous avons soulevé les limites quant à l'utilisation des modèles programmés dans le code ARTHUR. Nous avons démontré l'incohérence du modèle à écart de vitesses quant à la conservation de l'énergie. Aussi, nous avons présenté les imprécisions liées aux simplifications géométriques et aux hypothèses.

Nous avons suggéré l'introduction des inversions de flux massique, du calcul des propriétés thermodynamiques et d'échange de l'eau lourde et du flux de chaleur critique principalement.

Dans un second temps, nous avons développé les équations de transfert de chaleur pour le combustible et la gaine d'un crayon de réacteur CANDU. Là encore, nous avons utilisé fixé le coefficient de transfert de chaleur entre le combustible et la gaine et utilisé des corrélations pour le coefficient de transfert de chaleur entre la gaine et le caloporteur. Nous avons, là encore, utilisé le même schéma de discrétisation des équations que celui utilisé pour dans la partie écoulements. Aussi, nous avons utilisé les équations de l'International Nuclear Safety Center pour programmer un code Fortran 90 permettant le calcul des propriétés thermodynamiques et d'échange de l'oxyde d'uranium et du zircaloy-4. Nous avons validé la partie thermique du code de thermohydraulique ARTHUR en comparant les résultats fournis par ce dernier avec des valeurs théoriques. Là encore, nous avons montré les limites du code ARTHUR et présenté les développements futurs nécessaires, notamment la prise en considération du gonflement pour le calcul du transfert de chaleur entre le combustible et la gaine mais aussi l'utilisation de corrélations pour le calcul des propriétés thermodynamiques et d'échange de l' UO_2 lors du craquement des pastilles.

Dans un dernier temps, nous avons développé un schéma de calcul pour les calculs couplés neutronique/thermohydraulique avec le code DONJON-3 et ARTHUR pour des états transitoires et stationnaires. Ce schéma de calcul a été programmé dans l'environnement CLE2000 et testé en analysant les résultats de simulations d'états stationnaires et transitoires pour un réacteur CANDU et un réacteur fictif à quatre canaux. Pour le test de ce code, nous avons simulé une diminution de débit à l'entrée des canaux, une augmentation de la température du caloporteur à l'entrée des canaux et finalement, une baisse de la pression à la sortie des canaux. Par ailleurs, nous avons suggéré l'amélioration du couplage neutronique/thermohydraulique par l'amélioration de la base de donnée du FeedBack Model, l'utilisation d'un code de cinétique utilisant une modélisation numérique de différences finies centrées

implicites dans le temps pour une meilleure stabilité des calculs ainsi que l'introduction des conditions limites de thermohydraulique dues aux collecteurs.

ABSTRACT

Two-phase flow and heat transfer are two important subjects in the field of nuclear energy. Their understanding is crucial for the safety of nuclear power plants. In this thesis, we develop a thermalhydraulic code named ARTHUR (Advanced Routines of Thermal-Hydraulic for Unsteady-states Reactors) for simulating flows and thermal conditions in CANDU channels for stationary and transient states. We couple this code with the neutron-kinetic code DONJON-3 for fully coupled neutronic-thermalhydraulic simulations in steady and unsteady states. We simplified the thermalhydraulic calculations by considering that the flow around the central pin of the bundle is representative for flow around every other pin in the bundle. This consideration reduces the number of calculations needed.

In a first step, we develop the equations for mass, momentum and energy conservation with the drift-flux model. We use correlations to calculate void fraction and quality in subcooled boiling. We also use a correlation to calculate the velocity difference between the two phases in the flow. Then, we use a second-order finite-difference discretisation for space, and a first-order implicit discretisation for time, to implement the numerical model in Fortran 90. To evaluate water and steam thermodynamic and transport properties, we use American Society of Mechanical Engineering code of IAPWS-97IF implemented in Fortran 77. We validate that part of ARTHUR corresponding to thermalhydraulic flow by comparing our results with experimental data. We also discuss limits of said code related to the models used. We point out the incoherence of the energy conservation equation of the drift-flux model. Furthermore, we show the imprecision related to geometry simplifications and suggest some improvements to the code, especially those concerning mass flux inversion, heavy water thermodynamic and transport properties and critical heat flux.

In a second step, we developed the equations for heat transfer in fuel and cladding of CANDU reactors. We use a correlation to evaluate the heat transfer coefficient

between cladding and coolant, then we fix a heat transfer coefficient between fuel and cladding. Also, we use the same discretisation as previously. To calculate the thermodynamic and transport properties of the fuel and cladding, we use equations given by the International Nuclear Safety Center. Then, we validate the thermal part of ARTHUR by comparing the simulations with theoretical values. We point out some of the limits of this code and present future developments related to implementing a correlation for the coefficient of heat transfer between fuel and cladding. We also suggest developments related to the thermodynamic and transport properties of materials in swelling and cracking situations.

Finally, we develop a diagram for coupled calculations in steady and unsteady states with DONJON-3/ARTHUR. We analyse the results for some simulations of a CANDU and a four-channel reactor. We also suggest improvements related to using the Feed-Back cross-section Model (FBM) and to using a kinetic code based on fully implicit finite difference for better calculation stability. We suggest an enhancement for ARTHUR where the user would be able to model the reactor inlet and outlet headers by specifying the relevant boundary conditions of pressure, liquid and vapour enthalpies and void fraction.

TABLE DES MATIÈRES

DÉDICACE	iv
REMERCIEMENTS	v
RÉSUMÉ	vi
ABSTRACT	ix
TABLE DES MATIÈRES	xi
LISTE DES FIGURES	xiii
LISTE DES TABLEAUX	xvi
NOMENCLATURE	xvii
INTRODUCTION	1
CHAPITRE 1 MODÈLE THERMOHYDRAULIQUE	4
1 - Hypothèses et simplifications géométriques	4
2 - Équations de conservation d'écoulement monophasique	7
3 - Définitions et concepts	9
4 - Équations de conservation d'écoulement diphasique	12
5 - Modèle à écart de vitesses	13
6 - Équations constitutives	15
6.1 - Corrélation de Chexal-Lellouche :	15
6.2 - Corrélation de Saha-Zuber :	18
6.3 - Corrélation de Müller-Steinhagen :	20
CHAPITRE 2 MODÈLE NUMÉRIQUE ET VALIDATION POUR LA THERMOHYDRAULIQUE	24
1 - Modèle numérique pour la thermohydraulique	24
2 - Validation du modèle thermohydraulique	29
3 - Limites du modèle	38
CHAPITRE 3 MODÈLE DE TRANSFERT DE CHALEUR DANS LE CRAYON	41
1 - Hypothèses et simplifications	41
2 - Équation de transfert de chaleur	43

3 - Conditions aux limites	44
4 - Équations constitutives	46
4.1 - Corrélation de Markoczy :	46
4.2 - Corrélation de Chen (1966) :	47
CHAPITRE 4 MODÈLE NUMÉRIQUE ET VALIDATION DU TRANSFERT DE CHALEUR.....	50
1 - Modèle numérique de transfert de chaleur	50
2 - Validation du modèle.....	52
CHAPITRE 5 COUPLAGE NEUTRONIQUE/THERMOHYDRAULIQUE	54
1 - Aperçu de la neutronique.....	54
2 - Algorithmes de couplage	58
CHAPITRE 6 SIMULATIONS NEUTRONIQUE/THERMOHYDRAULIQUE .	61
1 - Simulation pour un état stationnaire	61
2 - Simulations pour des états transitoires.....	66
CONCLUSIONS.....	101
BIBLIOGRAPHIE	103
ANNEXES	119
ANNEXE 1 : Expériences d'écoulements.	119
ANNEXE 2 : Solution analytique en thermique.	124
ANNEXE 3 : Guide d'utilisateur des modules THERMO : et THADD :	125
ANNEXE 4 : Guide de la structure L_THERMO :	128
ANNEXE 5 : Guide du programmeur du code ARTHUR.....	132
ANNEXE 6 : Code source.	135
ANNEXE 7 : Fichiers d'entrée x2m et c2m.	242

LISTE DES FIGURES

Figure 1 : Schéma de la cuve du CANDU-6.....	5
Figure 2 : Vue normale d'une grappe.	5
Figure 3 : Géométrie de grappe simplifiée.....	6
Figure 4 : Schéma du canal discrétisé.	24
Figure 5 : Chute de pression dans l'expérience <i>m19</i> (Aubé, 1996).	30
Figure 6 : Chute de pression dans l'expérience <i>m37</i> (Aubé, 1996).	31
Figure 7 : Chute de pression dans l'expérience <i>m38</i> (Aubé, 1996).	31
Figure 8 : Chute de pression dans l'expérience 23 (Aubé, 1996).	33
Figure 9 : Chute de pression dans l'expérience 31 (Aubé, 1996).	33
Figure 10 : Chute de pression dans l'expérience 44v (Aubé, 1996).	34
Figure 11 : Taux de vide dans l'expérience 23 (Aubé, 1996).	34
Figure 12 : Taux de vide dans l'expérience 31 (Aubé, 1996).	35
Figure 13 : Taux de vide dans l'expérience 44v (Aubé, 1996).	35
Figure 14 : Titres dans l'expérience 23 (Aubé, 1996).	36
Figure 15 : Titres dans l'expérience 31 (Aubé, 1996).	36
Figure 16 : Titres dans l'expérience 44v (Aubé, 1996).	37
Figure 17 : Problème d'enthalpie du liquide dans l'expérience 44v (Aubé, 1996). ..	39
Figure 18 : Géométrie simplifiée du crayon central de la grappe.	42
Figure 19 : Schéma du crayon.....	44
Figure 20 : Distribution de température dans un crayon.....	53
Figure 21 : Algorithme de calcul d'ARTHUR.....	59
Figure 22 : Algorithme de couplage neutronique/thermohydraulique.....	59
Figure 23 : Puissance moyenne par grappe.....	62
Figure 24 : Température moyenne par grappe du combustible.	63
Figure 25 : Température du caloporteur.....	63
Figure 26 : Densité du caloporteur.....	64
Figure 27 : Écart relatif sur la puissance entre un calcul couplé et non couplé.	65
Figure 28 : Puissance pour une baisse de débit d'un CANDU.	67

Figure 29 : Température du combustible pour une baisse de débit d'un CANDU.....	67
Figure 30 : Température du caloporteur pour une baisse de débit d'un CANDU.....	68
Figure 31 : Densité du caloporteur pour une baisse de débit d'un CANDU.	68
Figure 32 : Puissance pour une augmentation de la température du caloporteur d'un CANDU.	69
Figure 33 : Température du combustible pour une augmentation de la température du caloporteur d'un CANDU.....	69
Figure 34 : Température du caloporteur pour une augmentation de la température du caloporteur d'un CANDU.....	70
Figure 35: Densité du caloporteur pour une augmentation de la température du caloporteur d'un CANDU.....	70
Figure 36 : Puissance pour une diminution de la pression d'un CANDU.	71
Figure 37 : Température du combustible pour une diminution de la pression d'un CANDU.	71
Figure 38 : Température du caloporteur pour une diminution de la pression d'un CANDU.	72
Figure 39 : Densité du caloporteur pour une diminution de la pression d'un CANDU.	72
Figure 40 : Puissance du réacteur lors d'une réduction de débit.....	74
Figure 41 : Densité de caloporteur lors d'une réduction de débit.	76
Figure 42 : Température du caloporteur lors d'une réduction de débit.....	78
Figure 43 : Température du combustible lors d'une réduction de débit.	80
Figure 44 : Puissance du réacteur lors d'une augmentation de température du caloporteur.	83
Figure 45 : Densité du caloporteur lors d'une augmentation de température du caloporteur.	85
Figure 46 : Température du caloporteur lors d'une augmentation de température du caloporteur.	87
Figure 47 : Température du combustible lors d'une augmentation de température du caloporteur.	89
Figure 48 : Puissance du réacteur lors d'une diminution de pression de sortie.	91
Figure 49 : Densité du caloporteur lors d'une diminution de pression de sortie.	93

Figure 50 : Température du caloporteur lors d'une diminution de pression de sortie.....	95
Figure 51 : Température du combustible lors d'une diminution de pression de sortie.....	97

LISTE DES TABLEAUX

Tableau 1 : Choix de la fonction ψ pour les équations de conservation.....	8
Tableau 2 : Indices des phases de l'écoulement.....	9
Tableau 3 : Limites des corrélations utilisées.	22
Tableau 4 : Propriétés physiques utilisées dans l'équation de conservation de l'énergie thermique.	43
Tableau 5 : Conditions nominales de fonctionnement des CANDU-6.....	62
Tableau 6 : Conditions de transitoires pour CANDU-6.....	66
Tableau 7 : Conditions de transitoires pour un réacteur fictif.....	73
Tableau 8 : Plage de validité du FBM, (Sissaoui <i>et al.</i> , 1999).....	99

NOMENCLATURE

Thermohydraulique :

S	Source
V	Volume de contrôle (m^3)
A	Aire d'une surface (m^2)
ψ	Propriété physique
$\overline{J}$	Flux
t	Temps (s)
z	Position axiale (m)
$\vec{n}$	Vecteur unitaire normal à une surface
ρ	Densité massique (kg / m^3)
m	Masse (kg)
$\dot{m}$	Débit massique (kg / s)
$\vec{v}$	Vitesse (m / s)
$\vec{G}$	Flux massique ($kg / s / m^2$)
$\overline{I}$	Matrice unitaire
p	Pression (Pa)
$\overline{\sigma}$	Contraintes (Pa)
θ	Angle entre la direction de l'écoulement et la verticale ($radians$)
u	Énergie interne (J / kg)
g	Constante de gravitation (m / s^2)
q''	Flux de chaleur (W / m^2)
q'''	Source de chaleur (W / m^3)
k	Indice de phase

l	Indice de la phase liquide
g	Indice de la phase gazeuse
m	Indice du mélange
ε	Taux de vide
x_d	Titre de l'écoulement
x_s	Titre statique
x_{th}	Titre thermodynamique
$\vec{j}$	Flux volumétrique (m/s)
i	Indice représentant la direction
z	Position longitudinale dans le canal (m)
sat	Propriété à l'état de saturation
τ_w	Contrainte due à la paroi (Pa)
A_w	Aire de la paroi (m^2)
D_h	Diamètre hydraulique (m)
P_e	Périmètre mouillé (m)
G_{lo}	Flux massique du liquide dans tout le volume considéré ($kg/s/m^2$)
ρ_M	Densité volumétrique de la quantité de mouvement (kg/m^3)
ρ_E	Densité volumétrique de l'énergie (kg/m^3)
C_0	Coefficient de distribution diphasique
v_{gi}	Écart de vitesse (m/s)
Δh_{lg}	Enthalpie de vaporisation (J/kg)
Re	Nombre de Reynolds
p_{crit}	Pression critique (Pa)
T	Température (K)
C_p	Chaleur spécifique à pression constante ($J/kg/K$)
k	Conductivité thermique ($W/m/K$)
Pe	Nombre de Peclet

μ	Viscosité ($Pa \cdot s$)
ε_r	Rugosité (m)
ζ	Facteur de friction monophasique
Pr	Nombre de Prandl
Nu	Nombre de Nusselt
B	Paramètre géométrique
ϕ	Coefficient de correction géométrique
h_t	Coefficient de transfert de chaleur ($W / m^2 / K$)
x_m	Paramètre de Martinelli
F	Facteur des nombres de Reynolds
σ	Tension de surface (N / m)
ΔT	Différence de température entre la gaine et le caloporteur (K)
Δp	Différence de pression de saturation entre la gaine et le caloporteur (Pa)
r	Positions radiale dans le crayon (m)
r_c	Rayon du combustible (m)
r_g	Rayon interne de la gaine (m)
r_s	Rayon externe de la gaine (m)
k_c	Conductivité thermique du combustible ($W / m / K$)
k_g	Conductivité thermique de la gaine ($W / m / K$)
h_{gap}	Coefficient de transfert de chaleur à l'interstice ($W / m^2 / K$)
T_c	Température du combustible à la surface de la pastille (K)
T_g	Température de la gaine au rayon interne (K)
T_s	Température de la gaine au rayon externe (K)
T_∞	Température du caloporteur (K)

Neutronique :

$\vec{r}$	Position spatiale des neutrons
-----------	--------------------------------

V_n	Vitesse des neutrons (m/s)
$\vec{\Omega}$	Direction de déplacement des neutrons
n	Densité neutronique (m^{-3})
ϕ	Flux neutronique (m^{-2}/s)
Σ	Section efficace macroscopique (m^2)
N	Composition isotopique
E	Énergie (J)
t	Temps (s)
m	Masse (kg)
C	Concentration isotopique
λ	Temps de demi-vie (s)
η	Rendement de fission

INTRODUCTION

La production énergétique par le nucléaire au Canada, durant les dernières décennies, a amené les recherches à mieux prévoir le comportement des réacteurs. Au Canada, ces recherches ont porté tant sur la neutronique que sur la thermohydraulique des réacteurs CANDU. En effet, plusieurs codes de simulations ont été développés, puis améliorés durant cette période. Nous pouvons noter parmi les nombreux codes existants, CATHENA (Hanna, 1998) pour la thermohydraulique et DONJON-3 (Varin *et al.*, 2005) pour la neutronique. Cependant, peu de travaux ont été réalisés sur le couplage entre la thermohydraulique et la neutronique pour les réacteurs CANDU. Pour cause, les effets d'un domaine sur l'autre ont toujours été considérés faibles et trop coûteux en calculs. L'amélioration des capacités de calcul ainsi que le désir de prévoir les effets des pertes de caloporteurs dans les réacteurs pour des raisons de sûreté ont relancé l'idée de couplage neutronique/thermohydraulique. C'est ainsi que plusieurs tentatives ont été faites, notamment avec DONJON-3/NDF et ASSERT-PV (Dionne, 2001) à l'état stationnaire. Le fait d'avoir des codes de thermohydraulique traitant la géométrie complexe des grappes de CANDU a limité grandement le couplage, les calculs étant trop longs.

Nous allons tenter, dans ce travail, de développer un code de thermohydraulique permettant le calcul des propriétés des écoulements dans les canaux ainsi que les transferts de chaleur entre les crayons et le caloporteur. Nous utiliserons des simplifications géométriques des grappes CANDU et des hypothèses permettant d'alléger le nombre de calculs. Ainsi, ce code nous permettra un couplage plus aisé de la thermohydraulique et de la neutronique. Le code de thermohydraulique, nommé ARTHUR (Advanced Routines of Thermal-Hydraulic for Unsteady-states Reactors), sera couplé avec le code de neutronique DONJON-3 pour les calculs stationnaires et transitoires.

Pour ce faire, nous débuterons ce mémoire par une présentation des simplifications et hypothèses liées aux écoulements monophasiques et diphasiques dans les canaux de réacteurs CANDU. Ceci nous amènera à développer les équations de conservation de la masse, de la quantité de mouvement et de l'énergie pour les mélanges selon le modèle à écart de vitesse. Aussi, nous devons compléter ce système d'équations par le choix de relations de fermetures données par des corrélations, notamment celles concernant l'ébullition sous-refroidie, la chute de pression par frottement ou encore la différence de vitesse entre les phases. Les propriétés thermodynamiques et d'échange du caloporteur nécessaires au calcul seront prises de banque de données, dont nous justifierons le choix. Par la suite, nous présenterons le modèle de discrétisation des équations choisi pour la programmation en Fortran 90. Cette présentation contiendra la forme matricielle choisie pour la solution des équations ainsi que la méthode de résolution d'équations matricielle. Finalement, nous validerons la partie écoulement du code de thermohydraulique en comparant les résultats de simulations aux résultats expérimentaux dont nous disposons. Ces résultats nous donneront un aperçu des limites de la partie reliée écoulements du code.

Également, nous traiterons la partie transfert de chaleur en commençant par présenter les hypothèses nécessaires pour simplifier les calculs avant de présenter les équations de transfert de chaleur dans le combustible et la gaine du crayon. Celles-ci nécessiteront des relations de fermeture pour le coefficient de transfert de chaleur entre le combustible et la gaine mais aussi entre la gaine et le caloporteur. Par la suite, nous montrerons le modèle de discrétisation des équations choisi ainsi que la formulation matricielle des équations à résoudre. Nous finirons cette partie par la présentation de quelques résultats de simulations pour la partie transfert de chaleur du code ARTHUR. Nous soulèverons, à la suite de ces simulations, les limites de la partie thermique du code.

Finalement, nous présenterons le schéma de calcul pour le code de thermohydraulique contenant la partie écoulement et la partie transfert de chaleur. Cette étape nous permettra d'envisager le couplage avec le code de neutronique

DONJON-3 (Varin *et al.*, 2005) en présentant un schéma de calcul pour le couplage. Ce schéma sera également validé en analysant les résultats de simulation. Nous finirons ce mémoire par une présentation des développements futurs nécessaires pour apporter des améliorations au code ARTHUR et au couplage DONJON-3/ARTHUR.

CHAPITRE 1

MODÈLE THERMOHYDRAULIQUE

La confirmation et l'amélioration des marges de sûreté dans l'exploitation des réacteurs CANDU s'appuient sur la compréhension des écoulements monophasiques et diphasiques. Cette compréhension amène aussi la possibilité d'améliorer leur rendement. Ainsi, nous nous intéresserons dans cette partie aux écoulements dans les canaux de réacteurs CANDU. Étant donnée la complexité géométrique des grappes utilisées dans les CANDU, nous procéderons à une simplification géométrique qui allégera les calculs pour les écoulements. Ces calculs seront faits en utilisant les équations de conservation pour des écoulements diphasiques. Nous présenterons ces équations sous la forme donnée par le modèle à écart de vitesses.

1 - Hypothèses et simplifications géométriques

Les réacteurs CANDU possèdent une géométrie singulière et particulièrement complexe. Ces derniers utilisent de l'eau lourde, comme caloporteur, véhiculée dans les tubes de force horizontaux. Nous comptons 380 tubes de force dans ces réacteurs dont les directions des écoulements sont opposées pour deux canaux voisins, selon un schéma en damier. De chaque côté du réacteur, nous avons des collecteurs d'entrée et de sortie. À cette complexité géométrique s'ajoute celle des grappes de combustible. En effet, nous retrouvons 12 grappes dans chaque canal, chacune composée de 37 crayons de combustible selon une géométrie, là encore complexe. Les deux figures 1 et 2 montrent clairement la complexité géométrique des réacteurs CANDU.

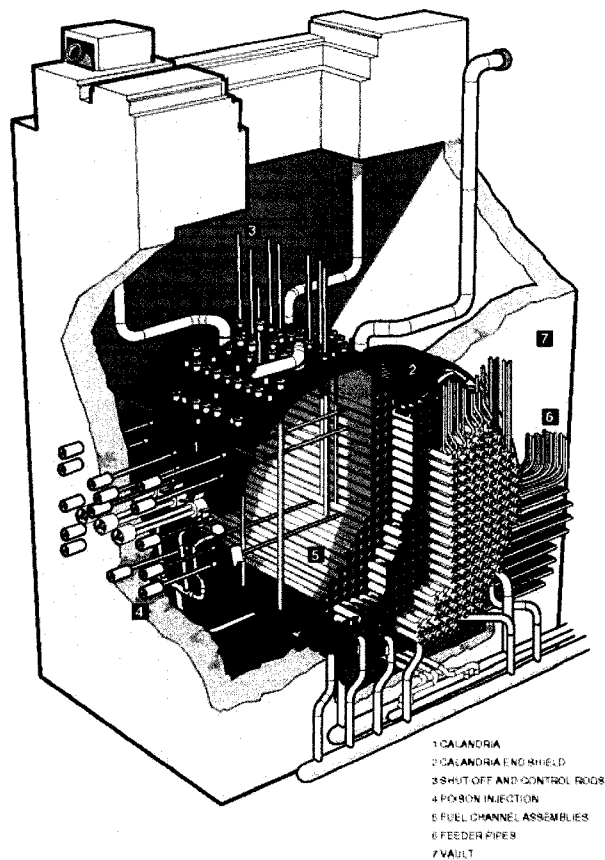


Figure 1 : Schéma de la cuve du CANDU-6.

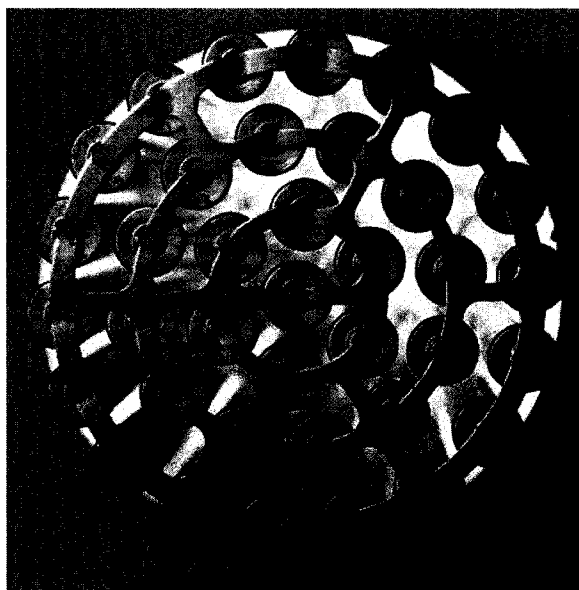


Figure 2 : Vue normale d'une grappe.

Il apparaît donc nécessaire de simplifier la géométrie considérée afin d'avoir des calculs plus simples à traiter. Pour cela, nous allons considérer l'écoulement qui se déroule autour du crayon central de la grappe et délimité l'hexagone formé par les centres des crayons du premier anneau. On supposera aussi que le crayon a la longueur du canal et qu'il n'y a pas d'entretoises. Cette géométrie simplifiée est présentée dans le schéma suivant.

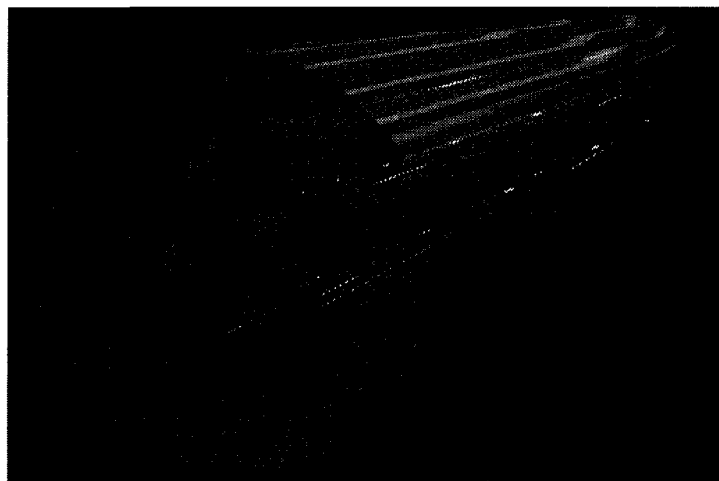


Figure 3 : Géométrie de grappe simplifiée.

L'hypothèse principale de ce mémoire consiste à supposer que l'écoulement autour du crayon central est représentative de l'écoulement dans le canal. Cette hypothèse simplifiera de beaucoup la quantité de calcul. Nous supposons par ailleurs que le flux massique à l'entrée de ce volume est proportionnel au débit massique. Cette proportionnalité est évaluée à partir du rapport des aires de l'entrée de l'écoulement de la géométrie simplifiée par rapport à celle de la géométrie réelle. Ceci découle de l'hypothèse selon laquelle le flux massique est uniforme à l'entrée du canal réel. Nous considérons également que les conditions frontières de l'écoulement sont fixées à l'entrée pour le flux massique et la température; à la sortie, nous fixons la pression. Finalement, nous supposons que seules les variations axiales ne sont pas négligeables et sont donc à calculer. Ces considérations seront donc prises en compte par la suite lors de la présentation des équations de conservation.

2 - Équations de conservation d'écoulement monophasique

Étant donné que l'objectif de ce mémoire n'est pas de redémontrer les équations de conservation, nous les présenterons de façon sommaire. Nous tenons néanmoins à référer au lecteur intéressé par les démonstrations, entre autres, les livres d'Ishii (2006), Kolev (2002) et de Todreas (1990). Ces ouvrages présentent le sujet de façon bien plus élégante et détaillée que nous ne pourrions prétendre le faire ici.

Pour obtenir une équation de conservation générale, nous allons faire le bilan d'une propriété ψ dans le volume V . La variation de cette propriété, dans le volume concerné, par rapport au temps est égale à la génération de cette même propriété, dans le volume considéré, à laquelle s'ajoute le flux net qui traverse la surface A délimitant ce volume. Cette formulation peut s'écrire de la façon suivante :

$$\frac{d}{dt} \int_{V(t)} \psi dV = \int_{V(t)} S dV - \int_{A(t)} \bar{\vec{J}} \cdot \vec{n} dA. \quad (1.1)$$

Cette équation peut être réécrite, en utilisant le théorème de transport de Reynolds ainsi que le théorème de Gauss sous la forme suivante :

$$\frac{\partial \psi}{\partial t} + \vec{\nabla} \cdot (\psi \vec{v}) + \vec{\nabla} \cdot \bar{\vec{J}} - S = 0. \quad (1.2)$$

Pour déduire de cette équation générale les équations de conservation de la masse, de la quantité de mouvement et d'énergie, nous allons remplacer la propriété abstraite ψ , sa source et son flux par ceux d'une propriété physique selon le tableau 1.

Tableau 1 : Choix de la fonction ψ pour les équations de conservation.

	Propriété ψ	Source S	Flux $\bar{\bar{J}}$
Équation de conservation de la masse	ρ	0	0
Équation de conservation de la quantité de mouvement	$\rho \vec{v}$	$\rho \vec{g} \cos(\theta)$	$p \bar{\bar{I}} - \bar{\bar{\sigma}}$
Équation de conservation de l'énergie	$\rho \left(u + \frac{\vec{v} \cdot \vec{v}}{2} \right)$	$\rho \vec{g} \cdot \vec{v} \cos(\theta) + q'''$	$q'' + \left(p \bar{\bar{I}} - \bar{\bar{\sigma}} \right) \vec{v}$

Ainsi, nous pouvons écrire les équations de conservation pour une phase. L'équation de conservation de la masse s'écrit donc :

$$\frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot (\rho \vec{v}) = 0. \quad (1.3)$$

Dans cette équation, nous pouvons définir le produit de la densité par la vitesse, le flux massique qui est noté $\vec{G} = \rho \vec{v}$.

Aussi, nous pouvons écrire l'équation de la conservation de la quantité de mouvement pour une phase de la façon suivante :

$$\frac{\partial}{\partial t} (\rho \vec{v}) + \vec{\nabla} \cdot (\rho \vec{v} \cdot \vec{v}) + \vec{\nabla} \cdot (p \bar{\bar{I}} - \bar{\bar{\sigma}}) - \rho \vec{g} \cos(\theta) = 0. \quad (1.4)$$

Finalement, nous pouvons écrire l'équation de conservation d'énergie de la façon suivante :

$$\begin{aligned} & \frac{\partial}{\partial t} \left[\rho \left(u + \frac{\vec{v} \cdot \vec{v}}{2} \right) \right] + \vec{\nabla} \cdot \left[\rho \left(u + \frac{\vec{v} \cdot \vec{v}}{2} \right) \vec{v} \right] \\ & + \vec{\nabla} \cdot \left[q'' + (p \bar{\bar{I}} - \bar{\bar{\sigma}}) \vec{v} \right] - \rho \vec{g} \cdot \vec{v} \cos(\theta) + q''' = 0 \end{aligned} \quad (1.5)$$

Là encore, nous pouvons introduire une nouvelle définition, celle de l'enthalpie de l'écoulement monophasique $h = u + \frac{P}{\rho}$.

Cette brève présentation est suffisante pour introduire les équations de conservations pour un mélange de deux phases. Cependant, il nous est nécessaire de développer des définitions et concepts qui faciliteront la généralisation de ces équations pour les mélanges à deux phases.

3 - Définitions et concepts

Nous avons jusque là présenté les équations de conservation pour une phase. Afin de généraliser ces équations aux écoulements diphasiques, nous allons introduire les indices représentant les phases de l'écoulement (voir tableau 2).

Tableau 2 : Indices des phases de l'écoulement.

Indice	Phase liquide	Phase gazeuse
k	l	g

Nous allons également introduire de nouvelles définitions qui nous permettront de présenter les équations de conservation pour les mélanges à deux phases. Ces définitions seront toutes basées sur la moyenne de propriétés dans un volume donné ou bien sur une aire considérée. Tout d'abord, nous allons présenter la notion de taux de vide, notée ε et définie par :

$$\varepsilon = \varepsilon_g = \frac{\int_V dV_g}{\int_V dV} = \frac{V_g}{V_g + V_l}, \quad (1.6)$$

$$1 - \varepsilon = \varepsilon_l = \frac{\int_V dV}{\int_V dV} = \frac{V_l}{V_g + V_l}. \quad (1.7)$$

Par ailleurs, il est nécessaire d'introduire la définition de la densité du mélange :

$$\rho_m = \sum_k \varepsilon_k \rho_k = (1 - \varepsilon) \rho_l + \varepsilon \rho_g. \quad (1.8)$$

Le titre statique est défini de la façon suivante :

$$x_s = \frac{\varepsilon \rho_g}{\varepsilon \rho_g + (1 - \varepsilon) \rho_l} = \frac{m_g}{m_g + m_l}. \quad (1.9)$$

Le titre de l'écoulement est également une définition nécessaire. Il est donné par :

$$x_d = \frac{\int_{A_g} \varepsilon_g \rho_g \vec{v}_g \cdot d\vec{A}}{\sum_k \int_{A_k} \varepsilon_k \rho_k \vec{v}_k \cdot d\vec{A}} = \frac{\dot{m}_g}{\dot{m}_g + \dot{m}_l}. \quad (1.10)$$

Lorsque la vitesse des deux phases est la même, les deux titres précédents deviennent égaux. Par ailleurs, nous définissons le flux massique du mélange de la façon suivante :

$$\vec{G}_m = \sum_k \vec{G}_k = \sum_k \varepsilon_k \rho_k \vec{v}_k. \quad (1.11)$$

Également, nous pouvons définir le flux volumétrique (ou vitesse superficielle totale) :

$$\vec{j} = \sum_k \vec{j}_k = \sum_k \frac{\vec{G}_k}{\rho_k} = \sum_k \varepsilon_k \vec{v}_k. \quad (1.12)$$

L'enthalpie du mélange est définie comme suit :

$$h_m = \frac{\varepsilon \rho_g h_g + (1 - \varepsilon) \rho_l h_l}{\varepsilon \rho_g + (1 - \varepsilon) \rho_l} . \quad (1.13)$$

Cette dernière est différente de l'enthalpie de l'écoulement qui est définie de la façon suivante :

$$h_d = x_d h_g + (1 - x_d) h_l . \quad (1.14)$$

Ces deux enthalpies sont égales lorsque la vitesse des deux phases est la même. Lorsque le gaz et le liquide sont à l'équilibre dans le mélange, on peut définir le titre thermodynamique comme suit :

$$x_{th} = \frac{h_m - (h_l)_{sat}}{(h_g)_{sat} - (h_l)_{sat}} . \quad (1.15)$$

En projetant les propriétés vectorielles selon la direction i (c'est-à-dire, selon la direction z par exemple), nous pouvons définir un rapport de glissement entre les phases comme suit :

$$s = \frac{(v_g)_i}{(v_l)_i} = \frac{x_d}{1 - x_d} \frac{\rho_l}{\rho_g} \frac{1 - \varepsilon}{\varepsilon} . \quad (1.16)$$

Cette relation clôt la présentation des définitions et concepts. Ainsi, nous pouvons développer les équations de conservation pour un mélange de deux phases.

4 - Équations de conservation d'écoulement diphasique

Pour le développement des équations de conservation pour les écoulements diphasique selon la direction z , nous allons reprendre l'équation générale de conservation (1.2) que nous allons appliquer aux propriétés du tableau 1. Contrairement à ce que l'on a fait précédemment, cette fois-ci, les propriétés sont d'abord moyennées dans le volume de contrôle contenant deux phases. Les résultats sont les suivants :

- Équation de conservation de la masse :

$$\frac{\partial}{\partial t}(\varepsilon \rho_g + (1-\varepsilon)\rho_l) + \frac{\partial}{\partial z}(\varepsilon v_g \rho_g + (1-\varepsilon)v_l \rho_l) = 0. \quad (1.17)$$

- Équation de conservation de la quantité de mouvement :

$$\begin{aligned} \frac{\partial}{\partial t}(\varepsilon v_g \rho_g + (1-\varepsilon)v_l \rho_l) + \frac{\partial}{\partial z}(\varepsilon v_g^2 \rho_g + (1-\varepsilon)v_l^2 \rho_l) + \frac{\partial p}{\partial z} \\ + (\varepsilon \rho_g + (1-\varepsilon)\rho_l)g \cos(\theta) + \left(\frac{\partial p}{\partial z}\right)_{\text{frottement}} = 0 \end{aligned} \quad (1.18)$$

Dans cette équation, nous avons réécrit le terme de contraintes au niveau de la paroi solide comme suit :

$$\frac{A_w}{V} \sum_k \frac{1}{A_{kw}} \int (\vec{n}_{kw} \cdot \vec{\sigma}_k) \cdot \vec{dA} = -\frac{A_w}{V} \tau_w = -\left(\frac{\partial p}{\partial z}\right)_{\text{frottement}}, \quad (1.19)$$

où nous avons introduit le diamètre hydraulique, $D_h = \frac{4A}{P_e}$ et le terme de perte de

pression par frottement $\left(\frac{\partial p}{\partial z}\right)_{\text{frottement}}$ qui sera calculé en utilisant la corrélation de

Müller-Steinhagen (Müller-Steinhagen & Heck, 1986).

- Équation de conservation de l'énergie :

$$\begin{aligned}
 & \frac{\partial}{\partial t} (\varepsilon \rho_g h_g + (1-\varepsilon) \rho_l h_l) + \frac{\partial}{\partial t} \left(\frac{\varepsilon \rho_g v_g^2}{2} + \frac{(1-\varepsilon) \rho_l v_l^2}{2} \right) \\
 & - \frac{\partial p}{\partial t} + \frac{\partial}{\partial z} (\varepsilon \rho_g h_g v_g + (1-\varepsilon) \rho_l h_l v_l) \\
 & + \frac{\partial}{\partial z} \left(\frac{\varepsilon \rho_g v_g^3}{2} + \frac{(1-\varepsilon) \rho_l v_l^3}{2} \right) = (\varepsilon \rho_g v_g + (1-\varepsilon) \rho_l v_l) g + \frac{A_w q''}{V}
 \end{aligned} \quad (1.20)$$

Pour l'obtention de cette équation, nous avons introduit les hypothèses suivantes : la vitesse du fluide à la paroi est nulle, la conduction du fluide est négligeable, les contraintes dues à la viscosité sont négligeables et la production de chaleur dans le fluide nulle.

5 - Modèle à écart de vitesses

La littérature concernant les fluides présente quatre modèles pour les écoulements à deux phases classés selon leur complexité en ordre croissant : homogène, séparé, à écart de vitesse et à deux fluides. Il est certain que plus le modèle utilisé est complexe, plus la quantité de calcul est grande mais meilleurs sont les résultats du fait que le modèle utilise moins d'hypothèses simplificatrices. Il est donc nécessaire de trouver un juste milieu entre les modèles existants afin de limiter le nombre de calculs tout en améliorant les résultats de ces calculs. C'est ainsi que le modèle à écart de vitesses a été choisi. Nous présenterons donc par la suite les équations liées au modèle à écart de vitesses pour une représentation 1D. Les équations de ce modèle sont :

- Équation de conservation de la masse :

$$\frac{\partial \rho_m}{\partial t} + \frac{\partial G}{\partial z} = 0, \quad (1.21)$$

- Équation de conservation de la quantité de mouvement :

$$\begin{aligned} \frac{\partial G}{\partial t} + \frac{\partial}{\partial z} \left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj}^2 \right) \\ + \frac{\partial p}{\partial z} + \rho_m g \cos(\theta) + \left(\frac{\partial p}{\partial z} \right)_{\text{frottement}} = 0 \end{aligned} \quad (1.22)$$

- Équation de conservation de l'énergie :

$$\begin{aligned} \frac{\partial}{\partial t} \left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m g z \cos(\theta) \right) + \\ \frac{\partial}{\partial z} \left(G h_m + \varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + G g z \cos(\theta) \right) = \frac{A_w q''}{V} \end{aligned} \quad (1.23)$$

où nous avons introduit l'écart de vitesse $\bar{v}_{gj}$ qui sera défini par la suite.

Dans ces équations, nous avons défini les entités suivantes :

$$\frac{1}{\rho_E^2} = \frac{x_d^3}{\varepsilon^2 \rho_g^2} + \frac{(1-x_d)^3}{(1-\varepsilon)^2 \rho_l^2}, \quad (1.24)$$

$$\frac{1}{\rho_M} = \frac{x_d^2}{\varepsilon \rho_g} + \frac{(1-x_d)^2}{(1-\varepsilon) \rho_l}, \quad (1.25)$$

$$\bar{v}_{gj} = v_{gj} + (C_o - 1)j, \quad (1.26)$$

$$\Delta h_{lg} = h_g - h_l, \quad (1.27)$$

où C_o est le coefficient de distribution de Zuber-Findlay (1965) et x_d le titre de l'écoulement donné par l'équation 1.10.

Le modèle à écart de vitesse nécessite l'utilisation de corrélations pour fermer le système d'équations. Ainsi, nous montrerons dans la section suivante les corrélations que nous allons utiliser dans nos calculs.

6 - Équations constitutives

Nous présenterons dans cette partie les corrélations utilisées comme équations constitutives pour le modèle à écart de vitesse. Nous utiliserons la corrélation de Chexal-Lellouche (Chexal *et al.*, 1992) pour calculer l'écart de vitesse entre les phases et le coefficient de distribution dans l'écoulement C_o . Également, nous utiliserons la corrélation de Saha-Zuber (1974) pour calculer le taux de vide et les conditions d'ébullition sous-refroidie, i.e. la température de sous-refroidissement. Finalement, nous utiliserons celle de Müller-Steinhagen (Müller-Steinhagen & Heck, 1986) pour le calcul de la perte de pression par frottement sur les parois.

6.1 - Corrélation de Chexal-Lellouche :

Pour le calcul de l'écart de vitesse et le coefficient de distribution C_o , la corrélation de Chexal-Lellouche (Chexal *et al.*, 1992) reste indéniablement la corrélation la plus précise et la plus recommandée pour effectuer ce type de calculs. Étant donné que nous ne considérons pas dans ce mémoire l'inversion de l'écoulement dans le réacteur, nous ne présenterons que la partie de la corrélation pour les écoulements co-courant. Les paramètres que nous allons utiliser dans cette corrélation sont les suivants :

Le nombre de Reynolds, pour chaque phase, s'écrit :

$$\text{Re}_g = \frac{x_d G D_h}{\mu_g}, \quad (1.28)$$

$$\text{Re}_l = \frac{(1-x_d) G D_h}{\mu_l}. \quad (1.29)$$

On procède de la manière suivante au calcul du coefficient de distribution C_o pour l'écoulement :

$$C_0 = F C_{0V} + F C_{0H}, \quad (1.30)$$

$$F = \frac{90^\circ - \frac{\theta}{\pi} \times 180^\circ}{90^\circ}, \quad (1.31)$$

$$C_{0V} = \frac{L}{K_0 + (1-K_0)\varepsilon^r}, \quad (1.32)$$

$$L = \frac{1 - e^{-C_1 \varepsilon}}{1 - e^{-C_1}}, \quad (1.33)$$

$$C_1 = \frac{4p_{crit}^2}{p(p_{crit} - p)}, \quad (1.34)$$

$$K_0 = B_1 + (1-B_1) \left(\frac{\rho_g}{\rho_l} \right)^{1/4}, \quad (1.35)$$

$$r = \left(1 + 1.57 \frac{\rho_g}{\rho_l} \right) / (1-B_1), \quad (1.36)$$

$$B_1 = \min(0.8, A_1), \quad (1.37)$$

$$A_1 = \frac{1}{1 + e^{-Re/60000}} , \quad (1.38)$$

$$Re = \begin{cases} Re_g, Re_g > Re_1 \\ Re_l, Re_g \leq Re_1 \end{cases} , \quad (1.39)$$

$$C_{0H} = (1 + \varepsilon^{0.05} (1 - \varepsilon)^2) C_{0V} . \quad (1.40)$$

Pour le calcul de l'écart de vitesse, nous avons :

$$v_{gj} = F v_{gfv} + (1 - F) v_{gfh} , \quad (1.41)$$

$$v_{gfv} = v_{gj}^0 C_9 , \quad (1.42)$$

$$v_{gj}^0 = 1.41 \left(\frac{(\rho_l - \rho_g) \sigma g}{\rho_l^2} \right)^{0.25} C_2 C_3 C_4 , \quad (1.43)$$

$$C_9 = (1 - \varepsilon)^{B_1} , \quad (1.44)$$

$$C_2 = \begin{cases} 0.4757 \left(\ln \left(\frac{\rho_l}{\rho_g} \right) \right)^{0.7} , \frac{\rho_l}{\rho_g} \leq 18 \\ 1, C_5 \geq 1 \\ \frac{1}{1 - e^{-C_5/(1-C_5)}} , C_5 < 1, \frac{\rho_l}{\rho_g} > 18 \end{cases} , \quad (1.45)$$

$$C_5 = (150 / (\rho_l / \rho_g))^{1/2} , \quad (1.46)$$

$$C_4 = \begin{cases} 1, C_7 \geq 1 \\ \frac{1}{1 - e^{-C_8}}, C_7 < 1 \end{cases} \quad (1.47)$$

$$C_7 = \left(\frac{D_2}{D_h} \right)^{0.6}, \quad (1.58)$$

$$C_8 = \frac{C_7}{1 - C_7}, \quad (1.49)$$

$$D_2 = 0.09144 \text{ m}, \quad (1.50)$$

$$C_3 = \max \left\{ \begin{array}{l} 0.5 \\ 2e^{-Re_f/60000} \end{array} \right., \quad (1.51)$$

$$v_{gh} \approx 0 \quad . \quad (1.52)$$

Pour nos calculs, les propriétés de la vapeur seront considérées à la saturation contrairement à celles du liquide qui seront considérées à l'état de sous refroidissement ou bien de saturation selon les valeurs de température et de pression locales données par les calculs.

6.2 - Corrélation de Saha-Zuber :

Nous avons porté notre choix sur l'utilisation de la corrélation de Saha-Zuber parce que celle-ci suggère que le détachement des bulles de la paroi est contrôlé thermiquement et hydrodynamiquement. Nous avons estimé que ces deux phénomènes seront présents dans le cas des réacteurs CANDU. Les calculs du taux de vide et de la température dans la zone d'ébullition sous-refroidie sont effectués selon la procédure suivante :

Nous définissons le sous-refroidissement par :

$$\Delta T = T_{sat} - T_l. \quad (1.53)$$

Aussi, nous introduisons le nombre de Peclet donné par :

$$Pe = \frac{GD_h C_{pl}}{k_l}. \quad (1.54)$$

La corrélation de Saha-Zuber définit deux régimes selon le nombre de Peclet. En effet, pour un flux massique faible, i.e. un nombre de Peclet faible, la diffusion de chaleur contrôle les phénomènes d'évaporation et de condensation. Ainsi, le détachement des bulles de la paroi, dû au taux de condensation de la vapeur et à celui d'évaporation à la paroi, ne dépend que des conditions thermiques locales données par le nombre de Nusselt. Par ailleurs, pour un flux massique élevé, les phénomènes d'évaporation et de condensation sont contrôlés par le transport hydrodynamique. En effet, le détachement des bulles à la paroi est régi par l'équilibre des forces exercées sur la bulle données par le nombre de Stanton. Ces deux régimes sont présentés par l'équation 1.55.

$$\begin{cases} Nu = \frac{q'' D_h}{k_l \Delta T} = 455, Pe \leq 70000 \\ St = \frac{q''}{GC_{pl} \Delta T}, Pe > 70000 \end{cases}. \quad (1.55)$$

De là, nous pouvons trouver le titre de l'écoulement (Kroeger & Zuber, 1968; Zuber *et al.*, 1966) :

$$x_d = \frac{x_{th} - x e^{(x_{th}/x-1)}}{1 - x e^{(x_{th}/x-1)}}, \quad (1.56)$$

$$\text{avec } x = -\frac{C_{pl}\Delta T}{(\Delta h_{lg})_{sat}}. \quad (1.57)$$

Aussi, le taux de vide peut être évalué selon :

$$\varepsilon = \frac{x_d}{C_0 \left(x_d + (1 - x_d) \frac{\rho_g}{\rho_l} \right) + \frac{\rho_g}{G} v_{gj}}. \quad (1.58)$$

Le coefficient de distribution C_0 dans l'écoulement ainsi que l'écart de vitesse entre les phases est calculé avec la corrélation de Chexal-Lellouche (Chexal *et al.*, 1992).

6.3 - Corrélation de Müller-Steinhagen :

Le calcul de la perte de pression en écoulement monophasique et diphasique peut être fait grâce à la corrélation de Müller-Steinhagen (Müller-Steinhagen & Heck, 1986). Le choix de cette corrélation a été fait pour la précision de ses résultats et sa simplicité. En effet, selon Thome *et al.* (2002), cette corrélation serait la plus intéressante à utiliser. La formulation de cette corrélation est la suivante :

$$\left(\frac{dp}{dz} \right)_{\text{frottement}} = (A + 2(B - A)x_d) \cdot (1 - x_d)^{1/C} + Bx_d^C, \quad (1.59)$$

où les coefficients sont donnés par :

$$A = \zeta_l \frac{G^2}{2\rho_l D_h}, \quad (1.60)$$

$$B = \zeta_g \frac{G^2}{2\rho_g D_h}, \quad (1.61)$$

$$C = 3. \quad (1.62)$$

Les facteurs frictions pour la phase liquide et gazeuse, ζ_l et ζ_g , sont respectivement donnés par les équations 1.63 et 1.64.

$$\begin{cases} \zeta_l = \frac{64}{Re_l}, Re_l \leq 1187 \\ \zeta_l = \frac{0.3164}{Re_l^{1/4}}, Re_l > 1187 \end{cases}, \quad (1.63)$$

$$\begin{cases} \zeta_g = \frac{64}{Re_g}, Re_g \leq 1187 \\ \zeta_g = \frac{0.3164}{Re_g^{1/4}}, Re_g > 1187 \end{cases}, \quad (1.64)$$

où les nombres de Reynolds pour chaque phase est calculé selon les expressions suivantes:

$$\begin{cases} Re_l = \frac{GD_h}{\mu_l} \\ Re_g = \frac{GD_h}{\mu_g} \end{cases}. \quad (1.65)$$

Pour l'équation 1.65, nous avons considéré que l'écoulement est monophasique ne contenant que du liquide pour le calcul du nombre de Reynolds du liquide, soit complètement gazeux pour le calcul du nombre de Reynolds du gaz. Nous pouvons retrouver l'équation 1.65 avec les équations 1.28 et 1.29 avec un titre de l'écoulement égale à 0 pour un écoulement monophasique de liquide et 1 pour un écoulement monophasique de gaz. Cette corrélation ne prend pas en compte la rugosité de la surface sur laquelle il y a frottement. Ainsi, nous apporterons à celle-ci une modification permettant de prendre en considération la rugosité du zircaloy. Cette modification est donnée par la relation de Coolbrooke, tirée du livre de Todreas et Kazimi (1990). La nouvelle formulation des coefficients de frottement devient donc :

$$\left\{ \begin{array}{l} \zeta_l = \frac{1}{4 \left(\log_{10} \left(\frac{\varepsilon_r}{3.7 D_h} + \frac{2.51}{\text{Re}_l \sqrt{\zeta_l}} \right) \right)^2}, \text{Re}_l > 1187 \\ \zeta_g = \frac{1}{4 \left(\log_{10} \left(\frac{\varepsilon_r}{3.7 D_h} + \frac{2.51}{\text{Re}_l \sqrt{\zeta_g}} \right) \right)^2}, \text{Re}_g > 1187 \end{array} \right. \quad (1.66)$$

Nous utiliserons cette nouvelle formulation pour nos calculs.

Les limites dans lesquelles ont été validés ces corrélations sont présenté dans le tableau 3. Elles nous permettront d'analyser la validité de nos calculs par la suite, notamment au chapitre 2.

Tableau 3 : Limites des corrélations utilisées.

Corrélation	Pression (kPa)	Flux massique (kg/m ² /s)	Flux de chaleur (kW/m ²)
Chexal-Lellouche (Chexal <i>et al.</i> ,1992)	100 - 18000	0.01 - 2550	1.3 – 1130
Saha-Zuber (1974)	101 - 13800	400 - 1050	280 – 1890
Müller-Steinhaggen (Müller-Steinhaggen & Heck,1986)	Données non fournies	291 - 2490	Données non fournies

Les corrélations précédentes seront donc toutes utilisées dans nos calculs. Celles-ci présentent les meilleures caractéristiques pour les calculs d'écoulements diphasique que nous allons effectuer. En effet, la corrélation de Chexal-Lellouche (Chexal *et al.*,1992) est dans la littérature de thermohydraulique la corrélation la plus recommandée. Aussi, étant donné que nous supposons que le détachement des bulles à la paroi est contrôlé thermiquement et hydrodynamiquement comme le suppose la corrélation de Saha-Zuber (1974), il va de soi que celle-ci est la plus adaptée à nos calculs. Finalement, les études récentes (Thome *et al.*, 2002) montrent que la perte de

pression par frottement sur les parois est bien déterminée par la corrélation de Müller-Steinhagen (Müller-Steinhagen & Heck, 1986). Pour les calculs, un choix du modèle numérique s'impose. Celui-ci sera présenté de façon détaillée dans le chapitre 2.

CHAPITRE 2

MODÈLE NUMÉRIQUE ET VALIDATION POUR LA THERMOHYDRAULIQUE

Nous présenterons dans cette partie la discrétisation numérique choisie ainsi que les schémas de calculs pour les états stationnaires et transitoires de la thermohydraulique. Également, nous validerons dans cette partie le modèle thermohydraulique choisi en comparant certaines simulations à des résultats expérimentaux tirés de la thèse d'Aubé (1996).

1 - Modèle numérique pour la thermohydraulique

Selon Doster (1989), une différenciation finie de second ordre dans l'espace et de premier ordre implicite dans le temps est une façon inconditionnellement stable et précise pour les calculs d'écoulements diphasiques. Nous utiliserons donc cette différenciation pour nos calculs. La formulation de cette discrétisation dépend de la position dans le canal du réacteur et des conditions frontières. En effet, nous connaissons le flux massique G et la température T de l'écoulement à l'entrée du canal et la pression P à la sortie. Les schémas de discrétisation seront donc centrés ou non dépendamment des variables connues et des inconnues du problème. En considérant une discrétisation en N point du canal (voir figure 4), à la position j et au temps n , la formulation s'écrit comme suit :

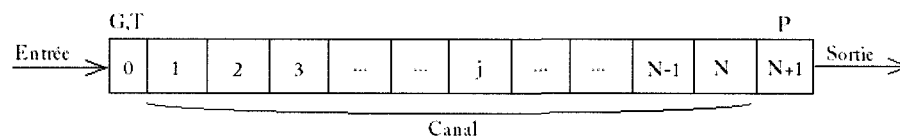


Figure 4 : Schéma du canal discrétisé.

- Équation de conservation de la masse :

$$\frac{G_{j+1}^{n+1} - G_j^{n+1}}{2\Delta z} = -\frac{(\rho_m)_j^{n+1} - (\rho_m)_j^n}{\Delta t}, \quad j=1, \dots, N-1, \quad (2.1)$$

à la frontière $j=N$, on réécrit cette équation comme suit :

$$\frac{G_j^{n+1} - 4G_{j-1}^{n+1} + 3G_{j-2}^{n+1}}{2\Delta z} = -\frac{(\rho_m)_j^{n+1} - (\rho_m)_j^n}{\Delta t}, \quad j=N. \quad (2.2)$$

- Équation de conservation de la quantité de mouvement :

$$\begin{aligned} & \frac{G_j^{n+1} - G_j^n}{\Delta t} \\ & + \frac{\left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g^{-2}}{\rho_m} v_{gj}^{-2} \right)_{j+1}^{n+1} - \left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g^{-2}}{\rho_m} v_{gj}^{-2} \right)_{j-1}^{n+1}}{2\Delta z}, \quad j=2, \dots, N-1, \quad (2.3) \\ & + \frac{p_{j+1}^{n+1} - p_{j-1}^{n+1}}{2\Delta z} + (\rho_m g \cos(\theta))_j^{n+1} + \left(\left(\frac{\partial p}{\partial z} \right)_{\text{frottement}} \right)_j^{n+1} = 0 \end{aligned}$$

à l'entrée, on écrit cette équation sous la forme suivante :

$$\begin{aligned} & \frac{G_j^{n+1} - G_j^n}{\Delta t} + \frac{-3p_{j+2}^{n+1} + 4p_{j+1}^{n+1} - p_j^{n+1}}{2\Delta z} + (\rho_m g \cos(\theta))_j^{n+1} \\ & + \frac{-3 \left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g^{-2}}{\rho_m} v_{gj}^{-2} \right)_{j+2}^{n+1} + 4 \left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g^{-2}}{\rho_m} v_{gj}^{-2} \right)_{j+1}^{n+1}}{2\Delta z}, \quad j=1, \quad (2.4) \\ & - \frac{\left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g^{-2}}{\rho_m} v_{gj}^{-2} \right)_j^{n+1}}{2\Delta z} + \left(\left(\frac{\partial p}{\partial z} \right)_{\text{frottement}} \right)_j^{n+1} = 0 \end{aligned}$$

à la sortie, l'équation de la conservation de la quantité de mouvement est exprimée par :

$$\begin{aligned}
 & \frac{G_j^{n+1} - G_j^n}{\Delta t} + \frac{p_{j+1}^{n+1} - p_{j-1}^{n+1}}{2\Delta z} + (\rho_m g \cos(\theta))_j^{n+1} \\
 & + \frac{\left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj}^{-2} \right)_j^{n+1} - 4 \left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj}^{-2} \right)_{j-1}^{n+1}}{2\Delta z} , \quad j = N . \quad (2.5) \\
 & + \frac{3 \left(\frac{G^2}{\rho_m} - \frac{\varepsilon}{1-\varepsilon} \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj}^{-2} \right)_{j-2}^{n+1}}{2\Delta z} + \left(\left(\frac{\partial p}{\partial z} \right)_{\text{frottement}} \right)_j^{n+1} = 0
 \end{aligned}$$

- Équation de conservation de l'énergie :

$$\begin{aligned}
 & \frac{\left(Gh_m + \varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_g + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_{j+1}^{n+1}}{2\Delta z} \\
 & - \frac{\left(Gh_m + \varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_g + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_{j-1}^{n+1}}{2\Delta z} , \quad j = 2, \dots, N-1 , \quad (2.6) \\
 & + \frac{\left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m gz \cos(\theta) \right)_j^{n+1}}{\Delta t} \\
 & - \frac{\left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m gz \cos(\theta) \right)_j^n}{\Delta t} = \left(\frac{A_w q''}{V} \right)_j^{n+1}
 \end{aligned}$$

en tenant en compte des conditions frontières à l'entrée, c'est-à-dire que la température est connue, alors cette équation peut s'écrire comme suit :

$$\begin{aligned}
& \frac{(Gh_m)_{j+1}^{n+1} - (Gh_m)_{j-1}^{n+1}}{2\Delta z} \\
& - 3 \frac{\left(\varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_{j+2}^{n+1}}{2\Delta z} \\
& + 4 \frac{\left(\varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_{j+1}^{n+1}}{2\Delta z} \\
& - \frac{\left(\varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_j^{n+1}}{2\Delta z} \\
& + \frac{\left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m gz \cos(\theta) \right)_j^{n+1}}{\Delta t} \\
& - \frac{\left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m gz \cos(\theta) \right)_j^n}{\Delta t} = \left(\frac{A_w q''}{V} \right)_j^{n+1}, \quad j=1, \quad (2.7)
\end{aligned}$$

et à la sortie du canal, elle s'exprime comme suit :

$$\begin{aligned}
& 3 \frac{\left(Gh_m + \varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_{j-2}^{n+1}}{2\Delta z} \\
& - 4 \frac{\left(Gh_m + \varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_{j-1}^{n+1}}{2\Delta z} \\
& + \frac{\left(Gh_m + \varepsilon \frac{\rho_l \rho_g}{\rho_m} \bar{v}_{gj} \Delta h_{lg} + \frac{G^3}{2\rho_E^2} + Ggz \cos(\theta) \right)_j^{n+1}}{2\Delta z} \\
& + \frac{\left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m gz \cos(\theta) \right)_j^{n+1}}{\Delta t}
\end{aligned}$$

$$-\frac{\left(\rho_m h_m + \frac{G^2}{\rho_M} - p + \rho_m g z \cos(\theta)\right)_j^n}{\Delta t} = \left(\frac{A_w q^n}{V}\right)_j^{n+1}, \quad j = N. \quad (2.8)$$

Les trois systèmes de N équations (équations 2.1 à 2.8) peuvent être résolus par une méthode matricielle utilisant la méthode de Gauss limitées aux bandes pleines de la matrice. Pour cela, nous pouvons formuler les systèmes sous la forme suivante :

$$[M_G][G] = [V_G], \quad (2.9)$$

pour la conservation de la masse,

$$[M_P][P] = [V_P], \quad (2.10)$$

pour la conservation de la quantité de mouvement et

$$[M_{Hm}][H_m] = [V_{Hm}], \quad (2.11)$$

pour la conservation de l'énergie.

Dans les équations 2.9, 2.10 et 2.11, les vecteurs des inconnues sont les suivants :

$$[G] = \begin{bmatrix} G_1^{n+1} \\ \vdots \\ G_j^{n+1} \\ \vdots \\ G_N^{n+1} \end{bmatrix}, \quad (1.80)$$

$$[P] = \begin{bmatrix} p_1^{n+1} \\ \vdots \\ p_j^{n+1} \\ \vdots \\ p_N^{n+1} \end{bmatrix}, \quad (1.81)$$

$$[H_m] = \begin{bmatrix} h_{m1}^{n+1} \\ \vdots \\ h_{mj}^{n+1} \\ \vdots \\ h_{mN}^{n+1} \end{bmatrix}. \quad (1.82)$$

Les systèmes d'équations que nous avons présentées ici ont été développés pour les calculs de transitoires. Pour les cas stationnaires, nous pouvons simplement annuler les dérivées temporelles et faire tout le calcul numérique au temps n . Nous ne présenterons pas la partie stationnaire ici car nous nous intéressons principalement aux calculs de transitoires. Par contre, les calculs stationnaires nous permettront de valider les calculs de transitoires.

Nous avons présenté ici la partie numérique du modèle de l'écoulement. Il est donc nécessaire de le valider dans la suite de notre exposé. Nous allons donc faire cette validation dans la partie suivante.

2 - Validation du modèle thermohydraulique

Dans cette partie, nous allons valider le modèle d'écoulement. Pour cela, nous allons comparer nos prédictions théoriques avec des expériences présentées dans la thèse d'Aubé (1996) que se soit pour des écoulements monophasique ou diphasique. Les conditions expérimentales ainsi que les données collectées sont présentées dans l'annexe 1 pour les expériences que nous avons choisies pour cette recherche.

Pour commencer, nous allons comparer les résultats obtenus par le code ARTHUR, développé ici, et les données expérimentales pour des écoulements monophasiques. Cette comparaison devrait nous donner un aperçu sur la fonctionnalité du code en écoulement monophasique, spécialement en ce qui a trait à la perte de pression par frottement. Les résultats pour les expériences *m19*, *m37* et *m38* (Aubé, 1996) sont présenté aux figures 5, 6 et 7 respectivement.

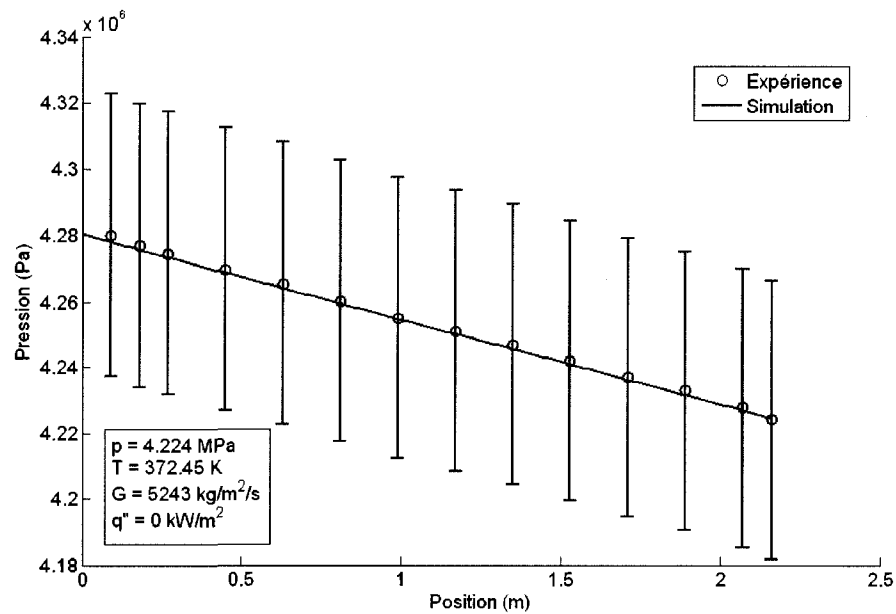


Figure 5 : Chute de pression dans l'expérience *m19* (Aubé, 1996).

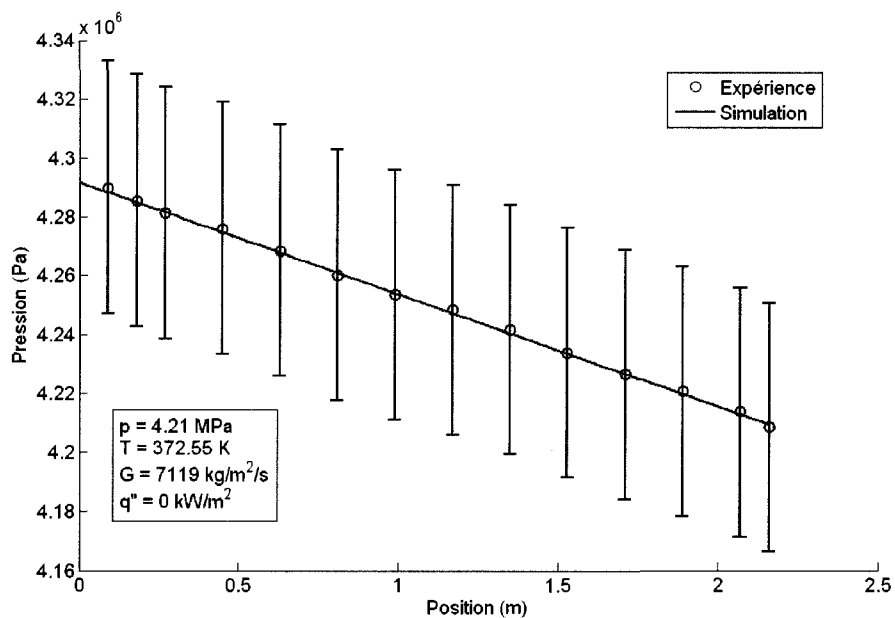


Figure 6 : Chute de pression dans l'expérience *m37* (Aubé, 1996).

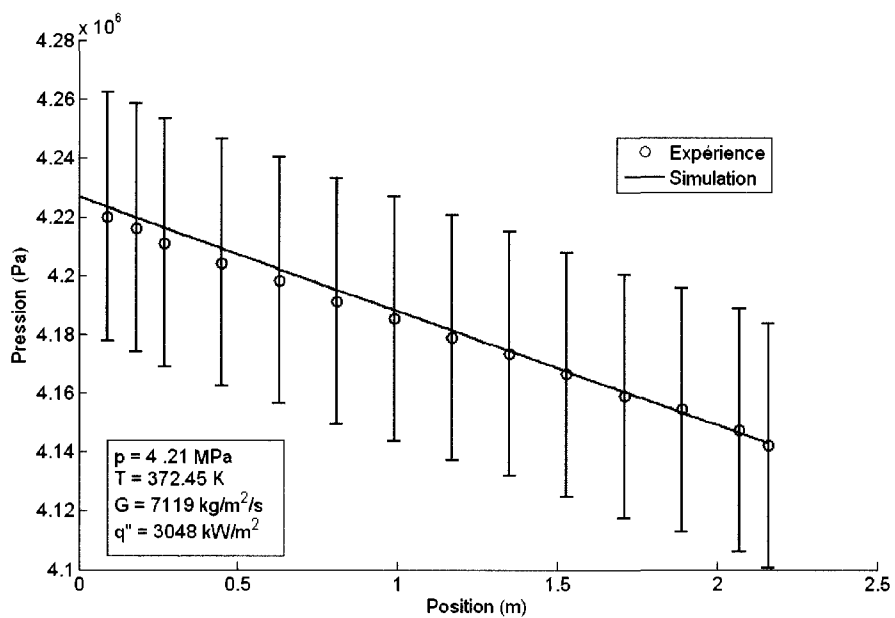


Figure 7 : Chute de pression dans l'expérience *m38* (Aubé, 1996).

Dans les trois cas présentés ci-dessus, les résultats du code ARTHUR ont été obtenus à l'état transitoire et validés pour l'état stationnaire. En effet, l'état stationnaire correspondant à chacune des conditions expérimentales a été simulé

comme un cas transitoire pour un temps suffisamment long, i.e. 2 minutes. Pendant cette procédure de calcul, nous avons vérifié que le taux de convergence numérique pour les variables de pression, de flux massique et d'enthalpie reste inférieur à 10^{-5} . Nous observons dans les figures 5, 6 et 7 que nous avons un décalage entre la valeur de la pression de sortie mesurée et celle calculée. Ceci est possiblement dû à l'arrondi des valeurs utilisées à l'entrée du code mais surtout au fait d'un décalage entre la discrétisation choisie pour la position de sortie et celle donnée par la mesure. Autrement dit, la mesure expérimentale serait l'équivalence du point de discrétisation $N+1$ alors que nous n'avons ici que le point N de discrétisation pour la simulation numérique. Il va donc de soi que cette différence prévue au début du calcul n'est pas une erreur de calcul dans le code. Pour les deux premières expériences, où nous n'avons pas de flux de chaleur dans le tube vers l'écoulement, la chute de pression est bien déterminée par les résultats de calculs. En effet, nous avons une erreur relative entre les pentes calculées et expérimentales d'environ 1%. Pour le cas avec flux de chaleur, soit la figure 7, l'erreur relative entre les valeurs calculées et celles mesurées augmente légèrement due à une surestimation des valeurs expérimentales. Cependant, les résultats prédits sont très proches des valeurs mesurées et à l'intérieur des erreurs de mesures expérimentales.

Regardons maintenant ce qu'il en est des expériences en écoulement diphasique. Cela nous permettra de voir si les corrélations utilisées pour la prédiction de la perte de pression en écoulements diphasique ainsi que la prédiction du taux de vide sont convenables. Pour cela, nous avons choisi de présenter les expériences 23, 31 et 44v de la thèse d'Aubé (Aubé, 1996). Les résultats obtenus pour les calculs de chute de pression sont donnés dans les figures 8, 9 et 10.

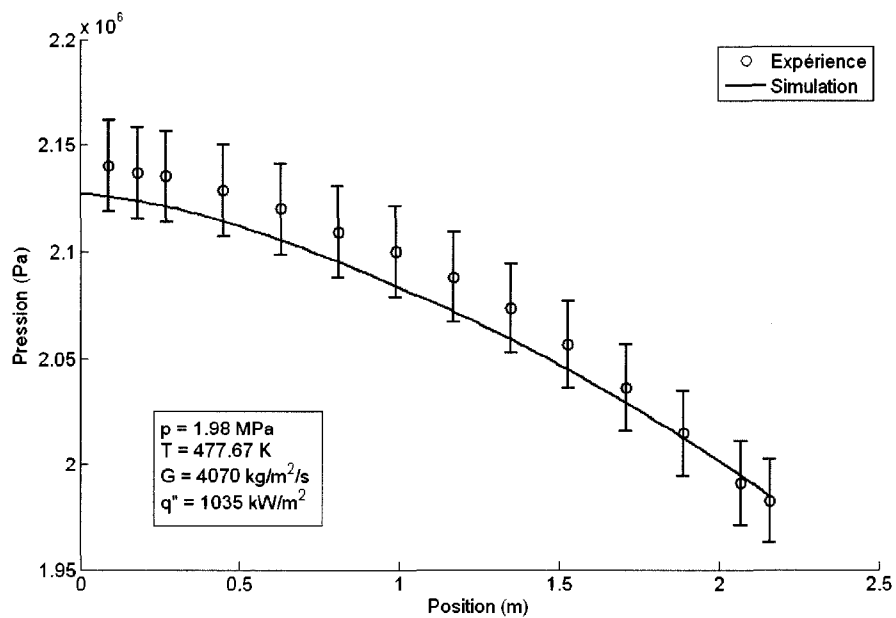


Figure 8 : Chute de pression dans l'expérience 23 (Aubé, 1996).

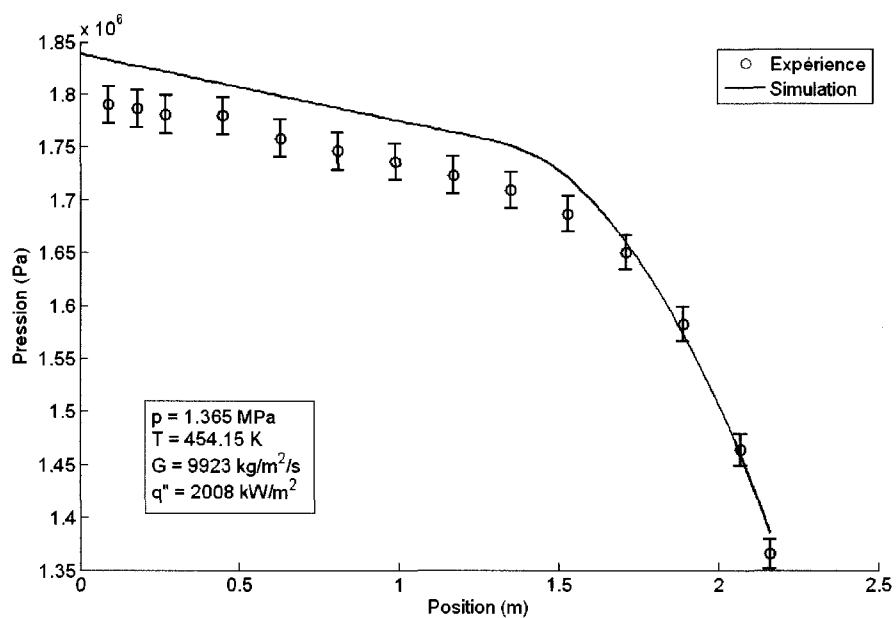


Figure 9 : Chute de pression dans l'expérience 31 (Aubé, 1996).

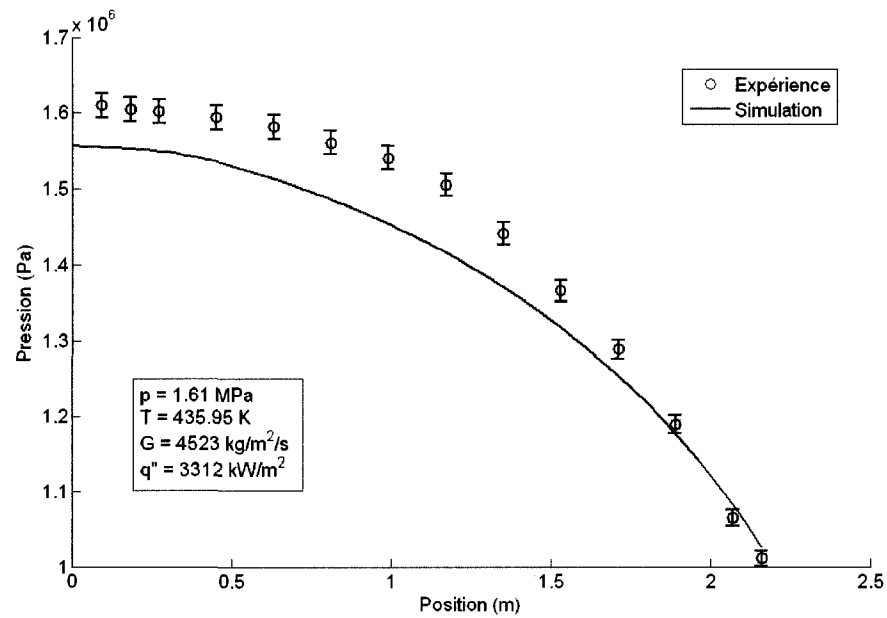


Figure 10 : Chute de pression dans l'expérience 44v (Aubé, 1996).

Nous avons aussi les résultats pour les calculs des taux de vide présentés aux figures 11, 12 et 13.

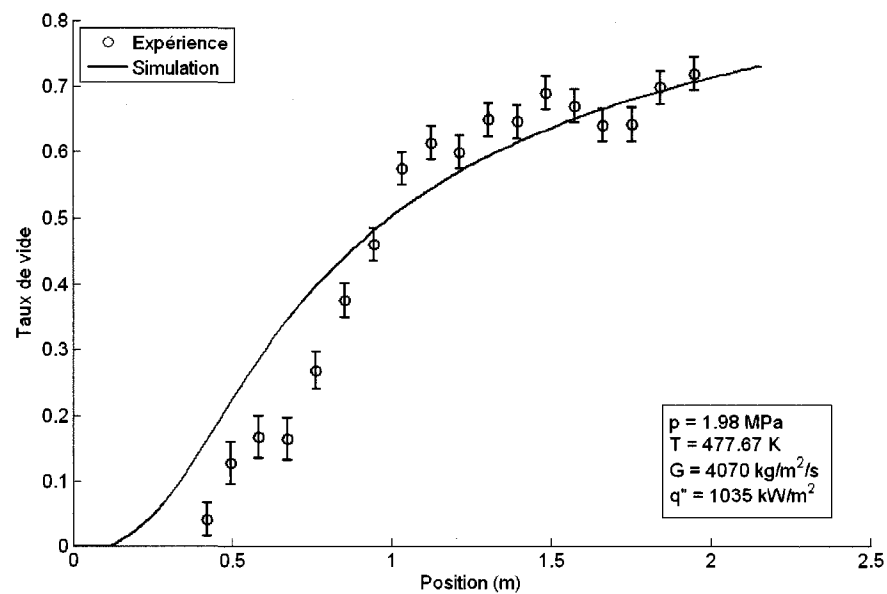


Figure 11 : Taux de vide dans l'expérience 23 (Aubé, 1996).

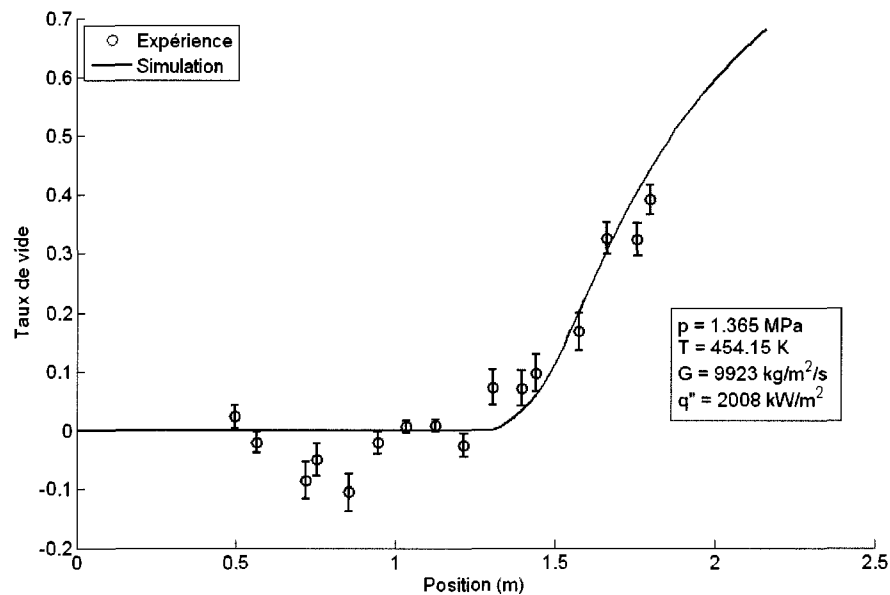


Figure 12 : Taux de vide dans l'expérience 31 (Aubé, 1996).

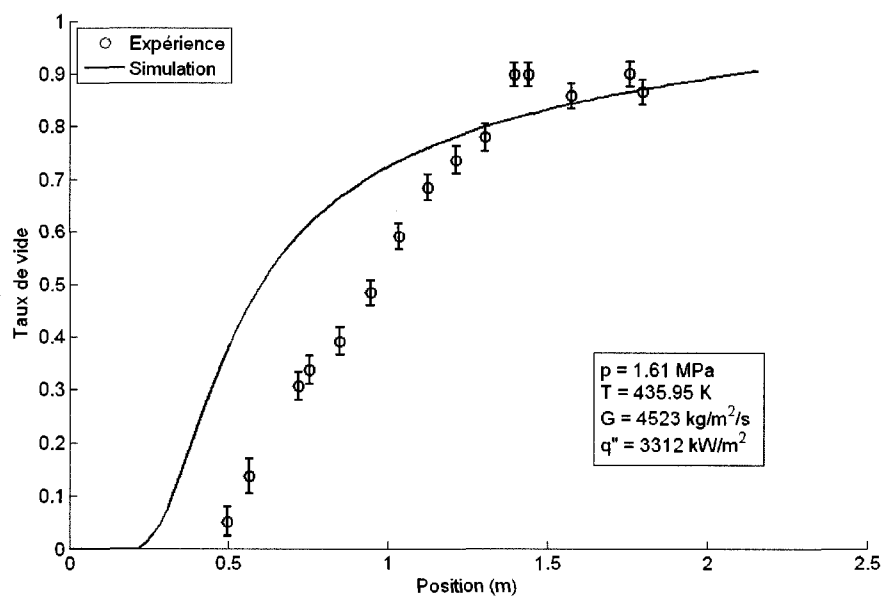


Figure 13 : Taux de vide dans l'expérience 44v (Aubé, 1996).

Finalement, nous obtenons les titres de l'écoulement et thermodynamique pour nos simulations que nous présentons aux figures 14, 15 et 16. Ces figures nous permettront d'analyser nos résultats.

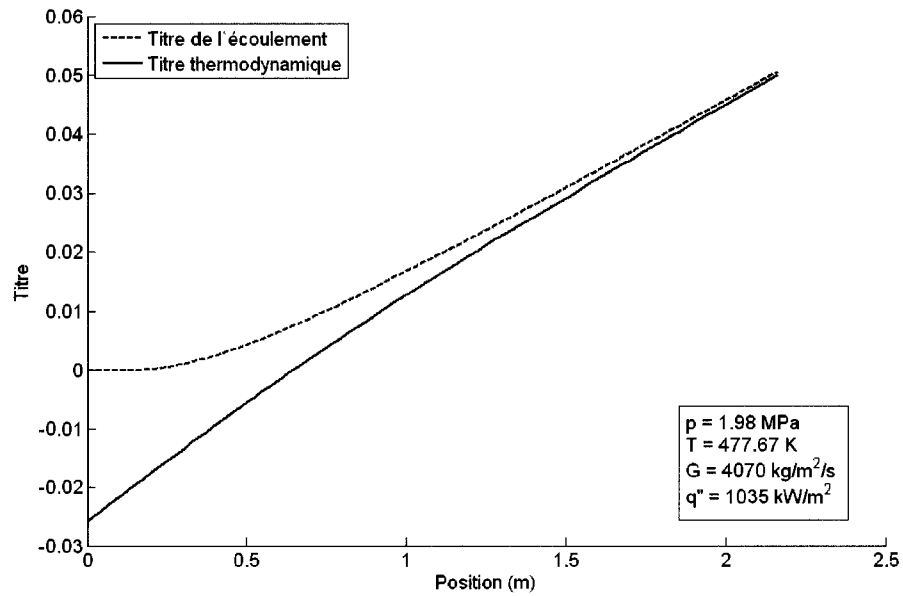


Figure 14 : Titres dans l'expérience 23 (Aubé, 1996).

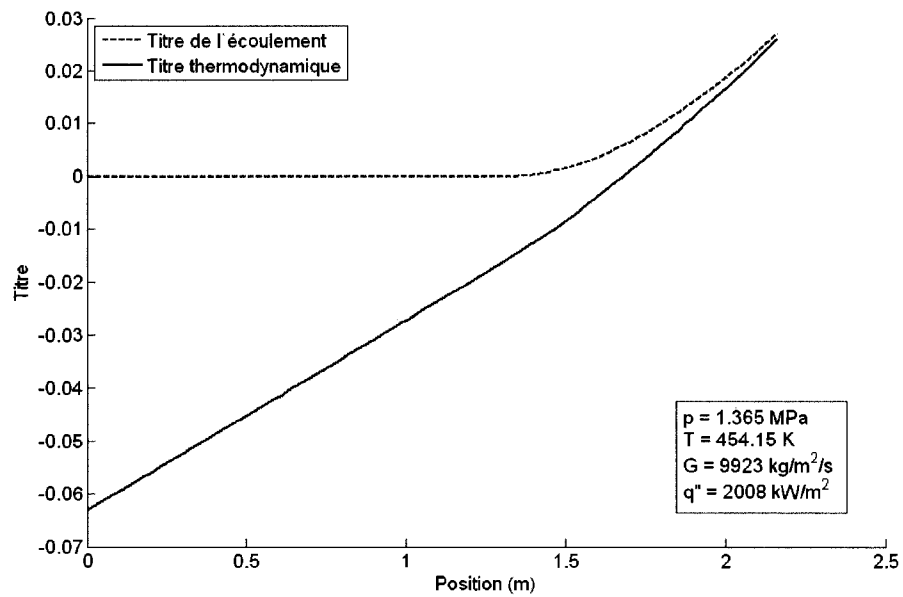


Figure 15 : Titres dans l'expérience 31 (Aubé, 1996).

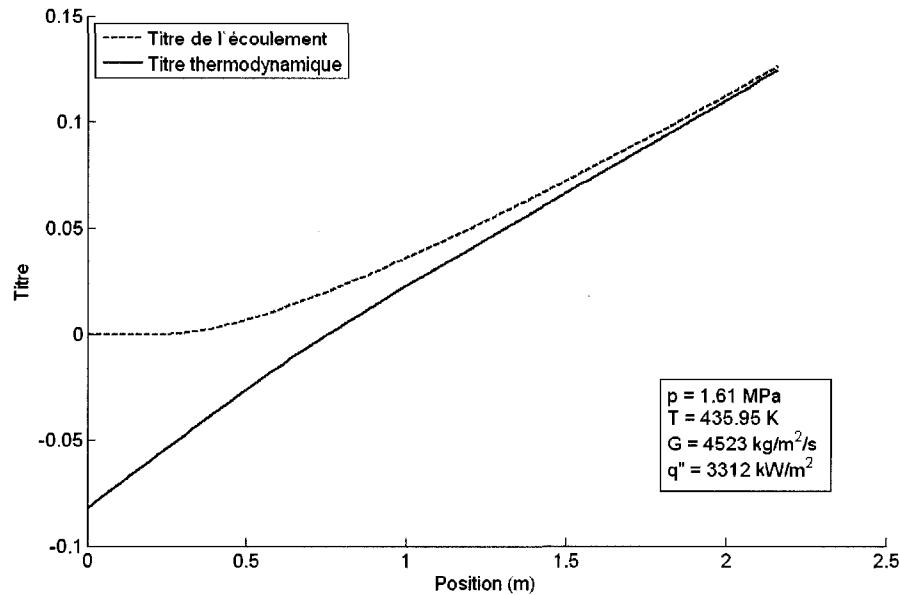


Figure 16 : Titres dans l'expérience 44v (Aubé, 1996).

Le décalage entre les pressions expérimentales à la sortie du tube et celle calculés est toujours présent comme prévu et nous en avons donné la raison plus haut, i.e. les points de discrétisation utilisés. Il est à noter que les pertes de pression en écoulement diphasique n'ont pas de profils semblables à ceux mesurés (voir figures 8, 9 et 10). Cependant, les prévisions théoriques restent sous le 10% d'erreur relative sur la pression à l'entrée. Les résultats sont donc convenables. Aussi, il faut prendre en considération que le début de l'ébullition sous-refroidie (voir figures 14, 15 et 16) a un effet très prononcé sur la perte de pression. Étant donné que la prédiction du début d'ébullition sous-refroidie n'est pas très précise dû aux corrélations utilisées, nous avons un impact notable sur la perte de pression. Il faut dire que nous avons néanmoins une prédiction du taux de vide correcte pour des situations d'écoulements diphasique (voir figures 11, 12 et 13). Il faut savoir, par ailleurs, que la précision des données de taux de vide est très peu fiable comme le montre les résultats négatifs mesurés (voir figure 12). Aussi, lorsque la chute de pression est bien prédite (figure 8), le taux de vide est mal prédit (figure 11) et vice-versa (figure 9 et 12). Cette remarque a été soulevée très souvent dans la littérature de thermohydraulique.

Il faut prendre en compte que les différences entre les résultats théoriques et ceux expérimentaux sont dues aussi bien au fait de l'incertitude liée aux corrélations de chute de pression par friction et à celles de la prédiction du taux de vide qu'à l'incertitude des mesures expérimentales (Teyssedou et *al.*, 1992). En effet, les incertitudes sur les pressions sont de 10% alors que pour le taux de vide elles varient de 100% pour de faibles taux de vide alors qu'elles sont de 4% pour des taux de vide élevés. Cependant, nous avons pu valider la partie thermohydraulique pour le code ARTHUR. En effet, les résultats sont convenables pour ce genre de calculs car proches des valeurs expérimentales. Par ailleurs, il est à noter que certaines limites sont apparues dans les calculs faits. Nous tenterons par la suite d'expliquer les limites des résultats obtenus pour le modèle que nous avons choisi.

3 - Limites du modèle

Le modèle suggéré ici présente des limites quant à sa précision et à son utilisation. En effet, nous pouvons soulever trois types de limites : limites des équations du modèle à écart de vitesse, limites des corrélations utilisées comme équations constitutives et limites des simplifications géométriques. Nous discuterons de chacune de ces limites dans les paragraphes suivants.

Pour ce qui est des limites soulevées concernant le modèle à écart de vitesse, nous avons remarqué une contradiction entre les équations des écoulements diphasiques et les corrélations utilisées. En effet, étant donné que les équations de conservation sont établies pour un système à l'équilibre, le titre de l'écoulement, ou le taux de vide, doit découler des équations comme indiqué dans le rapport de Tye *et al.* (1995) et non être imposé par une corrélation, ce qui révèle une incohérence. Ainsi, il existe une incompatibilité entre le fait d'avoir un écoulement sous refroidi et les équations de conservations, spécialement celle de la conservation de l'énergie. Prenons l'exemple de l'expérience 31 (Aubé, 1996). En traçant l'enthalpie du liquide calculée à partir de l'équation (1.13), où l'enthalpie du mélange est calculée par l'équation de conservation (1.23) et l'enthalpie de la vapeur est prise à saturation, nous observons les résultats présentés à la figure 17.

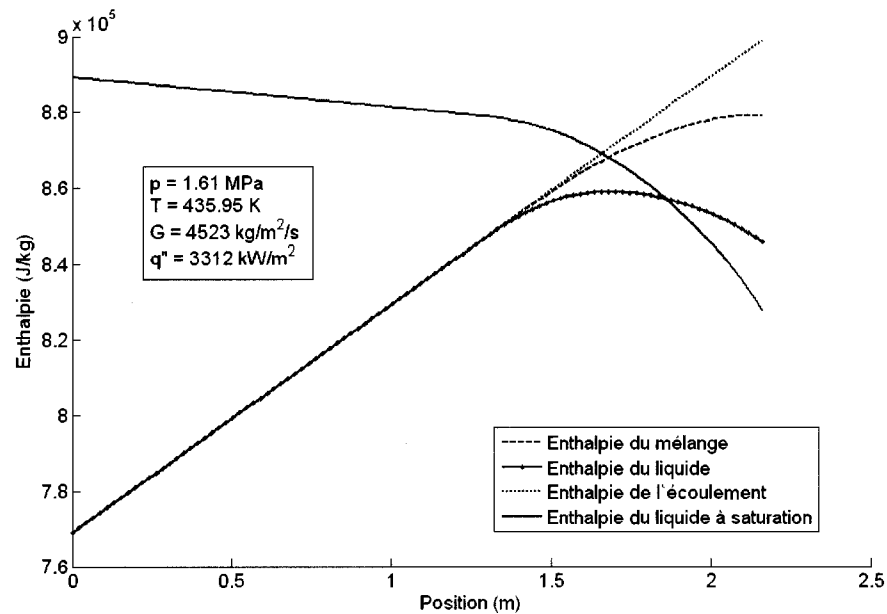


Figure 17 : Problème d'enthalpie du liquide dans l'expérience 44v (Aubé, 1996).

Cette figure montre les différentes enthalpies de l'écoulement calculées par le code ARTHUR. Nous pouvons y observer une erreur grossière dans l'enthalpie du liquide. En effet, il apparaît que l'enthalpie du liquide peut être supérieure à celle du liquide à saturation, ce qui n'a pas de sens physique. Pour ne pas tomber dans cette situation, nous fixerons dans le code les propriétés du liquide à la saturation une fois que nous nous retrouverons dans cette situation. Aussi, nous avons remarqué que certains articles de la littérature ont soulevé cette problématique liée à l'incohérence du modèle à écart de vitesse en une dimension. En effet, Collado (2000), montre l'incohérence de ce modèle et présente un autre modèle basé sur la vitesse relative appliquée à la conservation de la masse (Collado *et al.*, 2007b). Malheureusement, le modèle présenté par Collado est stationnaire et ne peut donc servir dans le cadre de ce travail.

Nous avons présenté, précédemment, les corrélations utilisées dans le cadre de ce travail avec les domaines dans lesquels elles ont été validées par les différents auteurs. Le tableau 3 présente un résumé des domaines dans lesquels les différentes

corrélations ont été validées. Les limites que nous avons, nous éclairent sur les résultats de validation du code. Les essais utilisés pour les comparaisons avec les simulations ont été faits en dehors du cadre où les corrélations ont été validées. De ce fait, il est très difficile de cerner la validité de nos tests.

Certaines considérations dans cette partie n'ont pas été clairement identifiées notamment celles qui concernent les limites voulues dans la conception du programme. Ces limites tiennent surtout à des hypothèses faites. En effet, il nous est impossible dans le code ARTHUR, à date, d'avoir des écoulements monophasique de vapeur surchauffée ou saturée. Aussi, comme nous l'avons spécifié plus haut, nous n'avons pas la possibilité dans ce code d'avoir une inversion de flux massique de l'écoulement comme c'est le cas dans les accidents pour les réacteurs CANDU. Également, nous avons utilisé les propriétés thermodynamiques et d'échange de l'eau légère alors que le caloporteur dans un CANDU est de l'eau lourde. Il serait important de modifier cette dernière situation pour accroître la précision du code.

Pour le couplage avec la neutronique, les paramètres qui seront transférés au code DONJON-3 (Varin *et al.*, 2005) sont : la densité et la température du caloporteur, mais aussi la température du combustible. La densité du caloporteur est celle du mélange moyennée sur la longueur d'une grappe et corrigée par un facteur 1.125 pour tenir compte du fait que nous avons de l'eau lourde dans les canaux des réacteurs CANDU. La température du caloporteur est celle du liquide (pour un écoulement monophasique ou diphasique) moyennée sur la longueur d'une grappe. Nous verrons dans la chapitre 3 comment est fait le calcul de la température du combustible ainsi que la moyenne utilisée dans le calcul de neutronique.

Dans cette partie, nous avons présenté les limites du modèle de thermohydraulique ainsi que les limites physiques à son utilisation. Il est certain qu'il serait important à l'avenir de prendre en considérations ces limitations et tenter de réduire leur impact sur le calcul, notamment en améliorant le modèle utilisé, en choisissant des corrélations plus appropriées et en réduisant les hypothèses simplificatrices que nous avons introduit.

CHAPITRE 3

MODÈLE DE TRANSFERT DE CHALEUR DANS LE CRAYON

La thermique des réacteurs est d'une importance particulière dans le domaine du nucléaire. En effet, ce domaine se retrouve à la jonction entre la thermohydraulique et la neutronique pour les réacteurs CANDU. Il représente donc un élément essentiel à la sécurité des réacteurs. Il est donc nécessaire de prévoir la température dans les crayons de combustible en fonction de l'écoulement du caloporteur mais aussi de la puissance dégagée dans ceux-ci. Ceci permettra d'évaluer les sections efficaces qui affectent la puissance dans les calculs de neutronique. Nous verrons donc dans cette partie les simplifications et hypothèses faites pour le calcul de la distribution de température dans le crayon, l'équation de transfert de chaleur et les corrélations utilisées comme relations de fermeture du système d'équations.

1 - Hypothèses et simplifications

La complexité géométrique et physique du système nous oblige à faire plusieurs hypothèses et simplifications afin de limiter la puissance de calcul requise. Nous présenterons par la suite les simplifications qui concernent la géométrie, les propriétés physiques des matériaux, ainsi que leurs implications dans les calculs.

En ce qui concerne la géométrie du crayon, nous allons supposer que le combustible d' UO_2 est un cylindre homogène de longueur égale à celle du canal. Aussi, le combustible est séparé de la gaine de zircaloy par un faible espacement (de l'ordre du micron). Les rayons et tailles du combustible, de la gaine et de l'espacement sont fixes (voir figure la 18). Le gonflement du combustible par la température est donc négligé. Ainsi, nous ne traiterons pas les variations géométriques des crayons. Aussi, les pressions à l'intérieur du crayon sont constantes. Ceci nous amènera à considérer

que le transfert de chaleur entre le combustible et la gaine comme étant fixe comme ce qui est fait dans les codes de calcul de l'Énergie Atomique du Canada Limitée (ÉACL). Nous fixerons la valeur de la conductivité thermique entre la surface externe de la pastille d' UO_2 et la surface interne de la gaine à $10 \text{ kW/m}^2/\text{K}$.

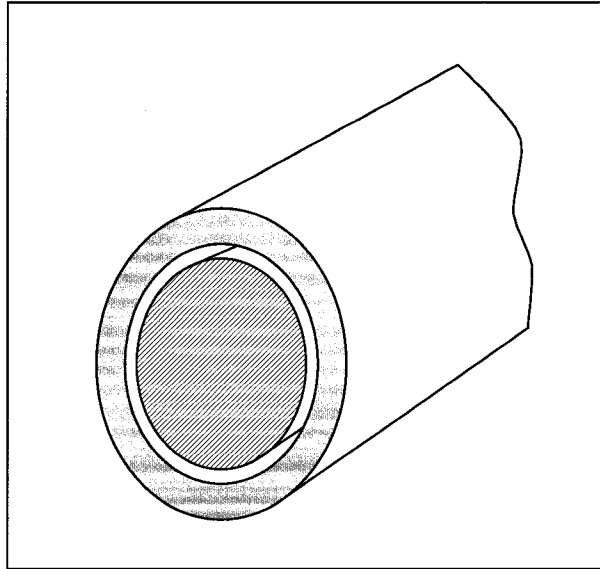


Figure 18 : Géométrie simplifiée du crayon central de la grappe.

En ce qui concerne les propriétés physiques des matériaux, nous allons supposer que celles-ci ne dépendent que de la température et non de la position radiale ou axiale dans le crayon. De plus, nous allons négliger le craquement et toute autre modification de l'état du combustible ou de la gaine. Cette hypothèse simplifiera grandement les calculs de la distribution radiale de température et l'utilisation de calculs mécaniques ou corrélations pour évaluer les phénomènes de modification de l'état des matériaux.

À ces deux suppositions, nous allons ajouter deux simplifications physiques qui nous permettront, elles aussi, de réduire la complexité du calcul. La première consiste à supposer que la conductivité thermique de l' UO_2 est suffisamment faible de façon à ce que nous puissions négliger le transfert de chaleur longitudinal et ne considérer que celui radial. Aussi, nous considérerons une symétrie angulaire du crayon, ce qui permettra un calcul unidimensionnel de l'équation de transfert de chaleur. Les

simplifications étant faites concernant la géométrie, les propriétés physiques des matériaux et du système, il nous faut présenter l'équation de transfert de l'énergie.

2 - Équation de transfert de chaleur

Nous présenterons dans cette partie l'équation de transfert de chaleur. Celle-ci sera déduite de l'équation de conservation généralisée ainsi que de l'utilisation de la loi de Fourier. En reprenant l'équation générale de conservation (1.2) présenté dans la chapitre 1 section 2, il nous est possible de trouver l'équation de transfert de chaleur en y remplaçant la propriété physique à évaluer, la source et le terme flux par ceux donnés dans le tableau 4.

Tableau 4 : Propriétés physiques utilisées dans l'équation de conservation de l'énergie thermique.

	Propriété	Source	Flux
Équation de transfert de chaleur	ρu	q'''	q''

Aussi, sachant que nous avons un solide indéformable, la vitesse de diffusion est nulle dans l'équation de conservation générale. L'équation que nous obtenons donc est la suivante :

$$\frac{\partial \rho u}{\partial t} + \vec{\nabla} \cdot \vec{q}'' - q''' = 0. \quad (3.1)$$

Tenant compte que l'équation de Fourier est donnée par :

$$\vec{q}'' = -k \cdot \vec{\nabla} T. \quad (3.2)$$

Celle-ci permet d'écrire l'équation de transfert de chaleur sous la forme suivante :

$$\frac{\partial \rho u}{\partial t} - \vec{\nabla} \cdot (k \cdot \vec{\nabla} T) - q''' = 0. \quad (3.3)$$

Dans ce travail, nous considérons constants la pression et le volume spécifique des solides. Ainsi, l'énergie interne pour un solide est donnée par $u = C_p T$, alors nous pouvons utiliser cette définition, l'indépendance de la conductivité thermique par rapport à la position et le fait que nous utilisons l'équation de transfert de chaleur en une dimension pour réécrire l'équation de transfert de chaleur de la façon suivante :

$$\frac{\partial(\rho C_p T)}{\partial t} = \frac{k}{r} \frac{\partial}{\partial r} \left(r \frac{\partial T}{\partial r} \right) + q''' . \quad (3.4)$$

Nous avons présenté dans cette partie l'équation de transfert de chaleur tel que nous allons l'utiliser dans la suite de notre étude. Nous allons voir dans la section suivante les équations constitutives dont nous avons besoin pour clore le système d'équations.

3 - Conditions aux limites

Afin de compléter l'équation de conservation, nous utiliserons des conditions aux limites. Les conditions aux limites se trouvent à la surface de la gaine, à l'interstice, des côtés de la gaine et du combustible, ainsi qu'au centre du crayon (voir figure 19).

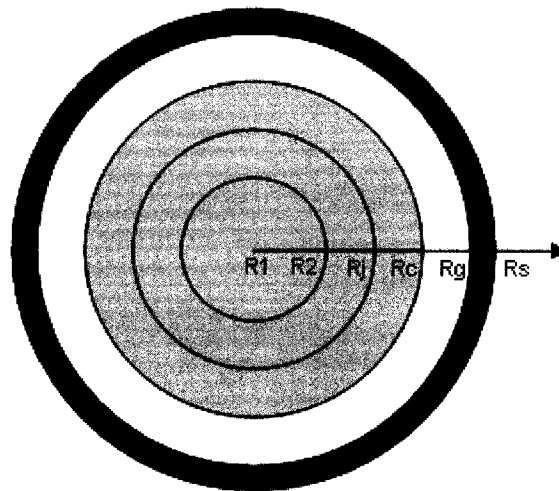


Figure 19 : Schéma du crayon.

Pour limiter les calculs sur la moitié du crayon, on utilise la symétrie du crayon. Ainsi, le flux de chaleur au centre du combustible est nul. Une première condition aux limites s'écrit donc:

$$\left. \frac{\partial T}{\partial r} \right|_{r=0} = 0. \quad (3.5)$$

Comme l'épaisseur de l'interstice est de l'ordre du micron, alors on considère que le flux de chaleur entre le combustible et la gaine est le même. Il est possible d'écrire cette condition sous la forme suivante :

$$-k_c \left. \frac{\partial T}{\partial r} \right|_{r=r_c} = h_{gap} (T_c - T_g), \quad (3.6)$$

$$r_g k_g \left. \frac{\partial T}{\partial r} \right|_{r=r_g} = r_c h_{gap} (T_c - T_g). \quad (3.7)$$

Étant donné que l'interstice est très mince, le rayon du combustible et le rayon interne de la gaine sont quasiment égaux. Pour éviter les problèmes numérique par la suite, nous allons simplifier les deux équations précédentes en une approximative donnant le transfert de chaleur entre la gaine et le combustible :

$$k_g \left. \frac{\partial T}{\partial r} \right|_{r=r_g} = h_{gap} (T_c - T_g). \quad (3.8)$$

Nous avons également à la surface de la gaine deux conditions, celle liée au flux de chaleur imposé étant donné que nous connaissons la puissance dégagée dans le combustible par la fission, ainsi que la condition liée à la température à la surface de la gaine. Ces conditions s'écrivent de la façon suivante :

$$q'' = -k_g \left. \frac{\partial T}{\partial r} \right|_{r=r_s} = q''' \frac{\pi r_c^2}{2\pi r_s}, \quad (3.9)$$

$$T_s = \frac{q''}{h_t} + T_\infty. \quad (3.10)$$

L'équation 3.9 est valide pour des régimes permanents, cependant, étant donné que les transitoires que nous allons simuler correspondent à des variations faibles, l'équation 3.9 est une approximation valide dans ces conditions. Par contre, dans le cas de perte de caloporteur ou de réduction de débit majeure, cette condition frontière n'est plus valide et il faudrait la modifier. Les conditions frontières étant définies, il nous reste à présenter les équations constitutives, ou corrélations, qui nous permettront de clore le système d'équations.

4 - Équations constitutives

L'équation de transfert de chaleur à elle seule n'est pas suffisante pour le calcul de la distribution de la température dans la gaine et le combustible. En effet, nous avons besoin de corrélations permettant de clore le système d'équations.

Pour le transfert de chaleur entre la gaine et l'écoulement, nous avons besoin d'utiliser deux corrélations. La première sert à prédire le coefficient de transfert de chaleur entre la gaine et l'écoulement monophasique et la seconde pour celui entre la gaine et l'écoulement diphasique, sous-refroidie ou saturé. Les corrélations sont les suivantes :

4.1 - Corrélation de Markoczy :

La corrélation de Markoczy (1972) est une reprise de la corrélation de Dittus-Boelter (1930) adaptée à différentes géométries, notamment celle hexagonale. Celle-ci est donnée par les formules suivantes :

$$Nu = \frac{h_t D_h}{k} = 0.023 \phi Re^{0.8} Pr^{0.4}, \quad (3.11)$$

$$\phi = 1 + 0.912 Re^{-0.1} Pr^{0.4} (1 - 2.0043 e^{-B}), \quad (3.12)$$

$$B = \frac{D_h}{d}, \quad (3.13)$$

$$Pr = \frac{\mu C_p}{k}. \quad (3.14)$$

Nous pouvons donc effectuer le calcul de la température à la surface de la gaine connaissant le flux de chaleur, issu de la puissance thermique dégagée dans le combustible, et la température du caloporteur à l'état de liquide comprimé.

4.2 - Corrélation de Chen (1966) :

La corrélation de Chen (1966) considère deux mécanismes de transfert de chaleur, l'un microconvective, donné par l'équation de Forster-Zuber (1955), et l'autre macroconvective, donné par l'équation de Dittus-Boelter (1930). Elle est donnée par les formules suivantes :

$$h_{mac} = 0.023 Re^{0.8} Pr^{0.4} \frac{k}{D_h} F, \quad (3.15)$$

$$x_m = \left(\frac{x}{1-x} \right)^{0.9} \left(\frac{\rho_l}{\rho_g} \right)^{0.5} \left(\frac{\mu_g}{\mu_l} \right)^{0.1}, \quad (3.16)$$

$$F = 2.35 \times (0.231 + x_m)^{0.8}, \quad (3.17)$$

$$h_{mic} = 0.00122 \left(\frac{k_l^{0.79} C_{pl}^{0.45} \rho_l^{0.49}}{\sigma^{0.5} \mu_l^{0.29} h_{lg}^{0.24} \rho_g^{0.24}} \right) (\Delta T)^{0.24} (\Delta P)^{0.75} S, \quad (3.18)$$

$$S = \frac{1}{1.0 + 2.53 \times 10^{-6} \text{Re}^{1.17}}, \quad (3.19)$$

$$\text{Re} = \text{Re}_l \cdot F^{1.25}. \quad (3.20)$$

La somme des coefficients de transfert de chaleur microscopique et macroscopique nous donnent le coefficient de transfert de chaleur total pour l'ébullition sous-refroidie et saturée. Afin d'avoir une continuité du transfert de chaleur macroscopique durant la transition monophasique-diphasique, nous allons inclure un coefficient de correction géométrique (voir l'équation 3.12). Ainsi, il est possible d'écrire la corrélation sous la forme suivante :

$$h_t = \phi(h_{mic} + h_{mac}) \quad (3.21)$$

La corrélation de Chen nous permettra donc de calculer de façon itérative la température à la surface de la gaine connaissant le flux de chaleur issu de la puissance thermique dégagée dans le combustible, et la température du caloporteur à l'état d'ébullition sous-refroidie et saturée.

Pour ce qui est du coefficient de transfert de chaleur entre la gaine et le combustible, nous avons décidé de le fixer à $10 \text{ kW/m}^2/\text{K}$ selon les informations fournies par l'Énergie Atomique du Canada Limitée (ÉACL). En effet, le manque de données ne nous a pas permis l'utilisation de corrélations plus précises, notamment celle de Ross-Stoute (1962) ou bien celle de Campbell *et al.* (1977).

En ce qui concerne la simplification de la géométrie de la grappe, il faut noter que celle-ci n'a pas été testé jusqu'à date. Il est donc nécessaire de relativiser les résultats qui seront obtenus par la suite avec cette simplification. En effet, il faut savoir que la

chaleur dégagée par les grappes dans le caloporteur se fait de façon tridimensionnelle ce qui implique des gradients de température dans le caloporteur pour une section transversale d'un canal. Ces gradients entraînent certainement des migrations du caloporteur dépendamment de la température de celui-ci de façon transversale au canal. De plus, le flux massique dans les sections du canal n'est pas homogène ce qui amplifie les migrations. Ces migrations impliquent forcément une baisse de la température au centre de la section transversale du canal, chose que nous ne calculons pas ici.

Dans cette partie, nous avons présenté les équations de transfert de chaleur qui serviront au calcul de la température dans les crayons de réacteurs CANDU. Ces équations seront présentées, dans le chapitre 4, sous la forme qui sera utilisée pour les calculs numériques. Également, nous validerons ces équations en les comparant à une solution analytique pour un problème connu.

CHAPITRE 4

MODÈLE NUMÉRIQUE ET VALIDATION DU TRANSFERT DE CHALEUR

Ce chapitre a pour objectif de présenter les équations de transfert de chaleur sous leur forme de différences finies. Cette forme sera utilisée dans les simulations que nous allons faire avec le code ARTHUR. En effet, nous verrons dans une première partie le modèle numérique de façon détaillé que nous validerons dans un second temps pour un cas simple ayant une solution analytique. Cette validation sera donc faite en comparant la solution numérique du code ARTHUR à une solution analytique.

1 - Modèle numérique de transfert de chaleur

L'équation de conservation et les équations constitutives vont nous permettre de présenter un modèle numérique. Nous allons donc présenter par la suite les équations de conservation ainsi que les conditions aux limites sous la forme discrétisées et matricielle.

Comme nous l'avons fait pour la partie écoulements, nous avons choisi une discrétisation du second ordre dans l'espace et du premier ordre, implicite, dans le temps. Cette discrétisation nous permet, là encore, d'avoir une stabilité du schéma numérique. En supposant que nous avons M_c points de discrétisation radiale dans la pastille d' UO_2 et M_g points de discrétisation de la gaine, les équations que nous obtenons sont les suivantes :

$$-3T_j^{n+1} + 4T_{j+1}^{n+1} - T_{j+2}^{n+1} = 0, \quad j=1, \quad (4.1)$$

$$\begin{aligned}
& -[k_c]_j^{n+1} \left(1 - \frac{\Delta r_c}{2r_j} \right) T_{j-1}^{n+1} + [k_c]_j^{n+1} \left(2 + \left[\frac{C_{pc} \rho_c}{k_c} \right]_j^{n+1} \frac{\Delta r_c^2}{\Delta t} \right) T_j^{n+1} \\
& - [k_c]_j^{n+1} \left(1 + \frac{\Delta r_c}{2r_j} \right) T_{j+1}^{n+1} = [C_{pc} \rho_c]_j^n \frac{\Delta r_c^2}{\Delta t} T_j^n + [q''']^{n+1} \Delta r_c^2
\end{aligned}
, j = 2, M_c - 1, \quad (4.2)$$

$$\begin{aligned}
& 2\Delta r_g \left[\frac{h_{gap}}{k_g} \right]_j^{n+1} T_{j-1}^{n+1} - \left(3 + 2\Delta r_g \left[\frac{h_{gap}}{k_g} \right]_j^{n+1} \right) T_j^{n+1} \\
& + 4T_{j-1}^{n+1} - T_{j-2}^{n+1} = 0
\end{aligned}
, j = M_c + 1, \quad (4.3)$$

$$\begin{aligned}
& -[k_g]_j^{n+1} \left(1 - \frac{\Delta r_g}{2r_j} \right) T_{j-1}^{n+1} + [k_g]_j^{n+1} \left(2 + \left[\frac{C_{pg} \rho_g}{k_g} \right]_j^{n+1} \frac{\Delta r_g^2}{\Delta t} \right) T_j^{n+1} \\
& - [k_g]_j^{n+1} \left(1 + \frac{\Delta r_g}{2r_j} \right) T_{j+1}^{n+1} = [C_{pg} \rho_g]_j^n \frac{\Delta r_g^2}{\Delta t} T_j^n
\end{aligned}
, j = M_c + 1, M_g - 1, \quad (4.4)$$

$$T_{j-2}^{n+1} - 4T_{j-1}^{n+1} + \left(3 + 2\Delta r_g \left[\frac{h_t}{k_g} \right]_j^{n+1} \right) T_j^{n+1} = 2\Delta r_g \left[\frac{h_t}{k_g} \right]_j^{n+1} T_\infty^{n+1}, \quad j = M_g. \quad (4.5)$$

Avec ces équations, il nous est possible de présenter une formulation matricielle plus compacte et plus utile pour une programmation informatique. Les équations de transfert de chaleur peuvent donc s'écrire de la façon matricielle suivante :

$$[M_T][T] = [V_T]. \quad (4.6)$$

où $[M_T]$ est la matrices des coefficients et $[V_T]$ le vecteurs de constantes.

Jusqu'ici, les équations ont été présentées sous leur forme dépendante du temps, mais il serait possible d'étudier le cas stationnaire en annulant tout simplement la partie dépendante du temps. Comme pour la partie écoulement de notre présentation, nous

allons nous abstenir de présenter les équations stationnaires pour la partie transfert de chaleur.

Dans la suite de notre présentation, nous devons valider les équations que nous avons présentées ici afin de s'assurer de la fonctionnalité du modèle.

2 - Validation du modèle

Dans cette partie, nous validerons nos équations discrétisées en comparant les résultats issus de celles-ci à des solutions analytiques. Nous ferons ces comparaisons pour des calculs stationnaires et transitoires mais nous ne présenterons ici que la solution stationnaire car la solution transitoire donne exactement les mêmes valeurs que la solution transitoire pour un temps suffisamment long, i.e. 10 *min*.

Afin de résoudre analytiquement l'équation de transfert de chaleur, nous allons supposer un cas stationnaire dans lequel la conductivité thermique de la gaine et du combustible est indépendante de la température. La résolution détaillée et les données de ce problème se trouvent dans l'annexe 2 de ce mémoire, nous ne ferons que présenter les résultats des équations analytiques et celles numériques. Les résultats que nous obtenons sont présentés dans la figure 20.

Nous pouvons observer sur cette figure que les valeurs obtenues numériquement suivent bien la tendance des valeurs théoriques. Cependant, nous pouvons remarquer qu'au centre et à la surface externe du combustible, les valeurs numériques s'éloignent quelque peu des valeurs théoriques mais la différence reste en deçà des 2%. Cette dernière s'explique principalement par l'erreur numérique commise pour le nombre de point de discrétisation. Il est donc possible de considérer que le modèle numérique que nous avons ici est fonctionnel.

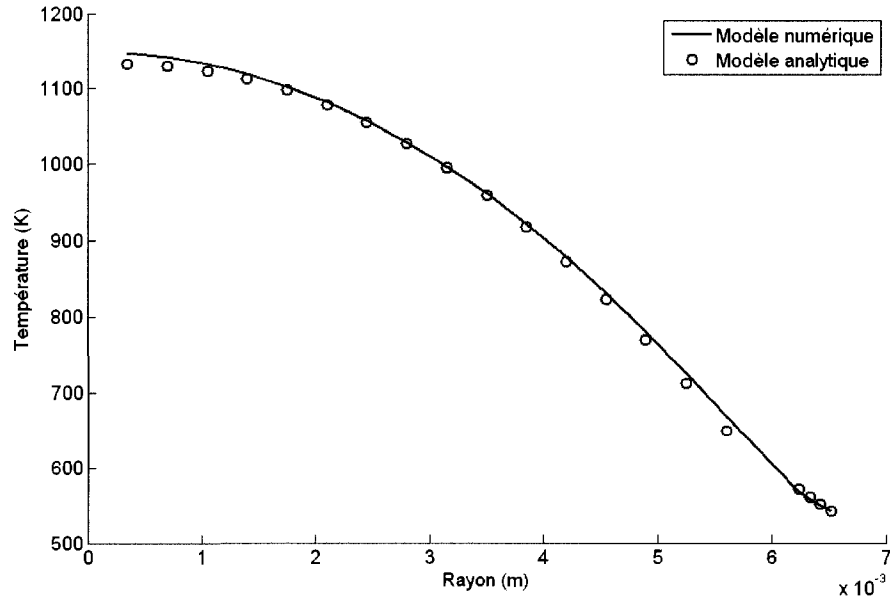


Figure 20 : Distribution de température dans un crayon.

Le paramètre qui sera transféré lors du couplage au code de neutronique DONJON-3 (Varin *et al.*, 2005) est la température moyenne du combustible de la grappe. Celle-ci est basée sur la moyenne axiale de la température sur la longueur d'une grappe mais aussi sur la moyenne radiale calculée grâce à l'équation 4.7.

$$\bar{T} = \sum_{i=1}^{Mc-1} \frac{T_{i+1} + T_i}{2} \times \frac{R_{i+1}^2 - R_i^2}{R_c^2}. \quad (4.7)$$

Cette partie nous a donc permis la validation du modèle numérique pour le transfert thermique par conduction. Bien évidemment, afin d'améliorer le modèle que nous avons présenté ici, il serait important d'inclure dans un développement futur toute la partie mécanique liée au crayon afin de prendre en considération le craquement du combustible, la dilatation thermique et un meilleur modèle de transfert de chaleur entre la combustible et la gaine. Également, il serait nécessaire d'y inclure la notion de flux de chaleur critique que se produit lors d'accident dans le réacteur. Cette partie a été un tremplin entre la section thermohydraulique que nous avons développé ici et la partie couplage neutronique que nous présenterons dans la section suivante.

CHAPITRE 5

COUPLAGE NEUTRONIQUE/THERMOHYDRAULIQUE

Le couplage entre la neutronique et la thermohydraulique est un élément essentiel permettant l'obtention de résultats de calculs représentant de façon réaliste les phénomènes physiques dans les réacteurs. Ainsi, nous serons amenés dans cette section à présenter une brève introduction de la neutronique qui nous permettra de mettre en lumière l'importance des calculs de thermohydraulique en neutronique. Nous verrons également, les algorithmes de calculs en thermohydraulique et en couplage neutronique/thermohydraulique dont nous nous servirons pour les calculs de transitoires des réacteurs CANDU.

1 - Aperçu de la neutronique

L'introduction à la neutronique se fera à travers la présentation des équations de transport et de diffusion. L'équation de transport permet de prédire le comportement des neutrons dans les réacteurs. Étant donné la complexité de cette dernière, elle n'est utilisée que pour les calculs de cellules (i.e. une petite région représentative du réacteur) ou pour de petits réacteurs. Afin de pouvoir faire un calcul de cœur complet pour un réacteur CANDU, certaines simplifications sont faites à l'équation de transport. Celles-ci nous permettent d'obtenir l'équation de diffusion. Nous commencerons donc par présenter quelques notions fondamentales en neutronique avant de présenter ces équations.

Il est possible de décrire une distribution de particules dans l'espace avec une approche de physique statistique. Celle-ci préconise l'utilisation d'un espace des phases à six dimensions auxquelles nous pouvons ajouter une septième liée au temps. Les coordonnées dans l'espace des phases sont :

- Vecteur position dans l'espace $\vec{r}$ (3 coordonnées).
- Vitesse V_n et la direction de déplacement $\vec{\Omega} = \frac{\vec{V}_n}{\|\vec{V}_n\|}$ (3 coordonnées).
- Temps t .

La distribution de particules peut donc s'écrire comme étant : $n(\vec{r}, V_n, \vec{\Omega}, t)$ et le nombre de particules : $n(\vec{r}, V_n, \vec{\Omega}, t) d^3r dV_n d^2\Omega$. Cependant en physique des réacteurs, nous utilisons le flux neutronique défini comme suit :

$$\phi(\vec{r}, V_n, \vec{\Omega}, t) = V_n n(\vec{r}, V_n, \vec{\Omega}, t) . \quad (5.1)$$

Il est possible de trouver l'équation de transport en faisant un bilan de la variation du nombre de particules dans le temps et dans l'hypervolume $d^3r dV_n d^2\Omega$ autour de l'état $\{\vec{r}, V_n, \vec{\Omega}\}$. Pour cela, nous allons considérer que la variation du nombre de neutrons dans le temps est égal au nombre de neutron créés auquel nous soustrayons le nombre de ceux perdus par collision et ceux quittant l'hypervolume. Ce bilan peut s'écrire de façon mathématique comme suit :

$$\frac{1}{V_n} \frac{\partial \phi(\vec{r}, V_n, \vec{\Omega}, t)}{\partial t} + \Sigma_{(\vec{r}, V_n)} \phi(\vec{r}, V_n, \vec{\Omega}, t) + \vec{\Omega} \cdot \vec{\nabla} \phi(\vec{r}, V_n, \vec{\Omega}, t) = Q_{(\vec{r}, V_n, \vec{\Omega}, t)} . \quad (5.2)$$

La section efficace macroscopique est définie comme suit :

$$\Sigma_{(\vec{r}, V_n)} = \sum_x \sum_i \sigma_{i,x(T)} N_{i(T, \vec{r})} . \quad (5.3)$$

où $\sigma_{i,x}$ et N_i sont respectivement la section efficace microscopique et nombre de noyaux par unité de volume pour le type i de noyaux et x d'interaction (diffusion, absorption...). Nous voyons apparaître dans le calcul de la section efficace macroscopique l'intérêt de la thermohydraulique. En effet, la densité du caloporteur permet de connaître le nombre de noyaux par unité de volume du deutérium et de

l'oxygène tandis que la température du caloporteur et du combustible permet d'évaluer la section efficace microscopique du deutérium, de l'oxygène et de l'uranium. Dans les calculs de neutronique, nous utilisons l'énergie plutôt que la vitesse de la particule, c'est-à-dire :

$$E = \frac{1}{2} m V_n^2 . \quad (5.4)$$

Aussi, les calculs de cellules se font en résolvant l'équation de transport pour des états stationnaires, donc l'équation (5.2) devient :

$$\Sigma_{(\vec{r}, E)} \phi_{(\vec{r}, E, \vec{\Omega})} + \vec{\Omega} \cdot \vec{\nabla} \phi_{(\vec{r}, E, \vec{\Omega})} = Q_{(\vec{r}, E, \vec{\Omega})} . \quad (5.5)$$

Le terme de source est défini comme étant la somme de la diffusion de neutrons d'un état vers celui considéré et de la fission. On peut donc l'écrire comme suit :

$$\begin{aligned} Q_{(\vec{r}, E, \vec{\Omega})} = & \int_{4\pi} d^2\Omega' \int_0^\infty dE' \Sigma_{s(\vec{r}, E \leftarrow E', \vec{\Omega} \leftarrow \vec{\Omega}')} \phi_{(\vec{r}, E', \vec{\Omega}')} \\ & + \frac{1}{4\pi k_{eff}} \sum_{j=1}^{J_{fiss}} \chi_{j(E)} \int_0^\infty dE' \nu \Sigma_{f, j(\vec{r}, E')} \int_{4\pi} d^2\Omega' \phi_{(\vec{r}, E', \vec{\Omega}')} \end{aligned} . \quad (5.6)$$

Le facteur k_{eff} sert au maintien de l'équilibre à l'état stationnaire. Cette équation est discrétisée en groupes d'énergie pour les calculs de cellules mais nous ne la présenterons pas dans ce document. Le lecteur est invité à consulter, entre autres, le livre de Stacey (2001) et de Rozon (1992).

Il est à noter que les calculs de cœur ne se font pas avec l'équation de transport mais plutôt avec l'équation de la diffusion. Celle-ci est une simplification de l'équation de transport grâce au développement du flux neutronique en harmonique sphérique d'ordre un :

$$\phi_{(\vec{r}, E, \vec{\Omega}, t)} \approx \frac{1}{4\pi} \left(\phi_{(\vec{r}, E, t)} + 3\vec{\Omega} \cdot \int_{4\pi} d^2\Omega \vec{\Omega} \phi_{(\vec{r}, E, \vec{\Omega}, t)} \right). \quad (5.7)$$

En multipliant par $\vec{\Omega}$ et en intégrant sur l'angle solide l'équation transport (5.2), nous obtenons :

$$\frac{1}{V_n} \frac{\partial \vec{J}_{(\vec{r}, E, t)}}{\partial t} + \Sigma_{(\vec{r}, V_n)} \vec{J}_{(\vec{r}, E, t)} + \frac{1}{3} \vec{\nabla} \phi_{(\vec{r}, V_n, t)} = \int_0^\infty dE' \Sigma_{s(\vec{r}, E \leftarrow E')} \phi_{(\vec{r}, E', t)} \vec{J}_{(\vec{r}, E, t)}, \quad (5.8)$$

où,

$$\vec{J}_{(\vec{r}, E, t)} = \int_{4\pi} d^2\Omega \vec{\Omega} \phi_{(\vec{r}, E, \vec{\Omega}, t)}. \quad (5.9)$$

La loi de Fick nous permet de trouver l'équation de diffusion :

$$\vec{J}_{(\vec{r}, E, t)} = -D_{(\vec{r}, E)} \vec{\nabla} \cdot \phi_{(\vec{r}, E, t)} = -\frac{1}{3\Sigma_{tr(\vec{r}, E, t)}} \vec{\nabla} \cdot \phi_{(\vec{r}, E, t)}. \quad (5.10)$$

Ainsi, en introduisant l'équation de Fick dans l'équation de continuité (5.8), on trouve l'équation de diffusion dépendante du temps :

$$\frac{1}{V_n} \frac{\partial \phi_{(\vec{r}, E, t)}}{\partial t} + \Sigma_{(\vec{r}, V_n)} \phi_{(\vec{r}, E, t)} - \vec{\nabla} \cdot D_{(\vec{r}, E)} \vec{\nabla} \cdot \phi_{(\vec{r}, E, t)} = \int_0^\infty dE' \Sigma_{s(\vec{r}, E \leftarrow E')} \phi_{(\vec{r}, E', t)}. \quad (5.11)$$

Là encore, cette équation est résolue avec une discrétisation en énergie pour les calculs de cœurs des réacteurs.

Pour le calcul de l'évolution des concentrations isotopiques C_i pour chaque isotope i , nous nous basons sur l'équation de Bateman donnée par :

$$\frac{dC_i}{dt} = \sum_j [(\sigma_{j \rightarrow i} + \eta_i \sigma_{f,j}) \phi - \lambda_{j \rightarrow i}] C_j - (\sigma_{a,i} \phi + \lambda_i) C_i. \quad (5.12)$$

Nous avons donc à itérer sur les équations de tous les isotopes i afin d'avoir la solution de l'évolution des concentrations isotopiques.

2 - Algorithmes de couplage

Nous présenterons dans cette partie l'algorithme des calculs pour un couplage neutronique/thermohydraulique. Pour cela, nous présenterons des schémas montrant tout d'abord comment fonctionne le module de thermohydraulique. Par la suite, nous présenterons le fonctionnement de l'algorithme de couplage entre le neutronique et la thermohydraulique.

Le calcul de thermohydraulique se fait par la résolution successive de l'équation de conservation de la masse, de la quantité de mouvement, de l'énergie et finalement l'équation de transfert de chaleur. Le schéma présenté dans la figure 21 montre la succession de calcul fait dans le code de thermohydraulique.

Pour le couplage de la neutronique avec la thermohydraulique, nous allons utiliser un algorithme de calcul basé sur le « Feedback Model », dit FBM (Sissaoui *et al.*, 1999). L'environnement CLE2000 (Roy, 1999) ainsi que la GANLIB (Roy & Hébert, 2000) seront utilisés pour le couplage avec le code de neutronique DONJON-3 (Varin *et al.*, 2005). L'algorithme calcule les sections efficaces à différents états et paramètres locaux en l'utilisant des corrélations du FBM (Sissaoui *et al.*, 1999). L'objectif, ici, est de tester l'utilisation de l'algorithme de calculs. Celui-ci est présenté dans la figure 22.

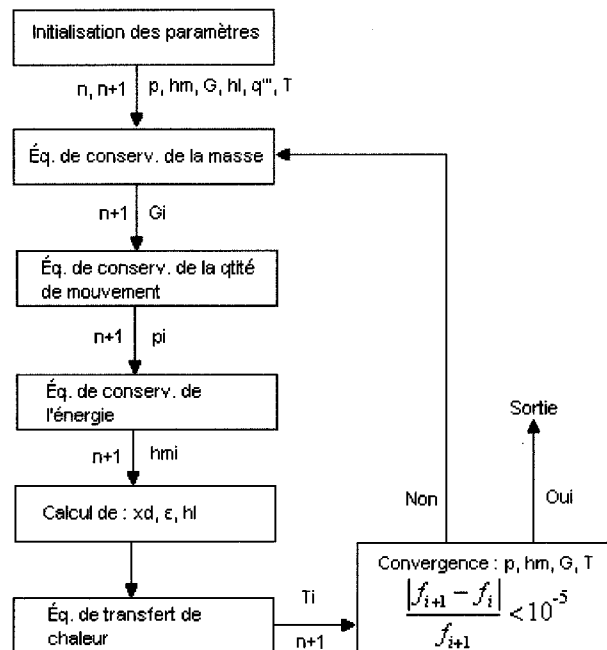


Figure 21 : Algorithme de calcul d'ARTHUR.

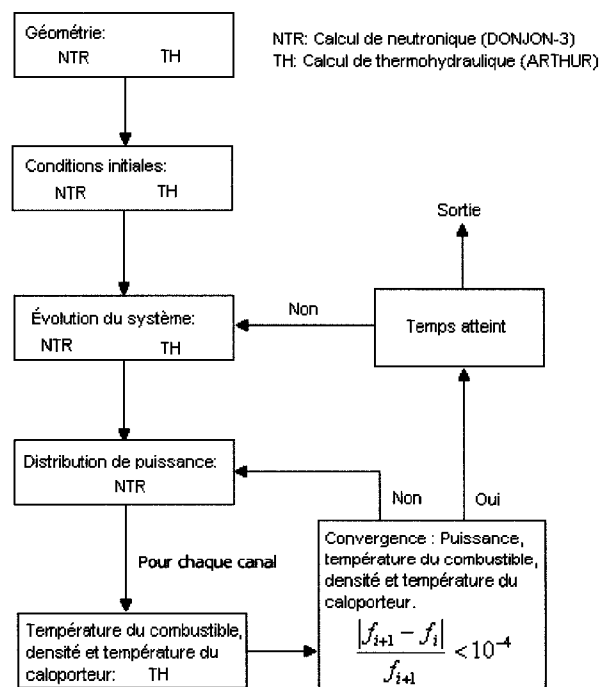


Figure 22 : Algorithme de couplage neutronique/thermohydraulique.

Nous avons présenté dans ce chapitre l'algorithme de calcul utilisé dans le code ARTHUR mais aussi l'algorithme de calcul que nous utiliserons pour les simulations de couplage neutronique/thermohydraulique. Nous verrons dans la suite du mémoire les résultats que nous avons obtenus en analysant la qualité de ces derniers. Ceux-ci seront donnés pour des états stationnaires et transitoires de réacteurs CANDU-6 mais aussi pour un réacteur fictif de type CANDU à quatre canaux.

CHAPITRE 6

SIMULATIONS NEUTRONIQUE/THERMOHYDRAULIQUE

L'impact de la thermohydraulique en neutronique montre une importance particulière lors de transitoires de thermohydraulique, notamment en cas d'accident. En effet, dans ces cas, nous pouvons avoir des variations thermohydrauliques brusques, i.e. une chute de pression, une diminution du débit du caloporteur mais aussi une augmentation de sa température. Dans ce travail, nous nous limiterons à présenter des simulations avec des variations thermohydrauliques lentes afin de vérifier la fonctionnalité du code ARTHUR. Néanmoins, ces simulations permettront de montrer la faisabilité de transitoires avec des variations brutales des propriétés thermohydrauliques comme c'est le cas pendant les accidents. Nous verrons donc, dans ce chapitre, les résultats de simulations à l'état stationnaire pour des CANDU-6. Nous verrons également plusieurs courtes transitoires pour un CANDU-6 avec des variations des paramètres de thermohydraulique. Finalement, nous étudierons en détails des simulations de transitoires pour un réacteur CANDU fictif à quatre canaux.

1 - Simulation pour un état stationnaire

Nous présenterons dans cette partie les résultats obtenus pour le couplage neutronique/thermohydraulique tout d'abord pour un état stationnaire avant de le présenter pour un état transitoire.

Nos calculs ont été faits pour un CANDU-6 dans les conditions de fonctionnement suivantes :

Tableau 5 : Conditions nominales de fonctionnement des CANDU-6.

Puissance totale (kW)	2061.4
Température du caloporteur à l'entrée des canaux ($^{\circ}C$)	262.4
Pression à la sortie des canaux (MPa)	10.8
Flux massique à l'entrée des canaux ($kg/m^2/s$)	7007.8
Température du modérateur ($^{\circ}C$)	72.5

Pour ces conditions et en utilisant le modèle FBM, les calculs couplés pour un état stationnaire donnent les résultats présentés aux figures 23, 24, 25 et 26.

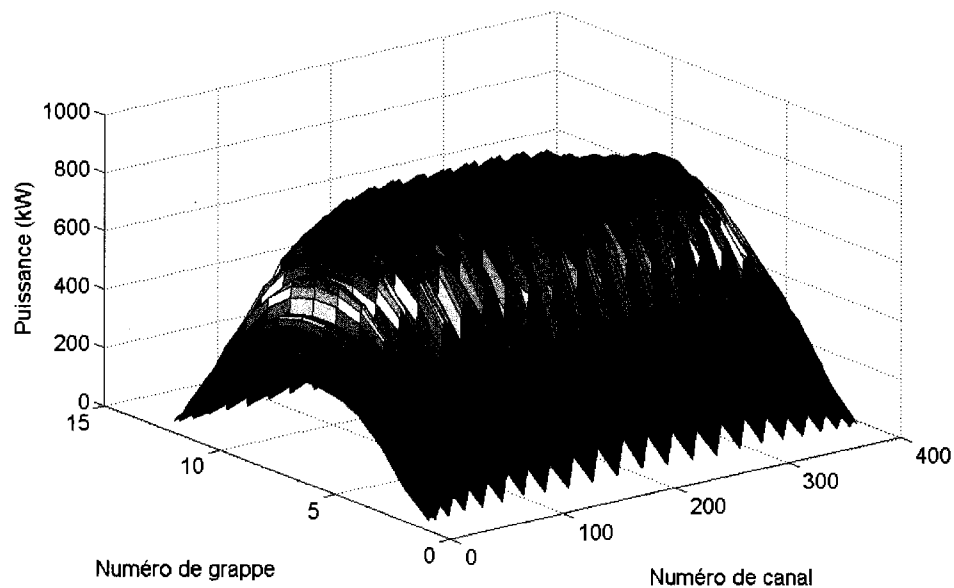


Figure 23 : Puissance moyenne par grappe.

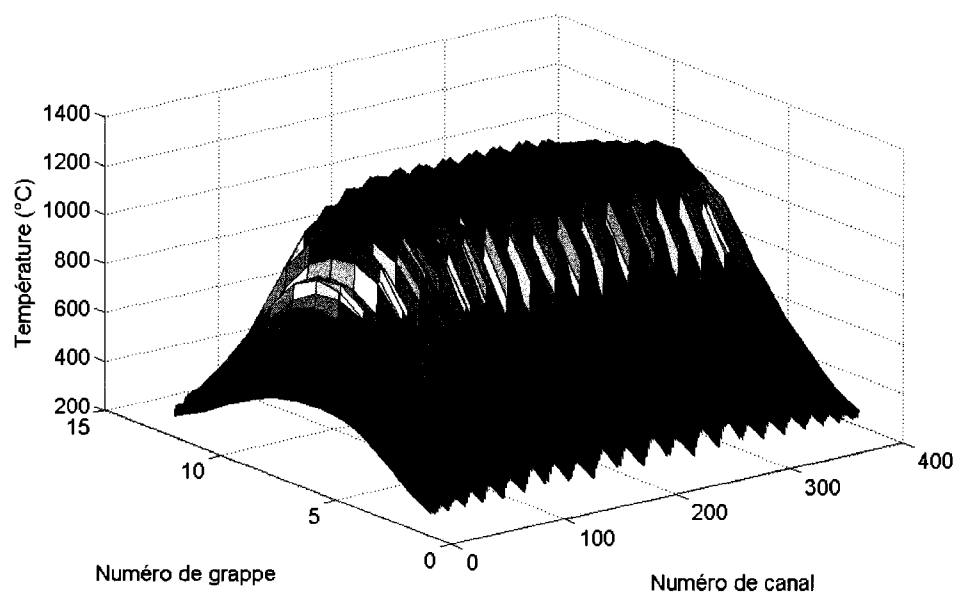


Figure 24 : Température moyenne par grappe du combustible.

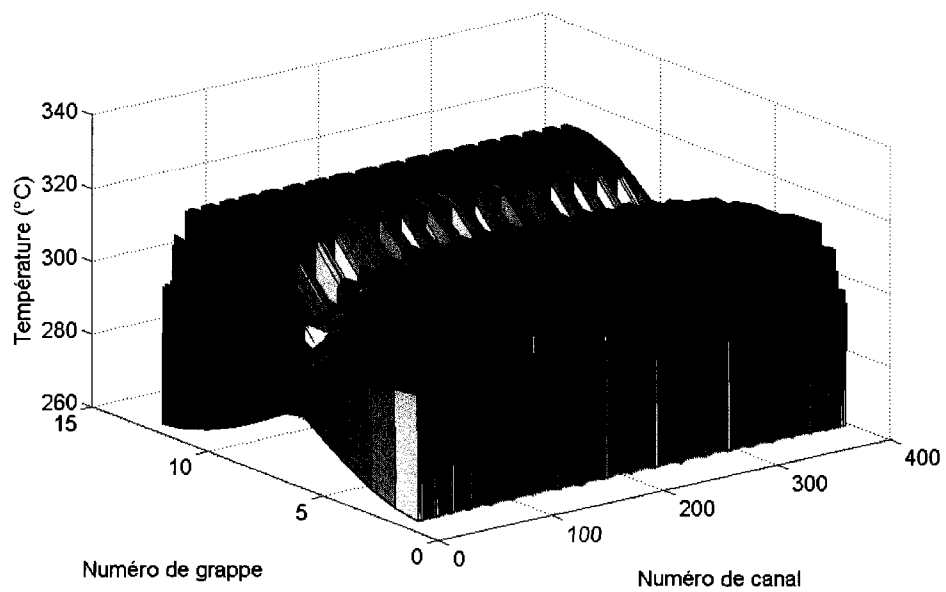


Figure 25 : Température du caloporteur.

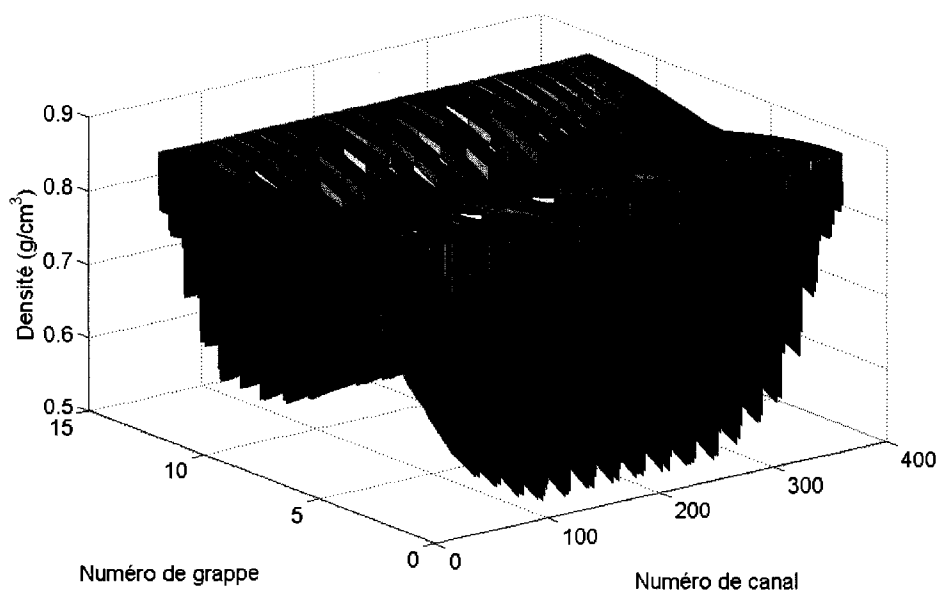


Figure 26 : Densité du caloporteur.

Les figures 23 et 24 montrent le lien qui existe entre la température du combustible et la puissance dégagée. En effet, la forme de la puissance dans le réacteur est très semblable à celle de la température du combustible. Pour ce qui est des figures 25 et 26, celle de la température et de la densité du caloporteur, nous remarquons un effet papillon lié au fait que les écoulements dans des canaux voisins se font dans des directions opposées. Il est à noter que la température du caloporteur atteint la saturation (voir la figure 25) à la fin des canaux centraux d'où la diminution de la densité du caloporteur due à l'ébullition sous-refroidie et saturée tel que montré à la figure 26. Pour compléter cette étude, il nous faudrait voir la rétroaction que nous avons entre la thermohydraulique et la neutronique. Pour cela, nous allons comparer un calcul classique de neutronique, sans rétroaction de la thermohydraulique, avec un calcul couplé entre la neutronique et la thermohydraulique. Dans ce cas, nous obtenons les différences sur la puissance présentées à la figure 27.

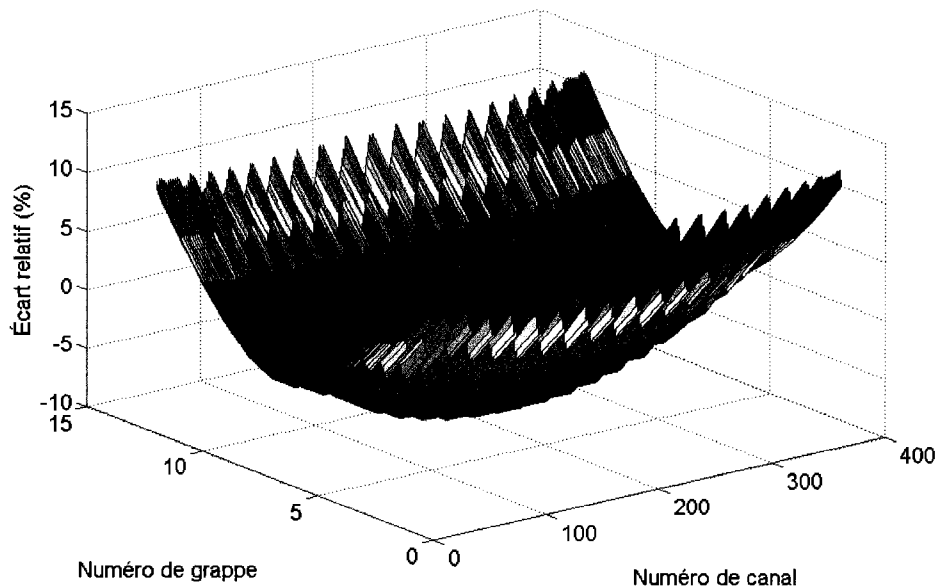


Figure 27 : Écart relatif sur la puissance entre un calcul couplé et non couplé.

L'effet du couplage sur la puissance est montré par l'écart relatif calculé de la façon suivante :

$$\acute{E}R = \frac{P_{coupl\acute{e}} - P_{noncoupl\acute{e}}}{P_{coupl\acute{e}}} \times 100 \quad (6.1)$$

Ici, nous nous basons sur deux calculs, l'un standard, sans rétroaction de la thermohydraulique sur la neutronique et vice-versa, et l'autre avec cette rétroaction. La figure 27 nous permet de voir une rétroaction négative (-5%) de la température du combustible sur la puissance au centre du réacteur. La température du combustible étant sous-estimée sans couplage, nous trouvons des puissances plus élevées qu'elles ne devraient l'être. En effet, sans couplage, la température du combustible est fixée généralement à 900°C dans les calculs standards alors que les calculs de thermohydraulique nous montrent des températures allant jusqu'à 1400°C environ. Grâce au couplage, nous avons des résultats plus réalistes au centre du coeur. Cet effet est contraire aux extrémités. Nous avons une importante rétroaction positive de plus de 10%. Ceci est dû à la surestimation de la température du combustible sans

couplage et par conséquent à une sous-estimation de la puissance produite. La température du combustible aux extrémités calculée par ARTHUR est d'environ 400°C. À la rétroaction de la température du combustible sur la puissance s'ajoute la rétroaction de la densité du caloporteur sur la puissance. Celle-ci est positive avec la diminution de la densité du caloporteur. Cet effet amplifie la rétroaction positive que nous avons avec la température du combustible à l'extrémité du réacteur, c'est pourquoi nous avons des erreurs relatives de plus de 10% aux extrémités.

2 - Simulations pour des états transitoires

Présentons maintenant les résultats obtenus pour des états transitoires. Tout d'abord, nous présenterons des transitoires très courtes, i.e. sur 0.5s, pour des réacteurs CANDU-6. Par la suite, nous verrons des transitoires plus longues, d'une minute, pour des réacteurs fictifs à quatre canaux. Étant donné que la puissance de calcul requise pour effectuer des calculs couplés de CANDU-6 est grande, nous nous sommes limités à des transitoires de 5 pas de temps de 0.1s permettant de valider l'algorithme de calcul. Cependant, nos calculs pour des réacteurs fictifs à quatre canaux seront fait pour des transitoires de une minute avec le même pas de temps que précédemment. Les transitoires consisteront en une baisse du débit à l'entrée, une augmentation de la température du caloporteur à l'entrée et une baisse de pression à la sortie des canaux.

Pour les calculs de transitoires courtes de CANDU-6, nous utiliserons les variations suivantes par rapport à l'état nominal d'opération donné dans le tableau 5 :

Tableau 6 : Conditions de transitoires pour CANDU-6.

Cas 1 : Variation du flux massique à l'entrée des 380 canaux	-10%/min
Cas 2 : Variation de la température du caloporteur à l'entrée des 380 canaux	+10%/min
Cas 3 : Variation de la pression à la sortie des 380 canaux	-10%/min

Pour les courtes transitoires de CANDU-6 que nous avons simulé avec un couplage complet du coeur, nous obtenons les résultats présentés aux figures 28 à 39.

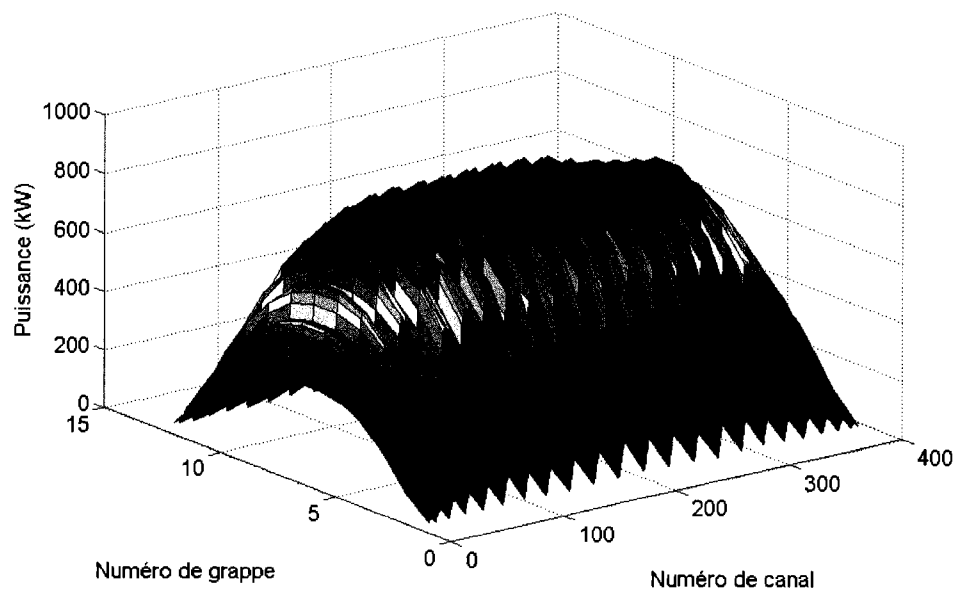


Figure 28 : Puissance pour une baisse de débit d'un CANDU.

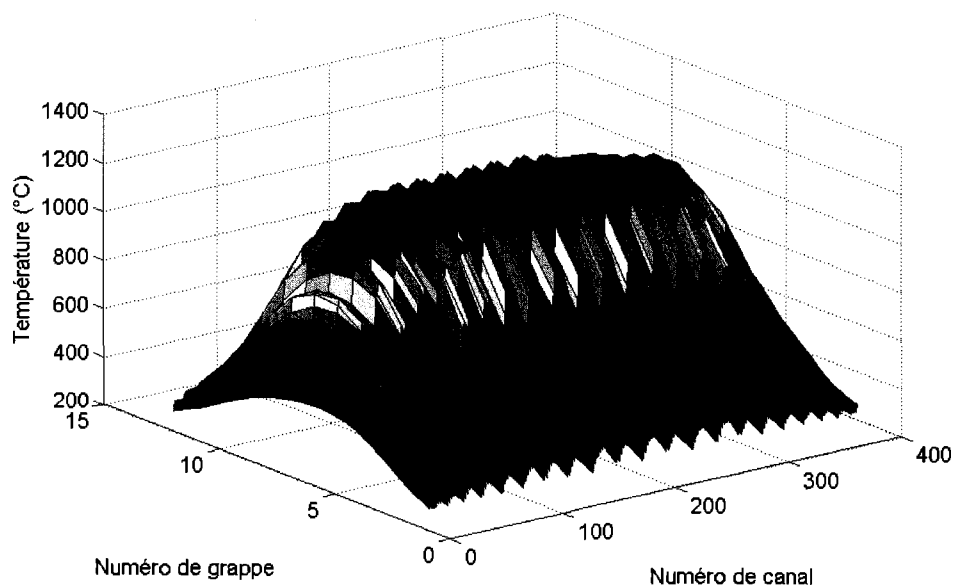


Figure 29 : Température du combustible pour une baisse de débit d'un CANDU.

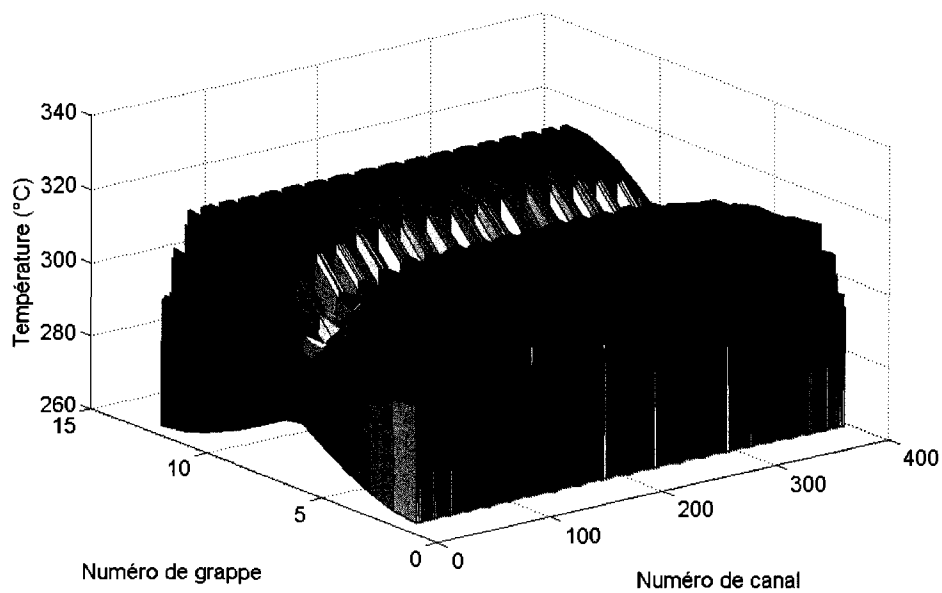


Figure 30 : Température du caloporteur pour une baisse de débit d'un CANDU.

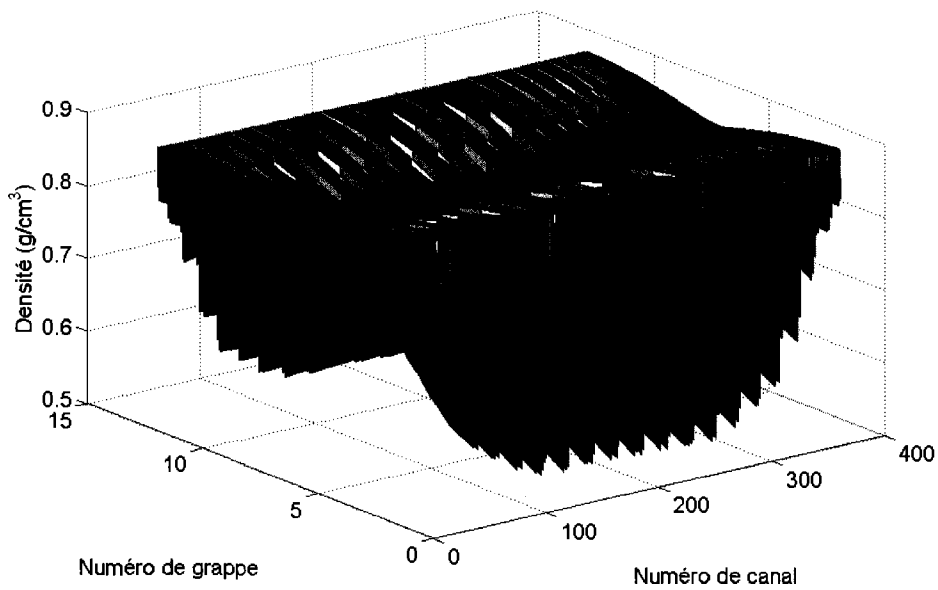


Figure 31 : Densité du caloporteur pour une baisse de débit d'un CANDU.

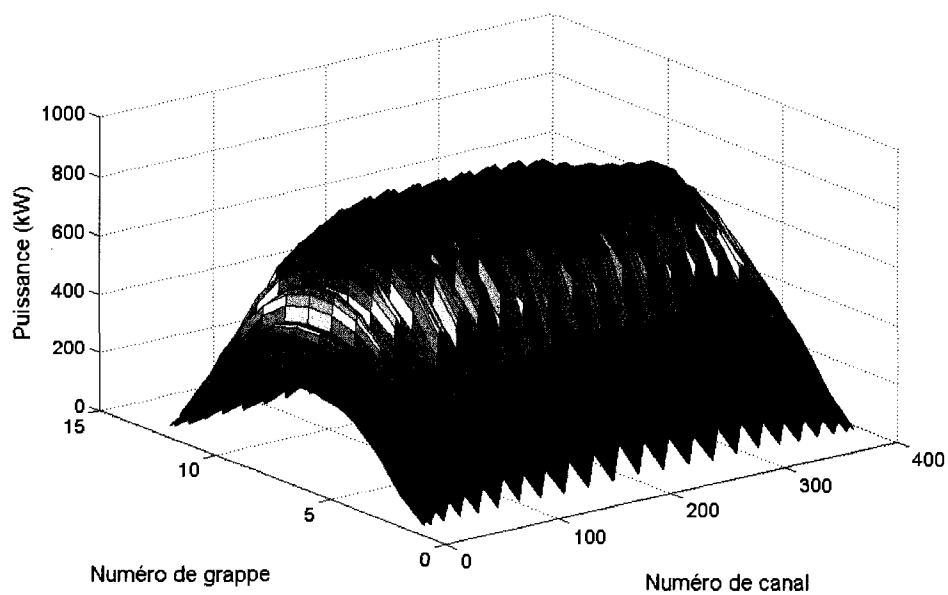


Figure 32 : Puissance pour une augmentation de la température du caloporteur d'un CANDU.

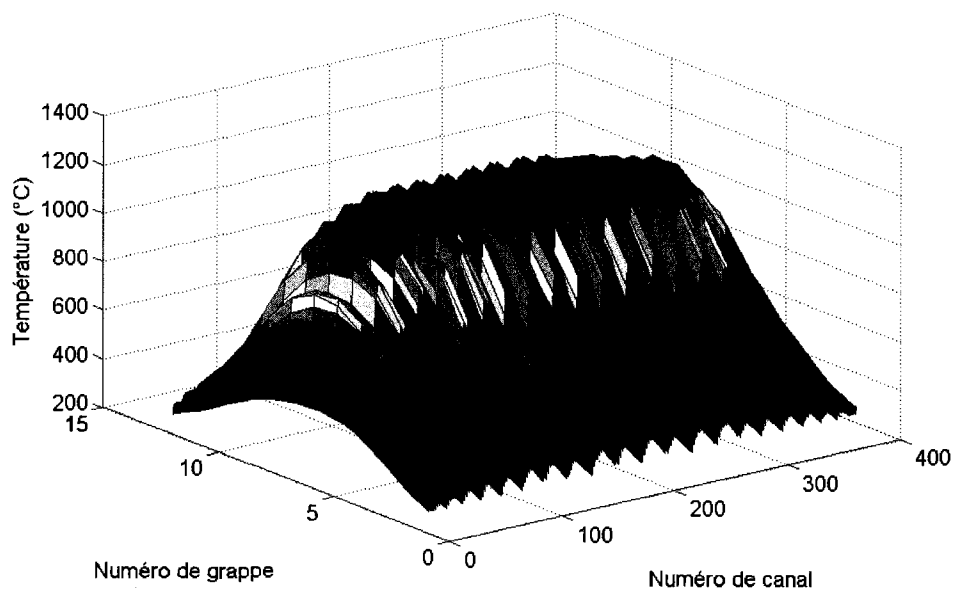


Figure 33 : Température du combustible pour une augmentation de la température du caloporteur d'un CANDU.

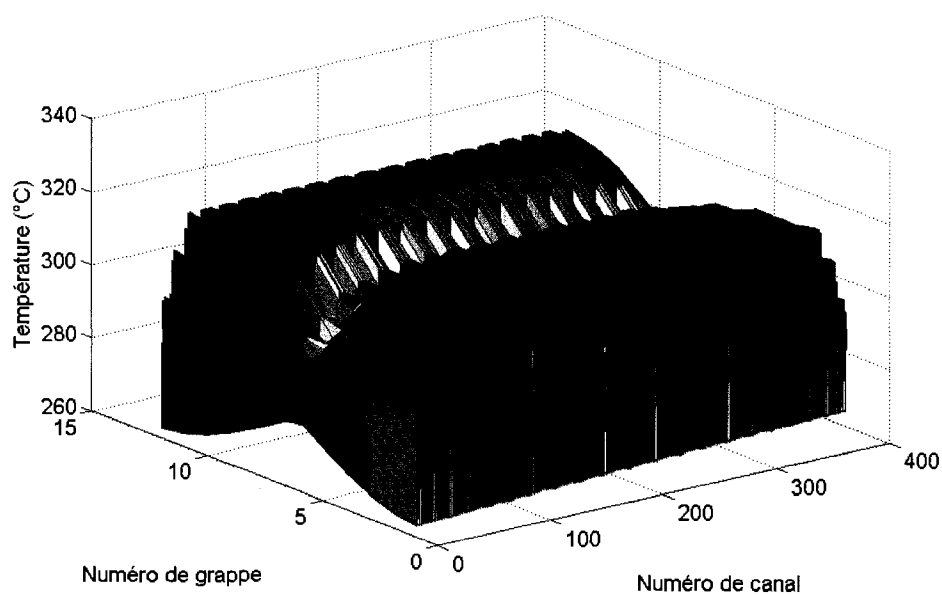


Figure 34 : Température du caloporteur pour une augmentation de la température du caloporteur d'un CANDU.

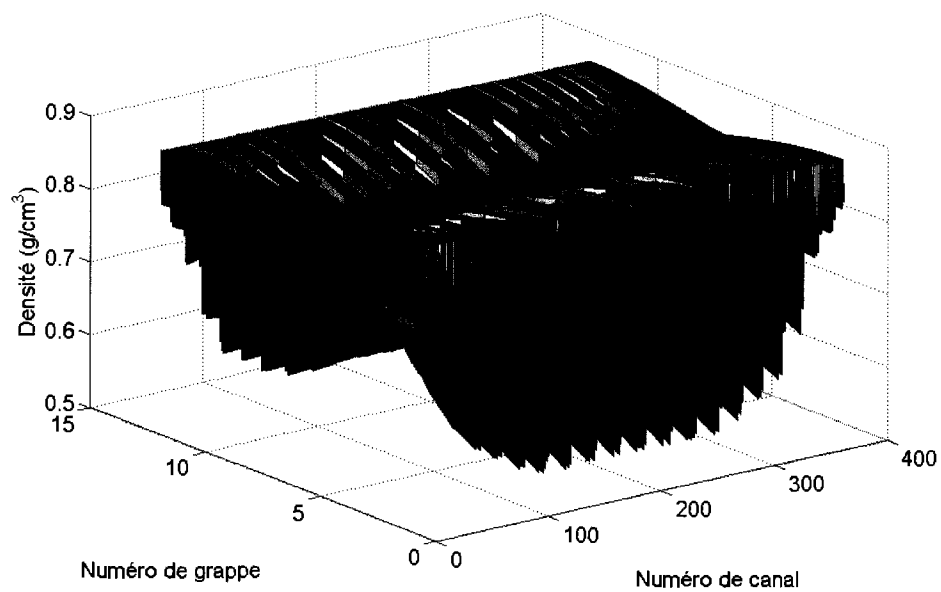


Figure 35: Densité du caloporteur pour une augmentation de la température du caloporteur d'un CANDU.

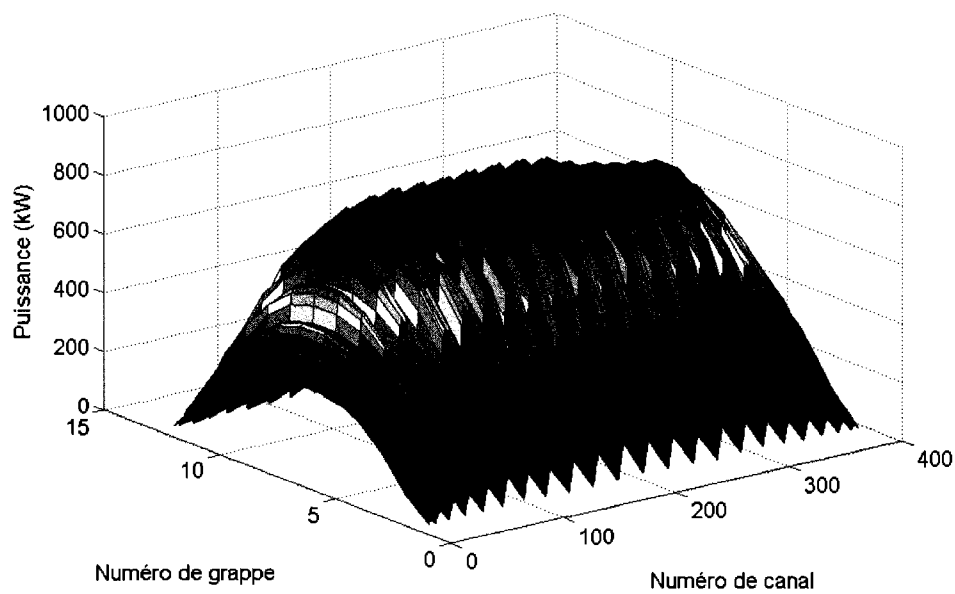


Figure 36 : Puissance pour une diminution de la pression d'un CANDU.

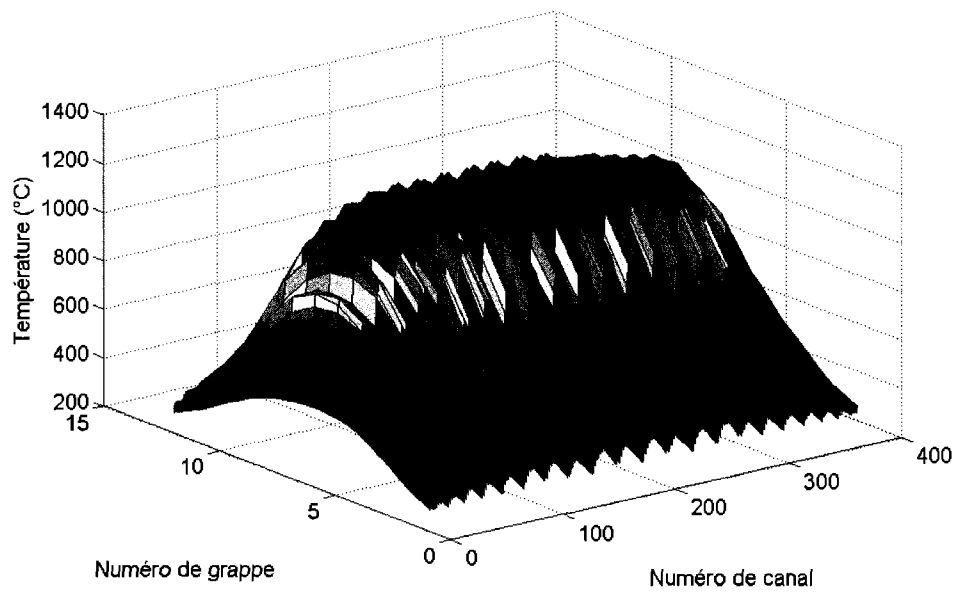


Figure 37 : Température du combustible pour une diminution de la pression d'un CANDU.

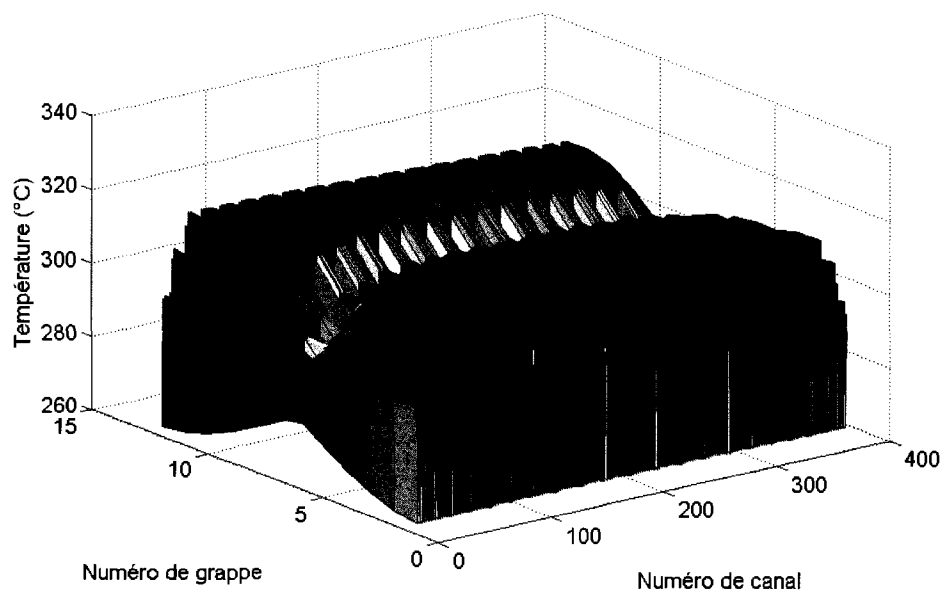


Figure 38 : Température du caloporteur pour une diminution de la pression d'un CANDU.

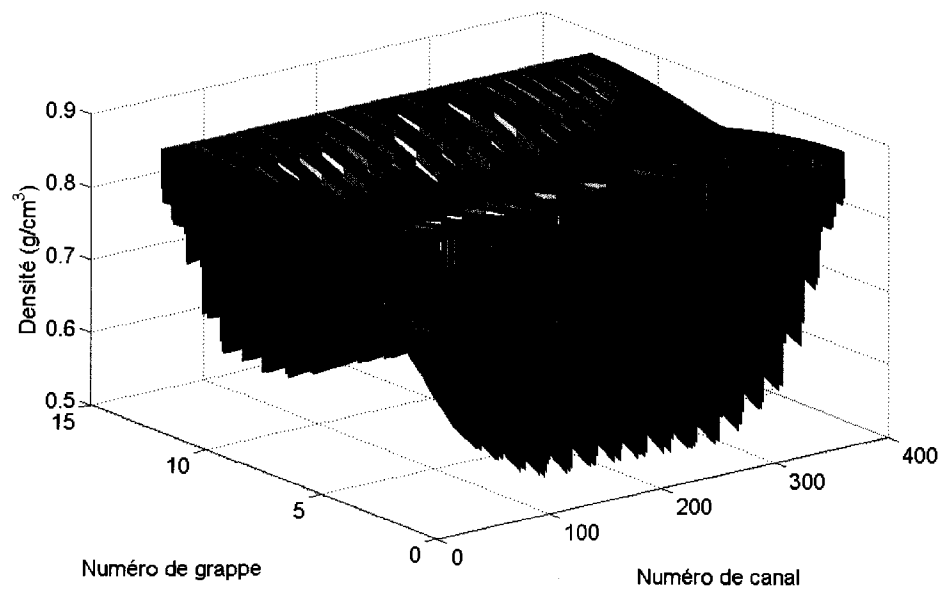


Figure 39 : Densité du caloporteur pour une diminution de la pression d'un CANDU.

Dans les trois cas de que nous avons présentés aux figures 28 à 39, les variations que nous pouvons observer durant la transitoire sont négligeables étant donné la grande inertie thermique du système par rapport à l'intervalle de temps considéré. En effet,

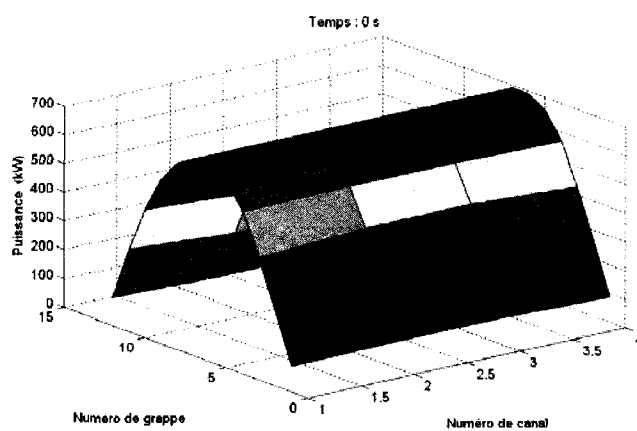
dans le cas d'une diminution linéaire de 0.1% du débit sur 0.5s, nous obtenons une diminution globale de la puissance du réacteur de 2.0% ainsi qu'une réduction de la température moyenne du combustible de 4.2%. Quant aux variations de la densité et de la température du caloporteur, elles restent négligeables, soit 0.9% d'augmentation de la densité moyenne du caloporteur mais une baisse de 0.2% de la température moyenne du caloporteur. Également, pour une transitoire de température, i.e. une augmentation linéaire de 0.5°C de la température du caloporteur à l'entrée des 380 canaux en 0.5s, nous observons une diminution de puissance de 2.1%, une augmentation de la densité moyenne du caloporteur de 0.9%, une diminution de la température moyenne du combustible de 4.9% et une stagnation de la température moyenne du caloporteur. Finalement, pour une diminution de 0.1% de la pression de façon linéaire sur 0.5s, nous observons une chute de puissance de 2.3%, une augmentation de la densité moyenne du caloporteur de 1.0%, une diminution des températures moyennes du caloporteur et du combustible de respectivement 0.4% et 4.3%. Il est difficile d'analyser ces résultats étant donnée l'intervalle de temps restreint de nos calculs et la grandeur de l'inertie thermique. Aussi, il est à noter que la précision du calcul est de 0.1% d'écart relatif donc très proche des petites valeurs de variations. Pour une analyse de la rétroaction, nous étudierons plus bas le cas d'un réacteur fictif à quatre canaux. Néanmoins, cette partie nous a permis de vérifier la fonctionnalité de l'algorithme de calculs couplés.

Pour les transitoires d'un réacteur fictif formée par 4 canaux selon une configuration 2×2, nous utiliserons des conditions d'évolution indiquées dans le tableau 7. Il faut noter que les variations que nous utilisons s'appliquent à l'ensemble des canaux du réacteur. Les canaux sont numérotés de droite à gauche sur une même ligne de canaux et de haut en bas pour les lignes de canaux suivantes.

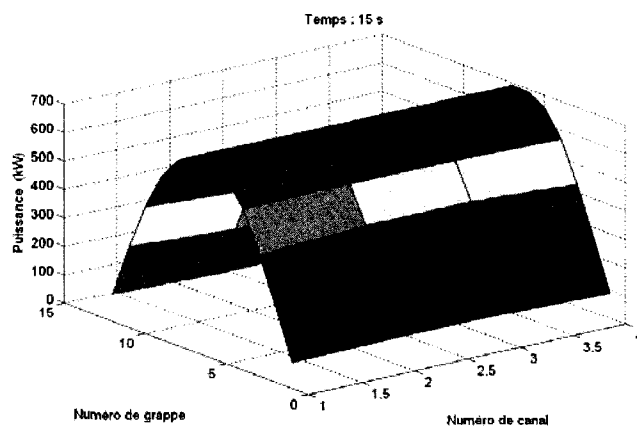
Tableau 7 : Conditions de transitoires pour un réacteur fictif.

Cas 1 : Variation du flux massique à l'entrée des 4 canaux	-4.3%/min
Cas 2 : Variation de la température du caloporteur à l'entrée des 4 canaux	0.6%/min
Cas 3 : Variation de la pression à la sortie des 4 canaux	-5.0%/min

Les résultats des simulations de transitoires lentes pour le réacteur fictif à quatre canaux sont présentés aux figures 40 à 51.

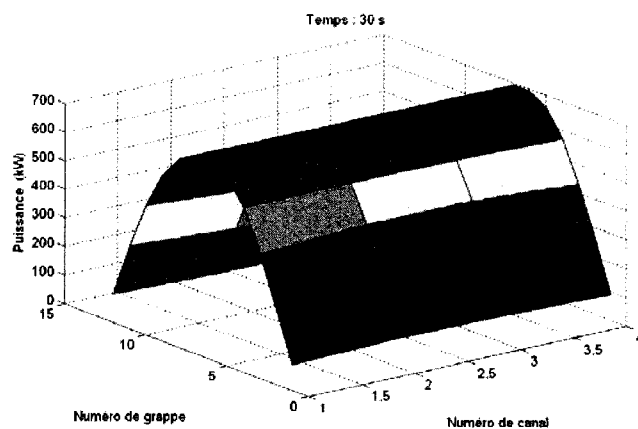


(a)

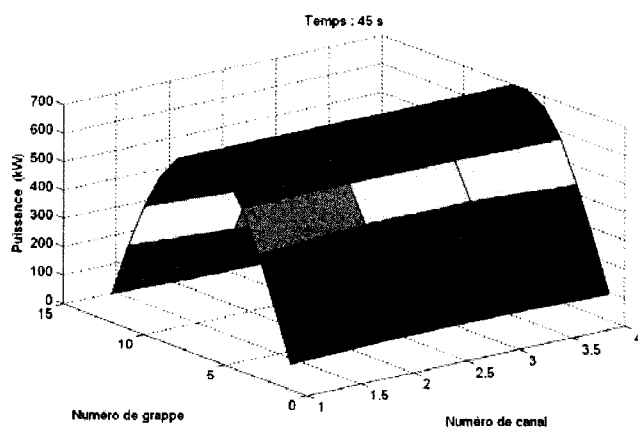


(b)

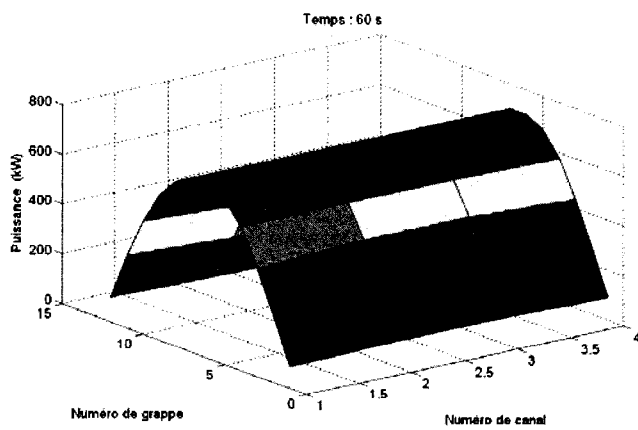
Figure 40 : Puissance du réacteur lors d'une réduction de débit.



(c)

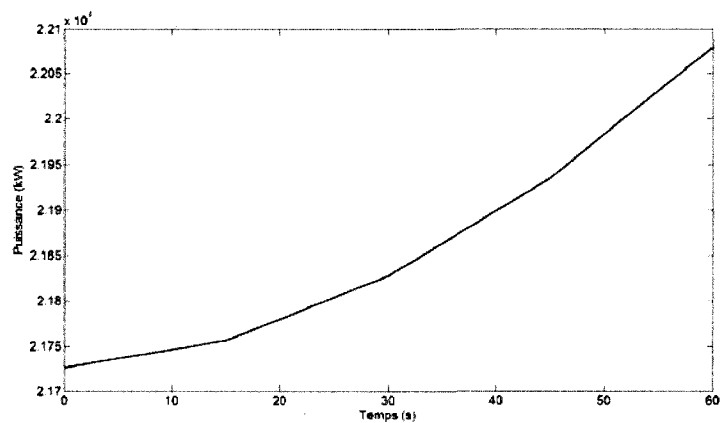


(d)



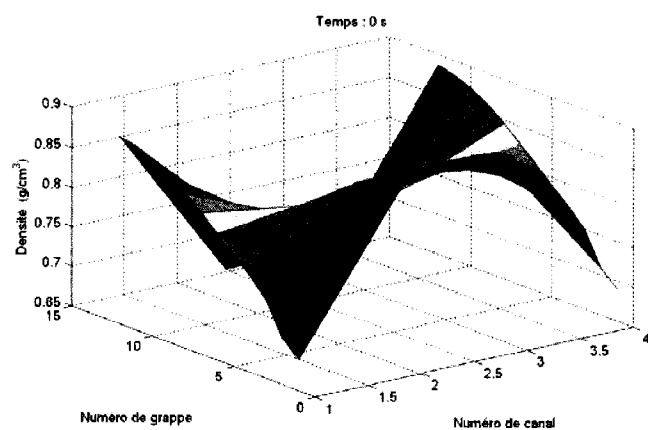
(e)

Figure 40. (Suite)

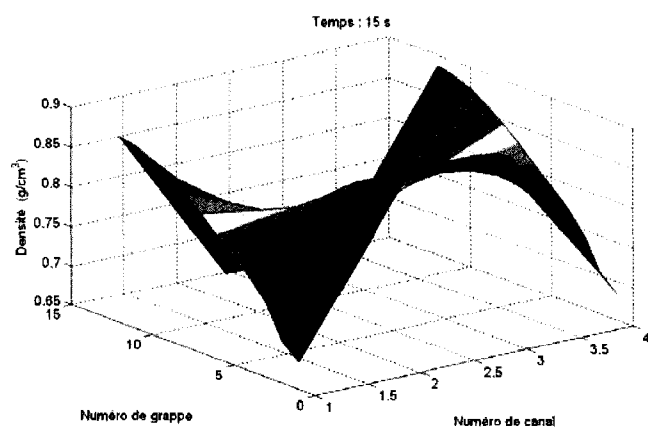


(f)

Figure 40. (Suite 2)

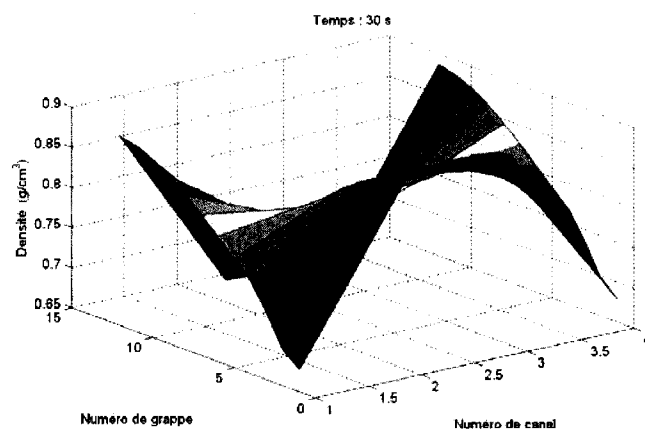


(a)

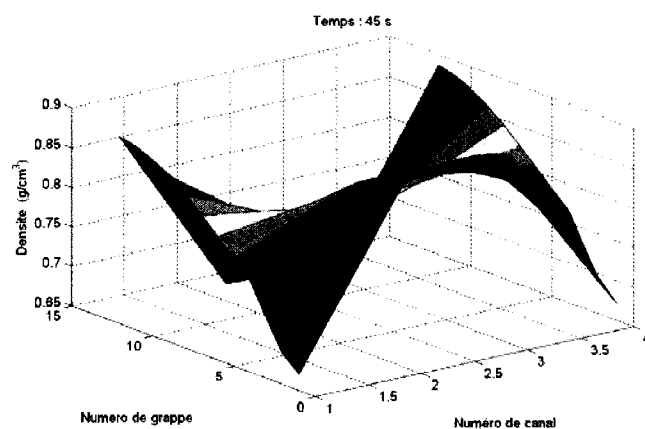


(b)

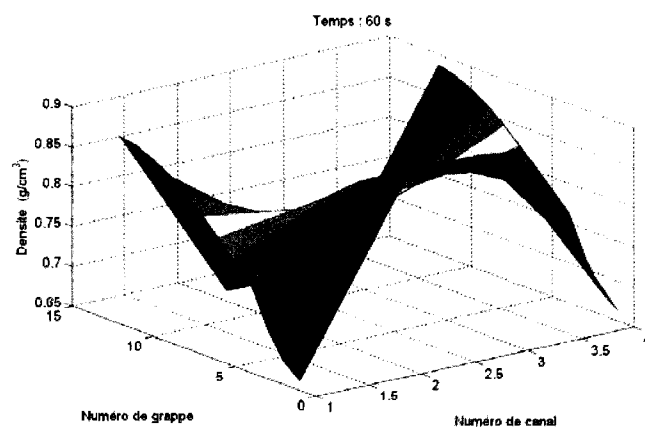
Figure 41 : Densité de caloporteur lors d'une réduction de débit.



(c)

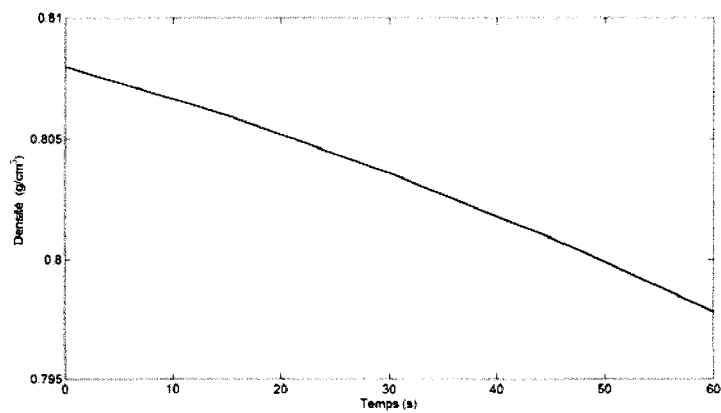


(d)



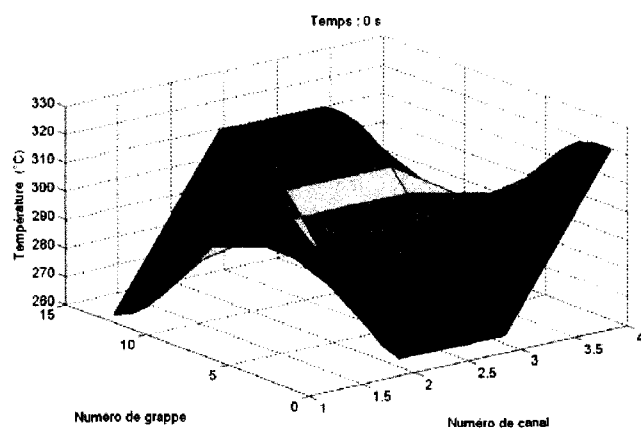
(e)

Figure 41. (Suite)

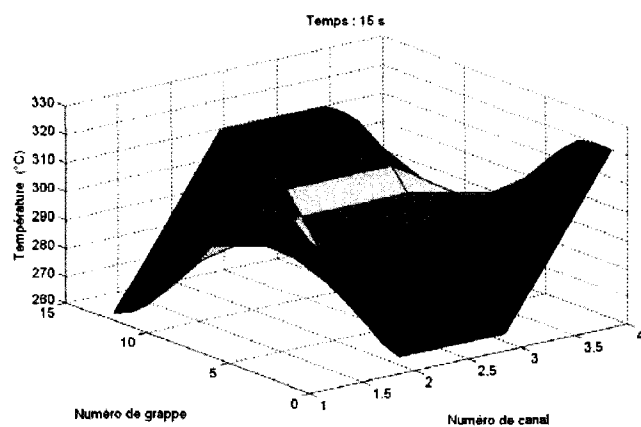


(f)

Figure 41. (Suite 2)

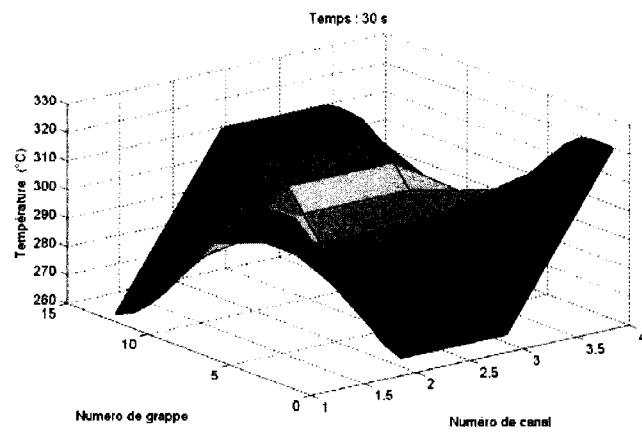


(a)

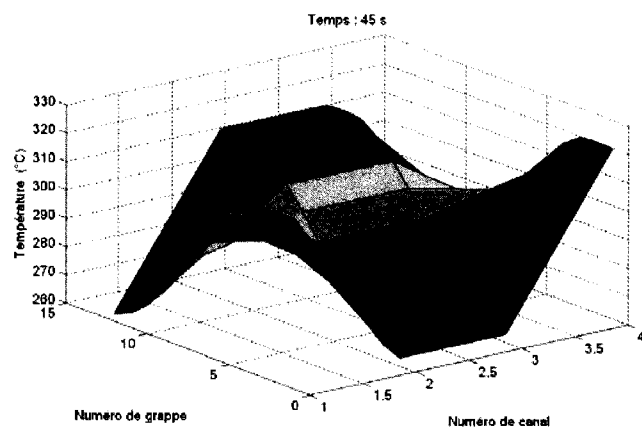


(b)

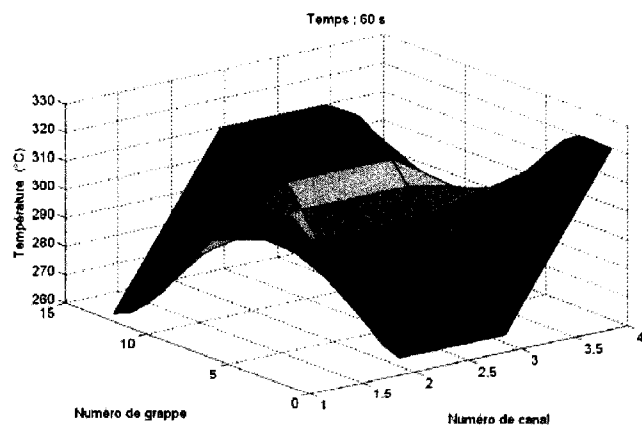
Figure 42 : Température du caloporteur lors d'une réduction de débit.



(c)

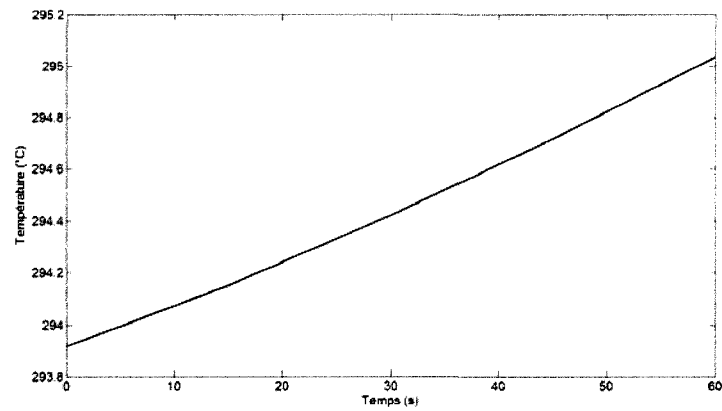


(d)



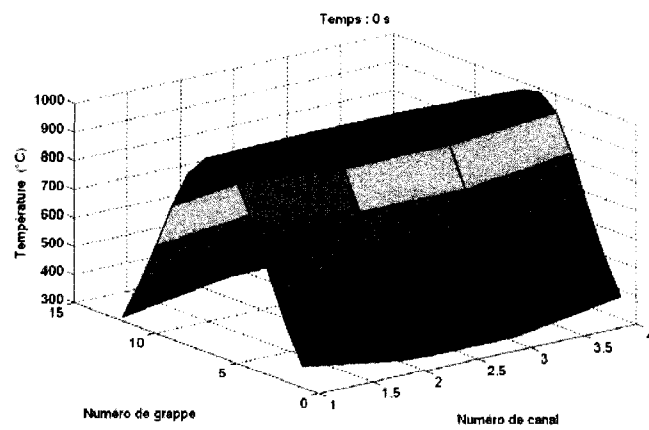
(e)

Figure 42. (Suite)

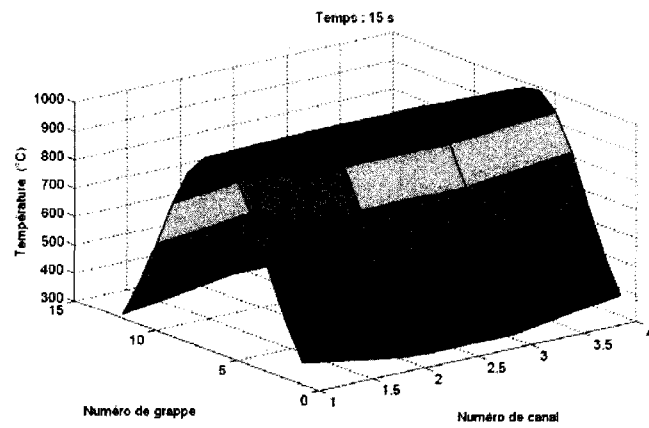


(f)

Figure 42. (Suite 2)

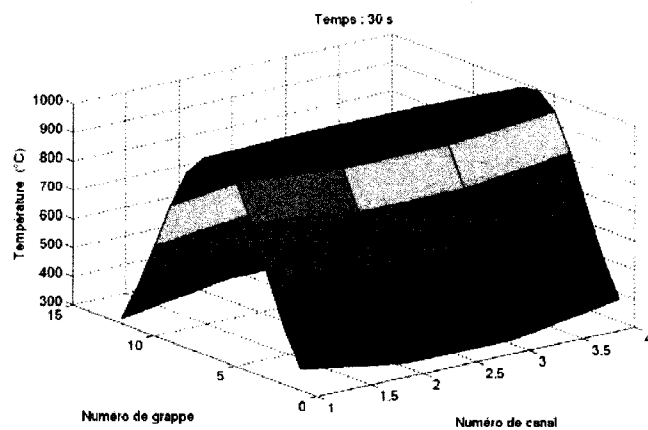


(a)

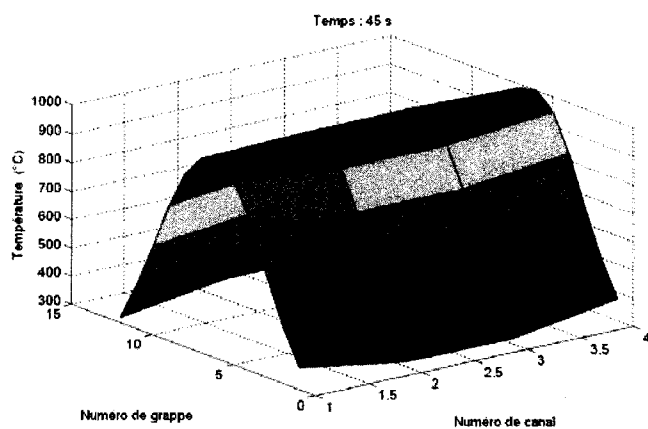


(b)

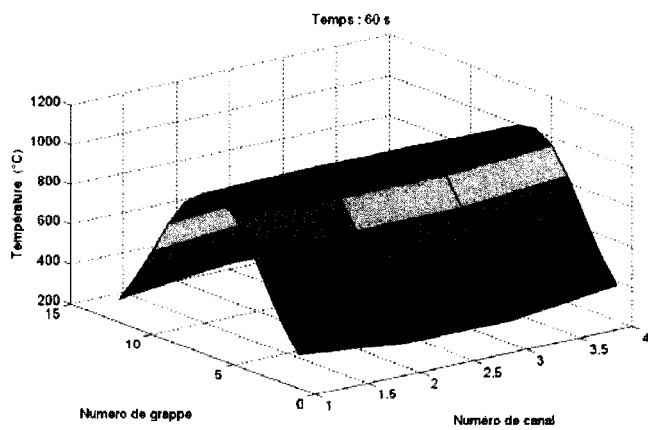
Figure 43 : Température du combustible lors d'une réduction de débit.



(c)

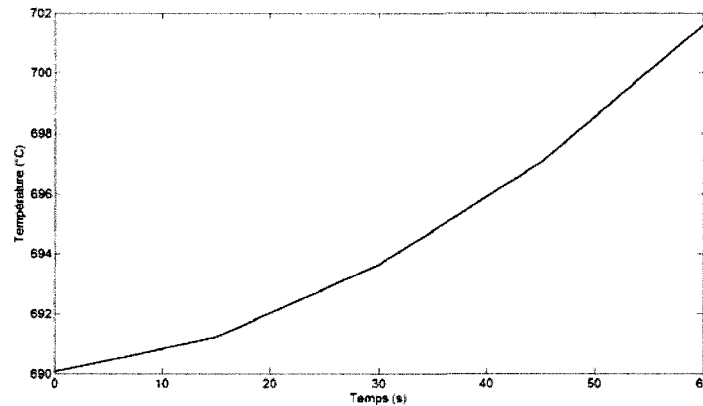


(d)



(e)

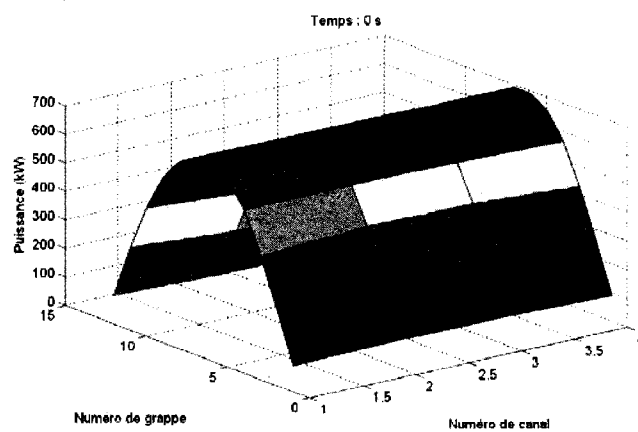
Figure 43. (Suite)



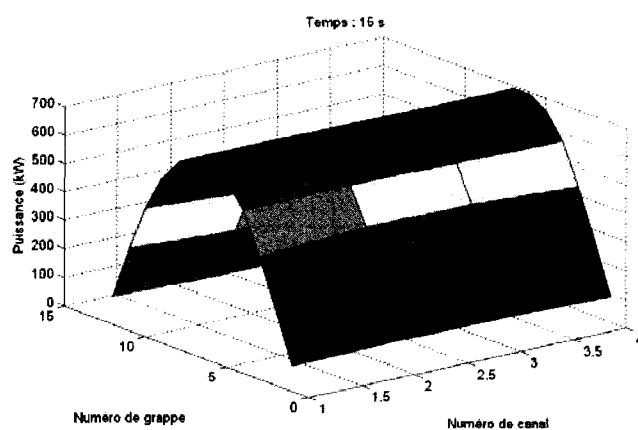
(f)

Figure 43. (Suite 2)

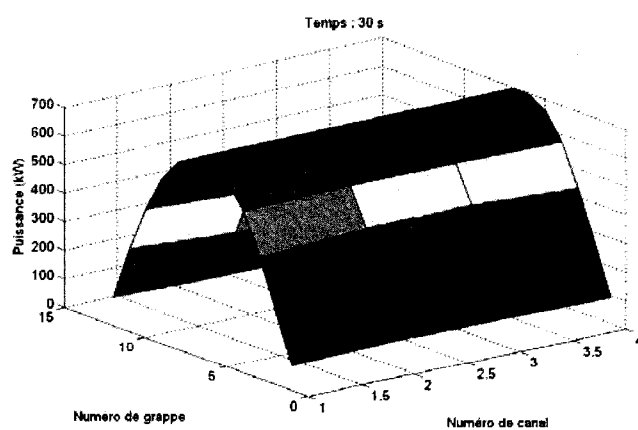
Suite à une baisse progressive du débit dans les canaux, nous pouvons observer une augmentation de la température du caloporteur (figure 42f) ainsi qu'une diminution de la densité du caloporteur (figure 41f). L'augmentation de la température du caloporteur entraîne une augmentation de la température du combustible (figure 43f). La rétroaction de la thermohydraulique entraîne une augmentation de la puissance du réacteur (figure 40f). À son tour, l'augmentation de la puissance amplifie la variation des paramètres thermohydrauliques. Pour le cas d'une augmentation de température du caloporteur à l'entrée des quatre canaux d'un réacteur fictif, nous obtenons les résultats présentés aux figures 44, 45, 46 et 47. Dans ce cas, nous avons augmenté la température à l'entrée des canaux. L'augmentation de la température du caloporteur (figure 46f) fait en sorte que la densité du caloporteur diminue (figure 45f) et que la température du combustible (figure 47f) augmente. Ces deux paramètres introduisent une réactivité positive dans le réacteur qui se traduit par une augmentation de la puissance comme le montre la figure 44f. Cette élévation de puissance augmente la température du combustible et diminue la densité du caloporteur, d'où l'accélération de l'augmentation de la puissance dans le temps. Pour une diminution de la pression à la sortie des quatre canaux du réacteur fictif, nous obtenons les résultats présentés aux figures 48, 49, 50 et 51.



(a)

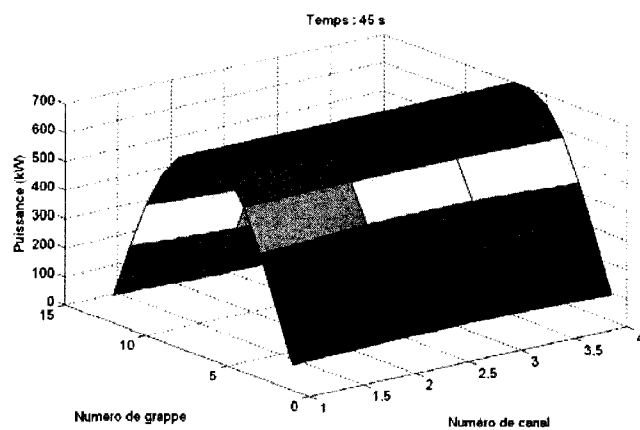


(b)

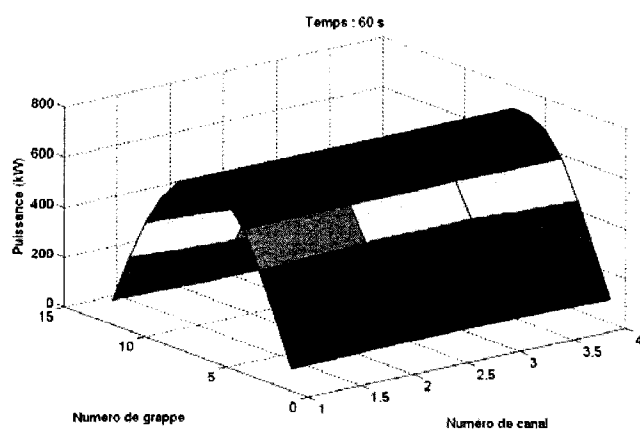


(c)

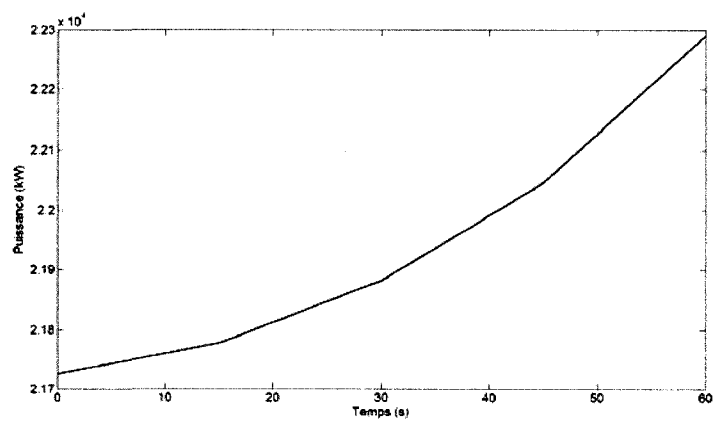
Figure 44 : Puissance du réacteur lors d'une augmentation de température du caloporteur.



(d)

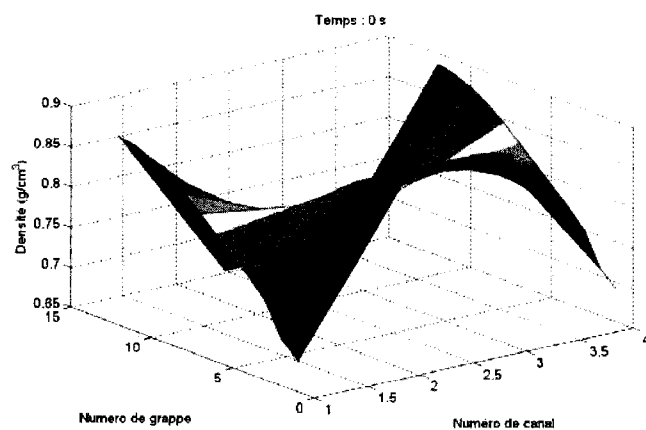


(e)

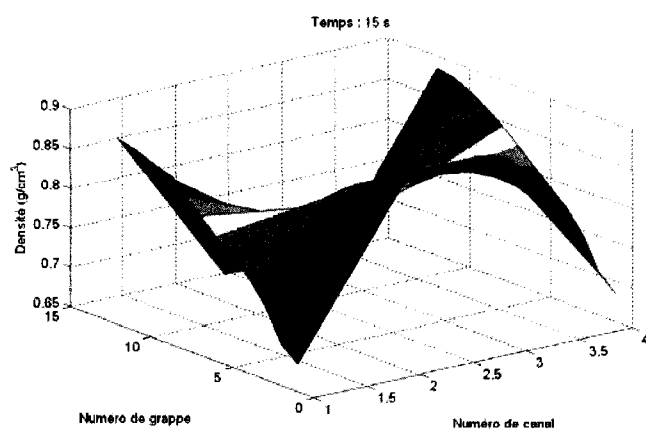


(f)

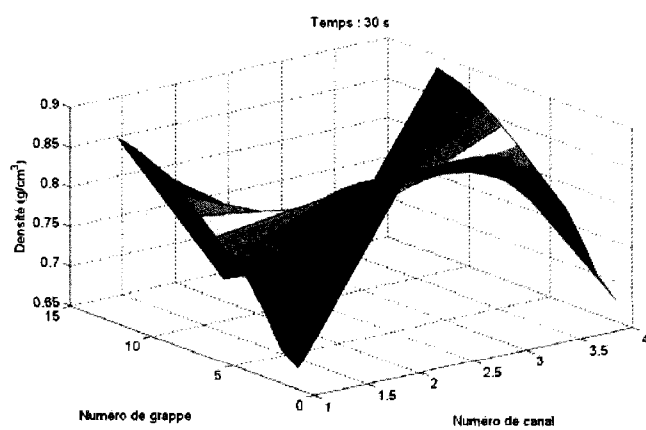
Figure 44. (Suite)



(a)

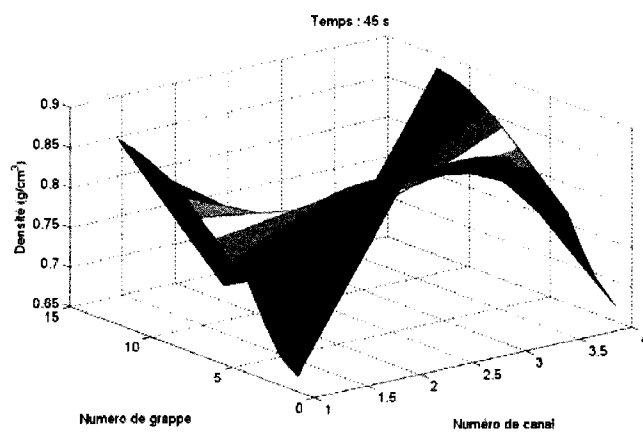


(b)

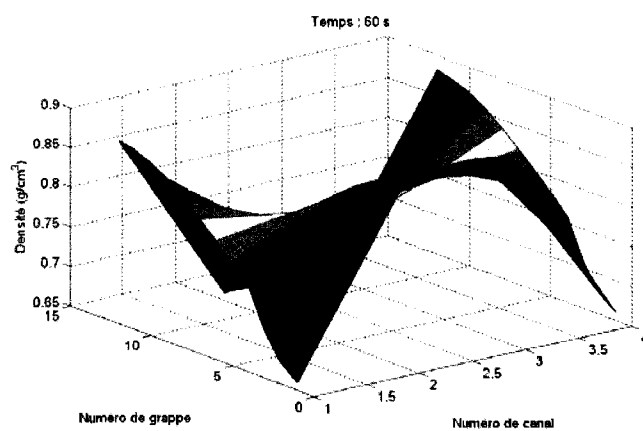


(c)

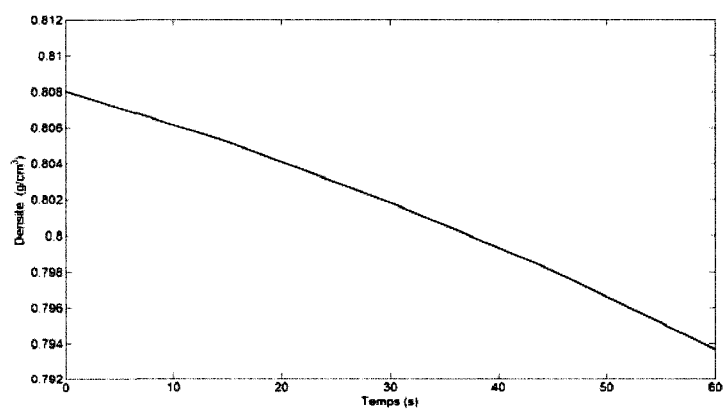
Figure 45 : Densité du caloporteur lors d'une augmentation de température du caloporteur.



(d)

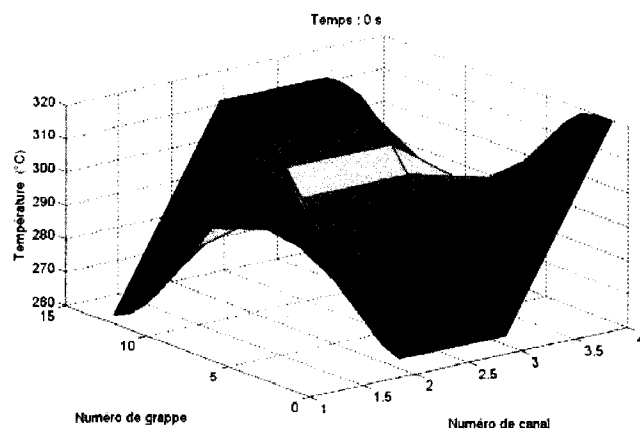


(e)

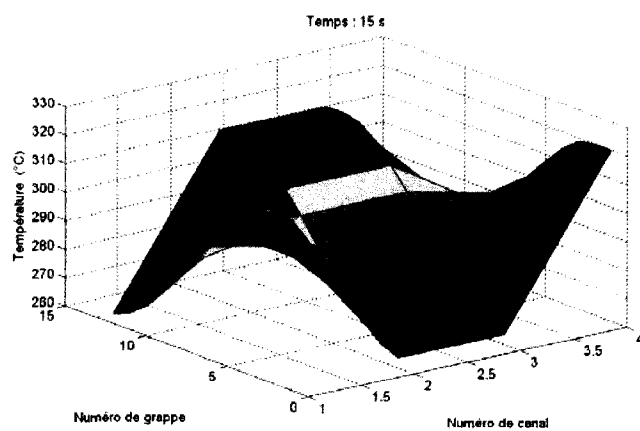


(f)

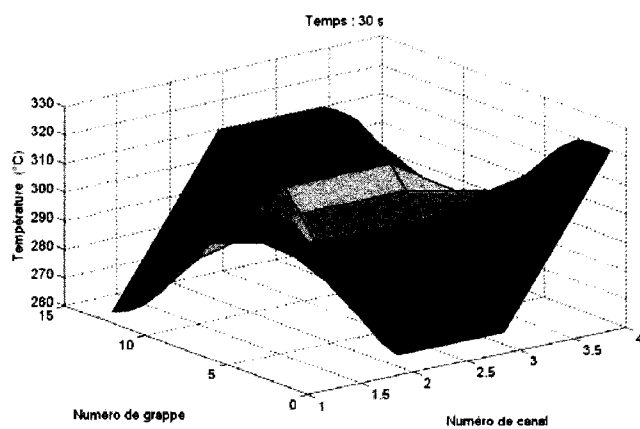
Figure 45. (Suite)



(a)

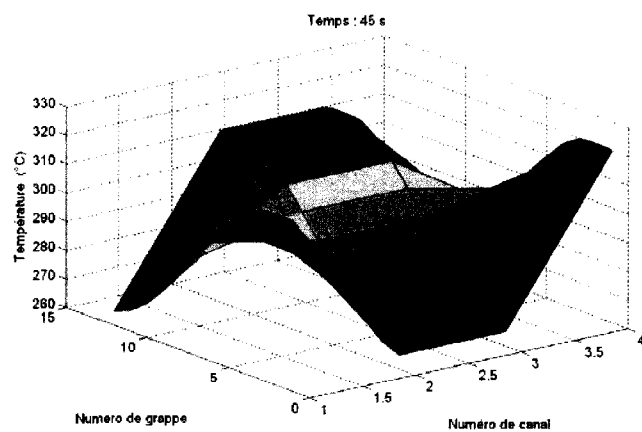


(b)

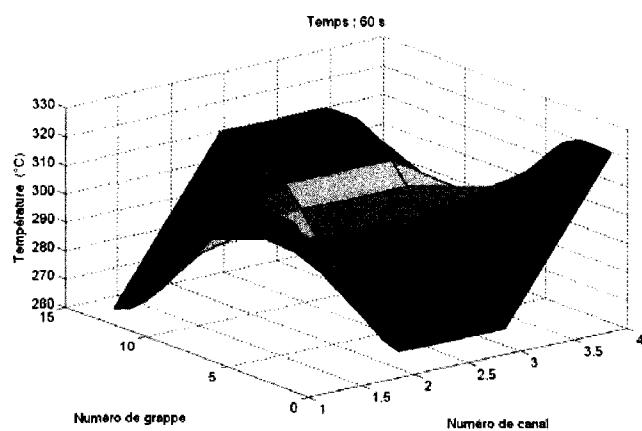


(c)

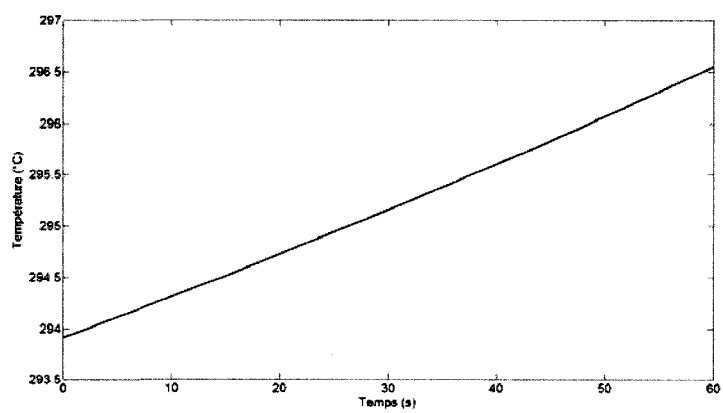
Figure 46 : Température du caloporteur lors d'une augmentation de température du caloporteur.



(d)

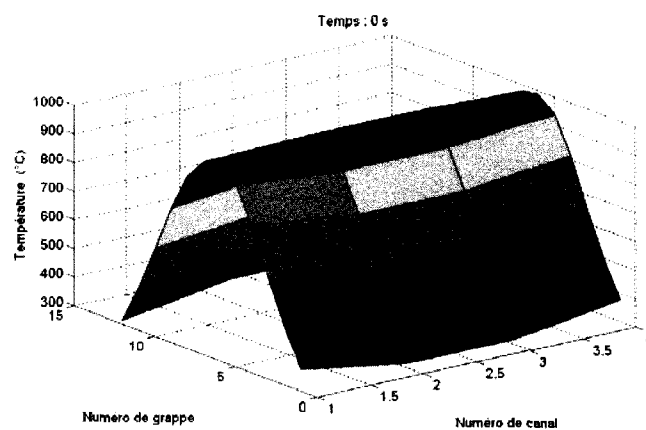


(e)

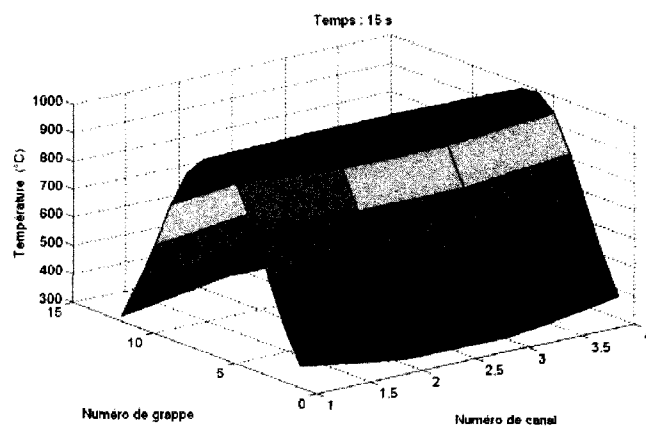


(f)

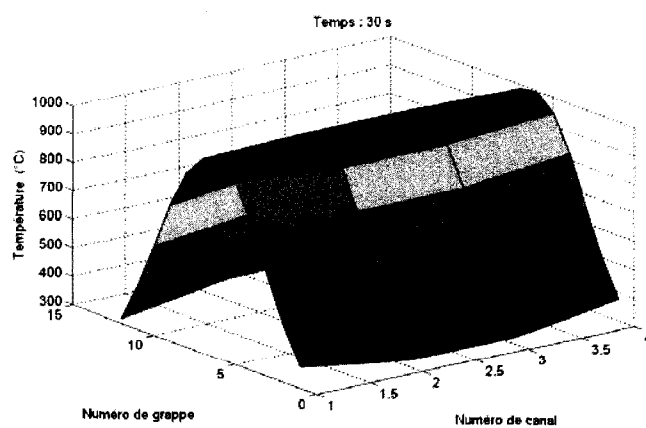
Figure 46. (Suite)



(a)

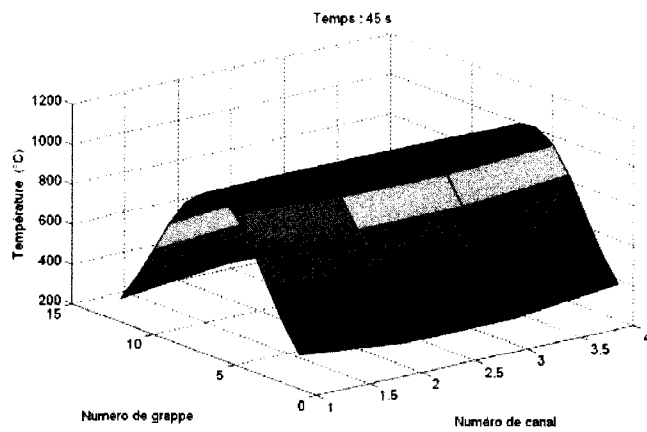


(b)

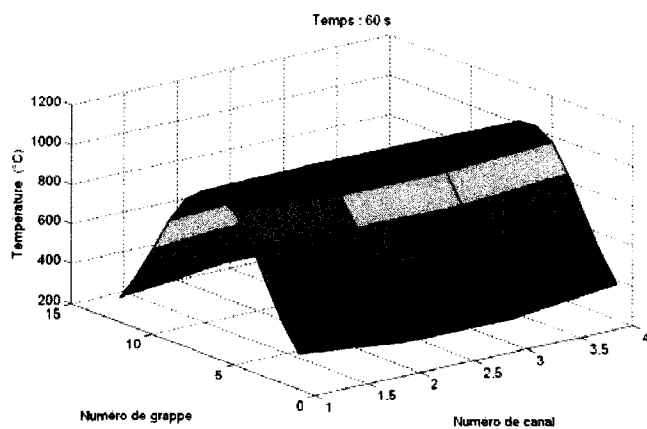


(c)

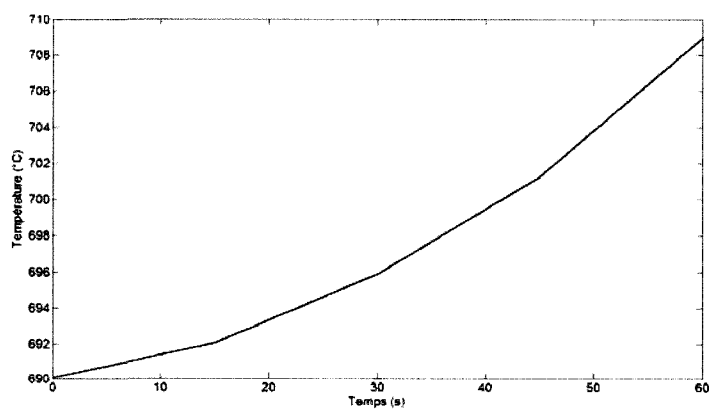
Figure 47 : Température du combustible lors d'une augmentation de température du caloporteur.



(d)

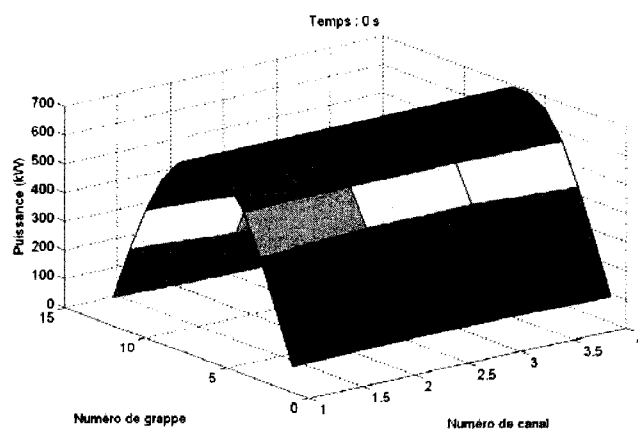


(e)

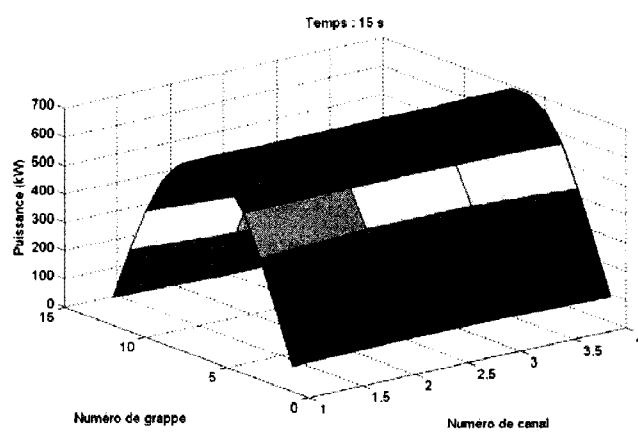


(f)

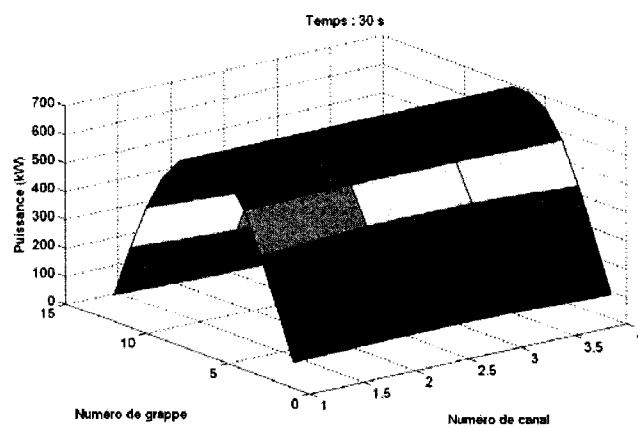
Figure 47. (Suite)



(a)

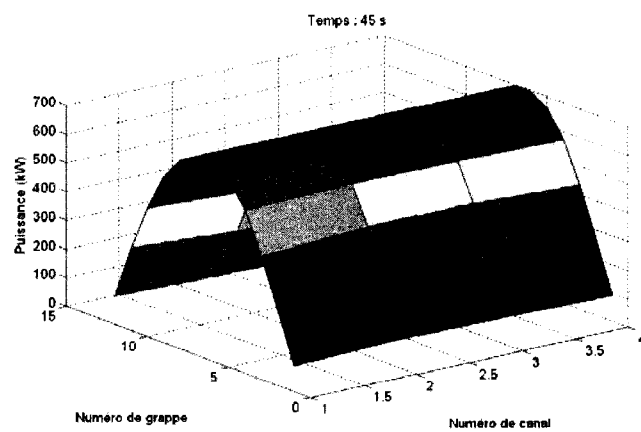


(b)

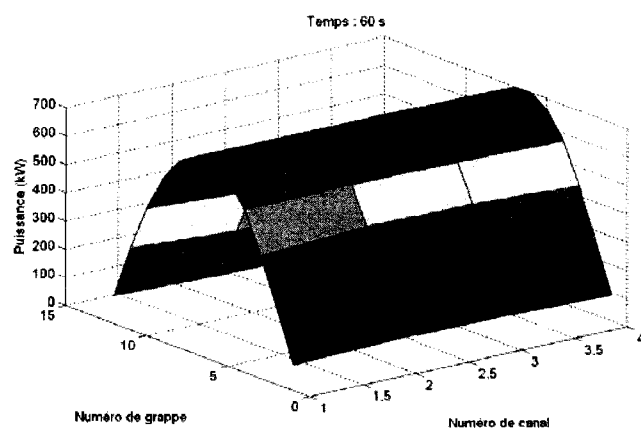


(c)

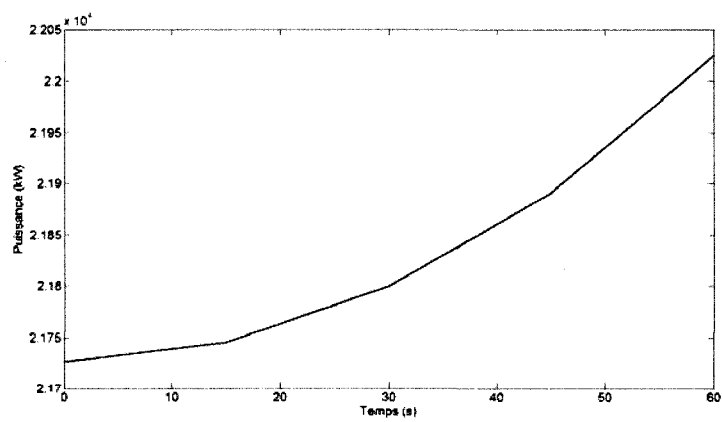
Figure 48 : Puissance du réacteur lors d'une diminution de pression de sortie.



(d)

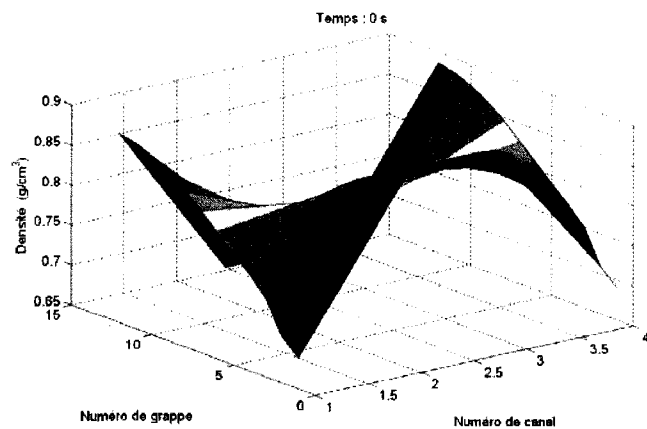


(e)

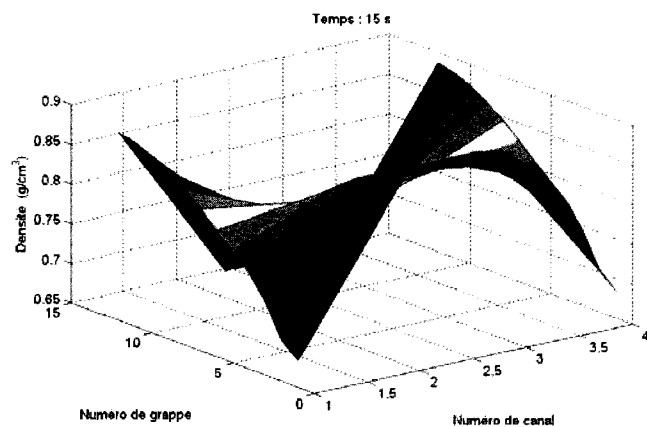


(f)

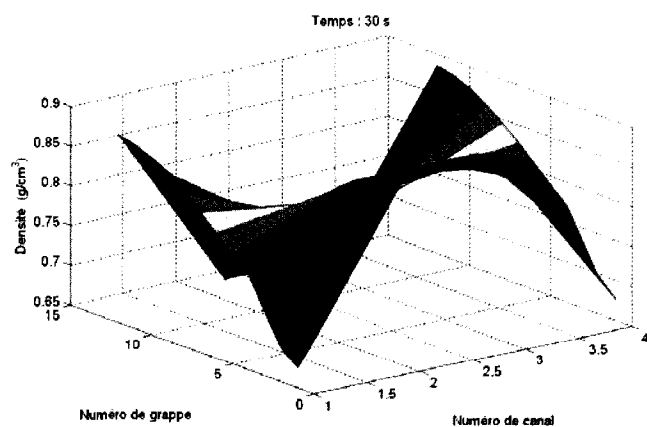
Figure 48. (Suite)



(a)

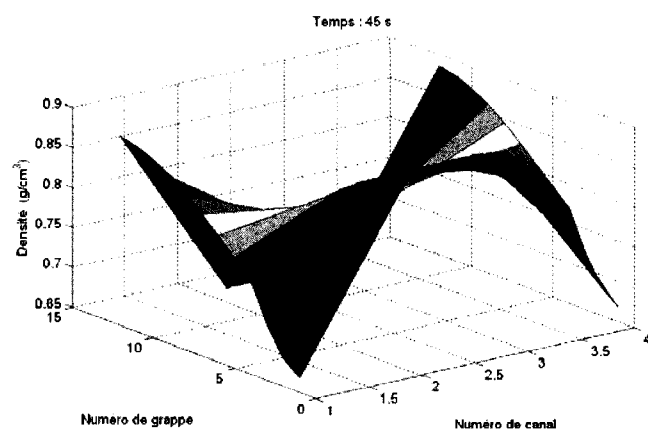


(b)

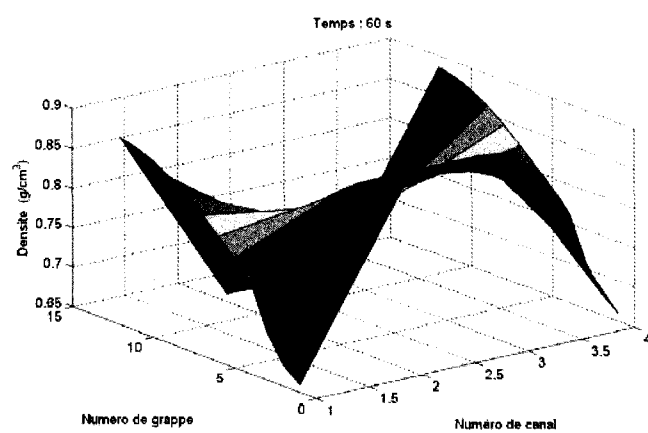


(c)

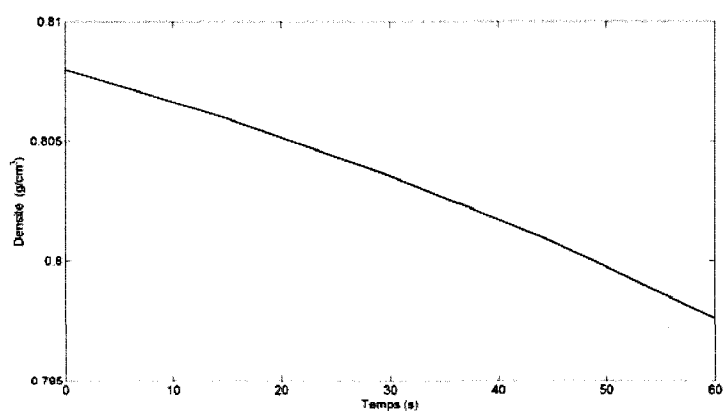
Figure 49 : Densité du caloporteur lors d'une diminution de pression de sortie.



(d)

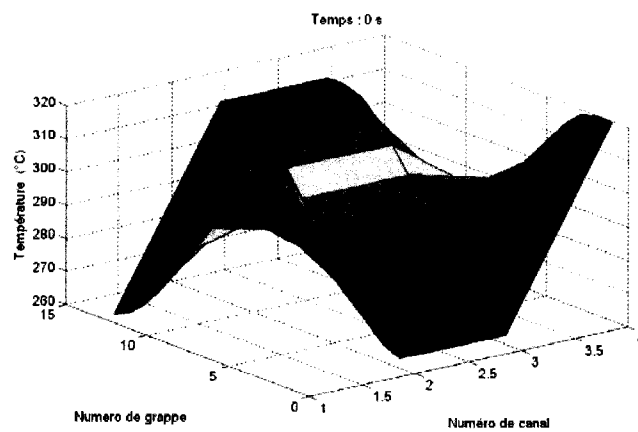


(e)

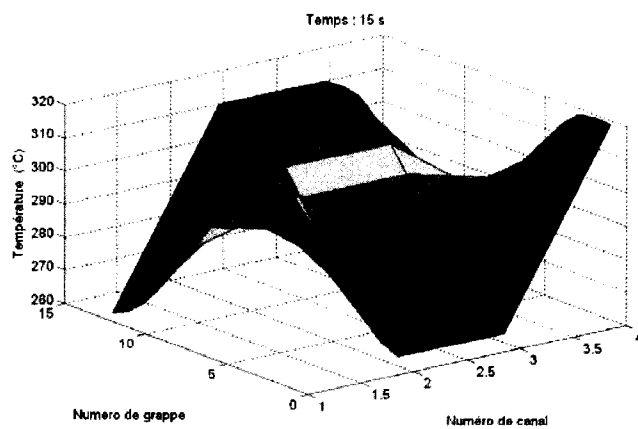


(f)

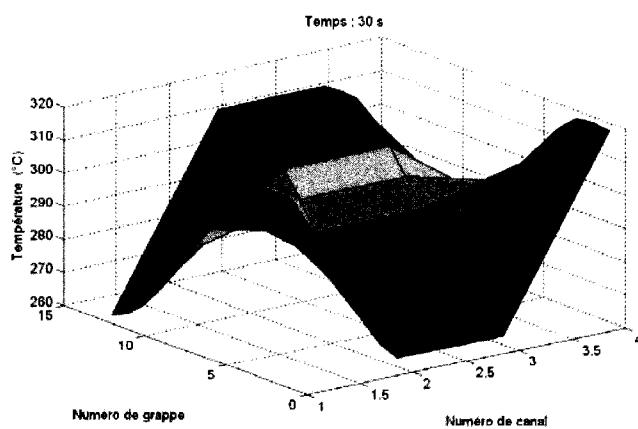
Figure 49. (Suite)



(a)

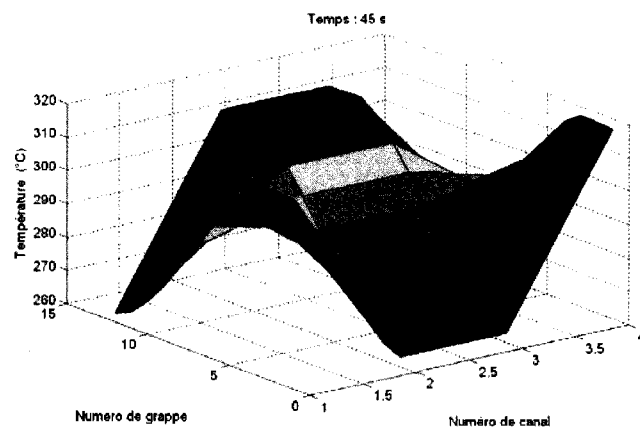


(b)

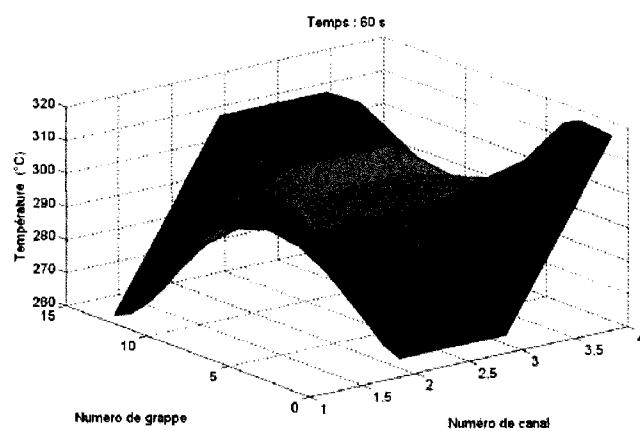


(c)

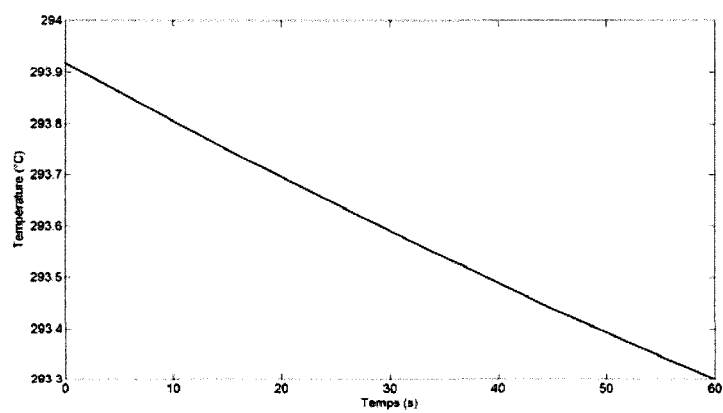
Figure 50 : Température du caloporteur lors d'une diminution de pression de sortie.



(d)

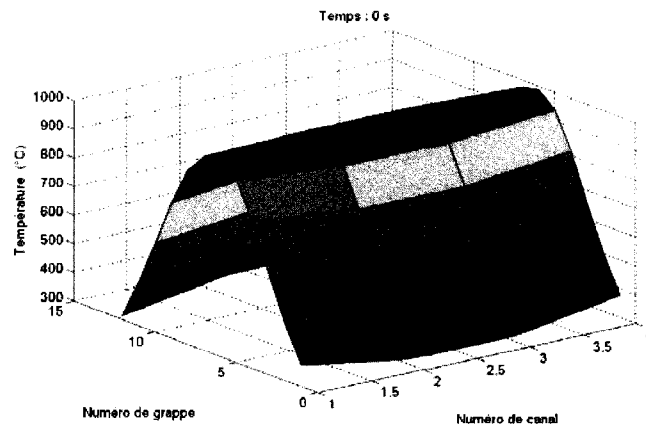


(e)

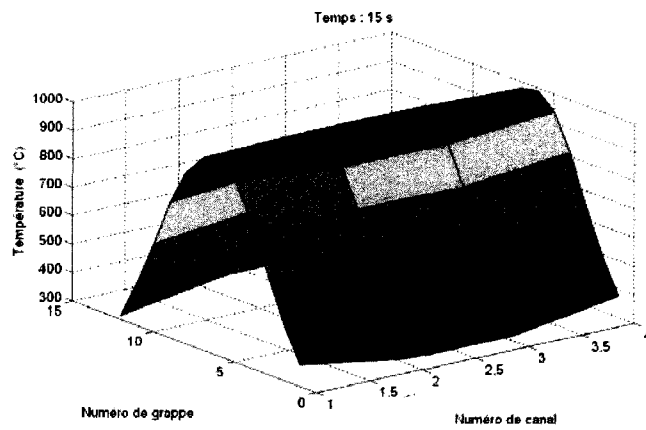


(f)

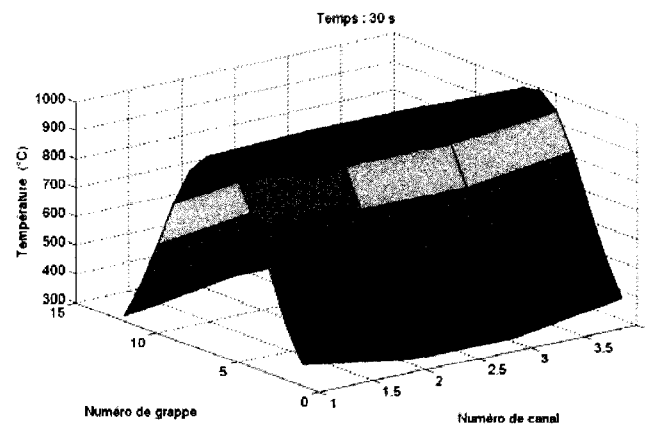
Figure 50. (Suite)



(a)

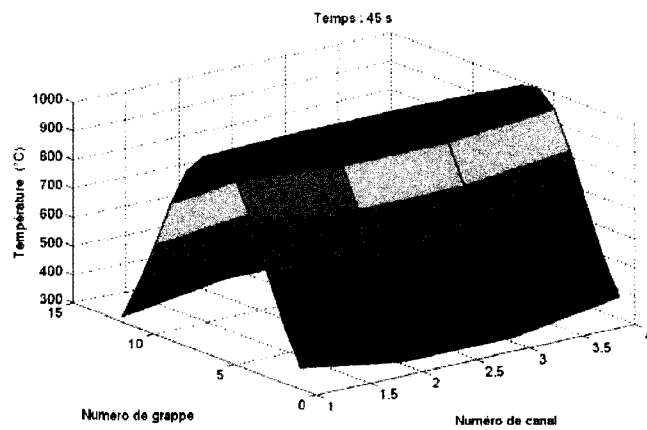


(b)

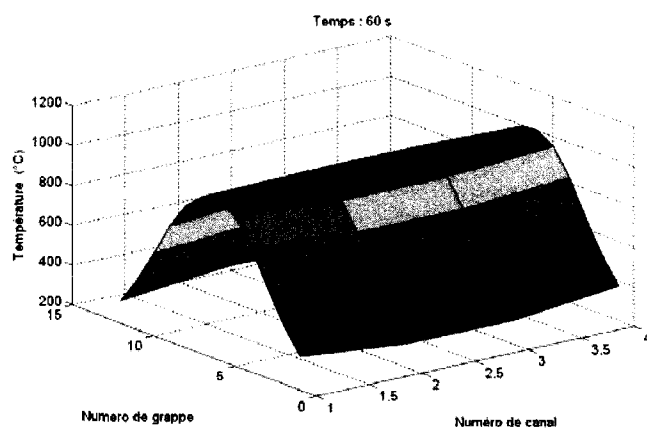


(c)

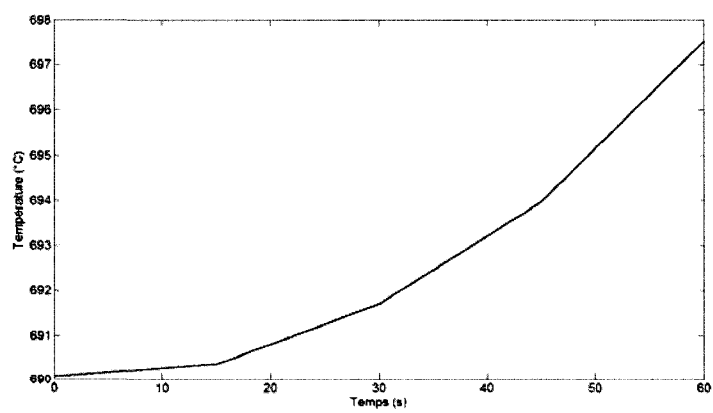
Figure 51 : Température du combustible lors d'une diminution de pression de sortie.



(d)



(e)



(f)

Figure 51. (Suite)

Dans ce cas, nous avons une diminution de pression à la sortie des quatre canaux du réacteur. Celle-ci entraîne une diminution de la température du caloporteur (figure 50f) ainsi que la diminution de la densité moyenne du caloporteur (figure 49f). La chute de la densité du caloporteur provoque une réactivité positive dans le réacteur d'où l'augmentation de la puissance montrée à la figure 48f. L'augmentation de la puissance entraîne une augmentation de la température du combustible (figure 51f). Ainsi, nous obtenons un accroissement accéléré de la puissance durant la transitoire.

Les résultats que nous obtenons ici semblent cohérents. Le modèle FBM est utilisé par le module AFM de DONJON-3. Il est à noter que la base de données utilisée par la corrélation du modèle FBM (Sissaoui *et al.*, 1999) est restreinte à un petit intervalle et que malheureusement il s'est avéré que nos calculs de thermohydraulique fournissent des résultats, notamment pour la température du combustible, en dehors du domaine du modèle (voir tableau 8). En dehors des plages de la corrélation du FBM, nous évaluons les sections efficaces en extrapolant.

Tableau 8 : Plage de validité du FBM, (Sissaoui *et al.*, 1999).

Paramètre local	Plage du modèle
Température du caloporteur (K)	300.7-660.7
Densité du caloporteur (g/cm^3)	0.081-0.812
Température du combustible (K)	541.3-1241.3

Il donc nécessaire d'élargir, dans un futur développement de la partie neutronique, la plage de validité de la base de donnée du modèle FBM et la comparer à une base de données multiparamétrée. Également, il faut noter que le code de neutronique DONJON-3 possède un module de cinétique basé sur une approche quasi-statique améliorée. Dans nos simulations, nous n'avons pas observé d'instabilités de l'approche quasi-statique améliorée. Il serait donc important de vérifier les résultats de ses simulations avec un code de cinétique basé sur des différences finies centrées et totalement implicites dans le temps. Ces dernières sont connues pour leur stabilités. Ainsi, il sera possible d'effectuer l'étude de réacteurs de façon plus sûre.

Par ailleurs, étant donné que nous ne prenons pas en compte les conditions frontières aux collecteurs, nous n'avons pas des conditions frontière réalistes à l'entrée et à la sortie des canaux du réacteur. Dans nos simulations, les débits à l'entrée et les pressions à la sortie des canaux sont fixés. Ceci doit être changé en tenant compte de la perte de charge comme condition aux limites. En effet, cela permettra de prendre en considération les collecteurs et d'avoir une distribution réaliste des débits à l'entrée des canaux mais aussi une pression unique à l'entrée et une autre à la sortie des canaux, dues à la perte de charge.

CONCLUSIONS

L'objectif de ce mémoire a été de développer un code de thermohydraulique pour effectuer des calculs couplés avec un code de neutronique. Le modèle de thermohydraulique ainsi que le modèle de transfert thermique sont développés de manière à permettre le traitement des problèmes transitoires. Le code de thermohydraulique nommé ARTHUR permet donc la simulation d'états transitoires pour des réacteurs CANDU-6 en tenant en compte de la thermohydraulique de l'ensemble des canaux du réacteur.

ARTHUR est composé d'une partie permettant le calcul des états d'écoulements avec le modèle à écart de vitesses dans une géométrie simplifiée des canaux de réacteurs de type CANDU-6. Ce calcul se fait de façon complètement implicite en prenant en compte l'ébullition sous-refroidie. La seconde partie du code ARTHUR traite le transfert de chaleur dans les crayons des réacteurs CANDU-6. Là encore, le calcul est basé sur un schéma implicite prenant en compte différents modes de transfert de chaleur vers l'écoulement dépendamment des conditions de ce dernier. Le code de thermohydraulique a été par la suite couplé avec le code de neutronique DONJON-3 en utilisant la GANLIB dans l'environnement CLE2000. Les différentes parties du code ont été testées indépendamment les unes des autres en comparant les résultats de chacune avec des essais expérimentaux ou des calculs théoriques. À la suite de ces essais de validation, nous avons relevé la nécessité d'améliorer le code de thermohydraulique ainsi que le couplage avec DONJON-3.

Concernant la partie écoulement du code, le modèle à écart de vitesses présente une grave lacune quant à la conservation de l'énergie. Aussi, nous utilisons des corrélations en dehors de leur plage de validité ce qui présente une incertitude sur les valeurs obtenues lors des calculs. Par ailleurs, nous avons négligé la complexité géométrique, l'inversion de flux massique associée à la présence d'obstruction ainsi que le flux de chaleur critique. D'autre part, nous avons les propriétés thermodynamiques et d'échange de l'eau légère plutôt que celles de l'eau lourde. Il

serait important de voir l'impact de la simplification géométrique et d'introduire l'inversion du flux massique dans les écoulements et modifier les propriétés de l'eau légère pour celles de l'eau lourde. Finalement, il serait nécessaire de programmer les conditions frontières liées aux collecteurs notamment la prise en compte de la perte de charge comme condition aux frontières.

Concernant la partie thermique, nous avons considéré les propriétés thermodynamiques et d'échange du crayon indépendamment des conditions mécaniques. Autrement dit, nous n'avons pas tenu compte des effets dus au craquement, au gonflement de la gaine, etc. Ceci nous donne une mauvaise information sur les propriétés thermodynamiques et d'échange effectives du crayon, et nous devons, dans un développement futur du code ARTHUR, prendre en considération ce point. Ceci nous conduirait à un calcul plus réaliste du coefficient de transfert de chaleur entre la gaine et le combustible. Finalement, il faudrait introduire des modèles pour la prédiction du flux de chaleur critique.

En ce qui a trait au couplage neutronique/thermohydraulique, il serait important d'étirer la plage de fonctionnement de la base de données utilisant le module AFM de DONJON-3. En effet, nos calculs ont été faits en dehors de la plage de la corrélation. Par ailleurs, étant donné que les calculs thermohydrauliques des réacteurs sont indépendants d'un canal à l'autre, la parallélisation sur plusieurs processeurs de ces calculs pourrait être un gain de temps important. Finalement, il serait important de vérifier les résultats obtenus à partir d'un code de différences finies centrées et implicite dans le temps pour la cinétique et basé sur des bases de données multiparamétrées pour les sections efficaces. C'est dans cette optique qu'il serait intéressant d'utiliser le code DONJON-4 (Sekki & Hébert, 2007) pour le couplage neutronique/thermohydraulique et sa validation.

BIBLIOGRAPHIE

Ahmad, S. Y. (1970). Axial Distribution of Bulk Temperature and Void Fraction in a Heated Channel with Inlet Subcooling. *92 Ser C* (4), pp. 595-609.

Aube, F. (1996). *Effets du flux de chaleur sur les pertes de pression par frottement dans des écoulements monophasiques et diphasiques*. Thèse de doctorat, École Polytechnique de Montréal, Canada.

Awad, M. M., & Muzychka, Y. S. (2004). A Simple Two-Phase Frictional Multiplier Calculation Method. *Calgary, Alta., Canada* (Vol. 1, pp. 475-483) American Society of Mechanical Engineers, New York, NY, United States.

Baigorria, C., Harriague, S., & Lassmann, K. (1982). Computational Experience Showing the Influence of the Fuel-To-Sheath Thermal Conductance in Reactor Fuel Temperature Predictions. *Nuclear Engineering and Design*, 73 (1), pp. 95-99.

Bailly, H., Ménessier, D., & Prunier, C. (1999). *The Nuclear Fuel of Pressurized Water Reactors and Fast Neutron Reactors: Design and Behaviour*. Andover, U.K.

Bandel, J., & Schluender, E. U. (1986). Pressure Drop and Heat Transfer by Vaporization of Boiling Refrigerants in a Horizontal Pipe. In (pp. 365-387), Hemisphere Publ Corp, Washington, DC, USA.

Baroczy, C. J. (1966). Systematic Correlation for Two-Phase Pressure Drop. *Chemical Engineering Progress Symposium Series*, 62 (64), pp. 232-249.

Boelter, L. M. K., Young, G., & Iversen, H. W. (1948). *Investigation of Aircraft Heaters. XXVII - Distribution of Heat-Transfer Rate in Entrance Section of Circular Tube*, National Advisory Committee For Aeronautics, Washington, DC, United

States.

Campbell, F. R., Bourque, L. R., Deshaies, R., Sills, R., & Notley, M. J. F. (1977). *In-Reactor Measurement of Fuel-To-Sheath Heat Transfer Coefficients between UO₂ and Stainless Steel* (Copyright 1978, IEE AECL-5400), Atomic Energy Canada Ltd., Chalk River, Ont., Canada.

Carey, J. J., & Valentin, R. A. (1971). On Thermal Stresses in Clad, Pellet Stacks and the Problem of Interface Stress States. *Berlin, West Germany*, Bundesanstalt für Materialprüfung.

Chen, J. C. (1966). Correlation for Boiling Heat Transfer to Saturated Fluids in Convective Flow. *Industrial and Engineering Chemistry - Process Design and Development*, 5 (3), pp. 322-329.

Cheon, J.-S., Lee, B.-H., Koo, Y.-H., Oh, J.-Y., & Sohn, D.-S. (2004). Evaluation of a Pellet-Clad Mechanical Interaction in Mixed Oxide Fuels During Power Transients by Using Axisymmetric Finite Element Modeling. *Nuclear Engineering and Design*, 231 (1), pp. 39-50.

Chexal, B., Lellouche, G., Horowitz, J., & Healy, J. (1992). A Void Fraction Correlation for Generalized Applications. *Progress in Nuclear Energy*, 27(4), pp. 255-295.

Coddington, P., & Macian, R. (2002). A Study of the Performance of Void Fraction Correlations Used in the Context of Drift-Flux Two-Phase Flow Models. *Nuclear Engineering and Design*, 215 (3), pp. 199-216.

Collado, F. J. (2000). Criticism of the Classical Mass Balance for Multiphase Flow. *Madrid, Spain* (Vol. 3, pp. 257-266) Witley & Sons Press, Southampton, United Kingdom.

Collado, F. J. (2001). Mass Quality, Void Fraction, and Slip Ratio in Bulk Flow Boiling. *New York, NY, United States* (Vol. 369, pp. 143-151) American Society of Mechanical Engineers, New York, NY, United States.

Collado, F. J., Monne, C., Pascau, A., Fuster, D., & Medrano, A. (2006a). Thermodynamics of Void Fraction in Saturated Flow Boiling. *Journal of Heat Transfer*, 128 (6), pp. 611-615.

Collado, F. J., Monne, C., & Pascau, A. (2006b). Changes of Enthalpy Slope in Subcooled Flow Boiling. *Heat and Mass Transfer*, 42 (5), pp. 437-448.

Collado, F. J. (2007a). Reynolds Transport Theorem for a Two-Phase Flow. *Applied Physics Letters*, 90 (2), pp. 24101-24101.

Collado, F. J., Monne, C., & Pascau, A. (2007b). A New Heat Balance for Flow Boiling. *AIChE Journal*, 53 (8), pp. 2123-2130.

Delhaye, J. M., Maugin, F., & Ochterbeck, J. M. (2004). Void Fraction Predictions in Forced Convective Subcooled Boiling of Water between 10 And 18 Mpa. *International Journal of Heat and Mass Transfer*, 47 (19-20), pp. 4415-4425.

Dionne, B. (2001). *Couplage neutronique-thermohydraulique pour l'analyse de réacteurs CANDU à l'état stationnaire*. M.Sc.A., École Polytechnique de Montréal, Canada.

Dionne, B., Koclas, J., & Teyssedou, A. (2001). Parallel Coupling of Thermal-Hydraulics and Reactor Physics for Applications in a CANDU-6. Canadian Nuclear Society, Toronto, ON, Canada.

Dittus, F. W. L. and Boelter, M. k. (1930). Heat Transfer in Automobile Radiators of the Tubular Type, *Univ. Calif. Publ. Eng.*, 2 (13), pp. 443-446.

Doster, J. M. (1989). Numerical Solutions of Multiphase Flow Problems. *Tampa, FL, USA* (Vol. 21, pp. 59-66) Published by Society for Computer Simulation Int., San Diego, CA, USA.

Fenech, H. (1986). Thermal Conductance at Solid Interfaces: An Application of the Fenech-Rohsenow Model. *International Journal of Heat and Mass Transfer*, 29 (8), pp. 1103-1108.

Fink, J. K. (2000). Thermophysical Properties of Uranium Dioxide. *Journal of Nuclear Materials*, 279 (1), pp. 1-18.

Forster, H. K., & Zuber, N. (1955). Dynamics of Vapor Bubbles and Boiling Heat Transfer. *Chemical Engineering Progress*, 1 (4), pp. 531-535.

Friedel, L. (1980). Pressure Drop during Gas/Vapor-Liquid Flow in Pipes. *International Chemical Engineering*, 20 (3), pp. 352-367.

Ha, K. S., Lee, Y. B., & No, H. C. (2005). Improvements in Predicting Void Fraction in Subcooled Boiling. *Nuclear Technology*, 150 (3), pp. 283-292.

Han, G. Y., & Stanley, T. P. (1999). Thermal-Hydraulic Transient Analysis of Light Water Reactors Using Moving Boundaries. *Annals of Nuclear Energy*, 26 (4), pp. 301-326.

Han, G. Y., Stanley, T. P., & Secker, P. A., Jr. (1999). Thermal-Hydraulic Modeling and Transient Analysis of Pressurized Water Reactors. *International Communications in Heat and Mass Transfer*, 26 (7), pp. 909-918.

Hanna, B. N. (1998). CATHENA: A Thermalhydraulic Code for CANDU Analysis. *Nuclear Engineering and Design*, 180 (2), pp. 113-131.

Hassan, Y. A., & Salim, P. (1990). Analysis of a Nuclear Power Plant Using

RELAP5/MOD2 with Modified Bundle Heat Transfer Correlations. *Nuclear Technology*, 92 (1), pp. 141-149.

Hibiki, T., & Ishii, M. (2003). One-Dimensional Drift-Flux Model and Constitutive Equations for Relative Motion between Phases in Various Two-Phase Flow Regimes. *International Journal of Heat and Mass Transfer*, 46 (25), pp. 4935-4948.

Hibiki, T., & Ishii, M. (2005). Erratum: One-Dimensional Drift-Flux Model and Constitutive Equations for Relative Motion between Phases in Various Two-Phase Flow Regimes. *International Journal of Heat and Mass Transfer*, 48 (6), pp. 1222-1223.

Hibiki, T., Takamasa, T., Ishii, M., & Gabriel, K. S. (2006). One-Dimensional Drift-Flux Model at Reduced Gravity Conditions. *AIJA Journal*, 44 (7), pp. 1635-1642.

Hoeld, A. (2007). Coolant Channel Module an Universally Applicable Thermal-Hydraulic Drift-Flux Based Mixture-Fluid 1D Model and Code. *Nuclear Engineering and Design*, 237 (15-17 Spec. ISS), pp. 1952-1967.

Hoffman, J. D. (2001). *Numerical Methods for Engineers and Scientists* (2nd^e Éd.). New York, Marcel Dekker.

Hsu, Y. Y., & Massoud, M. (1986). New Treatment of Transition Boiling. *Heat and Technology*, 4 (3-4), pp. 7-31.

Huang, L., & Witte, L. C. (1994). Influences of Subcooling and Velocity on Boiling Heat Transfer across Horizontal Cylinders. *Colorado Springs, CO, USA* (Vol. 273, pp. 65-71) ASME, New York, USA.

Hudina, M., & Markoczy, G. (1977). Hexagonal Bundle Heat Transfer and Fluid Flow Experiment Agathe Hex. *Nuclear Engineering and Design*, 40 (1), pp. 121-131.

Huo, X., Xie, Z., & Liao, C. (2004). Development of Core Fuel Management Code System for Candu Reactor Based on Coupled Neutronics and Thermal-Hydraulic Advanced Nodal Method. *Annals of Nuclear Energy*, 31 (10), pp. 1083-1100.

IAEA. (2006). *Thermophysical Properties Database of Materials for Light Water Reactors and Heavy Water Reactors* (Iaea-Tecdoc-1496). Austria, IAEA.

In, W. K., Oh, D. S., & Chun, T. H. (2002). Empirical and Computational Pressure Drop Correlations for Pressurized Water Reactor Fuel Spacer Grids. *Nuclear Technology*, 139 (1), 72-79.

Ishii, M., & Hibiki, T. (2006). *Thermo-Fluid Dynamics of Two-Phase Flow*. New York, Springer.

Ivanov, K., & Avramova, M. (2007). Challenges in Coupled Thermal-Hydraulics and Neutronics Simulations for LWR Safety Analysis. *Annals of Nuclear Energy*, 34 (6), pp. 501-513.

Jacobs, G., & Todreas, N. E. (1973). Thermal Contact Conductance in Reactor Fuel Elements. *Nuclear Science and Engineering*, 50, pp. 283-306.

Jaluria, Y., & Torrance, K. E. (2003). *Computational Heat Transfer* (2nd^e Éd.). New York, Taylor & Francis.

Jewer, S., Thompson, A., Hoeld, A., & Beeley, P. A. (2006). Initial Version of an Integrated Thermal Hydraulics and Neutron Kinetics 3D Code X3D. *Nuclear Engineering And Design*, 236 (14-16), pp. 1533-1546.

Jiyang, Y., Songtao, W., & Baoshan, J. (2007). Development of Sub-Channel Analysis Code for CANDU-SCWR. *Progress in Nuclear Energy*, 49 (4), pp. 334-350.

Kaempfer, H., & Karsten, G. (1970). Effects of Different Types of Void Volumes on the Radial Temperature Distribution of Fuel Pins. *9* (3), pp. 288-300.

Kim, D., & Ghajar, A. J. (2002). Heat Transfer Measurements and Correlations for Air-Water Flow of Different Flow Patterns in a Horizontal Pipe. *Experimental Thermal and Fluid Science*, *25* (8), pp. 659-676.

Kolev, N. I. (2002). *Multiphase Flow Dynamics. 1, Fundamentals*. Berlin, Springer.

Kolev, N. I. (2002). *Multiphase Flow Dynamics. 2, Thermal and Mechanical Interactions*. Berlin, Springer.

Kroeger, P. G., & Zuber, N. (1968). Analysis of Effects of Various Parameters on Average Void Fractions in Subcooled Boiling. *International Journal of Heat and Mass Transfer*, *11* (2), pp. 211-233.

Kumar, R. K., & Kroeger, V. D. (1996). Contact Heat Transfer and Thermal Contact Conductance between Nonconforming Surfaces in an Abrupt Contact. *New Orleans, LA, USA* (Vol. 1, pp. 647-656) ASME, New York, NY, USA.

Kumbaro, A., Toumi, I., & Seignole, V. (2002). Numerical Modeling of Two-Phase Flows using Advanced Two Fluid Systems. *Arlington, VA, United States* (Vol. 3, pp. 533-540) American Society of Mechanical Engineers.

Kureta, M., Hibiki, T., Mishima, K., & Akimoto, H. (2003). Study on Point of Net Vapor Generation by Neutron Radiography in Subcooled Boiling Flow along Narrow Rectangular Channels with Short Heated Length. *International Journal of Heat and Mass Transfer*, *46* (7), pp. 1171-1181.

Kuznetsov, V. (1979). The Temperature Distribution in a Fuel Pin and Sheath with Radiative Heat Transfer. *Atomnaya Énergiya*, *47*(6), pp. 410-411.

Lassmann, K., & Hohlefeld, F. (1987). Revised Urgap Model to Describe the Gap Conductance between Fuel and Cladding. *Nuclear Engineering and Design*, 103 (2), pp. 215-221.

Lassmann, K., & Pazdera, F. (1983). URGAP, a Gap Conductance Model for Transient Conditions. *Preston, S. AFR.* (pp. 97-113) Applied Science Publication, London, England.

Lebon, G., & Mathieu, P. (1979). Numerical Calculation of Nonlinear Transient Heat Conduction in the Fuel Elements of a Nuclear Reactor. *International Journal of Heat and Mass Transfer*, 22 (8), pp. 1187-1198.

Lee, K. (1995). Evaluation of Gap Heat Transfer Model in ELESTRES for CANDU Fuel Element under Normal Operating Conditions. *Journal of the Korean Nuclear Society*, 27, pp. 344-357.

Lee, K. M., Ohn, M. Y., Lim, H. S., Choi, J. H., & Hwang, S. T. (1995). Study on Models for Gap Conductance between Fuel and Sheath for CANDU Reactors. *Annals of Nuclear Energy*, 22 (9), pp. 601.

Lemczyk, T. F., & Yovanovich, M. M. (1987). New Models and Methodology for Predicting Thermal Contact Resistance in Compound Cylinders and Finned Tubes. *Heat Transfer Engineering*, 8 (2), pp. 35-48.

Leung, L. K. H., Groeneveld, D. C., Teyssedou, A., & Aube, F. (2005). Pressure Drops for Steam and Water Flow in Heated Tubes. *Nuclear Engineering and Design*, 235 (1), pp. 53-65.

Levy, S. (1967). Forced Convection Subcooled Boiling Prediction of Vapor Volumetric Fraction. *International Journal of Heat and Mass Transfer*, 10, pp. 951-965.

Liu, Q. S., Shiotsu, M., & Sakurai, A. (1992). Correlation for Forced Convection Film Boiling Heat Transfer from a Horizontal Cylinder under Subcooled Conditions. *Anaheim, CA, USA* (Vol. 217, pp. 21-32) Published by ASME, New York, NY, USA.

Lombardi, C., & Ceresa, I. (1978). Generalized Pressure Drop Correlation in Two-Phase Flow. *Energia Nucleare (Milan)*, 25 (4), pp. 181-198.

Lucas, S. (1988). Finite Element Applications of Two Phase Flow. *Chicago, IL, USA* (Vol. 71, pp. 31-33) Published by American Society of Mechanical Engineers (ASME), New York, NY, USA.

Mandhane, J. M., Gregory, G. A., & Aziz, K. (1974). A Flow Pattern Map for Gas-Liquid Flow in Horizontal Pipes. *International Journal of Multiphase Flow*, 1, pp. 537-553.

Marek, J., Maubach, K., & Rehme, K. (1973). Heat Transfer and Pressure Drop Performance of Rod Bundles Arranged in Square Arrays. *International Journal of Heat and Mass Transfer*, 16 (12), pp. 2215-2228.

Markoczy, G. (1972). Convective Heat Transfer in Rod Clusters with Turbulent Axial Coolant Flow, Part I: Mean Values over the Rod Perimeter. *Wärme Und Stoffübertragung*, 5, pp. 204-212.

Marleau, G., & Varin, E. (2004). *An History Interface between DRAGON And DONJON-3: The HST: Module*. Canada, École Polytechnique de Montréal.

Martinelli, R. C. (1947). Heat Transfer to Molten Metals. *American Society of Mechanical Engineers -- Transactions*, 69 (8), pp. 947-956.

Massoud, M. (2005). *Engineering Thermofluids: Thermodynamics, Fluid Mechanics*,

and Heat Transfer. Berlin, Springer.

Mesmous, N. (2005). *Modelisation thermohydraulique et simulation numerique d'un canal de combustible des reacteurs nucleaires CANDU*. M.Sc.A., École Polytechnique de Montréal, Canada.

Müeller-Steinhagen, H., & Heck, K. (1986). Simple Friction Pressure Drop Correlation for Two-Phase Flow in Pipes. *Chemical Engineering and Processing*, 20 (6), pp. 297-308.

Orenzini, E. (1976). Temperature Distributions in Cylindrical Fuel Pins with Gap and Cladding when Coolant Temperature is Variable. *Tecnica Italiana*, 46, pp. 299-306.

Park, J.-W. (1999). Subchannel Analysis of Dupic Fuel Bundle for the CANDU Reactor. *Annals of Nuclear Energy*, 26 (1), pp. 29-46.

Peggs, I. D., Stadnyk, A. M., & Godin, D. P. (1976). Thermophysical Properties of Zirconium-Alloy Fuel-Channel Components. *High Temperatures - High Pressures*, 8 (4), pp. 441-450.

Peterson, C. E., Shatford, J. G., Harrison, J. F., & Agee, L. J. (2003). Evaluation of the RETRAN-3D Wall Friction Models and Heat Transfer Coefficient Correlations. *Nuclear Technology*, 142 (1), pp. 64-76.

Petropoulos, N. P. (2000). *Thermophysical Properties of Nuclear Power Reactor Coolants: Comparative Study and Calculation Codes*. Ph.D. Thesis, National Technical University Of Athens, Greece.

Reddy, D. G., Fighetti, C. F., & Merilo, M. (1983). Evaluation of Two-Phase Pressure Drop Correlations for High Pressure Steam-Water Systems. *Honolulu, HI, USA* (Vol. 1, pp. 251-259) ASME, New York, NY, Usa.

Rehme, K. (1972). Pressure Drop Performance of Rod Bundles in Hexagonal Arrangements. *International Journal of Heat and Mass Transfer*, 15, pp. 2499-2517.

Rehme, K. (1973). Pressure Drop Correlations For Fuel Element Spacers. *Nuclear Technology*, 17 (1), pp. 15-23.

Rehme, K. (1973). Simple Method of Predicting Friction Factors of Turbulent Flow in Noncircular Channels. *International Journal of Heat and Mass Transfer*, 16 (5), pp. 933-950.

Roache, P. J. (1985). *Computational Fluid Dynamics*. Albuquerque, N.M., Hermosa Publishers.

Ross, A. M., & Stoute, R. L. (1962). *Heat Transfer Coefficient between UO₂ and Zircaloy-2*. Chalk River (ON), Canada, AECL.

Rouben, B., Brunner, K. S., & Jenkins, D. A. (1988). Calculation of Three-Dimensional Flux Distributions in Candu Reactors Using Lattice Properties Dependent on Several Local Parameters. *Saratoga Springs, NY, USA* (Vol. 98, pp. 139-148).

Rouhani, S. Z., & Axelsson, E. (1969). Calculation of Void Volume Fraction on the Subcooled and Quality Boiling Regions. *International Journal of Heat and Mass Transfer*, 13, pp. 383-393.

Roy, R. (1999). *The Cle-2000 Tool-Box* (IGE-163). Canada, École Polytechnique de Montréal.

Roy, R., & Hébert, A. (2000). *The GAN Generalized Driver* (IGE-158). Canada, École Polytechnique de Montréal.

Rozon, D. (1992). *Introduction à la cinétique des réacteurs nucléaires*. Montréal, Éditions de L'École Polytechnique de Montréal.

Saha, P., & Zuber, N. (1974). Point Of Net Vapor Generation and Vapor Void Fraction in Subcooled Boiling. *Muanyag Es Gumi*, pp. 175-179.

Sarma, N. V. L. S., & Bhaskar Rao, P. (1979). Analytical Method for Predicting the Temperature Distribution in an Irradiated Fuel Pin Bundle. *Atomkernenergie/Kerntechnik*, 34 (1), pp. 16-19.

Sekki, D. & Hébert, A. (2007). *A User Guide for DONJON Version 4* (IGE-300). Canada, École Polytechnique de Montréal.

Sermer, P. (1986). Analytical Model for Heat Transfer within a Candu Fuel Bundle Residing In Air. *Chalk River, Ont, Can* (pp. 428-443) Canadian Nuclear Society, Toronto, ON, Canada.

Sha, W. T. (1980). An Overview on Rod-Bundle Thermalhydraulic Analysis. *Nuclear Engineering and Design*, 62, pp. 1-24.

Shih-Kuei, C., & Todreas, N. E. (1986). Hydrodynamic Models and Correlations for Bare and Wire-Wrapped Hexagonal Rod Bundles-Bundle Friction Factors, Subchannel Friction Factors and Mixing Parameters. *Nuclear Engineering and Design*, 92 (2), pp. 227-251.

Sissaoui, M. T., & Marleau, G. (1995). Application of the Feedback Model for the History Base Using DRAGON. *19th Annual Conference of the CNS*, Canadian Nuclear Society, Hamilton, ON., Canada.

Sissaoui, M. T., Marleau, G., & Rozon, D. (1999). Candu Reactor Simulations Using the Feedback Model with Actinide Burnup History. *Nuclear Technology*, 125 (2), pp. 197-212.

Song, S., Yovanovich, M. M., & Nho, K. (1992). Thermal Gap Conductance. Effects of Gas Pressure and Mechanical Load. *Journal of Thermophysics and Heat Transfer*, 6 (1), pp. 62-68.

Song, S., & Yovanovitch, M. M. (1988). Relative Contact Pressure: Dependence on Surface Roughness and Vickers Microhardness. *Journal of Thermophysics*, 2 (1), pp. 43-47.

Stacey, W. M. (2001). *Nuclear Reactor Physics*. New York; Toronto, Wiley.

Staub, F. W. (1967). Void Fraction in Subcooled Boiling -- Prediction of Initial Point of Net Vapor Generation. (pp. 7) American Society of Mechanical Engineers (ASME), New York, NY, United States.

Szpunar, B., Lewis, B. J., Arimescu, V. I., Dickson, R. S., & Dickson, L. W. (2001). Multi-Component Gas Transport in the Fuel-To-Clad Gap of Candu Fuel Rods during Severe Accidents. *Journal of Nuclear Materials*, 294 (3), pp. 315-329.

Taitel, Y., & Dukler, A. E. (1975). Model for Predicting Flow Regime Transitions in Horizontal and Near Horizontal Gas-Liquid Flow. *American Society Of Mechanical Engineers (Paper)(75-Wa/Ht-29)*, pp. 9.

Tamai, H., Kureta, M., Ohnuki, A., Sato, T., & Akimoto, H. (2006). Pressure Drop Experiments Using Tight-Lattice 37-Rod Bundles. *Journal of Nuclear Science and Technology*, 43 (6), pp. 699-706.

Teyssedou, A., Aubé, F., & Champagne, P. (1992). Void Fraction Measurement System for High Temperature Flows. *Measurement Science and Technology*, 3, pp. 485-494.

Thome, J. R., Ould Didi, M. B., & Kattan, N. (2002). Prediction of Two-Phase

Pressure Gradients of Refrigerants in Horizontal Tubes. *International Journal of Refrigeration*, 25 (7), pp. 935-947.

Todreas, N. E., & Kazimi, M. S. (1990). *Nuclear Systems*. New York, Hemisphere Pub. Corp.

Tong, L.-S., & Tang, Y. S. (1997). *Boiling Heat Transfer and Two-Phase Flow* (2nd^e Éd.). Washington, D.C., Taylor & Francis.

Tribbe, C., & Muller-Steinhagen, H. M. (2000). Evaluation of the Performance of Phenomenological Models for Predicting Pressure Gradient during Gas-Liquid Flow in Horizontal Pipelines. *International Journal of Multiphase Flow*, 26 (6), pp. 1019-1036.

Tye, P., Teyssedou, A., & Troche, N. (1995). *A Review of Void Quality Relationships Applicability to Assert* (IGE-202). Canada, École Polytechnique De Montréal.

Ulmann, A. (1974). Thermal Accommodation Coefficients of Inert Gases on Stainless and UO₂. *Journal of Nuclear Materials*, 51, pp. 277-279.

Varin, E., & Hébert, A. (2003). *Data Structures of DONJON-3* (IGE-226). Canada, École Polytechnique de Montréal.

Varin, E., Hébert, A., Roy, R., & Koclas, J. (2005). *A User Guide for DONJON-3* (IGE-208). Canada, École Polytechnique de Montréal.

Varin, E., & Marleau, G. (2005). Validating the History-Based Method in DRAGON/DONJON Using G2 Core Follow-Up. (pp. 939-951) Canadian Nuclear Society, Toronto, ON, Canada.

Varin, E., & Marleau, G. (2006). Candu Reactor Core Simulations Using Fully Coupled DRAGON and DONJON Calculations. *Annals of Nuclear Energy*, 33 (8),

pp. 682-691.

Varin, E., Roy, R., Baril, R., & Hotte, G. (2004). Candu-6 Operation Post-Simulations Using the Reactor Physics Codes DRAGON/DONJON. *Annals of Nuclear Energy*, 31 (18), pp. 2139-2155.

Vijayan, P. K., Pilkhwal, D. S., Saha, D., & Raj, V. V. (1999). Experimental Studies on the Pressure Drop across the Various Components of a PHWR Fuel Channel. *Experimental Thermal and Fluid Science*, 20 (1), pp. 34-44.

Wagner, W., Cooper, J. R., Dittmann, A., Kijima, J., Kretschmar, H. J., Kruse, A., et al. (2000). The IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam. *Journal of Engineering for Gas Turbines and Power*, 122 (1), pp. 150-180.

Wahid, S. M. S., & Madhusudana, C. V. (2000). Gap Conductance In Contact Heat Transfer. *International Journal Of Heat And Mass Transfer*, 43 (24), pp. 4483-4487.

Wang, G., Byers, W. A., Karoutas, Z. E., Hochreiter, L. E., Young, M. Y., & Jacko, R. J. (2006). Single Rod Heat Transfer Tests to Study the Effects of Crud Deposition. *Miami, Fl, United States* (Vol. 2006, pp. 8) American Society Of Mechanical Engineers, New York, NY, United States.

Wesley, D. A., & Yovanovich, M. M. (1986). New Gaseous Gap Conductance Relationship. *Nuclear Technology*, 72 (1), pp. 70-74.

Wright, S. A., & Houts, M. (2001). Coupled Reactor Kinetics and Heat Transfer Model For Heat Pipe Cooled Reactor. *Aip* (Vol. 552, pp. 815-821).

Yu, S. D., & Xu, S. (2002). Modeling Of Three-Dimensional Steady State Non-Linear Heat Transfer In Candu Nuclear Fuel. *Nuclear Engineering and Design*, 216 (1-3), pp. 165-181.

Zuber, N., Staub, F. W., & Bijwaard, G. (1966). Vapor void fraction in subcooled boiling and in saturated boiling systems. *International Heat Transfer Conference*, Chicago, IL, United States (Vol. 5, pp. 24-38).

Zuber, N., & Findlay, J. A. (1965). Average Volumetric Concentration in Two-Phase Flow Systems. *Journal Of Heat Transfer*, 87, pp. 453-468.

ANNEXES

ANNEXE 1 : Expériences d'écoulements.

Diamètre du tube : 13.4 *mm*

Expérience 19m :

Pression de sortie: 4.224 *MPa*

Température d'entrée: 372.45 *K*

Flux massique: 5243 *kg/m²/s*

Flux de chaleur: 0 *W/m²*

Position (<i>m</i>)	Perte de pression (<i>kPa</i>)
0.09	56.0
0.18	52.8
0.27	50.4
0.45	45.7
0.63	41.3
0.81	36.
0.99	30.9
1.17	26.9
1.35	22.7
1.53	17.9
1.71	12.8
1.89	8.9
2.07	3.7
2.16	0.0

Expérience 37m :

Pression de sortie: 4.2085 *MPa*

Température d'entrée: 372.55 *K*

Flux massique: $7119 \text{ kg/m}^2/\text{s}$

Flux de chaleur: 0 W/m^2

Position (m)	Perte de pression (kPa)
0.09	81.5
0.18	77.1
0.27	72.8
0.45	67.6
0.63	60.0
0.81	51.8
0.99	45.0
1.17	39.9
1.35	33.2
1.53	25.3
1.71	17.9
1.89	12.2
2.07	5.2
2.16	0.0

Expérience 38m :

Pression de sortie: 4.212 MPa

Température d'entrée: 372.45 K

Flux massique: $7119 \text{ kg/m}^2/\text{s}$

Flux de chaleur: 3048 kW/m^2

Position (m)	Perte de pression (kPa)
0.09	77.9
0.18	74.1
0.27	69.0
0.45	62.2
0.63	56.4

0.81	49.2
0.99	43.1
1.17	36.7
1.35	31.3
1.53	24.2
1.71	16.8
1.89	12.4
2.07	5.3
2.16	0.0

Expérience 23 :

Pression de sortie: 1.9824 MPa

Température d'entrée: 477.67 K

Flux massique: 4070 kg/m²/s

Flux de chaleur: 1035 kW/m²

Position (m)	Taux de vide	Position (m)	Perte de pression (kPa)
0.420	0.040	0.09	157.6
0.495	0.127	0.18	154.4
0.585	0.167	0.27	152.8
0.675	0.164	0.45	146.3
0.765	0.268	0.63	137.6
0.855	0.375	0.81	126.9
0.945	0.460	0.99	117.5
1.035	0.575	1.17	105.6
1.125	0.614	1.35	91.3
1.215	0.600	1.53	74.1
1.305	0.649	1.71	53.8
1.395	0.646	1.89	32.1
1.485	0.690	2.07	8.5

1.575	0.670	2.16	0.0
1.665	0.641		
1.755	0.642		
1.845	0.698		
1.950	0.719		

Expérience 31 :

Pression de sortie: 1.365 MPa

Température d'entrée: 454.15 K

Flux massique: 9923 kg/m²/s

Flux de chaleur: 2008 kW/m²

Position (m)	Taux de vide	Position (m)	Perte de pression (kPa)
0.495	0.024	0.09	424.5
0.565	-0.020	0.18	420.6
0.720	-0.085	0.27	415.2
0.755	-0.050	0.45	414.2
0.855	-0.105	0.63	392.8
0.945	-0.021	0.81	380.6
1.035	0.007	0.99	370.5
1.125	0.008	1.17	358.2
1.215	-0.025	1.35	343.4
1.305	0.073	1.53	321.1
1.395	0.071	1.71	284.5
1.440	0.097	1.89	216.8
1.575	0.168	2.07	98.3
1.665	0.325	2.16	0.0
1.760	0.323		
1.800	0.391		

Expérience 44v :

Pression de sortie: 1.61 MPa

Température d'entrée: 435.95 K

Flux massique: 4523 kg/m²/sFlux de chaleur: 3312 kW/m²

Position (m)	Taux de vide	Position (m)	Perte de pression (kPa)
0.495	0.051	0.09	598.9
0.565	0.137	0.18	594.0
0.720	0.306	0.27	590.0
0.755	0.337	0.45	582.2
0.850	0.391	0.63	570.5
0.945	0.483	0.81	549.9
1.035	0.591	0.99	529.9
1.125	0.684	1.17	494.4
1.215	0.736	1.35	430.7
1.305	0.779	1.53	355.1
1.395	0.898	1.71	277.9
1.440	0.898	1.89	177.7
1.575	0.857	2.07	53.4
1.760	0.899	2.16	0.0
1.800	0.865		

ANNEXE 2 : Solution analytique en thermique.

$$T(r) = -\frac{q'''}{4k_c}r^2 + A\ln(r) + B, 0 \leq r \leq r_c$$

$$T(r) = C\ln(r) + D, r_g \leq r \leq r_s$$

$$\left. \frac{\partial T}{\partial r} \right|_{r=0} = 0$$

$$k_g \left. \frac{\partial T}{\partial r} \right|_{r=r_g} = h_{gap} (T_c - T_g)$$

$$q'' = -k_g \left. \frac{\partial T}{\partial r} \right|_{r=r_s} = q''' \frac{\pi r_c^2}{2\pi r_s \Delta z}$$

$$T_s = T_\infty$$

Densité de puissance (GW/m^3)	0.3
Rayon du combustible (mm)	5.6
Rayon interne de la gaine (mm)	6.14
Rayon externe de la gaine (mm)	6.52
Conductivité thermique du combustible ($W/m/K$)	5.0
Conductivité thermique de la gaine ($W/m/K$)	10.0
Coefficient de transfert de chaleur de l'interstice ($kW/m^2/K$)	10.0
Température à la surface de la gaine (K)	541.8

ANNEXE 3 : Guide d'utilisateur des modules THERMO : et THADD : .

Définitions :

Structure :

RESEAU : structure signée L_MAP

THER : structure signée L_THERMO

THERP : structure signée L_THERMO

[...] : option

Entier :

NZ : Nombre de discrétisation du canal

NF : Nombre de discrétisation du combustible

NC : Nombre de discrétisation de la gaine

CANI : Numéro du premier canal

CANF : Numéro du dernier canal

CV : Convergence de calcul couplé (>0 si convergé et <0 sinon)

Réal double précision :

MFIN : Flux massique à l'entrée des canaux ($\text{kg/m}^2/\text{s}$)

POUT : Pression à la sortie des canaux (Pa)

TIN : Température du caloporteur à l'entrée des canaux (K)

TCOM : Température du combustible pour initialisation (K)

LZ : Longueur du canal (m)

DF : Diamètre du combustible (m)

TG : Taille de l'interstice (m)

DC : Diamètre externe de la gaine (m)

DP : Distance entre deux centres de combustibles voisins (m)

THETA : Angle d'inclinaison du réacteur par rapport à la verticale (radians)

RUF : Rugosité de la gaine (m)

EVOLG : Variation du flux massique à l'entrée des canaux ($\text{kg/m}^2/\text{s}^2$)

EVOLT : Variation de la température du caloporteur à l'entrée des canaux (K/s)

EVOLP : Variation de la pression à la sortie des canaux (Pa/s)

DT : Pas de temps (s)

PREC : précision requise sur les calculs couplés entre deux itérations ($>10^{-5}$)

Module THERMO :

Le module THERMO : sert au calcul des états thermohydraulique dans les canaux de CANDU. L'appel de ce module se fait de la façon suivante :

```

THER RESEAU := THERMO : [THER] RESEAU ::

```

```

[...]

```

```

;

```

Pour la création de la structure L_THERMO, il est nécessaire de définir les dimensions ainsi que les conditions de fonctionnement du réacteur aux limites. Cela se fait en utilisant l'option suivante :

```

DEFCANDU

```

```

    DISCRET <<NZ>> <<NF>> <<NC>>

```

```

    GEO <<LZ>> <<DF>> <<TG>> <<DC>> <<DP>> <<THETA>> <<RUF>>

```

```

    THCOND <<MFIN>> <<POUT>> <<TIN>> <<TCOMB>>

```

```

    EVOLC <<EVOLG>> <<EVOLP>> <<EVOLT>>

```

```

    TIMEPAR <<DT>>

```

```

ENDDEF

```

Pour un calcul thermohydraulique à l'état stationnaire, l'appel se fait de la façon suivante :

```

THSTAT [SINGLP] ou [MULTIP <<CANI>> <<CANF>>]

```

Pour un calcul thermohydraulique à l'état transitoire, l'appel se fait de la façon suivante :

```

THKINEC [SINGLP] ou [MULTIP <<CANI>> <<CANF>>]

```

L'option SINGLP permet de faire les calculs thermohydrauliques pour tous les canaux tandis que MULTIP permet de faire un calcul thermohydraulique partiel limité aux canaux allant du numéro CANI au numéro CANF. L'option MULTIP sert aux calculs sur plusieurs processeurs.

L'option THEVOL permet de passer au prochain pas de temps dans un calcul transitoire.

L'option CONVER permet de vérifier la convergence entre deux itérations d'un calcul couplé entre la thermohydraulique et la neutronique. L'appel de cette option se fait de la façon suivante :

```
CONVER <<PREC>> >>CV<<
```

Module THADD :

Le module THADD permet l'addition de deux structures signées L_THERMO calculées avec l'option MULTIP. L'appel de ce module se fait comme suit :

```
THER := THADD : THER THERP ;
```

Module THUPDT :

Le module THUPDT permet de mettre à jour les données thermohydraulique contenues dans la structure signée L_MAP à partir d'une structure signée L_THERMO complète. L'appel de ce module est nécessaire après un calcul, sur plusieurs processeurs, des propriétés thermohydraulique et l'addition de ceux-ci avec le module THADD. L'appel de ce module se fait comme suit :

```
RESEAU := THUPDT : RESEAU THER ;
```

ANNEXE 4 : Guide de la structure L_THERMO : .

La structure utilisée pour le module de thermohydraulique est signée L_THERMO. Les discrétisations utilisées, et donc les tailles des matrices utilisées, sont répertoriées dans l'espace «STATE_VECTOR». Celui-ci contient les éléments suivants :

Position	Commentaire
1	Discrétisation axiale du canal (NZ)
2	Discrétisation radiale du combustible (NF)
3	Discrétisation radiale de la gaine (NC)
4	Nombre de canaux dans le réacteur (NCHAN)
5	Nombre de grappes dans le réacteur (NBUN)
6	Numéro du premier canal (utilisé uniquement pour les calculs partiels)
7	Numéro du dernier canal (utilisé uniquement pour les calculs partiels)

I : Entier

C : Caractère

R : Réel simple précision

D : Réel double précision

F : Dossier

(...) : Taille du vecteur ou du tenseur

Nom	Type	Niveau	Commentaire
SIGNATURE	C(12)	1	Signature de la structure (L_THERMO)
STATE-VECTOR	I(40)	1	Taille des vecteurs et tenseurs
GEOMETRY	D(7)	1	Dimensions du canal

BOUNDCOND	D(5)	1	Conditions aux limites des canaux
EVOLUTION	D(3)	1	Paramètres d'évolution
TIMEPARAM	D(1)	1	Discretisation temporelle (s)
CHANNEL X	F	1	Dossier des calculs de canaux (X est le numéro du canal)
MIX-ENTH	D(NZ,2)	2	Enthalpie du mélange (J/kg)
PRESSURE	D(NZ,2)	2	Pression dans le canal (Pa)
MASS-FLUX	D(NZ,2)	2	Flux massique dans le canal (kg/m ² /s)
VOID-FRAC	D(NZ,2)	2	Taux de vide
FLOW-QLTY	D(NZ,2)	2	Titre de l'écoulement
LIQ-ENTH	D(NZ,2)	2	Enthalpie du liquide (J/kg)
PIN-TEMP	D(NC+NF,NZ,2)	2	Température du crayon (K)
PWR-DSTY	D(NZ)	2	Densité de puissance (W/m ³)
B-TCOOL-I	D(NCHAN,NBUN)	1	Température du caloporteur par grappe avant calcul (K)
B-DCOOL-I	D(NCHAN,NBUN)	1	Densité du caloporteur par grappe avant calcul (g/cm ³)
B-TFUEL-I	D(NCHAN,NBUN)	1	Température du combustible par grappe avant calcul (K)

B-PWR-I	D(NCHAN,NBUN)	1	Puissance par grappe avant calcul (W)
B-TCOOL-F	D(NCHAN,NBUN)	1	Température après calcul du caloporteur par grappe (K)
B-DCOOL-F	D(NCHAN,NBUN)	1	Densité du caloporteur par grappe après calcul (g/cm^3)
B-TFUEL-F	D(NCHAN,NBUN)	1	Température du combustible par grappe après calcul (K)
B-PWR-F	D(NCHAN,NBUN)	1	Puissance par grappe après calcul (W)

Dans l'espace «GEOMETRY», nous avons les paramètres suivants :

Position	Commentaire
1	Longueur du canal (m)
2	Diamètre du combustible (m)
3	Dimension de l'interstice (m)
4	Diamètre externe de la gaine (m)
5	Dimension entre les centres de deux crayons voisins (m)
6	Inclinaison du réacteur par rapport à la verticale (radians)
7	Rugosité de la gaine (m)

Dans l'espace «BOUNDCOND», nous avons les paramètres suivants :

Position	Commentaire
1	Flux massique à l'entrée des canaux ($\text{kg/m}^2/\text{s}$)
2	Pression à la sortie des canaux (Pa)
3	Température du caloporteur à l'entrée des canaux (K)
4	Température initiale du combustible (K)

Dans l'espace «EVOLUTION», nous avons les paramètres suivants :

Position	Commentaire
1	Variation du flux massique à l'entrée des canaux ($\text{kg/m}^2/\text{s}^2$)

2	Variation de la pression à la sortie des canaux (Pa/s)
3	Variation de la température du caloporteur à l'entrée des canaux (K/s)

ANNEXE 5 : Guide du programmeur du code ARTHUR.

Fichier	Routine(R)/Fonction (F)	Description
arthur.f90	arthur (R)	Routine principale
	Pcplstat (R)	Algorithme stationnaire
	Pcpldyn (R)	Algorithme transitoire
Initial.f90	Initparam (R)	Initialisation des paramètres
Inoutdat.f90	Neutro (R)	Conversion des données pour la neutronique
Matrice.f90	Initmatcm (R)	Initialisation de la matrice pour la conservation de masse
	Initmatcqm (R)	Initialisation de la matrice pour la conservation de quantité de mouvement
	Initmatcestat (R)	Initialisation de la matrice pour la conservation d'énergie (à l'état stationnaire)
	Initmatce (R)	Initialisation de la matrice pour la conservation d'énergie
	Initmattigstat (R)	Initialisation de la matrice pour le transfert de chaleur (à l'état stationnaire)
	Initmattig (R)	Initialisation de la matrice pour le transfert de chaleur
Vecteur.f90	Initvectcmstat (R)	Initialisation du vecteur pour la conservation de masse (à l'état stationnaire)
	Initvectcm (R)	Initialisation du vecteur pour la conservation de masse
	Initvectcqmstat (R)	Initialisation du vecteur pour la conservation de quantité de mouvement (à l'état stationnaire)

	Initvectcqm (R)	Initialisation du vecteur pour la conservation de quantité de mouvement
	Initvectcestat (R)	Initialisation du vecteur pour la conservation d'énergie (à l'état stationnaire)
	Initvectce (R)	Initialisation du vecteur pour la conservation d'énergie
	Initvecttigstat (R)	Initialisation du vecteur pour le transfert de chaleur (à l'état stationnaire)
	Initvecttig (R)	Initialisation du vecteur pour le transfert de chaleur
Calculs.f90	Gausscm (R)	Méthode de Gauss pour la conservation de masse
	Gausscqm (R)	Méthode de Gauss pour la conservation de quantité de mouvement
	Gaussce (R)	Méthode de Gauss pour la conservation d'énergie
	Gausstig (R)	Méthode de Gauss pour le transfert de chaleur
	Converge (R)	Convergence sur la partie écoulements
	Convergt (R)	Convergence sur la partie thermique
	Converg (R)	Convergence complète
	Rhom (F)	Densité du mélange
	Rhoe (F)	Densité d'énergie
	Rho_m (F)	Densité de la quantité de mouvement
	Hliq (F)	Enthalpie du liquide
	Twal (F)	Température externe de la gaine
	Upxepts (R)	Taux de vide et titre de l'écoulement
Mprop.f90	Propur (F)	Propriétés thermodynamiques et d'échange du dioxyde d'uranium
	Propzir (F)	Propriétés thermodynamiques et

		d'échange du zircaloy
	Protev (F)	Propriétés thermodynamiques et d'échange de l'eau et de la vapeur

ANNEXE 6 : Code source.

Arthur.f90

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!
!
!
!                                     CODE ARTHUR
!
!       (Advanced Routines of Thermal-Hydraulic for Unsteady-states
Reactors)      !
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale
!
! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
! prnt : impression
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! Ttig : temperature du crayon en K
!
! dt : pas de temps en s
!
! qinit : puissance en W
!
! Ginit : flux massique à l'entrée du canal en kg/m2/s
!
! hinit : enthalpie a l'entree du canal en J/kg
!

```



```

! pinit: pression a la sortie du canal en Pa
!
! tinit : temperature a l'entree du canal en K
!
! tinic : temperature initiale du combustible en K
!
! evolg : variation du flux massique à l'entree du canal en kg/m2/s2
!
! evolt : variation de la temperature à l'entree du canal en K/s
!
! evolp : variation de la pression à la sortie du canal en Pa/s
!
! theta : inclinaison du canal par rapport a la verticale en radian
!
! Dcomb : diametre du combustible en m
!
! tg : taille de l'interstice en m
!
! Dgext : diametre externe de la gaine en m
!
! Dinter : distance entre le centre de deux crayons voisins en m
!
! Ltig : longueur du canal en m
!
! epsr : rugosite de la gaine en m
!
! Tcal : temperature du caloporteur en K
!
! Rcal : densite du caloporteur en kg/m3
!
! Tcomb : temperature du combustible en K
!
!Caractere :
!
! act : action
!
!
!
!*****!
*****!

subroutine arthur(Ndisc,Mcdisc,Mgdisc,Mdisc,dt,&
  &hent,ppres, Gmel, epsm, xtit, hlq,Ttig,&
  &qinit,Ginit,hinit,pinit,tinit,tcini,evolg,evolp,evolt,&
  &theta,Dcomb, tg, Dgext, Dinter, Ltig,
  epsr,Tcal,Rcal,Tcomb,act,prnt)
  implicit none
  !Discrétisation
  integer :: i
  integer, intent (in):: Ndisc, Mcdisc, Mgdisc, Mdisc, prnt,act
  double precision, intent(in) :: dt
  !Matrices des propriétés thermodynamiques et des températures de
  la tige
  double precision, dimension (Ndisc,2), intent(inout) :: hent,
  ppres, Gmel, epsm, xtit, hlq
  double precision, dimension (Mdisc,Ndisc,2), intent(inout) ::
  Ttig

```

```

!Conditions initiales et d'évolution
double precision, intent (inout) ::
Ginit,hinit,pinit,tinit,tcini,evolg,evolt,evolp
double precision, dimension (Ndisc), intent(in) :: qinit
!Angle de l'écoulement
double precision, intent(in) :: theta
!Géométrie
double precision, intent(in) :: Dcomb,tg,Dgext,Dinter,epsr,Ltig
!Valeurs à calculer
double precision :: Dgint,Atothex,Atigehex,A,Pe,Dh
double precision :: dz,drc,drg
double precision, dimension (Ndisc) :: qwv,qws !Puissance
volumique et surfacique
!Matrice et vecteur
double precision, dimension (Ndisc,Ndisc) :: matcm, matcqm,
matce !matrices des coefficients thermodynamiques
double precision, dimension (Ndisc) :: vectcm, vectcqm, vectce
!vecteurs des propriétés thermodynamiques
double precision, dimension (Mdisc,Mdisc,Ndisc) :: mattg
!matrices des coefficients de transfert de chaleur
double precision, dimension (Mdisc,Ndisc) :: vecttig !vecteurs
des propriétés thermiques
!Vecteurs temporaires
double precision, dimension (Ndisc) :: henttps, pprests,
Gmeltps !vecteurs temporaires pour la convergence
double precision, dimension (Mdisc,Ndisc) :: Ttigtps !vecteurs
temporaires pour la convergence
!Sortie neutronique
double precision, dimension (Ndisc), intent(out) ::
Tcal,Rcal,Tcomb
!Constantes
double precision, parameter :: pi=3.14159265359
!Calcul de la géométrie
Dgint=Dgext-2.0*tg
Atothex=3.0*(Dinter)**2.0*(3.0)**(1.0/2.0)/2.0
Atigehex=pi*Dgext**2.0*3.0/4.0
A=Atothex-Atigehex
Pe=3.0*pi*Dgext
Dh=4*A/Pe
!Discrétisation
drc=Dcomb/2.0/(dble(Mdisc)-1.0)
drg=tg/(dble(Mdisc)-1.0)
dz=Ltig/(dble(Ndisc)-1.0)
!Flux de chaleur et densité de puissance
qwv=qinit/(pi*(Dcomb**2.0)/4.0*dz*(dble(Ndisc)-1.0))
qws=qinit/(pi*Dgext*dz*(dble(Ndisc)-1.0))
select case (act)
case (0)
call
initparam(Ndisc,Mdisc,Gmel,ppres,hent,epsm,xtit,hlq,Ttig,&
&Ginit,pinit,hinit,tinit,tcini)
call
neutro(Ndisc,Mdisc,Mdisc,Mdisc,Gmel,ppres,hent,xtit,&
&epsm,hlq,Ttig,Tcal,Rcal,Tcomb,1)
case (1)
Gmeltps(:)=0.0
henttps(:)=0.0
pprestps(:)=0.0

```

```

      Ttigtps(:, :)=0.0
      call
pcplstat(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,epsm,hlq,Tti
g,&
      &Ginit,hinit,pinit,tinit,tcini,evolg,evolt,evolp,&

&theta,Dcomb,tg,Dgext,epsr,Dgint,A,Pe,Dh,dz,drc,drg,qwv,qws,&

&matcm,matcqm,matce,vectcm,vectcqm,vectce,mattg,vecttig,&
      &Gmeltps,pprestps,henttps,Ttigtps,prnt)
      call
neutro(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,&
      &epsm,hlq,Ttig,Tcal,Rcal,Tcomb,1)
      case (2)
      Gmeltps(:)=0.0
      henttps(:)=0.0
      pprestps(:)=0.0
      Ttigtps(:, :)=0.0
      ppres(:,2)=ppres(:,1)
      hent(:,2)=hent(:,1)
      Gmel(:,2)=Gmel(:,1)
      hlq(:,2)=hlq(:,1)
      xtit(:,2)=xtit(:,1)
      epsm(:,2)=epsm(:,1)
      Ttig(:,2)=Ttig(:,1)
      call
pcpldyn(Ndisc,Mdisc,Mcdisc,Mgdisc,dt,Gmel,ppres,hent,xtit,epsm,hlq,T
tig,&
      &Ginit,pinit,hinit,tinit,tcini,evolg,evolt,evolp,&

&theta,Dcomb,tg,Dgext,epsr,Dgint,A,Pe,Dh,dz,drc,drg,qwv,qws,&

&matcm,matcqm,matce,vectcm,vectcqm,vectce,mattg,vecttig,&
      &Gmeltps,pprestps,henttps,Ttigtps,prnt)
      call
neutro(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,&
      &epsm,hlq,Ttig,Tcal,Rcal,Tcomb,2)
      end select
end subroutine arthur

```

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!                                CALCUL STATIONNAIRE
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale
!

```

```

! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
! prnt : impression
!
! Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! Ttig : temperature du crayon en K
!
! qinit : puissance en W
!
! Ginit : flux massique à l'entrée du canal en kg/m2/s
!
! hinit : enthalpie a l'entree du canal en J/kg
!
! pinit: pression a la sortie du canal en Pa
!
! tinit : temperature a l'entree du canal en K
!
! tinic : temperature initiale du combustible en K
!
! evolg : variation du flux massique à lentree du canal en kg/m2/s2
!
! evolt : variation de la temperature à lentree du canal en K/s
!
! evolp : variation de la pression à la sortie du canal en Pa/s
!
! theta : inclinaison du canal par rapport a la verticale en radian
!
! Dcomb : diametre du combustible en m
!
! tg : taille de l'interstice en m
!
! Dgext : diametre externe de la gaine en m
!
! epsr : rugosite de la gaine en m
!
! Dgint : diametre interne de la gaine en m
!
! A : aire d'ecoulement en m2
!
! Pe : perimetre mouille en m
!
! Dh : diametre hydraulique en m
!

```



```

! dz : pas axial en m
!
! drc : pas pour le combustible en m
!
! drg : pas pour la gaine en m
!
! qwv : densite de puissance en W/m3
!
! qws : flux de puissance en W/m2
!
! matcm : matrice du systeme Ax=b      pour la conservation de la
masse !
! matcqm : matrice du systeme Ax=b      pour la conservation de la
qtite de movvmt !
! matce : matrice du systeme Ax=b      pour la conservation de
lenergie !
! mattg : matrice du systeme Ax=b      pour le transfert de chaleur
!
! vectcm : vecteur du systeme Ax=b      pour la conservation de la
masse !
! vectcqm : vecteur du systeme Ax=b      pour la conservation de la
qtite de movvmt !
! vectce : vecteur du systeme Ax=b      pour la conservation de
lenergie !
! vecttig : vecteur du systeme Ax=b      pour le transfert de chaleur
!
! Gmel : valeurs temporaires du flux massique en kg/m2/s
!
! ppres : valeurs temporaires de la pression en Pa
!
! hent : valeurs temporaires de l'enthalpie en J/kg
!
! Ttig : temperature temporaires du crayon en K
!
!
! *****
*****!

```

subroutine

pcplstat (Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,eps,hlq,Ttig,&

```

g,&
    &Ginit,hinit,pinit,tinit,tcini,evol,evolt,evolp,&
    &theta,Dcomb,tg,Dgext,epsr,Dgint,A,Pe,Dh,dz,drc,drq,qwv,qws,&
    &matcm,matcqm,matce,vectcm,vectcqm,vectce,mattg,vecttig,&
    &Gmeltps,pprestps,henttps,Ttigtps,prnt)
implicit none
integer, intent(in) :: Ndisc, Mcdisc, Mgdisc, Mdisc, prnt
double precision, dimension (Ndisc,2), intent(inout) :: hent,
ppres, Gmel, eps, xtit, hlq
double precision, dimension (Mdisc,Ndisc,2), intent(inout) ::
Ttig
double precision, intent(inout) ::
Ginit,hinit,pinit,tinit,tcini,evol,evolt,evolp
double precision, intent(in) :: theta
double precision, intent(in) :: Dcomb, tg, Dgext, epsr
double precision, intent(in) :: Dgint,A,Pe,Dh

```

```

double precision, intent(in) :: dz,drc,drg
double precision, dimension (Ndisc), intent(in) :: qwv,qws
double precision, dimension (Ndisc,Ndisc), intent(inout) ::
matcm, matcqm, matce
double precision, dimension (Ndisc), intent(inout) :: vectcm,
vectcqm, vectce
double precision, dimension (Mdisc,Mdisc,Ndisc), intent(inout)
:: mattg
double precision, dimension (Mdisc,Ndisc), intent(inout) ::
vecttig
double precision, dimension (Ndisc), intent(inout) :: henttps,
pprestps, Gmeltps
double precision, dimension (Mdisc,Ndisc), intent(inout) ::
Ttigtps
integer :: crg, itr, i, j
double precision, external :: propev
itr=0
crg=0
boucle : do
    itr=itr+1
    call initmatcqm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matcqm)
    call
initvectcqmstat(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcqm,pinit,dz
,Dh,epsr,theta)
    call gausscqm(1,Ndisc,ppres,matcqm,vectcqm)
    call
initmatcestat(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matce)
    call
initvectcestat(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectce,qws,Ginit,
hinit,dz,Dh,Pe,A,epsr,theta)
    call gaussce(1,Ndisc,hent,matce,vectce)
    call
upxeps(crg,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,qws,Dh,theta)
    if (crg==1) then
        crg=0
    else
        print *, "CHEVALIER: Void fraction not converged"
        exit boucle
    end if
    call
initmattigstat(Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,drc,drg,Dcomb,tg
)
    call
initvecttigstat(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,epsm,
hlq,Ttig,&
&vecttig,qws,qwv,drc,drg,Dh,Dgext)
    call
gausstig(1,Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,vecttig)
    call convrg(crg,1,Ndisc,Mdisc,Gmel,ppres,hent,Ttig,Gmeltps,&
&pprestps,henttps,Ttigtps,prnt)
    if (crg==1) then
        exit boucle
    else if (itr>100000) then
        print *, "CHEVALIER: Static calculus not converged"
        exit boucle
    end if
    if (prnt>0) then
        print *,"Iteration : ",itr

```

```

        end if
        hinit=propev(ppres(1,1),tinit,hinit,"HPTL")
    end do boucle
end subroutine pcplstat

```

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!                                     CALCUL TRANSITOIRE
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
!  Ndisc : discretisation axiale du canal
!
!  Mdisc : discretisation radiale
!
!  Mcdisc : discretisation du combustible
!
!  Mgdisc : discretisation de la gaine
!
!  prnt : impression
!
!Double precision :
!
!  Gmel : valeurs du flux massique en kg/m2/s
!
!  ppres : valeurs de la pression en Pa
!
!  hent : valeurs de l'enthalpie en J/kg
!
!  xtit : titre de l'ecoulement
!
!  epsm : taux de vide
!
!  hlq : enthalpie du liquide J/kg
!
!  Ttig : temperature du crayon en K
!
!  qinit : puissance en W
!
!  Ginit : flux massique à l'entrée du canal en kg/m2/s
!
!  hinit : enthalpie a l'entree du canal en J/kg
!
!  pinit: pression a la sortie du canal en Pa
!
!  tinit : temperature a l'entree du canal en K
!
!  tinic : temperature initiale du combustible en K
!
!  evolg : variation du flux massique à l'entree du canal en kg/m2/s2
!

```

```

! evolt : variation de la temperature à l'entree du canal en K/s
!
! evolp : variation de la pression à la sortie du canal en Pa/s
!
! theta : inclinaison du canal par rapport a la verticale en radian
!
! Dcomb : diametre du combustible en m
!
! tg : taille de l'interstice en m
!
! Dgext : diametre externe de la gaine en m
!
! epsr : rugosite de la gaine en m
!
! Dgint : diametre interne de la gaine en m
!
! A : aire d'ecoulement en m2
!
! Pe : perimetre mouille en m
!
! Dh : diametre hydraulique en m
!
! dt : pas de temps en s
!
! dz : pas axial en m
!
! drc : pas pour le combustible en m
!
! drg : pas pour la gaine en m
!
! qwv : densite de puissance en W/m3
!
! qws : flux de puissance en W/m2
!
! matcm : matrice du systeme Ax=b      pour la conservation de la
masse                                !
! matcqm : matrice du systeme Ax=b      pour la conservation de la
qtite de movvmt                    !
! matce : matrice du systeme Ax=b      pour la conservation de
lenergie                            !
! mattg : matrice du systeme Ax=b      pour le transfert de chaleur
!
! vectcm : vecteur du systeme Ax=b      pour la conservation de la
masse                                !
! vectcqm : vecteur du systeme Ax=b      pour la conservation de la
qtite de movvmt                    !
! vectce : vecteur du systeme Ax=b      pour la conservation de
lenergie                            !
! vecttig : vecteur du systeme Ax=b      pour le transfert de chaleur
!
! Gmel : valeurs temporaires du flux massique en kg/m2/s
!
! ppres : valeurs temporaires de la pression en Pa
!
! hent : valeurs temporaires de l'enthalpie en J/kg
!
! Ttig : temperature temporaires du crayon en K
!

```

```

!
!
! *****
*****!

```

```

subroutine
pcpldyn(Ndisc,Mdisc,Mcdisc,Mgdisc,dt,Gmel,ppres,hent,xtit,epsm,hlq,T
tig,&
    &Ginit,pinit,hinit,tinit,tcini,evolp,evolt,volp,&
    &theta,Dcomb,tg,Dgext,epsr,Dgint,A,Pe,Dh,dz,drc,drg,qwv,qws,&
    &matcm,matcqm,matce,vectcm,vectcqm,vectce,mattg,vecttig,&
    &Gmeltps,pprestps,henttps,Ttigtps,prnt)
implicit none
integer, intent(in):: Ndisc, Mcdisc, Mgdisc, Mdisc,prnt
double precision, intent(in):: dt
double precision, dimension (Ndisc,2), intent(inout):: hent,
ppres, Gmel, epsm, xtit, hlq
double precision, dimension (Mdisc,Ndisc,2), intent(inout)::
Ttig
double precision, intent(inout)::
Ginit,hinit,pinit,tinit,tcini,evolp,evolt,volp
double precision, intent(in):: theta
double precision, intent(in):: Dcomb, tg, Dgext, epsr
double precision, intent(in):: Dgint,A,Pe,Dh
double precision, intent(in):: dz,drc,drg
double precision, dimension (Ndisc), intent(in):: qwv,qws
double precision, dimension (Ndisc,Ndisc), intent(inout)::
matcm, matcqm, matce
double precision, dimension (Ndisc), intent(inout):: vectcm,
vectcqm, vectce
double precision, dimension (Mdisc,Mdisc,Ndisc), intent(inout)
:: mattg
double precision, dimension (Mdisc,Ndisc), intent(inout)::
vecttig
double precision, dimension (Ndisc), intent(inout):: henttps,
pprestps, Gmeltps
double precision, dimension (Mdisc,Ndisc), intent(inout)::
Ttigtps
double precision, external :: propev
integer :: crg, itr, j, i
itr=0
crg=0
pinit=pinit+evolp*dt
tinit=tinit+evolt*dt
Ginit=Ginit+evolG*dt
hinit=propev(ppres(1,2),tinit,hinit,"HPTL")
boucle : do
    itr=itr+1
    call initmatcm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matcm)
    call
initvectcm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcm,Ginit,dz,dt)
    call gausscm(2,Ndisc,Gmel,matcm,vectcm)
    call initmatcqm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matcqm)
    call
initvectcqm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcqm,pinit,dz,dt,
Dh,epsr,theta)
    call gausscqm(2,Ndisc,ppres,matcqm,vectcqm)

```

```

        call
initmatce (Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matce,dz,dt)
        call
initvectce (Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectce,qws,Ginit,hini
t,dz,dt,Dh,Pe,A,epsr,theta)
        call gaussce(2,Ndisc,hent,matce,vectce)
        call
upxeps (crg,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,qws,Dh,theta)
        if (crg==1) then
            crg=0
        else
            print *, "CHEVALIER: Void fraction not converged"
            exit boucle
        end if
        call
initmattig (Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,drc,drg,dt,Dcomb,tg)
        call
initvecttig (Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,epsm,hlq,
Ttig,&
            &vecttig,qws,qwv,drc,drg,dt,Dh,Dgext)
        call
gausstig(2,Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,vecttig)
        call convrg(crg,2,Ndisc,Mdisc,Gmel,ppres,hent,Ttig,Gmeltps,&
            &pprestps,henttps,Ttigtps,prnt)
        if (crg==1) then
            exit boucle
        else if (itr>100000) then
            print *, "CHEVALIER: Kinetic calculus not converged"
            exit boucle
        end if
        if (prnt>0) then
            print *,"Iteration : ",itr
        end if
        hinit=propev(ppres(1,2),tinit,hinit,"HPTL")
    end do boucle
end subroutine pcpldyn

```

calculs.f90

```

!*****
!*****!
!
!
!           Methode de Gauss pour le systeme Ax=B de l'équation
!
!           de conservation de la masse
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!

```

```

! val : =1 ou 2 pour un calcul stationnaire ou transitoire
!
! Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! matcm : matrice du systeme Ax=B
!
! vectcm : vecteur du systeme Ax=b
!
!
!
! *****
! *****!

```

```

subroutine gausscm(val,Ndisc,Gmel,matcm,vectcm)
  implicit none
  integer :: i,j,limit
  integer, intent (in) :: val, Ndisc
  double precision, dimension (Ndisc,2), intent (inout) :: Gmel
  double precision, dimension (Ndisc,Ndisc), intent (inout) ::
matcm
  double precision, dimension (Ndisc), intent (inout) :: vectcm
  double precision :: tp
  !Élimination
  do i=Ndisc,2,-1
    if (i<Ndisc) then
      limit=i-1
    else
      limit=i-2
    end if
    vectcm(i)=vectcm(i)/matcm(i,i)
    matcm(i,limit:i)=matcm(i,limit:i)/matcm(i,i)
    tp=matcm(i-1,i)
    matcm(i-1,limit:i)=matcm(i-1,limit:i)-tp*matcm(i,limit:i)
    vectcm(i-1)=vectcm(i-1)-tp*vectcm(i)
  end do
  vectcm(1)=vectcm(1)/matcm(1,1)
  matcm(1,1)=matcm(1,1)/matcm(1,1)
  !Descente
  do i=1,Ndisc
    tp=vectcm(i)
    if (i>1 .AND. i<Ndisc) then
      limit=i-1
    else if (i==Ndisc) then
      limit=i-2
    else
      limit=Ndisc
    end if
    do j=limit,i-1
      tp=tp-matcm(i,j)*Gmel(j,val)
    end do
    Gmel(i,val)=tp
  end do
end subroutine gausscm

```

```

*****
*****!
!
!
!           Methode de Gauss pour le systeme Ax=B de l'équation
!
!           de conservation de la quantite de mouvement
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! val : =1 ou 2 pour un calcul stationnaire ou transitoire
!
!Double precision :
!
! ppres : valeurs de la pression en Pa
!
! matcqm : matrice du systeme Ax=B
!
! vectcqm : vecteur du systeme Ax=b
!
!
!
*****
*****!

subroutine gausscqm(val,Ndisc,ppres,matcqm,vectcqm)
  implicit none
  integer :: i,j,limit
  integer, intent (in) :: val, Ndisc
  double precision, dimension (Ndisc,2), intent (inout) :: ppres
  double precision, dimension (Ndisc,Ndisc), intent (inout) ::
matcqm
  double precision, dimension (Ndisc), intent (inout) :: vectcqm
  double precision :: tp
  !Élimination
  do i=1,Ndisc-1
    if (i>1) then
      limit=i+1
    else
      limit=i+2
    end if
    vectcqm(i)=vectcqm(i)/matcqm(i,i)
    matcqm(i,i:limit)=matcqm(i,i:limit)/matcqm(i,i)
    tp=matcqm(i+1,i)
    matcqm(i+1,i:limit)=matcqm(i+1,i:limit)-tp*matcqm(i,i:limit)
    vectcqm(i+1)=vectcqm(i+1)-tp*vectcqm(i)
  end do
  vectcqm(Ndisc)=vectcqm(Ndisc)/matcqm(Ndisc,Ndisc)
  matcqm(Ndisc,Ndisc)=matcqm(Ndisc,Ndisc)/matcqm(Ndisc,Ndisc)
  !Remontée
  do i=Ndisc,1,-1

```



```

        tp=vectcqm(i)
        if (i>1 .AND. i<Ndisc) then
            limit=i+1
        else if (i==1) then
            limit=i+2
        else
            limit=Ndisc
        end if
        do j=i+1,limit
            tp=tp-matcqm(i,j)*ppres(j,val)
        end do
        ppres(i,val)=tp
    end do
end subroutine gausscqm

```

```

!*****
!*****!
!
!
!           Methode de Gauss pour le systeme Ax=B de l'équation
!
!           de conservation de l'energie
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! val : =1 ou 2 pour un calcul stationnaire ou transitoire
!
!Double precision :
!
! hent : valeurs de l'enthalpie en J/kg
!
! matce : matrice du systeme Ax=B
!
! vectce : vecteur du systeme Ax=b
!
!
!
!*****
!*****!

```

```

subroutine gaussce(val,Ndisc,hent,matce,vectce)
    implicit none
    integer :: i,j,limit
    integer, intent (in) :: val , Ndisc
    double precision, dimension (Ndisc,2), intent (inout) :: hent
    double precision, dimension (Ndisc,Ndisc), intent (inout) ::
matce
    double precision, dimension (Ndisc), intent (inout) :: vectce
    double precision :: tp
    !Élimination

```

```

do i=Ndisc,2,-1
  if (i<Ndisc) then
    limit=i-1
  else
    limit=i-2
  end if
  vectce(i)=vectce(i)/matce(i,i)
  matce(i,limit:i)=matce(i,limit:i)/matce(i,i)
  tp=matce(i-1,i)
  matce(i-1,limit:i)=matce(i-1,limit:i)-tp*matce(i,limit:i)
  vectce(i-1)=vectce(i-1)-tp*vectce(i)
end do
vectce(1)=vectce(1)/matce(1,1)
matce(1,1)=matce(1,1)/matce(1,1)
!Descente
do i=1,Ndisc
  tp=vectce(i)
  if (i>1 .AND. i<Ndisc) then
    limit=i-1
  else if (i==Ndisc) then
    limit=i-2
  else
    limit=Ndisc
  end if
  do j=limit,i-1
    tp=tp-matce(i,j)*hent(j,val)
  end do
  hent(i,val)=tp
end do
end subroutine gaussce

!*****
!*****!
!
!
!           Methode de Gauss pour le systeme Ax=B de l'équation
!
!           de transfert de chaleur
!
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale du crayon
!
! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
! val : =1 ou 2 pour un calcul stationnaire ou transitoire
!

```

```

!Double precision :
!
! Ttig : valeurs de la temperature en K
!
! mattg : matrice du systeme Ax=B
!
! vecttig : vecteur du systeme Ax=b
!
!
!*****
*****!

```

```

subroutine
gausstig(val,Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,vecttig)
  implicit none
  integer :: i,j,k,limit
  integer, intent (in) :: val, Ndisc, Mdisc, Mcdisc, Mgdisc
  double precision, dimension (Mdisc,Ndisc,2), intent (inout) ::
Ttig
  double precision, dimension (Mdisc,Mdisc,Ndisc), intent (inout)
:: mattg
  double precision, dimension (Mdisc,Ndisc), intent (inout) ::
vecttig
  double precision :: tp
  do k=1,Ndisc
    !Élimination
    do i=1,Mdisc-1
      if (i<Mcdisc) then
        limit=i+2
      else if (i>Mcdisc .AND. i<Mdisc-1) then
        limit=i+2
      else if (i==Mdisc-1) then
        limit=i+1
      else if (i==Mcdisc) then
        limit=i+3
      else
        limit=Mdisc
      end if
      vecttig(i,k)=vecttig(i,k)/mattg(i,i,k)
      mattg(i,i:limit,k)=mattg(i,i:limit,k)/mattg(i,i,k)
      tp=mattg(i+1,i,k)
      mattg(i+1,i:limit,k)=mattg(i+1,i:limit,k)-
tp*mattg(i,i:limit,k)
      vecttig(i+1,k)=vecttig(i+1,k)-tp*vecttig(i,k)
    end do
    vecttig(Mdisc,k)=vecttig(Mdisc,k)/mattg(Mdisc,Mdisc,k)
  end do
  mattg(Mdisc,Mdisc,k)=mattg(Mdisc,Mdisc,k)/mattg(Mdisc,Mdisc,k)
  !Remontée
  do i=Mdisc,1,-1
    tp=vecttig(i,k)
    if (i/=Mdisc .AND. i/=Mdisc-1 .AND. i/=Mdisc-2) then
      limit=i+3
    else if (i==Mdisc-2) then
      limit=i+2
    else if (i==Mdisc-1) then

```

```

        limit=i+1
    else
        limit=Mdisc
    end if
    do j=i+1,limit
        tp=tp-mattg(i,j,k)*Ttig(j,k,val)
    end do
    Ttig(i,k,val)=tp
end do
end do
end subroutine gausstig

!*****
!*****!
!
!
!           Convergence de la partie ecoulement
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! val : 1 ou 2 pour un calcul stationnaire ou transitoire
!
! prnt : affichage des erreurs
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! Gmeltps : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! pprests : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! henttps : valeurs de l'enthalpie en J/kg
!
!
!
!*****
!*****!

subroutine convrge(crg,val,Ndisc,Gmel,ppres,hent,&
&Gmeltps,pprestps,henttps,prnt)
    implicit none
    integer, intent (out) :: crg
    double precision :: errg ,errp, errh
    integer, intent (in) :: val,Ndisc, prnt

```

```

      double precision, dimension (Ndisc,2), intent(in) :: hent,
      ppres, Gmel
      double precision, dimension (Ndisc), intent(inout) :: henttps,
      ppresteps, Gmeltps
      crg=0
      errg=abs(Gmel(Ndisc,val)-Gmeltps(Ndisc))/Gmel(Ndisc,val)
      errp=abs(ppres(1,val)-pprestps(1))/ppres(1,val)
      errh=abs(hent(Ndisc,val)-henttps(Ndisc))/hent(Ndisc,val)
      if (prnt>1) then
        print *, "Error Mass Flux: ", errg
        print *, "Error Pressure: ", errp
        print *, "Error Enthalpy: ", errh
      end if
      if (errg<1.0e-5 .AND. errp<1.0e-5 .AND. errh<1.0e-5) then
        crg=1
      end if
      ppresteps(:)=ppres(:,val)
      henttps(:)=hent(:,val)
      Gmeltps(:)=Gmel(:,val)
end subroutine convrge

!*****
!*****!
!
!
!           Convergence de la partie thermique
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale du crayon
!
! val : 1 ou 2 pour un calcul stationnaire ou transitoire
!
! prnt : affichage des erreurs
!
!Double precision :
!
! Ttig : valeurs de la temperature en K
!
! Ttigtps : valeurs de la temperature en K
!
!
!
!*****
!*****!

subroutine convrgt(crg,val,Ndisc,Mdisc,Ttig,Ttigtps,prnt)
  implicit none
  integer, intent (out) :: crg
  double precision :: errt
  integer, intent (in) :: val,Ndisc,Mdisc,prnt

```

```

      double precision, dimension (Mdisc,Ndisc,2), intent(in) :: Ttig
      double precision, dimension (Mdisc,Ndisc), intent(inout) ::
Ttigtps
      crg=0
      errt=abs(Ttig(1,int(Ndisc/2),val)-
Ttigtps(1,int(Ndisc/2)))/Ttig(1,int(Ndisc/2),val)
      if (prnt>1) then
        print *, "Error Tcomb:",errt
      end if
      if (errt<1.0e-5) then
        crg=1
      end if
      Ttigtps(:,:)=Ttig(:,:,val)
end subroutine convrgt

!*****
!*****!
!
!
!           Convergence de la partie thermique et ecoulement
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale du crayon
!
! val : 1 ou 2 pour un calcul stationnaire ou transitoire
!
! prnt : affichage des erreurs
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! Gmeltps : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! ppretps : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! henttps : valeurs de l'enthalpie en J/kg
!
! Ttig : valeurs de la temperature en K
!
! Ttigtps : valeurs de la temperature en K
!
!
!
!*****
!*****!

```

```

subroutine convrg(crg,val,Ndisc,Mdisc,Gmel,ppres,hent,Ttig,Gmeltps,&
&pprestps,henttps,Ttigtps,prnt)
  implicit none
  integer, intent (in) :: val,Ndisc,Mdisc,prnt
  double precision, dimension (Ndisc,2), intent(in) :: hent,
  ppres, Gmel
  double precision, dimension (Ndisc), intent(inout) :: henttps,
  pprestps, Gmeltps
  double precision, dimension (Mdisc,Ndisc,2), intent(in) :: Ttig
  double precision, dimension (Mdisc,Ndisc), intent(inout) ::
  Ttigtps
  integer, intent (out) :: crg
  integer :: crgt, crge
  call convrgt(crge,val,Ndisc,Mdisc,Ttig,Ttigtps,prnt)
  call
  convrge(crgt,val,Ndisc,Gmel,ppres,hent,Gmeltps,pprestps,henttps,prnt
  )
  if (crgt==1 .AND. crge==1) then
    crg=1
  else
    crg=0
  end if
end subroutine convrg

```

```

!*****
!*****!

```

```
!
```

```
!
```

```
!          Calcul de la densité du mélange (RHOM)
```

```
!
```

```
!
```

```
!
```

```
!Fait par: Majid FASSI FEHRI (31/12/2007)
```

```
!
```

```
!Entier:
```

```
!
```

```
! Ndisc : discretisation axiale du canal
```

```
!
```

```
! i : position dans le canal
```

```
!
```

```
! j : 1 ou 2 pour un calcul stationnaire ou transitoire
```

```
!
```

```
!Double precision :
```

```
!
```

```
! Gmel : valeurs du flux massique en kg/m2/s
```

```
!
```

```
! ppres : valeurs de la pression en Pa
```

```
!
```

```
! hent : valeurs de l'enthalpie en J/kg
```

```
!
```

```
! xtit : titre de l'ecoulement
```

```
!
```

```
! epsm : taux de vide
```

```
!
```



```

! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
!
!
! *****
! *****!

```

```

double precision function
rhoe(i,j,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)
    implicit none
    integer, intent (in) :: i,j,Ndisc
    double precision, dimension (Ndisc,2), intent(in) :: hent,
    ppres, Gmel, hlq
    double precision, dimension (Ndisc,2), intent(inout) :: epsm,
    xtit
    double precision, external :: propev
    double precision :: temp
    temp=0.0
    if (hlq(i,j)<propev(ppres(i,j),temp,hlq(i,j),"HFP ")) then
        temp=propev(ppres(i,j),temp,hlq(i,j),"TPHL")
        if (xtit(i,j)==0.0) then
            rhoe=propev(ppres(i,j),temp,hlq(i,j),"VPTL")
        else
            rhoe=1.0/sqrt(((xtit(i,j))**3.0)/(epsm(i,j)*propev(ppres(i,j),temp,0
            .0,"VGP "))**2.0+((1.0-xtit(i,j))**3.0)/((1.0-
            epsm(i,j))*propev(ppres(i,j),temp,hlq(i,j),"VPT&
            &L"))**2.0)
            end if
        else
            if (xtit(i,j)==0.0) then
                xtit(i,j)=1.0e-9
                epsm(i,j)=1.0e-9
            end if
            rhoe=1.0/sqrt(((xtit(i,j))**3.0)/(epsm(i,j)*propev(ppres(i,j),temp,0
            .0,"VGP "))**2.0+
            &((1.0-xtit(i,j))**3.0)/((1.0-
            epsm(i,j))*propev(ppres(i,j),temp,hlq(i,j),"VFP "))**2.&
            &0)
            end if
        end function rhoe

```

```

! *****
! *****!
!
!

```

```

!                               Calcul de la densité de la quantité de
mouvement (RHO_M)              !
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! i : position dans le canal
!
! j : 1 ou 2 pour un calcul stationnaire ou transitoire
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
!
!
!*****
*****!

double precision function
rho_m(i,j,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)
    implicit none
    integer, intent (in) :: i,j,Ndisc
    double precision, dimension (Ndisc,2), intent(in) :: hent,
    ppres, Gmel, hlq
    double precision, dimension (Ndisc,2), intent(inout) :: epsm,
    xtit
    double precision, external :: propev
    double precision :: temp
    temp=0.0
    if (hlq(i,j)<propev(ppres(i,j),temp,hlq(i,j),"HFP ")) then
        temp=propev(ppres(i,j),temp,hlq(i,j),"TPHL")
        if (xtit(i,j)==0.0) then
            rho_m=propev(ppres(i,j),temp,hlq(i,j),"VPTL")
        else

rho_m=1.0/(((xtit(i,j))**2.0)/(epsm(i,j)*propev(ppres(i,j),temp,0.0,
"VGP ")))+(1.&
&0-xtit(i,j))**2.0)/((1.0-
epsm(i,j))*propev(ppres(i,j),temp,hlq(i,j),"VPTL")))
        end if
    else
        if (xtit(i,j)==0.0) then

```

```

        xtit(i,j)=1.0e-9
        epsm(i,j)=1.0e-9
    end if

rho_m=1.0/(((xtit(i,j))**2.0)/(epsm(i,j)*propev(ppres(i,j),temp,0.0,
"VGP "))+((1.0-xtit(i,j))**2.0)/((1.0-
        epsm(i,j))*propev(ppres(i,j),temp,hliq(i,j),"VFP ")))
    end if
end function rho_m

!*****
!*****!
!
!
!                               Calcul de l'enthalpie du liquide
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! i : position dans le canal
!
! j : 1 ou 2 pour un calcul stationnaire ou transitoire
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
!
!
!*****
!*****!

double precision function hliq(i,j,Ndisc,Gmel,ppres,hent,xtit,epsm)
    implicit none
    integer, intent(in) :: i,j,Ndisc
    double precision, dimension (Ndisc,2), intent(in) :: hent,
    ppres, Gmel
    double precision, dimension (Ndisc,2), intent(inout) :: epsm,
    xtit
    double precision, external :: propev
    integer :: k
    double precision :: T,h

```

```

    if (epsm(i,j)/=0.0) then
        h=propev(ppres(i,j),0.0,0.0,"HFP ")
        k=0
        boucle : do
            k=k+1
            if (h<propev(ppres(i,j),0.0,h,"HFP ")) then
                T=propev(ppres(i,j),0.0,h,"TPHL")
                hliq=hent(i,j)+epsm(i,j)/(1.0-
epsm(i,j))*propev(ppres(i,j),T,h,"VGP ")/propev&
                &(ppres(i,j),T,h,"VPTL")*(hent(i,j)-
propev(ppres(i,j),0.0,h,"HGP "))
            else
                hliq=hent(i,j)+epsm(i,j)/(1.0-
epsm(i,j))*propev(ppres(i,j),T,h,"VGP ")/propev&
                &(ppres(i,j),0.0,h,"VFP ")*(hent(i,j)-
propev(ppres(i,j),0.0,h,"HGP "))
            end if
            if (abs((hliq-h)/hliq)<1.0e-5 .OR. k>1000) then
                exit boucle
            end if
            h=hliq
        end do boucle
        if (hliq>propev(ppres(i,j),0.0,0.0,"HFP ")) then
            hliq=propev(ppres(i,j),0.0,0.0,"HFP ")
        end if
    else
        hliq=hent(i,j)
    end if
end function hliq

```

```

!*****
!*****!

```

```

!
!
!
!
!
!
!

```

Calcul de la temperature a la surface de la gaine

```

!Fait par: Majid FASSI FEHRI (31/12/2007)

```

```

!Entier:

```

```

! Ndisc : discretisation axiale du canal

```

```

! i : position dans le canal

```

```

! j : 1 ou 2 pour un calcul stationnaire ou transitoire

```

```

!Double precision :

```

```

! Gm : valeurs du flux massique en kg/m2/s

```

```

! p : valeurs de la pression en Pa

```

```

! h : valeurs de l'enthalpie en J/kg

```

```

!

```

```
!
!  
!  
! Calcul du titre et du taux de vide  
!  
!  
!  
!  
! Fait par: Majid FASSI FEHRI (31/12/2007)  
!  
! Entier:  
!  
! Ndisc : discretisation axiale du canal  
!  
! crg : 1 si le calcul a converge, 0 sinon  
!
```

```

! val : 1 ou 2 pour un calcul stationnaire ou transitoire
!
! Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! qws : flux de chaleur en W/m2
!
! Dh : diametre hydraulique en m
!
! theta : inclinaison de l'ecoulement par rapport a la verticale en
radians
!
!
! *****
! *****!

subroutine
upxeps(crg,val,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,qws,Dh,theta)
  implicit none
  integer, intent (out) :: crg
  integer, intent (in) :: val,Ndisc
  double precision, dimension (Ndisc,2), intent(in) ::
Gmel,ppres,hent
  double precision, dimension (Ndisc,2), intent(inout) ::
hlq,xtit,epsm
  double precision, dimension (Ndisc), intent(in) :: qws
  double precision, intent(in) :: Dh,theta
  double precision, external :: titrem, tauv, hliq
  double precision :: erre, errx, xtp, epstp
  integer :: i,j
  boucle2 : do i=1,Ndisc
    j=0
    crg=0
    boucle1 : do
      j=j+1
      xtp=titrem(Gmel(i,val),ppres(i,val),hlq(i,val),qws(i),hent(i,val),Dh
)
      epstp=tauv(xtp,Gmel(i,val),ppres(i,val),hlq(i,val),Dh,theta)
      errx=abs(xtp-xtit(i,val))
      erre=abs(epstp-epsm(i,val))
      xtit(i,val)=xtp
      epsm(i,val)=epstp
      hlq(i,val)=hliq(i,val,Ndisc,Gmel,ppres,hent,xtit,epsm)
      if (i>100000) then

```

```

        exit boucle1
    else if (erre<1.0e-5 .AND. errx<1.0e-5) then
        crg=1
        exit boucle1
    end if
end do boucle1
if (crg==0) then
    exit boucle2
end if
end do boucle2
end subroutine upxeps

```

correl.f90

```

!*****
!*****!
!
!
!           Correlation de Chexal-Lellouche
!
!           pour un système sans inversion de flux
!
!           Paramètre de distribution
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!
! x : titre de l'écoulement
!
! eps : taux de vide
!
! Dh : diametre hydraulique en m
!
! theta : inclinaison du tube en radian
!
!
!*****
!*****!

double precision function C0(x,Gm,p,h,eps,Dh,theta)
    implicit none
    double precision, external :: propev
    double precision, intent (in) :: eps,x,Gm,p,h,Dh,theta
    double precision :: Reg, Rel, Re, COH, COV, L, r, K0, A1, B1,
    C1, T, Ts, Fr
    double precision , parameter :: pcrit=21.617e6, pi=3.14159265359
    Ts=propev(p,T,h,"TSAT")

```

```

if (h<propev(p,T,h,"HFP ")) then
  T=propev(p,T,h,"TPHL")
else
  T=Ts
end if
Reg=x*Gm*Dh/propev(p,Ts,h,"VISV")
Rel=(1.0-x)*Gm*Dh/propev(p,T,h,"VISL")
if (Reg>Rel) then
  Re=Reg
else
  Re=Rel
end if
A1=1.0/(1.0-exp(-Re/6.0e4))
B1=min(0.8,A1)
if (h<propev(p,T,h,"HFP ")) then
  r=(1.0+1.57*propev(p,Ts,h,"VGP
)/propev(p,T,h,"VPTL"))/(1.0-B1)
  K0=B1+(1.0-B1)*(propev(p,Ts,h,"VGP
)/propev(p,T,h,"VPTL"))**(1.0/4.0)
else
  r=(1.0+1.57*propev(p,Ts,h,"VGP ") /propev(p,T,h,"VFP
"))/(1.0-B1)
  K0=B1+(1.0-B1)*(propev(p,Ts,h,"VGP ") /propev(p,T,h,"VFP
"))**(1.0/4.0)
end if
C1=4.0*pcrit**2.0/(p*(pcrit-p))
L=(1.0-exp(-C1*eps))/(1.0-exp(-C1))
COV=L/(K0+(1-K0)*eps**r)
COH=(1.0+eps**0.05*(1.0-eps)**2.0)*COV
Fr=(90.0-theta*180.0/pi)/90.0
C0=Fr*COV+(1.0-Fr)*COH
end function C0

!*****
!*****!
!
!
!           Correlation de Chexal-Lellouche
!
!           pour un système sans inversion de flux
!
!           Écart de vitesses
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!

```



```

! x : titre de l'ecoulement
!
! eps : taux de vide
!
! Dh : diametre hydraulique en m
!
! theta : inclinaison du tube en radian
!
!
!
!*****
*****!

```

```

double precision function vgj(x,Gm,p,h,eps,Dh,theta)
  implicit none
  double precision, external :: propev
  double precision, intent (in) :: eps, x, Gm, p, h,Dh,theta
  double precision, parameter :: D2=0.09144, g=9.81
  double precision :: Reg, Rel, Re, C10, C3, C7, C8, C4, C5, C2,
C9, A1, B1, vgj0, T, Ts
  Ts=propev(p,T,h,"TSAT")
  if (h<propev(p,T,h,"HFP ")) then
    T=propev(p,T,h,"TPHL")
  else
    T=Ts
  end if
  Reg=x*Gm*Dh/propev(p,Ts,h,"VISV")
  Rel=(1.0-x)*Gm*Dh/propev(p,T,h,"VISL")
  if (Reg>Rel) then
    Re=Reg
  else
    Re=Rel
  end if
  A1=1.0/(1.0-exp(-Re/6.0e4))
  B1=min(0.8,A1)
  C10=2.0*exp(-abs(Rel)/6.0e4)
  C3=max(0.5,C10)
  C7=(D2/Dh)**0.6
  C8=C7/(1-C7)
  if (C7>=1.0) then
    C4=1
  else
    C4=1.0/(1.0-exp(-C8))
  end if
  if (h<propev(p,T,h,"HFP ")) then
    C5=sqrt(150.0/(propev(p,T,h,"VPTL")/propev(p,Ts,h,"VGP ")))
    if (propev(p,T,h,"VPTL")/propev(p,Ts,h,"VGP ")<=18) then
      C2=0.4757*(log(propev(p,T,h,"VPTL")/propev(p,Ts,h,"VGP
")))**0.7
    else
      if (C5>=1) then
        C2=1.0
      else
        C2=1.0/(1.0-exp(-C5/(1.0-C5)))
      end if
    end if
  end if
  C9=(1.0-eps)**B1

```

```

        vgj0=1.41*((propev(p,T,h,"VPTL")-propev(p,Ts,h,"VGP
"))*propev(p,Ts,h,"SGM ")*g/(prop&
        &ev(p,T,h,"VPTL"))**2)**0.25*C2*C3*C4
    else
        C5=sqrt(150.0/(propev(p,T,h,"VFP ")/propev(p,Ts,h,"VGP ")))
        if (propev(p,T,h,"VFP ")/propev(p,Ts,h,"VGP ")<=18) then
            C2=0.4757*(log(propev(p,T,h,"VFP ")/propev(p,Ts,h,"VGP
")))**0.7
        else
            if (C5>=1) then
                C2=1.0
            else
                C2=1.0/(1.0-exp(-C5/(1.0-C5)))
            end if
        end if
        C9=(1.0-eps)**B1
        vgj0=1.41*((propev(p,T,h,"VFP ")-propev(p,Ts,h,"VGP
"))*propev(p,Ts,h,"SGM ")*g/(prop&
        &ev(p,T,h,"VFP "))*2)**0.25*C2*C3*C4
    end if
    vgj=C9*vgj0*cos(theta)
end function vgj

!*****
!*****!
!
!
!           Correlation de Chexal-Lellouche
!
!           pour un système sans inversion de flux
!
!           Écart de vitesses avec distribution
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!
! x : titre de l'écoulement
!
! eps : taux de vide
!
! Dh : diametre hydraulique en m
!
! theta : inclinaison du tube en radian
!
!
!*****
!*****!

```

```

double precision function v_gj(x,Gm,p,h,eps,Dh,theta)
  implicit none
  double precision, intent (in) :: x,Gm,eps,p,h,Dh,theta
  double precision, external :: propev, C0, vgj
  double precision :: T, Ts
  Ts=propev(p,T,h,"TSAT")
  if (h<propev(p,T,h,"HFP ")) then
    T=propev(p,T,h,"TPHL")
  else
    T=Ts
  end if
  if (h<propev(p,T,h,"HFP ")) then
    v_gj=vgj(x,Gm,p,h,eps,Dh,theta)+(C0(x,Gm,p,h,eps,Dh,theta)-
1.0)&
    &*Gm*((1.0-x)/propev(p,T,h,"VPTL")+x/propev(p,Ts,h,"VGP "))
  else
    v_gj=vgj(x,Gm,p,h,eps,Dh,theta)+(C0(x,Gm,p,h,eps,Dh,theta)-
1.0)&
    &*Gm*((1.0-x)/propev(p,T,h,"VFP ") +x/propev(p,Ts,h,"VGP "))
  end if
end function v_gj

!*****
!*****!
!
!
! Correlation de Chexal-Lellouche
!
! pour un système sans inversion de flux
!
! Taux de vide
!
!
!
! Fait par: Majid FASSI FEHRI (31/12/2007)
!
! Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!
! x : titre de l'écoulement
!
! eps : taux de vide
!
! Dh : diametre hydraulique en m
!
! theta : inclinaison du tube en radian
!
!
!*****
!*****!

```

```

double precision function tauv(x,Gm,p,h,Dh,theta)
  implicit none
  double precision, intent (in) :: x,Gm,p,h,Dh,theta
  double precision, external :: propev, C0, vgj
  double precision :: eps1, eps2, T,Ts
  Ts=propev(p,T,h,"TSAT")
  if (h<propev(p,T,h,"HFP ")) then
    T=propev(p,T,h,"TPHL")
  else
    T=Ts
  end if
  if (x==0.0) then
    tauv=0.0
  else if (x>0.0 .AND. x<1.0) then
    eps2=1.0-1.0e-5
    boucle : do
      if (h<propev(p,T,h,"HFP ")) then
        eps1=x/(C0(x,Gm,p,h,eps2,Dh,theta)*(x+(1.0-
x)*propev(p,Ts,h,"VGP ")&
&/propev(p,T,h,"VPTL"))+propev(p,Ts,h,"VGP ")&
&/Gm*vgj(x,Gm,p,h,eps2,Dh,theta))
      else
        eps1=x/(C0(x,Gm,p,h,eps2,Dh,theta)*(x+(1.0-
x)*propev(p,Ts,h,"VGP ")&
&/propev(p,T,h,"VFP ")&
&/Gm*vgj(x,Gm,p,h,eps2,Dh,theta))
      end if
      if (abs((eps1-eps2)/eps1)<1.0e-5) then
        exit boucle
      end if
      eps2=(eps1+eps2)/2.0
    end do boucle
    tauv=eps1
  else
    tauv=1.0
  end if
end function tauv

```

```

!*****
!*****!
!
!
! Correlation de Saha et Zuber
!
! Calcul de l'écart de température entre la saturation et l'état
sous-refroidi !
!
!
! Fait par: Majid FASSI FEHRI (31/12/2007)
!
! Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!

```

```

! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!
! Dh : diametre hydraulique en m
!
! q : flux de chaleur en W/m2
!
!
!
!*****
*****!

double precision function deltaT(Gm,p,h,q,Dh)
  implicit none
  double precision, external :: propev
  double precision, intent (in) :: Gm,p,q,h,Dh
  double precision, parameter :: Nu=455.0, St=0.0065
  double precision :: Pl, Ts, T
  Ts=propev(p,T,h,"TSAT")
  if (h<propev(p,T,h,"HFP ")) then
    T=propev(p,T,h,"TPHL")
  else
    T=Ts
  end if
  Pl=Gm*propev(p,T,h,"CPL ") *Dh/propev(p,T,h,"CONL")
  if (Pl<=70000.0) then
    deltaT=q*Dh/(propev(p,T,h,"CONL") *Nu)
  else
    deltaT=q/(Gm*propev(p,T,h,"CPL ") *St)
  end if
end function deltaT

!*****
*****!

!
!
! Correlation de Saha et Zuber
!
! Calcul du titre d'écoulement
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! hl : valeur de l'enthalpie du liquide en J/kg
!
! he : valeur de l'enthalpie de l'ecoulement en J/kg
!

```

```

! Dh : diametre hydraulique en m
!
! q : flux de chaleur en W/m2
!
!
!*****
*****!

```

```

double precision function titrem(Gm,p,h1,q,he,Dh)
  implicit none
  double precision, external :: propev, deltaT
  double precision, intent (in) :: Gm,p,q,h1,he,Dh
  double precision :: xd,T, xth, dT, dev
  dT=deltaT(Gm,p,h1,q,Dh)
  T=propev(p,T,h1,"TSAT")
  if (he>=propev(p,T,h1,"HGP ")) then
    titrem=1.0
  else
    xd=-propev(p,T-dT,h1,"CPL ") *dT/propev(p,T,h1,"HGF ")
    xth=propev(p,T,he,"XPH ")
    titrem=(xth-xd*exp(xth/xd-1.0))/(1.0-xd*exp(xth/xd-1.0))
    dev=1.0-exp(xth/xd-1.0)
    if (dev<0.0) then
      titrem=0.0
    end if
  end if
end function titrem

```

```

!*****
*****!
!
!
! Correlation de Müller-Steinhagen (et Coolbrooke)
!
! calcul du terme de friction
!
!
!
!
! Fait par: Majid FASSI FEHRI (31/12/2007)
!
! Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!
! Dh : diametre hydraulique en m
!

```

```

! epsr : rugosite de la surface en m
!
!
!*****
*****!

```

```

double precision function dpfric(x,Gm,p,h,Dh,epsr)
  implicit none
  double precision, external :: propev
  double precision, intent (in) :: x,Gm,p,h,Dh,epsr
  double precision :: Rel, Reg, Cl, Cg, dpl, dpg, T, Ts ,Cll, Cgg
  Ts=propev(p,T,h,"TSAT")
  if (h<propev(p,T,h,"HFP ")) then
    T=propev(p,T,h,"TPHL")
  else
    T=Ts
  end if
  Reg=Gm*Dh/propev(p,Ts,h,"VISV")
  Rel=Gm*Dh/propev(p,T,h,"VISL")
  if (x<1.0) then
    if (Rel<=1187.0) then
      Cl=64.0/Rel
    else
      !Blasius
      !Cl=0.3164/Rel**(1.0/4.0)
      !Coolbrooke
      Cll=1.0
      boucle1 : do
        Cl=1.0/(4.0*(log10((epsr/Dh)/3.7+2.51/(sqrt(Cll)*Rel)))*2.0)
        if (abs(Cll-C1)/C1<1.0e-5) then
          Cll=C1
          exit boucle1
        else
          Cll=C1
        end if
      end do boucle1
      !Swamee-Jain
      !Cl=1.325/((log(epsr/Dh/3.7+5.74/(Rel**0.9)))*2.0)
    end if
  else
    Cl=0.0
  end if
  if (x>0.0) then
    if (Reg<=1187.0) then
      Cg=64.0/Reg
    else
      !Blasius
      !Cg=0.3164/Reg**(1.0/4.0)
      !Coolbrooke
      Cgg=1.0
      boucle2 : do

```

```

Cg=1.0/(4.0*(log10((epsr/Dh)/3.7+2.51/(sqrt(Cgg)*Reg))))**2.0
    if (abs(Cgg-Cg)/Cg<1.0e-5) then
        Cgg=Cg
        exit boucle2
    else
        Cgg=Cg
    end if
end do boucle2
!Swamee-Jain
!Cg=1.325/((log(epsr/Dh/3.7+5.74/(Reg**0.9))))**2.0
end if
else
    Cg=0.0
end if
if (h<propev(p,T,h,"HFP ")) then
    dpl=C1*Gm**2.0/(2.0*propev(p,T,h,"VPTL")*Dh)
else
    dpl=C1*Gm**2.0/(2.0*propev(p,T,h,"VFP ")*Dh)
end if
dpg=Cg*Gm**2.0/(2.0*propev(p,Ts,h,"VGP ")*Dh)
dpfric=(dpl+2.0*(dpg-dpl)*x)*(1.0-x)**(1.0/3.0)+dpg*x**3.0
end function dpfric

```

```

!*****
!*****!

```

```

!
```

```

!
```

```

! Correlation de Chen

```

```

!
```

```

! Coefficient de transfert de chaleur pour l'état sous-refroidi et saturé
!
```

```

!
```

```

!
```

```

!Fait par: Majid FASSI FEHRI (31/12/2007)

```

```

!
```

```

!Double precision :

```

```

!
```

```

! x : titre de l'ecoulement

```

```

!
```

```

! Gm : valeur du flux massique en kg/m2/s

```

```

!
```

```

! p : valeur de la pression en Pa

```

```

!
```

```

! h : valeur de l'enthalpie en J/kg

```

```

!
```

```

! Dh : diametre hydraulique en m

```

```

!
```

```

! Tw : temperature externe de la gaine en K

```

```

!
```

```

! Dgext : diametre externe de la gaine en m

```

```

!
```

```

!
```

```

!
```



```
!*****
*****!
```

```
double precision function hebu(x,Gm,p,h,Tw,Dh,Dgext)
  implicit none
  double precision, external :: propev
  double precision, intent (in) :: x,Gm,p,Tw,h,Dh,Dgext
  double precision :: T, Rel, F, S, ha, hb, dTsats, dp, xm, Re, Ts
  Ts=propev(p,T,h,"TSAT")
  if (h<propev(p,T,h,"HFP ")) then
    T=propev(p,T,h,"TPHL")
  else
    T=Ts
  end if
  Rel=Gm*(1.0-x)*Dh/propev(p,T-1.0e-5,h,"VISL")
  if (h<propev(p,T,h,"HFP ")) then
    !Parametre de Martinelli
    xm=(x/(1.0-x))**0.9*(propev(p,T,h,"VPTL")/propev(p,Ts,h,"VGP
  ))**0.5*(propev(p,Ts+1.&
    &0e-5,h,"VISV")/propev(p,T-1.0e-5,h,"VISL"))**0.1
    !Facteur des nombres de Reynolds
    !if (xm>0.1) then
      F=2.35*(0.213+xm)**0.8 !**0.736
    !else
      ! F=1.0
    !end if
    Re=Rel*F**1.25
    !Facteur de suppression
    S=(1.0+2.53e-6*Re**1.17)**(-1.0)
    dTsats=Tw-T
    dp=propev(p,Tw,h,"PSAT")-propev(p,T,h,"PSAT")
    !Equation de Dittus-Boelter
    ha=0.023*propev(p,T,h,"CONL")/Dh*(Rel)**0.8*(propev(p,T-
    1.0e-5,h,"NPR "))**0.4*F
    !Equation de Zuber-Forster
    hb=0.00122*(propev(p,T,h,"CONL"))**0.79*(propev(p,T,h,"CPL
    "))**0.45*(propev(p,T,h,"V&
    &PTL"))**0.49/((propev(p,T,h,"VISL"))**0.29*(propev(p,Ts,h,"SGM
    "))**0.5*(propev(p,Ts&
      &h,"HGP ") -h)**0.24*(propev(p,Ts,h,"VGP
    "))**0.24)*dTsats**0.24*dp**0.75*S
  else
    !Parametre de Martinelli
    xm=(x/(1.0-x))**0.9*(propev(p,T,h,"VFP ") /propev(p,Ts,h,"VGP
  ))**0.5*(propev(p,Ts+1.&
    &0e-5,h,"VISV")/propev(p,T-1.0e-5,h,"VISL"))**0.1
    !Facteur des nombres de Reynolds
    !if (xm>0.1) then
      F=2.35*(0.213+xm)**0.8 !**0.736
    !else
      ! F=1.0
    !end if
    Re=Rel*F**1.25
    !Facteur de suppression
    S=(1.0+2.53e-6*Re**1.17)**(-1.0)
    dTsats=Tw-T
```

```

dp=propev(p,Tw,h,"PSAT")-propev(p,T,h,"PSAT")
!Equation de Dittus-Boelter
ha=0.023*propev(p,T,h,"CONL")/Dh*(Rel)**0.8*(propev(p,T-
1.0e-5,h,"NPR "))**0.4*F
!Equation de Zuber-Forster
hb=0.00122*(propev(p,T,h,"CONL"))**0.79*(propev(p,T,h,"CPL
"))**0.45*(propev(p,T,h,"V&
&FP "))**0.49/( (propev(p,T-1.0e-
5,h,"VISL"))**0.29*(propev(p,Ts,h,"SGM "))**0.5*(prop&
&ev(p,Ts,h,"HGF "))**0.24*(propev(p,Ts,h,"VGP
"))**0.24)*dTsat**0.24*dp**0.75*S
end if
!Equation de Chen
hebu=(ha+hb)*(1.0+0.912*Rel**(-0.1)*(propev(p,T-1.0e-5,h,"NPR
"))**0.4*(1.0-2.0043*exp(-D&
&h/Dgext)))
end function hebu

```

```

!*****
*****!
!
!
!
!
!
Correlation de Markoczy
!
Coefficient de transfert de chaleur pour un écoulement une
phase !
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Double precision :
!
! Gm : valeur du flux massique en kg/m2/s
!
! p : valeur de la pression en Pa
!
! h : valeur de l'enthalpie en J/kg
!
! Dh : diametre hydraulique en m
!
! Dgext : diametre externe de la gaine en m
!
!
!
!*****
*****!

```

```
double precision function hfsr(Gm,p,h,Dh,Dgext)
  implicit none
  double precision, external :: propev
  double precision, intent (in) :: Gm,p,h,Dh,Dgext
  double precision :: psi, B, Rel, T
  T=propev(p,T,h,"TPHL")
  B=Dh/Dgext
```

```

        Rel=Gm*Dh/propev(p,T,h,"VISL")
        psi=1.0+0.912*Rel**(-0.1)*(propev(p,T-1.0e-5,h,"NPR
"))**0.4*(1.0-2.0043*exp(-B))
        ! Dittus-Boelter
        hfsr=psi*0.023*Rel**0.8*(propev(p,T,h,"NPR
"))**0.4*propev(p,T,h,"CONL")/Dh
end function hfsr

```

```

!*****
!*****!
!
!
!           Conductivite thermique entre la gaine
!
!           et le combustible (AECL)
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Tgi : temperature de la gaine interne (K)
!
!Tgi : temperature du combustible externe (K)
!
!Tgi : temperature du combustible interne (K)
!
!
!*****
!*****!

```

!Coefficient de transfert de chaleur entre le combustible et la gaine

```

double precision function hgap(Tgi,Tcoe,Tcoi)
    implicit none
    ! double precision, external :: propur, propzir
    double precision, intent (in) :: Tgi,Tcoe,Tcoi
    ! double precision :: hf,hs,km,R,H,drp,Pinter,Tm,gap,Tgaz,AA,BB
    ! gap=(Dgint-Dcomb)/2.0
    ! Tm=(Tcoe+Tcoi)/2.0
    ! drp=propur(Tm,"d")-propur(298.15,"d")
    ! Pinter=2.0*tg*2.0e11*drp/(0.85*Dgint)
    ! Tgaz=(Tgi+Tcoe)/2.0
    ! AA=1.5*(Dgint+Dcomb)/2.0
    ! BB=gfo*(Tgaz/273.15)**(0.5+sfo))*(0.1013e6/Pgaz)
    ! hf=kfo/(AA+BB+gap)
    !
    km=2.0*propur(Tcoe,"k")*propzir(Tgi,"k")/(propur(Tcoe,"k")+propzir(T
    gi,"k"))
    ! R=sqrt(((Dgint/2.0)**2.0+(Dcomb/2.0)**2.0)/2.0)
    ! H=3.0*YS
    ! hs=km/(8.6e-3*sqrt(R))*sqrt(Pinter)/H
    ! hgap=hf+hs
    hgap=10.0e3

```

```
end function hgap
```

initial.f90

```
!*****
!*****!
!
!
!               Initialise les paramètres de calculs
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale du crayon
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! Ttig : valeurs de la temperature en K
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! Ginit : flux massique a l'entree du canal en kg/m2/s
!
! pinit : pression a la sortie du canal en Pa
!
! hinit : enthalpie a l'entree du canal en J/kg
!
! tinit : temperature a l'entree du canal en K
!
! tinic : temperature initiale du combustible en K
!
!
!
!*****
!*****!
```

```
subroutine
```

```
initparam(Ndisc,Mdisc,Gmel,ppres,hent,epsm,xtit,hlq,Ttig,Ginit,pinit  
,hinit,tinit,tcini)
```

```

        implicit none
        integer, intent(in) :: Ndisc,Mdisc
        double precision, dimension (Ndisc,2), intent(inout) :: hent,
        ppres, Gmel, epsm, xtit, hlq
        double precision, dimension (Mdisc,Ndisc,2), intent(inout) ::
        Ttig
        double precision, intent(inout) :: Ginit,hinit,pinit,tinit,tcini
        double precision, external :: propev
        hinit=propev(pinit,tinit,0.0,'HPTL')
        Gmel(:,:)=Ginit
        hent(:,:)=hinit
        hlq(:,:)=hinit
        ppres(:,:)=pinit
        epsm(:,:)=0.0
        xtit(:,:)=0.0
        Ttig(:,:,:)=tcini
    end subroutine initparam

```

inoutdat.f90

```

!*****
!*****!
!
!
!                               Sortie densite et temperature du caloporteur
!
!                               et temperature du combustible
!
!
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale du crayon
!
! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
! val : 1 ou 2 pour un calcul stationnaire ou transitoire
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!

```

```

! Ttig : valeurs de la temperature en K
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! Tcal : temperature du caloporteur en K
!
! Rcal : densite du caloporteur en kg/m3
!
! Tcomb : temperature du combustible en K
!
!
! *****
! *****!

```

```

subroutine
neutro(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,epsm,hlq&
&,Ttig,Tcal,Rcal,Tcomb,val)
implicit none
integer, intent (in) :: Ndisc,Mdisc,Mcdisc,Mgdisc,val
double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,xtit,epsm,hlq
double precision, dimension(Mdisc,Ndisc,2),intent(in) :: Ttig
double precision, dimension(Ndisc), intent(out) ::
Tcal,Rcal,Tcomb
double precision, external :: rhom, propev
integer :: i,j
double precision :: tempcomb
do i=1,Ndisc
Rcal(i)=rhom(i,val,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)
tempcomb=0.0
do j=1,Mcdisc-1
tempcomb=tempcomb+(Ttig(j+1,i,val)+Ttig(j,i,val))/2.0*&
&(((db1e(j))*2.0-(db1e(j-1))*2.0)/(db1e(Mcdisc-
1))**2.0)
end do
Tcomb(i)=tempcomb
if (hlq(i,val)<propev(ppres(i,val),0.0,0.0,"HFP ")) then
Tcal(i)=propev(ppres(i,val),0.0,hlq(i,val),"TPHL")
else
Tcal(i)=propev(ppres(i,val),0.0,hlq(i,val),"TSAT")
end if
end do
end subroutine neutro

```

matrice.f90

```

! *****
! *****!

```

```

!
!
!                               Remplissage de la matrice de
!                               conservation de la masse (MATCM)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! matcm : matrice du systeme Ax=b
!
!
!
!*****
!*****!

subroutine initmatcm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matcm)
  implicit none
  integer, intent(in) :: Ndisc
  double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,xtit,epsm,hlq
  double precision, dimension(Ndisc,Ndisc), intent(out) :: matcm
  integer :: i,j
  do i=1,Ndisc-1
    do j=1,Ndisc
      if (i==j-1) then
        matcm(i,j)=1.0
      elseif (i-1==j) then
        matcm(i,j)=-1.0
      else
        matcm(i,j)=0.0
      endif
    end do
  end do
  do i=1,Ndisc-3
    matcm(Ndisc,i)=0.0
  end do
  matcm(Ndisc,Ndisc-2)=1.0

```

```

      matcm(Ndisc,Ndisc-1)=-4.0
      matcm(Ndisc,Ndisc)=3.0
end subroutine initmatcm

```

```

!*****
!*****!
!
!
!           Remplissage de la matrice de
!
!           conservation de la quantité de mouvement (MATCQM)
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! matcqm : matrice du systeme Ax=b
!
!
!
!*****
!*****!

```

```

subroutine initmatcqm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matcqm)
  implicit none
  integer, intent (in) :: Ndisc
  double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,xtit,epsm,hlq
  double precision, dimension(Ndisc,Ndisc), intent(out) :: matcqm
  integer :: i,j
  do i=2,Ndisc
    do j=1,Ndisc
      if (i==j-1) then
        matcqm(i,j)=1.0
      elseif (i-1==j) then
        matcqm(i,j)=-1.0
      else
        matcqm(i,j)=0.0
      end if
    end do
  end do

```



```

        endif
      end do
    end do
    do i=4,Ndisc
      matcqm(1,i)=0.0
    end do
    matcqm(1,1)=-3.0
    matcqm(1,2)=4.0
    matcqm(1,3)=-1.0
  end subroutine initmatcqm

```

```

!*****
!*****!
!
!
!                               Remplissage de la matrice de
!
!                               conservation de l'énergie à l'état statique (MATCE)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! matce : matrice du systeme Ax=b
!
!
!*****
!*****!

```

```

subroutine initmatcestat(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,matce)
  implicit none
  integer, intent(in) :: Ndisc
  double precision, dimension(Ndisc,2), intent(in) ::
    Gmel,ppres,hent,xtit,epsm,hlq
  double precision, dimension(Ndisc,Ndisc), intent(out) :: matce
  integer :: i,j

```



```

!
!
!*****
*****!

```

```

subroutine
initmatce(Ndisc,Gmel,ppres,hent,xtit,eps,h,q,matce,dz,dt)
  implicit none
  integer, intent(in) :: Ndisc
  double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,xtit,eps,h,q
  double precision, dimension(Ndisc,Ndisc), intent(out) :: matce
  double precision, intent(in) :: dt,dz
  double precision, external :: propev
  double precision, external :: rhom
  integer :: i,j
  do i=1,Ndisc-1
    do j=1,Ndisc
      if (j-1==i) then
        matce(i,j)=Gmel(j,2)
      else if (i==j) then
        matce(i,j)=2.0*dz/dt*rhom(i,2,&
&Ndisc,Gmel,ppres,hent,xtit,eps,h,q)
      else if (i-1==j) then
        matce(i,j)=-Gmel(j,2)
      else
        matce(i,j)=0.0
      end if
    end do
  end do
  do i=1,Ndisc-3
    matce(Ndisc,i)=0.0
  end do
  matce(Ndisc,Ndisc-2)=Gmel(Ndisc-2,2)
  matce(Ndisc,Ndisc-1)=-4.0*Gmel(Ndisc-1,2)
  matce(Ndisc,Ndisc)=3.0*Gmel(Ndisc,2)+2.0*dz/dt*rhom(Ndisc,2,&
&Ndisc,Gmel,ppres,hent,xtit,eps,h,q)
end subroutine initmatce

```

```

!*****
*****!

```

```

!
!
!
!          Remplissage de la matrice de transfert
!
!          de chaleur à l'état statique (MATG)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!

```

```

! Mdisc : discretisation radiale
!
! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
!Double precision :
!
! Ttig : temperature du crayon en K
!
! mattg : matrice du systeme Ax=b
!
! drc : pas pour le combustible en m
!
! drg : pas pour la gaine en m
!
! Dcomb : diametre du combustible en m
!
! tg : taille de l'interstice en m
!
!
!
!*****
*****!

```

```

subroutine
initmattgstat(Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,drc,drg,Dcomb,tg
)
  implicit none
  integer, intent(in) :: Ndisc,Mdisc,Mcdisc,Mgdisc
  double precision, dimension(Mdisc,Ndisc,2), intent(in) :: Ttig
  double precision, dimension(Mdisc,Mdisc,Ndisc), intent(out) ::
mattg
  double precision, intent(in) :: Dcomb,tg,drc,drg
  double precision, external :: propur, propzir, hgap
  integer :: i,j,k
  do k=1,Ndisc
    do i=1,Mdisc
      do j=1,Mdisc
        if (i==j+1 .AND. i>1 .AND. i<Mcdisc) then
          mattg(i,j,k)=-propur(Ttig(i,k,1),"k")*(1.0-
drc/(2.0*dble(i-1)*drc))
        else if (i==j .AND. i>1 .AND. i<Mcdisc) then
          mattg(i,j,k)=2.0*propur(Ttig(i,k,1),"k")
        else if (i==j-1 .AND. i>1 .AND. i<Mcdisc) then
          mattg(i,j,k)=-
propur(Ttig(i,k,1),"k")*(1.0+drc/(2.0*dble(i-1)*drc))
        else if (i==j .AND. i>Mcdisc .AND. i<Mdisc-1) then
          mattg(i,j,k)=-propzir(Ttig(i+1,k,1),"k")*(1.0-
drg/(2.0*(Dcomb/2.0+tg+drg*&
&dble(i-Mcdisc))))
        else if (i==j-1 .AND. i>Mcdisc .AND. i<Mdisc-1) then
          mattg(i,j,k)=2.0*propzir(Ttig(i+1,k,1),"k")
        else if (i==j-2 .AND. i>Mcdisc .AND. i<Mdisc-1) then
          mattg(i,j,k)=-
propzir(Ttig(i+1,k,1),"k")*(1.0+drg/(2.0*(Dcomb/2.0+tg+drg*&
&dble(i-Mcdisc))))

```



```

! Mgdisc : discretisation de la gaine
!
!Double precision :
!
! Ttig : temperature du crayon en K
!
! mattg : matrice du systeme Ax=b
!
! drc : pas pour le combustible en m
!
! drg : pas pour la gaine en m
!
! dt : pas de temps en s
!
! Dcomb : diametre du combustible en m
!
! tg : taille de l'interstice en m
!
!
!
!*****
*****!

```

```

subroutine
initmattg(Ndisc,Mdisc,Mcdisc,Mgdisc,Ttig,mattg,drc,drg,dt,Dcomb,tg)
  implicit none
  integer, intent (in) :: Ndisc,Mdisc,Mcdisc,Mgdisc
  double precision, dimension(Mdisc,Ndisc,2), intent(in) :: Ttig
  double precision, dimension(Mdisc,Mdisc,Ndisc), intent(out) ::
mattg
  double precision, intent (in) :: Dcomb,tg,drc,drg,dt
  double precision, external :: propur, propzir, hgap
  double precision :: T
  integer :: i,j,k
  do k=1,Ndisc
    do i=1,Mdisc
      do j=1,Mdisc
        if (i==j+1 .AND. i>1 .AND. i<Mcdisc) then
          mattg(i,j,k)=-propur(Ttig(i,k,2),"k")*(1.0-
drc/(2.0*dble(i-1)*drc))
        else if (i==j .AND. i>1 .AND. i<Mcdisc) then
          mattg(i,j,k)=(2.0+propur(Ttig(i,k,2),"c")*propur(Ttig(i,k,2),"r")/pr
opur(&
&Ttig(i,k,2),"k")*drc**2.0/dt)*propur(Ttig(i,k,2),"k")
          else if (i==j-1 .AND. i>1 .AND. i<Mcdisc) then
            mattg(i,j,k)=-
propur(Ttig(i,k,2),"k")*(1.0+drc/(2.0*dble(i-1)*drc))
          else if (i==j .AND. i>Mcdisc .AND. i<Mdisc-1) then
            mattg(i,j,k)=-propzir(Ttig(i+1,k,2),"k")*(1.0-
drg/(2.0*(Dcomb/2.0+tg+drg*&
&dble(i-Mcdisc))))
          else if (i==j-1 .AND. i>Mcdisc .AND. i<Mdisc-1) then
            mattg(i,j,k)=(2.0+propzir(Ttig(i+1,k,2),"c")*propzir(Ttig(i+1,k,2),"
r")/p&

```

```

&propzir(Ttig(i+1,k,2),"k")*drg**2.0/dt)*propzir(Ttig(i+1,k,2),"k")
      else if (i==j-2 .AND. i>Mcdisc .AND. i<Mdisc-1) then
        mattg(i,j,k)=-
propzir(Ttig(i+1,k,2),"k")*(1.0+drg/(2.0*(Dcomb/2.0+tg+drg*&
      &dble(i-Mcdisc))))
      else if (i==1 .AND. j==1) then
        mattg(i,j,k)=-3.0
      else if (i==1 .AND. j==2) then
        mattg(i,j,k)=4.0
      else if (i==1 .AND. j==3) then
        mattg(i,j,k)=-1.0
      else if (i==Mcdisc .AND. j==Mcdisc) then

mattg(i,j,k)=2.0*drg*hgap(Ttig(Mcdisc+1,k,2),Ttig(Mcdisc,k,1),Ttig(1
,k,2)&
      &)/propzir(Ttig(Mcdisc+1,k,2),"k")
      else if (i==Mcdisc .AND. j==Mcdisc+1) then
        mattg(i,j,k)=-3.0-
2.0*drg*hgap(Ttig(Mcdisc+1,k,2),Ttig(Mcdisc,k,2),Ttig(1&
      &k,2))/propzir(Ttig(Mcdisc+1,k,2),"k")
      else if (i==Mcdisc .AND. j==Mcdisc+2) then
        mattg(i,j,k)=4.0
      else if (i==Mcdisc .AND. j==Mcdisc+3) then
        mattg(i,j,k)=-1.0
      else if (j==Mdisc-2 .AND. i==Mdisc-1) then
        mattg(i,j,k)=1.0
      else if (j==Mdisc-1 .AND. i==Mdisc-1) then
        mattg(i,j,k)=-4.0
      else if (j==Mdisc .AND. i==Mdisc-1) then
        mattg(i,j,k)=3.0
      else if (i==Mdisc .AND. j==Mdisc) then
        mattg(i,j,k)=1.0
      else
        mattg(i,j,k)=0.0
      end if
    end do
  end do
end do
end subroutine initmattg

```

mprop.f90

```

!*****
!*****!
!
!
!           Calcul des proprietes du dioxyde d'uranium
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!

```

```

!Double precision :
!
!T : temperature en K
!
!Caractere :
!
!prop : choix du calcul:
!
!  "k" conductivite thermique W/m/K
!
!  "c" chaleur specifique J/kg/K
!
!  "r" densite kg/m3
!
!  "d" elongation
!
!*****
!*****!

double precision function propur(T,prop)
  implicit none
  double precision , parameter :: rho_273=10963.0,
MUO2=270.02771/1000.0
  double precision , parameter :: C1=81.613, theta=548.68,
C2=2.285e-3, C3=2.360e7, Ea=185&
&31.7
  double precision :: rapp
  double precision , intent (in) :: T
  character ,intent (in) :: prop*1
  select case (prop)
    case ("k") ! Conductivite thermique de l'UO2 en W/(m.K)

propur=100.0/(7.5408+17.692*T/1000.0+3.6142*(T/1000.0)**2.0)+6400.0/
((T/1000.0)**&
&(5.0/2.0))*exp(-16.35/(T/1000.0))
    case ("r") ! Densite de l'UO2 en kg/m3  273K<T<3120K
      if (T<=923.0) then
        rapp=1.0/(0.9973+9.082e-6*T-2.705e-10*T**2.0+4.391e-
13*T**3.0)
      else if (T>923.0) then
        rapp=1.0/(0.99672+1.179e-5*T-2.429e-9*T**2.0+1.219e-
12*T**3.0)
      end if
      propur=rho_273*(rapp)**3.0
    case ("c") ! Chaleur specifique de l'UO2 en J/(kg.K)
298.15K<T<3120K

propur=MUO2*(C1*theta**2.0*exp(theta/T)/(T**2.0*(exp(theta/T)-
1.0)**2.0)+2.0*C&
&2*T+C3*Ea*exp(-Ea/T)/T**2.0) !semi-analytique
    case ("d") ! Ellongation de l'UO2  273K<T<3120K
      if (T<=923.0) then
        rapp=0.9973+9.082e-6*T-2.705e-10*T**2.0+4.391e-
13*T**3.0
      else if (T>923.0) then
        rapp=0.99672+1.179e-5*T-2.429e-9*T**2.0+1.219e-
12*T**3.0

```



```

        end if
        propur=rapp-1.0
    end select
end function propur

```

```

!*****
!*****!
!
!
!           Calcul des proprietes du zircaloy-2/4
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Double precision :
!
!T : temperature en K
!
!Caractere :
!
!prop : choix du calcul:
!
!   "k" conductivite thermique W/m/K
!
!   "c" chaleur specifique J/kg/K
!
!   "r" densite kg/m3
!
!*****
!*****!

```

```

double precision function propzir(T,prop)
    implicit none
    double precision, intent (in) :: T
    character :: prop*1
    select case (prop)
        case ("k") ! Conductivite thermique de Zircaloy-2/4 en
W/(m.K) 300K<T<1800K
            propzir=12.767-5.4348e-4*T+8.9818e-6*T**2.0
        case ("r") ! Densite du Zircaloy-2 en kg/m3
            if (T<=1113.5) then
                propzir=6595.2-0.1477*T
            else if (T>1113.5) then
                propzir=6690.0-0.1855*T
            end if
        case ("c") ! Chaleur specifique du Zircaloy-2 en J/(kg.K)
273K<T<2000K
            if (T<=1100.0) then
                propzir=255.66+0.1024*T
            else if (T>1320.0) then
                propzir=597.1-0.4088*T+1.565e-4*T**2.0
            else if (1100.0<T .AND. T<=1214.0) then
                propzir=255.66+0.1024*T+1058.4*exp(((T-
1213.8)**2.0)/719.61)

```

```

        else if (1214.0<T .AND. T<=1320.0) then
            propzir=597.1-0.4088*T+1.565e-
4*T**2.0+1058.4*exp(((T-1213.8)**2.0)/719.61)
        end if
    end select
end function propzir

```

```

!*****
!*****!

```

```

!
!
!          Calcul des proprietes de l'eau et de la vapeur (IAPWS-
IF97)          !
!
!

```

```

!Fait par: Majid FASSI FEHRI (31/12/2007)
!

```

```

!Double precision :
!

```

```

!p : pression en Pa
!

```

```

!T : temperature en K
!

```

```

!h : enthalpie en J/kg
!

```

```

!Caractere :
!

```

```

!act : choix du calcul:
!

```

```

!      "VISL" viscosite du liquide en Pa.s
!

```

```

!      "VISV" viscosite de la vapeur en Pa.s
!

```

```

!      "SGM " tension de surface N/m
!

```

```

!      "NPR " nombre de Prandl
!

```

```

!      "TPHL" temperature du liquide en K
!

```

```

!      "TPH " temperature a saturation en K
!

```

```

!      "HPTL" enthalpie du liquide en J/kg
!

```

```

!      "HPTD" enthalpie de la vapeur en J/kg
!

```

```

!      "HFP " enthalpie du liquide sature en J/kg
!

```

```

!      "HGP " enthalpie de la vapeur saturee en J/kg
!

```

```

!      "HGF " chaleur latente de vaporisation en J/kg
!

```

```

!      "VPTL" densite du liquide en kg/m3
!

```

```

!      "VPTD" densite de la vapeur en kg/m3
!

```

```

!      "VFP " densite du liquide sature en kg/m3
!
!      "VGP " densite de la vapeur saturee en kg/m3
!
!      "XPH " titre thermodynamique
!
!      "TSAT" temperature de saturation en K
!
!      "PSAT" pression a saturation en Pa
!
!      "CONL" conductivite thermique du liquide W/m/K
!
!      "CONV" conductivite thermique de la vapeur W/m/K
!
!      "CPL " chaleur specifique du liquide J/kg/K
!
!      "CPD " chaleur specifique de la vapeur J/kg/K
!
!*****
*****!

```

```

double precision function propev(p,T,h,choix)
  implicit none
  integer:: ircnt
  character :: cright*60, title1*40, title2*40, version*10, year*4
  double precision , intent (in) :: p, T, h
  double precision :: pin, Tin, hin
  character, intent (in) :: choix*4
  double precision, external :: VISL, VISV, SURFT, PRANDL, TPHL,
  TPH, HPTL, HPTD, HFP, HGP,&
  & VPTL, VPTD, VFP, VGP, TSATP, CONDL, CONDV, CPPTL, CPPTD, PSATT
  call SPINIT(ircnt,cright,title1,title2,version,year)
  pin=p*1.45037738e-4
  Tin=(T-273.15)*9.0/5.0+32.0
  hin=h*0.429922614e-3
  select case (choix)
    case ("VISL")
      propev=1.48816394*VISL(pin,Tin) !Pa.s
    case ("VISV")
      propev=1.48816394*VISV(pin,Tin) !Pa.s
    case ("SGM ")
      propev=14.59390294*SURFT(TSATP(pin)) !N/m
    case ("NPR ")
      propev=PRANDL(pin,Tin)
    case ("TPHL")
      propev=(TPHL(pin,hin)-32.0)/1.8+273.15 !K
    case ("TPH ") !NU
      propev=(TPH(pin,hin)-32.0)/1.8+273.15 !K
    case ("HPTL")
      propev=HPTL(pin,Tin)*2.326e3 !J/kg
    case ("HPTD") !NU
      propev=HPTD(pin,Tin)*2.326e3 !J/kg
    case ("HFP ")
      propev=HFP(pin)*2.326e3 !J/kg
    case ("HGP ")
      propev=HGP(pin)*2.326e3 !J/kg
    case ("HGF ")

```

```

        propev=(HGP(pin)-HFP(pin))*2.326e3 !J/kg
    case ("VPTL")
        propev=1.0/VPTL(pin,Tin)*16.01846337 !kg/m^3
    case ("VPTD") !NU
        propev=1.0/VPTD(pin,Tin)*16.01846337 !kg/m^3
    case ("VFP ")
        propev=1.0/VFP(pin)*16.01846337 !kg/m^3
    case ("VGP ")
        propev=1.0/VGP(pin)*16.01846337 !kg/m^3
    case ("XPH ")
        propev=(hin-HFP(pin))/(HGP(pin)-HFP(pin)) !xth<=1
    case ("TSAT")
        propev=(TSATP(pin)-32.0)/1.8+273.15 !K
    case ("PSAT")
        propev=PSATT(Tin)/1.45037738e-4 !Pa
    case ("CONL")
        propev=CONDL(pin,Tin)*1.73073467 !W/m/K
    case ("CONV") !NU
        propev=CONDV(pin,Tin)*1.73073467 !W/m/K
    case ("CPL ")
        propev=CPPTL(pin,Tin)*4.1868e3 !J/kg/K
    case ("CPD ") !NU
        propev=CPPTD(pin,Tin)*4.1868e3 !J/kg/K
    end select
end function propev

```

vecteur.f90

```

!*****
!*****!
!
!
!           Remplissage du vecteur de conservation
!
!           de la masse à l'état statique (VECTCM)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!

```

```

! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! vectcm : vecteur du systeme Ax=b
!
! Ginit: flux massique à l'entrée du canal en kg/m2/s
!
! dz : pas axial en m
!
!
!
! *****
! *****!

```

```

subroutine
initvectcmstat (Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcm,Ginit,dz)
    implicit none
    integer, intent (in) :: Ndisc
    double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
    double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
    double precision, dimension(Ndisc), intent(out) :: vectcm
    double precision, intent(in):: Ginit,dz
    integer :: i
    vectcm(1)=Ginit
    do i=2,Ndisc
        vectcm(i)=0.0
    end do
end subroutine initvectcmstat

```

```

! *****
! *****!
!
!
!                               Remplissage du vecteur de conservation
!
!                               de la masse à l'état dynamique (VECTCM)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!

```

```

! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! vectcm : vecteur du systeme Ax=b
!
! Ginit: flux massique à l'entrée du canal en kg/m2/s
!
! dz : pas axial en m
!
! dt : pas de temps en s
!
!
!
!*****
*****!

subroutine
initvectcm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcm,Ginit,dz,dt)
    implicit none
    integer, intent (in) :: Ndisc
    double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
    double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
    double precision, dimension(Ndisc), intent(out) :: vectcm
    double precision, intent(in):: Ginit,dz,dt
    double precision, external :: rhom
    integer :: i
    vectcm(1)=-
2.0*dz/dt*(rhom(1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&-rhom(1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq))+Ginit
    do i=2,Ndisc
        vectcm(i)=-
2.0*dz/dt*(rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&-rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq))
    end do
end subroutine initvectcm

!*****
*****!
!
!
!                               Remplissage du vecteur de conservation
!
!                               de la quantité de mouvement à l'état statique (VECTCQM)
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!

```

```

!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! vectcqm : vecteur du systeme Ax=b
!
! pinit: pression a la sortie du canal en Pa
!
! dz : pas axial en m
!
! Dh : diametre hydraulique en m
!
! epsr : rugosite de la gaine en m
!
! theta : inclinaison du canal en radians
!
!
!
!*****
*****!

subroutine
initvectcqmstat(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcqm,pinit,dz
,Dh,epsr,theta)
    implicit none
    integer, intent (in) :: Ndisc
    double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
    double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
    double precision, dimension(Ndisc), intent(out) :: vectcqm
    double precision, intent(in):: pinit,dz,Dh,epsr, theta
    double precision, external :: propev, rhom, dpfric, v_gj
    double precision :: dpfr, dpacc, dpacc1, dpacc2, dpacc3, dpgrav,
T
    double precision, parameter :: g=9.81
    integer :: i
    T=0.0
    do i=1,Ndisc

dpfr=dpfric(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),Dh,epsr)*2.0*dz

```

```

dpgrav=rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)*g*cos(theta)*2.
0*dz
      if (i>1 .AND. i<Ndisc) then
        if (hlq(i-1,1)<propev(ppres(i-1,1),T,hent(i-1,1),"HFP
")) then
          T=propev(ppres(i-1,1),T,hlq(i-1,1),"TPHL")
          dpacc1=((Gmel(i-1,1))**2.0)/rhom(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &+eps(i-1,1)/(1.0-eps(i-1,1))*propev(ppres(i-
1,1),T,hent(i-1,1),"VPTL")&
          &*propev(ppres(i-1,1),T,hent(i-1,1),"VGP ")&
          &/rhom(i-1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &*(v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),eps(i-1,&
          &1),Dh,theta))**2.0
        else
          dpacc1=((Gmel(i-1,1))**2.0)/rhom(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &+eps(i-1,1)/(1.0-eps(i-1,1))*propev(ppres(i-
1,1),T,hent(i-1,1),"VFP ")&
          &*propev(ppres(i-1,1),T,hent(i-1,1),"VGP ")&
          &/rhom(i-1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &*(v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),eps(i-1,&
          &1),Dh,theta))**2.0
        end if
        if (hlq(i+1,1)<propev(ppres(i+1,1),T,hent(i+1,1),"HFP
")) then
          T=propev(ppres(i+1,1),T,hlq(i+1,1),"TPHL")

dpacc2=((Gmel(i+1,1))**2.0)/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &+eps(i+1,1)/(1.0-
eps(i+1,1))*propev(ppres(i+1,1),T,hent(i+1,1),"VPTL")&
          &*propev(ppres(i+1,1),T,hent(i+1,1),"VGP ")&
          &/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &*(v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),eps(i+1,&
          &1),Dh,theta))**2.0
        else
dpacc2=((Gmel(i+1,1))**2.0)/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &+eps(i+1,1)/(1.0-
eps(i+1,1))*propev(ppres(i+1,1),T,hent(i+1,1),"VFP ")&
          &*propev(ppres(i+1,1),T,hent(i+1,1),"VGP ")&
          &/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
          &*(v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),eps(i+1,&
          &1),Dh,theta))**2.0
        end if
        dpacc=(dpacc2-dpacc1)
      elseif (i==1) then
        if (hlq(i,1)<propev(ppres(i,1),T,hent(i,1),"HFP ")) then
          T=propev(ppres(i,1),T,hlq(i,1),"TPHL")

```



```

dpacc1=((Gmel(i,1))**2.0)/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
      &+epsm(i,1)/(1.0-
epsm(i,1))*propev(ppres(i,1),T,hent(i,1),"VPTL")&
      &*propev(ppres(i,1),T,hent(i,1),"VGP ")&
      &/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta))
**2.0
      else

dpacc1=((Gmel(i,1))**2.0)/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
      &+epsm(i,1)/(1.0-
epsm(i,1))*propev(ppres(i,1),T,hent(i,1),"VFP ")&
      &*propev(ppres(i,1),T,hent(i,1),"VGP ")&
      &/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta))
**2.0
      end if
      if (hlq(i+1,1)<propev(ppres(i+1,1),T,hent(i+1,1),"HFP
")) then
          T=propev(ppres(i+1,1),T,hlq(i+1,1),"TPHL")

dpacc2=((Gmel(i+1,1))**2.0)/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq)&
      &+epsm(i+1,1)/(1.0-
epsm(i+1,1))*propev(ppres(i+1,1),T,hent(i+1,1),"VPTL")&
      &*propev(ppres(i+1,1),T,hent(i+1,1),"VGP ")&
      &/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),epsm(i+1,&
&1),Dh,theta))**2.0
      else

dpacc2=((Gmel(i+1,1))**2.0)/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq)&
      &+epsm(i+1,1)/(1.0-
epsm(i+1,1))*propev(ppres(i+1,1),T,hent(i+1,1),"VFP ")&
      &*propev(ppres(i+1,1),T,hent(i+1,1),"VGP ")&
      &/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),epsm(i+1,&
&1),Dh,theta))**2.0
      end if
      if (hlq(i+2,1)<propev(ppres(i+2,1),T,hent(i+2,1),"HFP
")) then
          T=propev(ppres(i+2,1),T,hlq(i+2,1),"TPHL")

dpacc3=((Gmel(i+2,1))**2.0)/rhom(i+2,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq)&
      &+epsm(i+2,1)/(1.0-
epsm(i+2,1))*propev(ppres(i+2,1),T,hent(i+2,1),"VPTL")&
      &*propev(ppres(i+2,1),T,hent(i+2,1),"VGP ")&
      &/rhom(i+2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

```

```

&*(v_gj(xtit(i+2,1),Gmel(i+2,1),ppres(i+2,1),hlq(i+2,1),epsm(i+2,&
&1),Dh,theta))**2.0
    else

dpacc3=((Gmel(i+2,1))**2.0)/rhom(i+2,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq)&
    &+epsm(i+2,1)/(1.0-
epsm(i+2,1))*propev(ppres(i+2,1),T,hent(i+2,1),"VFP ")&
    &*propev(ppres(i+2,1),T,hent(i+2,1),"VGP ")&
    &/rhom(i+2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i+2,1),Gmel(i+2,1),ppres(i+2,1),hlq(i+2,1),epsm(i+2,&
&1),Dh,theta))**2.0
    end if
    dpacc=(-3.0*dpacc1+4.0*dpacc2-dpacc3)
    else
        if (hlq(i,1)<propev(ppres(i,1),T,hent(i,1),"HFP ")) then
            T=propev(ppres(i,1),T,hlq(i,1),"TPHL")

dpacc1=((Gmel(i,1))**2.0)/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
    &+epsm(i,1)/(1.0-
epsm(i,1))*propev(ppres(i,1),T,hent(i,1),"VPTL")&
    &*propev(ppres(i,1),T,hent(i,1),"VGP ")&
    &/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta))
**2.0
    else

dpacc1=((Gmel(i,1))**2.0)/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
    &+epsm(i,1)/(1.0-
epsm(i,1))*propev(ppres(i,1),T,hent(i,1),"VFP ")&
    &*propev(ppres(i,1),T,hent(i,1),"VGP ")&
    &/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta))
**2.0
    end if
    if (hlq(i-1,1)<propev(ppres(i-1,1),T,hent(i-1,1),"HFP
")) then
        T=propev(ppres(i-1,1),T,hlq(i-1,1),"TPHL")
        dpacc2=((Gmel(i-1,1))**2.0)/rhom(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
        &+epsm(i-1,1)/(1.0-epsm(i-1,1))*propev(ppres(i-
1,1),T,hent(i-1,1),"VPTL")&
        &*propev(ppres(i-1,1),T,hent(i-1,1),"VGP ")&
        &/rhom(i-1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
        &*(v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),epsm(i-1,&
        &1),Dh,theta))**2.0
    else
        dpacc2=((Gmel(i-1,1))**2.0)/rhom(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
        &+epsm(i-1,1)/(1.0-epsm(i-1,1))*propev(ppres(i-
1,1),T,hent(i-1,1),"VFP ")&

```

```

&*propev(ppres(i-1,1),T,hent(i-1,1),"VGP ")&
&/rhom(i-1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*(v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),epsm(i-1,&
&1),Dh,theta))**2.0
end if
if (hlq(i-2,1)<propev(ppres(i-2,1),T,hent(i-2,1),"HFP
")) then
T=propev(ppres(i-2,1),T,hlq(i-2,1),"TPHL")
dpacc3=((Gmel(i-2,1))**2.0)/rhom(i-
2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&+epsm(i-2,1)/(1.0-epsm(i-2,1))*propev(ppres(i-
2,1),T,hent(i-2,1),"VPTL")&
&*propev(ppres(i-2,1),T,hent(i-2,1),"VGP ")&
&/rhom(i-2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*(v_gj(xtit(i-2,1),Gmel(i-2,1),ppres(i-2,1),hlq(i-
2,1),epsm(i-2,&
&1),Dh,theta))**2.0
else
dpacc3=((Gmel(i-2,1))**2.0)/rhom(i-
2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&+epsm(i-2,1)/(1.0-epsm(i-2,1))*propev(ppres(i-
2,1),T,hent(i-2,1),"VFP ")&
&*propev(ppres(i-2,1),T,hent(i-2,1),"VGP ")&
&/rhom(i-2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*(v_gj(xtit(i-2,1),Gmel(i-2,1),ppres(i-2,1),hlq(i-
2,1),epsm(i-2,&
&1),Dh,theta))**2.0
end if
dpacc=(3.0*dpacc1-4.0*dpacc2+dpacc3)
end if
vectcqm(i)=(-dpfr-dpgrav-dpacc)
end do
vectcqm(Ndisc)=vectcqm(Ndisc)-pinit
end subroutine initvectcqmstat

```

```

!*****
!*****!

```

```

!
!
!           Remplissage du vecteur de conservation
!
!           de la quantité de mouvement à l'état dynamique
(VECTCQM)           !
!
!

```

```

!Fait par: Majid FASSI FEHRI (31/12/2007)
!

```

```

!Entier:
!

```

```

! Ndisc : discretisation axiale du canal
!

```

```

!Double precision :
!

```

```

! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! vectcqm : vecteur du systeme Ax=b
!
! pinit: pression a la sortie du canal en Pa
!
! dz : pas axial en m
!
! dt : pas de temps en s
!
! Dh : diametre hydraulique en m
!
! epsr : rugisite de la gaine en m
!
! theta : inclinaison du canal en radians
!
!
!
!*****
*****!

```

```

subroutine
initvectcqm(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectcqm,pinit,dz,dt,
Dh,epsr,theta)
  implicit none
  integer, intent (in) :: Ndisc
  double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
  double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
  double precision, dimension(Ndisc), intent(out) :: vectcqm
  double precision, intent(in):: pinit,dz,dt,Dh,epsr, theta
  double precision, external :: propev, rhom, dpfric, v_gj
  double precision :: dpfr, dpacc, dpacc1, dpacc2, dpacc3, dpgrav,
dpflu, T
  double precision, parameter :: g=9.81
  integer :: i
  T=0.0
  do i=1,Ndisc

dpfr=dpfric(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),Dh,epsr)*2.0*dz

dpgrav=rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)*g*cos(theta)*2.
0*dz

    dpflu=(Gmel(i,2)-Gmel(i,1))*2.0*dz/dt
    if (i>1 .AND. i<Ndisc) then

```

```

        if (hlq(i-1,2)<propev(ppres(i-1,2),T,hent(i-1,2),"HFP
")) then
            T=propev(ppres(i-1,2),T,hlq(i-1,2),"TPHL")
            dpacc1=((Gmel(i-1,2))*2.0)/rhom(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
            &+epsm(i-1,2)/(1.0-epsm(i-1,2))*propev(ppres(i-
1,2),T,hent(i-1,2),"VPTL")&
            &*propev(ppres(i-1,2),T,hent(i-1,2),"VGP ")&
            &/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
            &*(v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,&
            &2),Dh,epsr))*2.0
        else
            dpacc1=((Gmel(i-1,2))*2.0)/rhom(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
            &+epsm(i-1,2)/(1.0-epsm(i-1,2))*propev(ppres(i-
1,2),T,hent(i-1,2),"VFP ")&
            &*propev(ppres(i-1,2),T,hent(i-1,2),"VGP ")&
            &/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
            &*(v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,&
            &2),Dh,epsr))*2.0
        end if
        if (hlq(i+1,2)<propev(ppres(i+1,2),T,hent(i+1,2),"HFP
")) then
            T=propev(ppres(i+1,2),T,hlq(i+1,2),"TPHL")

dpacc2=((Gmel(i+1,2))*2.0)/rhom(i+1,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq)&
            &+epsm(i+1,2)/(1.0-
epsm(i+1,2))*propev(ppres(i+1,2),T,hent(i+1,2),"VPTL")&
            &*propev(ppres(i+1,2),T,hent(i+1,2),"VGP ")&
            &/rhom(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

            &*(v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,&
            &2),Dh,epsr))*2.0
        else

dpacc2=((Gmel(i+1,2))*2.0)/rhom(i+1,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq)&
            &+epsm(i+1,2)/(1.0-
epsm(i+1,2))*propev(ppres(i+1,2),T,hent(i+1,2),"VFP ")&
            &*propev(ppres(i+1,2),T,hent(i+1,2),"VGP ")&
            &/rhom(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

            &*(v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,&
            &2),Dh,epsr))*2.0
        end if
        dpacc=(dpacc2-dpacc1)
    elseif (i==1) then
        if (hlq(i,2)<propev(ppres(i,2),T,hent(i,2),"HFP ")) then
            T=propev(ppres(i,2),T,hlq(i,2),"TPHL")

dpacc1=((Gmel(i,2))*2.0)/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
            &+epsm(i,2)/(1.0-
epsm(i,2))*propev(ppres(i,2),T,hent(i,2),"VPTL")&
            &*propev(ppres(i,2),T,hent(i,2),"VGP ")&

```

```

&/rho(m(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&*(v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,epsr)) *
*2.0
else
dpacc1=((Gmel(i,2))**2.0)/rho(m(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&+epsm(i,2)/(1.0-
epsm(i,2))*propev(ppres(i,2),T,hent(i,2),"VFP ") &
&*propev(ppres(i,2),T,hent(i,2),"VGP ") &
&/rho(m(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&*(v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,epsr)) *
*2.0
end if
if (hlq(i+1,2)<propev(ppres(i+1,2),T,hent(i+1,2),"HFP
")) then
T=propev(ppres(i+1,2),T,hlq(i+1,2),"TPHL")
dpacc2=((Gmel(i+1,2))**2.0)/rho(m(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&+epsm(i+1,2)/(1.0-
epsm(i+1,2))*propev(ppres(i+1,2),T,hent(i+1,2),"VPTL") &
&*propev(ppres(i+1,2),T,hent(i+1,2),"VGP ") &
&/rho(m(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&*(v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,&
&2),Dh,epsr))**2.0
else
dpacc2=((Gmel(i+1,2))**2.0)/rho(m(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&+epsm(i+1,2)/(1.0-
epsm(i+1,2))*propev(ppres(i+1,2),T,hent(i+1,2),"VFP ") &
&*propev(ppres(i+1,2),T,hent(i+1,2),"VGP ") &
&/rho(m(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&*(v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,&
&2),Dh,epsr))**2.0
end if
if (hlq(i+2,2)<propev(ppres(i+2,2),T,hlq(i+2,2),"HFP "))
then
T=propev(ppres(i+2,2),T,hlq(i+2,2),"TPHL")
dpacc3=((Gmel(i+2,2))**2.0)/rho(m(i+2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&+epsm(i+2,2)/(1.0-
epsm(i+2,2))*propev(ppres(i+2,2),T,hent(i+2,2),"VPTL") &
&*propev(ppres(i+2,2),T,hent(i+2,2),"VGP ") &
&/rho(m(i+2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &
&*(v_gj(xtit(i+2,2),Gmel(i+2,2),ppres(i+2,2),hlq(i+2,2),epsm(i+2,&
&2),Dh,epsr))**2.0
else
dpacc3=((Gmel(i+2,2))**2.0)/rho(m(i+2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq) &

```

```

&+epsm(i+2,2)/(1.0-
epsm(i+2,2))*propev(ppres(i+2,2),T,hent(i+2,2),"VFP ")&
&*propev(ppres(i+2,2),T,hent(i+2,2),"VGP ")&
&/rhom(i+2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i+2,2),Gmel(i+2,2),ppres(i+2,2),hlq(i+2,2),epsm(i+2,&
&2),Dh,epsr))**2.0
end if
dpacc=(-3.0*dpacc1+4.0*dpacc2-dpacc3)
else
if (hlq(i,2)<propev(ppres(i,2),T,hent(i,2),"HFP ")) then
T=propev(ppres(i,2),T,hlq(i,2),"TPHL")

dpacc1=((Gmel(i,2))*2.0)/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
&+epsm(i,2)/(1.0-
epsm(i,2))*propev(ppres(i,2),T,hent(i,2),"VPTL")&
&*propev(ppres(i,2),T,hent(i,2),"VGP ")&
&/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,epsr))*
*2.0
else

dpacc1=((Gmel(i,2))*2.0)/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,h
lq)&
&+epsm(i,2)/(1.0-
epsm(i,2))*propev(ppres(i,2),T,hent(i,2),"VFP ")&
&*propev(ppres(i,2),T,hent(i,2),"VGP ")&
&/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*(v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,epsr))*
*2.0
end if
if (hlq(i-1,2)<propev(ppres(i-1,2),T,hent(i-1,2),"HFP
")) then
T=propev(ppres(i-1,2),T,hlq(i-1,2),"TPHL")
dpacc2=((Gmel(i-1,2))*2.0)/rhom(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&+epsm(i-1,2)/(1.0-epsm(i-1,2))*propev(ppres(i-
1,2),T,hent(i-1,2),"VPTL")&
&*propev(ppres(i-1,2),T,hent(i-1,2),"VGP ")&
&/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*(v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,&
&2),Dh,epsr))**2.0
else
dpacc2=((Gmel(i-1,2))*2.0)/rhom(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&+epsm(i-1,2)/(1.0-epsm(i-1,2))*propev(ppres(i-
1,2),T,hent(i-1,2),"VFP ")&
&*propev(ppres(i-1,2),T,hent(i-1,2),"VGP ")&
&/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*(v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,&
&2),Dh,epsr))**2.0
end if

```

```

        if (hlq(i-2,2)<propev(ppres(i-2,2),T,hent(i-2,2),"HFP
")) then
            T=propev(ppres(i-2,2),T,hlq(i-2,2),"TPHL")
            dpacc3=((Gmel(i-2,2))**2.0)/rhom(i-
2,2,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
            &+eps(i-2,2)/(1.0-eps(i-2,2))*propev(ppres(i-
2,2),T,hent(i-2,2),"VPTL")&
            &*propev(ppres(i-2,2),T,hent(i-2,2),"VGP ")&
            &/rhom(i-2,2,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
            &*(v_gj(xtit(i-2,2),Gmel(i-2,2),ppres(i-2,2),hlq(i-
2,2),eps(i-2,2),
            &2),Dh,epsr))**2.0
        else
            dpacc3=((Gmel(i-2,2))**2.0)/rhom(i-
2,2,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
            &+eps(i-2,2)/(1.0-eps(i-2,2))*propev(ppres(i-
2,2),T,hent(i-2,2),"VFP ")&
            &*propev(ppres(i-2,2),T,hent(i-2,2),"VGP ")&
            &/rhom(i-2,2,Ndisc,Gmel,ppres,hent,xtit,eps,hlq)&
            &*(v_gj(xtit(i-2,2),Gmel(i-2,2),ppres(i-2,2),hlq(i-
2,2),eps(i-2,2),
            &2),Dh,epsr))**2.0
        end if
        dpacc=(3.0*dpacc1-4.0*dpacc2+dpacc3)
    end if
    vectcqm(i)=(-dpfr-dpgrav-dpacc-dpflu)
end do
vectcqm(Ndisc)=vectcqm(Ndisc)-pinit
end subroutine initvectcqm

```

```

!*****
!*****!
!
!
!           Remplissage du vecteur de conservation
!
!           de l'énergie à l'état statique (VECTCE)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!

```



```

! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! vectce : vecteur du systeme Ax=b
!
! Ginit : flux massique à l'entrée du canal en kg/m2/s
!
! hinit : enthalpie a l'entree du canal en J/kg
!
! qws : flux de puissance en W/m2
!
! dz : pas axial en m
!
! Dh : diametre hydraulique en m
!
! Pe : perimetre equivalent en m
!
! A : aire en m2
!
! epsr : rugisite de la gaine en m
!
! theta : inclinaison du canal en radians
!
!
!
!*****
*****!

```

```

subroutine
initvectcestat(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectce,qws,Ginit,
hinit,dz,Dh,Pe,A,epsr,theta)
    implicit none
    integer, intent (in) :: Ndisc
    double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
    double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
    double precision, dimension(Ndisc), intent(out) :: vectce
    double precision, dimension(Ndisc), intent(in) :: qws
    double precision, intent(in):: Ginit,hinit,dz,Dh,Pe,A,epsr,theta
    double precision, external :: propev, rhom, rho_m, rhoe, v_gj
    double precision, parameter :: g=9.81
    double precision :: vall, val2, val3, T
    double precision :: temp1,temp2,temp3,temp4
    integer :: i
    do i=1,Ndisc
        if (i>1 .AND. i<Ndisc) then
            if (hlq(i-1,1)<propev(ppres(i-1,1),T,hent(i-1,1),"HFP
")) then
                T=propev(ppres(i-1,1),T,hlq(i-1,1),"TPHL")
                vall=((Gmel(i-1,1))**3.0)/(2.0*(rhoe(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,1)*g*dble(i-1)*dz&

```

```

&*cos(theta)+epsm(i-1,1)*propev(ppres(i-
1,1),T,hent(i-1,1),"VGP ")*propev(ppr&
&es(i-1,1),T,hent(i-1,1),"VPTL")/rhom(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),epsm(i-1,1),Dh,theta)&
&*(propev(ppres(i-1,1),T,hent(i-1,1),"HGP ") -hlq(i-
1,1))
else
val1=((Gmel(i-1,1))*3.0)/(2.0*(rhoe(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))*2.0)+Gmel(i-1,1)*g*dbble(i-1)*dz&
&*cos(theta)+epsm(i-1,1)*propev(ppres(i-
1,1),T,hent(i-1,1),"VGP ")*propev(ppr&
&es(i-1,1),T,hent(i-1,1),"VFP ") /rhom(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),epsm(i-1,1),Dh,theta)&
&*propev(ppres(i-1,1),T,hent(i-1,1),"HGF ")
end if
if (hlq(i+1,1)<propev(ppres(i+1,1),T,hent(i+1,1),"HFP
")) then
T=propev(ppres(i+1,1),T,hlq(i+1,1),"TPHL")

val2=((Gmel(i+1,1))*3.0)/(2.0*(rhoe(i+1,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))*2.0)+Gmel(i+1,1)*g*dbble(i+1)*dz&
&*cos(theta)+epsm(i+1,1)*propev(ppres(i+1,1),T,hent(i+1,1),"VGP
")*propev(ppr&
&es(i+1,1),T,hent(i+1,1),"VPTL")/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&)*v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),epsm(i+1,1),
Dh,theta)&
&*(propev(ppres(i+1,1),T,hent(i+1,1),"HGP ") -
hlq(i+1,1))
else
val2=((Gmel(i+1,1))*3.0)/(2.0*(rhoe(i+1,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))*2.0)+Gmel(i+1,1)*g*dbble(i+1)*dz&
&*cos(theta)+epsm(i+1,1)*propev(ppres(i+1,1),T,hent(i+1,1),"VGP
")*propev(ppr&
&es(i+1,1),T,hent(i+1,1),"VFP
")/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),epsm(i+1,1),
Dh,theta)&
&*propev(ppres(i+1,1),T,hent(i+1,1),"HGF ")
end if
vectce(i)=Pe/A*qws(i)*2.0*dz-(val2-val1)
elseif (i==1) then
if (hlq(i,1)<propev(ppres(i,1),T,hent(i,1),"HFP ")) then
T=propev(ppres(i,1),T,hlq(i,1),"TPHL")

```

```

val1=((Gmel(i,1))**3.0)/(2.0*(rhoe(i,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
&))**2.0)+Gmel(i,1)*g*dbble(i)*dz*cos(the&
&ta)+epsm(i,1)*propev(ppres(i,1),T,hent(i,1),"VGP
")*propev(ppres(i,1),T,hent&
&(i,1),"VPTL")/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta)&
&*(propev(ppres(i,1),T,hent(i,1),"HGP ") -hlq(i,1))
else
val1=((Gmel(i,1))**3.0)/(2.0*(rhoe(i,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
&))**2.0)+Gmel(i,1)*g*dbble(i)*dz*cos(the&
&ta)+epsm(i,1)*propev(ppres(i,1),T,hent(i,1),"VGP
")*propev(ppres(i,1),T,hent&
&(i,1),"VFP
")/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta)&
&*(propev(ppres(i,1),T,hent(i,1),"HGF ")
end if
if (hlq(i+1,1)<propev(ppres(i+1,1),T,hent(i+1,1),"HFP
")) then
T=propev(ppres(i+1,1),T,hlq(i+1,1),"TPHL")
val2=((Gmel(i+1,1))**3.0)/(2.0*(rhoe(i+1,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))**2.0)+Gmel(i+1,1)*g*dbble(i+1)*dz&
&*cos(theta)+epsm(i+1,1)*propev(ppres(i+1,1),T,hent(i+1,1),"VGP
")*propev(ppr&
&es(i+1,1),T,hent(i+1,1),"VPTL")/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xt
it,epsm,hlq&
&)*v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),epsm(i+1,1),
Dh,theta)&
&*(propev(ppres(i+1,1),T,hent(i+1,1),"HGP ") -
hlq(i+1,1))
else
val2=((Gmel(i+1,1))**3.0)/(2.0*(rhoe(i+1,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))**2.0)+Gmel(i+1,1)*g*dbble(i+1)*dz&
&*cos(theta)+epsm(i+1,1)*propev(ppres(i+1,1),T,hent(i+1,1),"VGP
")*propev(ppr&
&es(i+1,1),T,hent(i+1,1),"VFP
")/rhom(i+1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i+1,1),Gmel(i+1,1),ppres(i+1,1),hlq(i+1,1),epsm(i+1,1),
Dh,theta)&
&*(propev(ppres(i+1,1),T,hent(i+1,1),"HGF ")
end if

```

```

        if (hlq(i+2,1)<propev(ppres(i+2,1),T,hent(i+2,1),"HFP
")) then
            T=propev(ppres(i+2,1),T,hlq(i+2,1),"TPHL")

val3=((Gmel(i+2,1))*3.0)/(2.0*(rhoe(i+2,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
            &))*2.0)+Gmel(i+2,1)*g*dbble(i+2)*dz&

&*cos(theta)+epsm(i+2,1)*propev(ppres(i+2,1),T,hent(i+2,1),"VGP
")*propev(ppr&

&es(i+2,1),T,hent(i+2,1),"VPTL")/rhom(i+2,1,Ndisc,Gmel,ppres,hent,xt
it,epsm,hlq&

&)*v_gj(xtit(i+2,1),Gmel(i+2,1),ppres(i+2,1),hlq(i+2,1),epsm(i+2,1),
Dh,theta)&
            &*(propev(ppres(i+2,1),T,hent(i+2,1),"HGP ") -
hlq(i+2,1))
        else

val3=((Gmel(i+2,1))*3.0)/(2.0*(rhoe(i+2,1,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
            &))*2.0)+Gmel(i+2,1)*g*dbble(i+2)*dz&

&*cos(theta)+epsm(i+2,1)*propev(ppres(i+2,1),T,hent(i+2,1),"VGP
")*propev(ppr&
            &es(i+2,1),T,hent(i+2,1),"VFP
")/rhom(i+2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&

&)*v_gj(xtit(i+2,1),Gmel(i+2,1),ppres(i+2,1),hlq(i+2,1),epsm(i+2,1),
Dh,theta)&
            &*propev(ppres(i+2,1),T,hent(i+2,1),"HGF ")
        end if
        vectce(i)=Pe/A*qws(i)*2.0*dz-(-3.0*val1+4.0*val2-
val3)+Ginit*hinit
    else
        if (hlq(i,1)<propev(ppres(i,1),T,hent(i,1),"HFP ")) then
            T=propev(ppres(i,1),T,hlq(i,1),"TPHL")

val1=((Gmel(i,1))*3.0)/(2.0*(rhoe(i,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
            &))*2.0)+Gmel(i,1)*g*dbble(i)*dz*cos(the&
            &ta)+epsm(i,1)*propev(ppres(i,1),T,hent(i,1),"VGP
")*propev(ppres(i,1),T,hent&

&(i,1),"VPTL")/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&

&)*v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta)&
            &*(propev(ppres(i,1),T,hent(i,1),"HGP ") -hlq(i,1))
        else

val1=((Gmel(i,1))*3.0)/(2.0*(rhoe(i,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
            &))*2.0)+Gmel(i,1)*g*dbble(i)*dz*cos(the&
            &ta)+epsm(i,1)*propev(ppres(i,1),T,hent(i,1),"VGP
")*propev(ppres(i,1),T,hent&
            &(i,1),"VFP
")/rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&

```

```

&)*v_gj(xtit(i,1),Gmel(i,1),ppres(i,1),hlq(i,1),epsm(i,1),Dh,theta)&
&*propev(ppres(i,1),T,hent(i,1),"HGF ")
end if
if (hlq(i-1,1)<propev(ppres(i-1,1),T,hent(i-1,1),"HFP
")) then
    T=propev(ppres(i-1,1),T,hlq(i-1,1),"TPHL")
    val2=((Gmel(i-1,1))**3.0)/(2.0*(rhoe(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,1)*g*dbble(i-1)*dz&
&*cos(theta)+epsm(i-1,1)*propev(ppres(i-
1,1),T,hent(i-1,1),"VGP ")*propev(ppr&
&es(i-1,1),T,hent(i-1,1),"VPTL")/rho(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),epsm(i-1,1),Dh,theta)&
&*(propev(ppres(i-1,1),T,hent(i-1,1),"HGP ") -hlq(i-
1,1))
    else
        val2=((Gmel(i-1,1))**3.0)/(2.0*(rhoe(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,1)*g*dbble(i-1)*dz&
&*cos(theta)+epsm(i-1,1)*propev(ppres(i-
1,1),T,hent(i-1,1),"VGP ")*propev(ppr&
&es(i-1,1),T,hent(i-1,1),"VFP ")/rho(i-
1,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i-1,1),Gmel(i-1,1),ppres(i-1,1),hlq(i-
1,1),epsm(i-1,1),Dh,theta)&
&*propev(ppres(i-1,1),T,hent(i-1,1),"HGF ")
    end if
    if (hlq(i-2,1)<propev(ppres(i-2,1),T,hent(i-2,1),"HFP
")) then
        T=propev(ppres(i-2,1),T,hlq(i-2,1),"TPHL")
        val3=((Gmel(i-2,1))**3.0)/(2.0*(rhoe(i-
2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-2,1)*g*dbble(i-2)*dz&
&*cos(theta)+epsm(i-2,1)*propev(ppres(i-
2,1),T,hent(i-2,1),"VGP ")*propev(ppr&
&es(i-2,1),T,hent(i-2,1),"VPTL")/rho(i-
2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i-2,1),Gmel(i-2,1),ppres(i-2,1),hlq(i-
2,1),epsm(i-2,1),Dh,theta)&
&*(propev(ppres(i-2,1),T,hent(i-2,1),"HGP ") -hlq(i-
2,1))
        else
            val3=((Gmel(i-2,1))**3.0)/(2.0*(rhoe(i-
2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-2,1)*g*dbble(i-2)*dz&
&*cos(theta)+epsm(i-2,1)*propev(ppres(i-
2,1),T,hent(i-2,1),"VGP ")*propev(ppr&
&es(i-2,1),T,hent(i-2,1),"VFP ")/rho(i-
2,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&)*v_gj(xtit(i-2,1),Gmel(i-2,1),ppres(i-2,1),hlq(i-
2,1),epsm(i-2,1),Dh,theta)&
&*propev(ppres(i-2,1),T,hent(i-2,1),"HGF ")
        end if
        vectce(i)=Pe/A*qws(i)*2.0*dz-(3.0*val1-4.0*val2+val3)
    end if

```

```

      end do
end subroutine initvectcestat

```

```

!*****
!*****!
!
!
!           Remplissage du vecteur de conservation
!
!           de l'énergie à l'état dynamique (VECTCE)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! vectce : vecteur du systeme Ax=b
!
! Ginit : flux massique à l'entrée du canal en kg/m2/s
!
! hinit : enthalpie a l'entree du canal en J/kg
!
! qws : flux de puissance en W/m2
!
! dz : pas axial en m
!
! dt : pas de temps en s
!
! Dh : diametre hydraulique en m
!
! Pe : perimetre equivalent en m
!
! A : aire en m2
!
! epsr : rugisite de la gaine en m
!
! theta : inclinaison du canal en radians
!

```

```

!
!
!*****
*****!

```

```

subroutine
initvectce(Ndisc,Gmel,ppres,hent,xtit,epsm,hlq,vectce,qws,Ginit,hini
t,dz,dt,Dh,Pe,A,epsr,theta)
    implicit none
    integer, intent (in) :: Ndisc
    double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
    double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
    double precision, dimension(Ndisc), intent(out) :: vectce
    double precision, dimension(Ndisc), intent(in) :: qws
    double precision, intent(in)::
Ginit,hinit,dz,dt,Dh,Pe,A,epsr,theta
    double precision, external :: propev, rhom, rho_m, rhoe, v_gj
    double precision :: val1, val2, val3, val4, val5, T
    double precision, parameter :: g=9.81
    integer :: i
    do i=1,Ndisc

val4=((Gmel(i,1))**2.0)/(2.0*rho_m(i,1,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq))&
&-
ppres(i,1)+rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)*g*db
le(i)*dz*cos(theta)

val5=((Gmel(i,2))**2.0)/(2.0*rho_m(i,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq))&
&-
ppres(i,2)+rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)*g*db
le(i)*dz*cos(theta)
        if (i>1 .AND. i<Ndisc) then
            if (hlq(i-1,2)<propev(ppres(i-1,2),T,hent(i-1,2),"HFP
") ) then
                T=propev(ppres(i-1,2),T,hlq(i-1,2),"TPHL")
                val1=((Gmel(i-1,2))**3.0)/(2.0*(rhoe(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,2)*g*db
le(i-1)*dz&
&*cos(theta)+epsm(i-1,2)*propev(ppres(i-
1,2),T,hent(i-1,2),"VGP ")*propev(ppr&
&es(i-1,2),T,hent(i-1,2),"VPTL")&
&/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,2),Dh,theta)&
&*(propev(ppres(i-1,2),T,hent(i-1,2),"HGP ") -hlq(i-
1,2))
            else
                val1=((Gmel(i-1,2))**3.0)/(2.0*(rhoe(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,2)*g*db
le(i-1)*dz&
&*cos(theta)+epsm(i-1,2)*propev(ppres(i-
1,2),T,hent(i-1,2),"VGP ")*propev(ppr&
&es(i-1,2),T,hent(i-1,2),"VFP ")&

```

```

&/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,2),Dh,theta)&
&*propev(ppres(i-1,2),T,hent(i-1,2),"HGF ")
end if
if (hlq(i+1,2)<propev(ppres(i+1,2),T,hent(i+1,2),"HFP
")) then
T=propev(ppres(i+1,2),T,hlq(i+1,2),"TPHL")

val2=((Gmel(i+1,2))*3.0)/(2.0*(rhoe(i+1,2,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))*2.0)+Gmel(i+1,2)*g*db1e(i+1)*dz&

&*cos(theta)+epsm(i+1,2)*propev(ppres(i+1,2),T,hent(i+1,2),"VGP
")*propev(ppr&
&es(i+1,2),T,hent(i+1,2),"VPTL")&
&/rhom(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,2),D
h,theta)&
&*(propev(ppres(i+1,2),T,hent(i+1,2),"HGP ") -
hlq(i+1,2))
else

val2=((Gmel(i+1,2))*3.0)/(2.0*(rhoe(i+1,2,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))*2.0)+Gmel(i+1,2)*g*db1e(i+1)*dz&

&*cos(theta)+epsm(i+1,2)*propev(ppres(i+1,2),T,hent(i+1,2),"VGP
")*propev(ppr&
&es(i+1,2),T,hent(i+1,2),"VFP ")&
&/rhom(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,2),D
h,theta)&
&*propev(ppres(i+1,2),T,hent(i+1,2),"HGF ")
end if
vectce(i)=Pe/A*qws(i)*2.0*dz-(val2-
vall1)+rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*hent(i,1)*2.0*dz/dt-(val5-vall4)*2.0*dz/dt
elseif (i==1) then
if (hlq(i,2)<propev(ppres(i,2),T,hent(i,2),"HFP ")) then
T=propev(ppres(i,2),T,hlq(i,2),"TPHL")

vall1=((Gmel(i,2))*3.0)/(2.0*(rhoe(i,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
&))*2.0)+Gmel(i,2)*g*db1e(i)*dz*cos(the&
&ta)+epsm(i,2)*propev(ppres(i,2),T,hent(i,2),"VGP
")*propev(ppres(i,2),T,hent&
&(i,2),"VPTL")/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,theta)&
&*(propev(ppres(i,2),T,hent(i,2),"HGP ") -hlq(i,2))
else

vall1=((Gmel(i,2))*3.0)/(2.0*(rhoe(i,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&

```



```

&))**2.0)+Gmel(i,2)*g*dbble(i)*dz*cos(the&
&ta)+epsm(i,2)*propev(ppres(i,2),T,hent(i,2),"VGP
")*propev(ppres(i,2),T,hent&
&(i,2),"VFP
")/rho(m(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,theta)&
&*propev(ppres(i,2),T,hent(i,2),"HGF ")&
end if
if (hlq(i+1,2)<propev(ppres(i+1,2),T,hent(i+1,2),"HFP
")) then
T=propev(ppres(i+1,2),T,hlq(i+1,2),"TPHL")

val2=((Gmel(i+1,2))**3.0)/(2.0*(rhoe(i+1,2,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))**2.0)+Gmel(i+1,2)*g*dbble(i+1)*dz&

&*cos(theta)+epsm(i+1,2)*propev(ppres(i+1,2),T,hent(i+1,2),"VGP
")*propev(ppr&
&es(i+1,2),T,hent(i+1,2),"VPTL")&
&/rho(m(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,2),D
h,theta)&
&*(propev(ppres(i+1,2),T,hent(i+1,2),"HGP ") -
hlq(i+1,2))
else

val2=((Gmel(i+1,2))**3.0)/(2.0*(rhoe(i+1,2,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))**2.0)+Gmel(i+1,2)*g*dbble(i+1)*dz&

&*cos(theta)+epsm(i+1,2)*propev(ppres(i+1,2),T,hent(i+1,2),"VGP
")*propev(ppr&
&es(i+1,2),T,hent(i+1,2),"VFP ")&
&/rho(m(i+1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i+1,2),Gmel(i+1,2),ppres(i+1,2),hlq(i+1,2),epsm(i+1,2),D
h,theta)&
&*propev(ppres(i+1,2),T,hent(i+1,2),"HGF ")&
end if
if (hlq(i+2,2)<propev(ppres(i+2,2),T,hent(i+2,2),"HFP
")) then
T=propev(ppres(i+2,2),T,hlq(i+2,2),"TPHL")

val3=((Gmel(i+2,2))**3.0)/(2.0*(rhoe(i+2,2,Ndisc,Gmel,ppres,hent,xti
t,epsm,hlq&
&))**2.0)+Gmel(i+2,2)*g*dbble(i+2)*dz&

&*cos(theta)+epsm(i+2,2)*propev(ppres(i+2,2),T,hent(i+2,2),"VGP
")*propev(ppr&
&es(i+2,2),T,hent(i+2,2),"VPTL")&
&/rho(m(i+2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i+2,2),Gmel(i+2,2),ppres(i+2,2),hlq(i+2,2),epsm(i+2,2),D
h,theta)&
&*(propev(ppres(i+2,2),T,hent(i+2,2),"HGP ") -
hlq(i+2,2))

```

```

else

val3=((Gmel(i+2,2))**3.0)/(2.0*(rhoe(i+2,2,Ndisc,Gmel,ppres,hent,xtit,
epsm,hlq&
&))**2.0)+Gmel(i+2,2)*g*dbble(i+2)*dz&

&*cos(theta)+epsm(i+2,2)*propev(ppres(i+2,2),T,hent(i+2,2),"VGP
")*propev(ppr&
&es(i+2,2),T,hent(i+2,2),"VFP ")&
&/rhom(i+2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i+2,2),Gmel(i+2,2),ppres(i+2,2),hlq(i+2,2),epsm(i+2,2),D
h,theta)&
&*propev(ppres(i+2,2),T,hent(i+2,2),"HGF ")
end if
vectce(i)=Pe/A*qws(i)*2.0*dz-(-3.0*val1+4.0*val2-val3)&

&+rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)*hent(i,1)*2.0*dz/dt&
&-(val5-val4)*2.0*dz/dt+Ginit*hinit
else
if (hlq(i,2)<propev(ppres(i,2),T,hent(i,2),"HFP ")) then
T=propev(ppres(i,2),T,hlq(i,2),"TPHL")

val1=((Gmel(i,2))**3.0)/(2.0*(rhoe(i,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
&))**2.0)+Gmel(i,2)*g*dbble(i)*dz*cos(the&
&ta)+epsm(i,2)*propev(ppres(i,2),T,hent(i,2),"VGP
")*propev(ppres(i,2),T,hent&
&(i,2),"VPTL")/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,theta)&
&*(propev(ppres(i,2),T,hent(i,2),"HGP ") -hlq(i,2))
else

val1=((Gmel(i,2))**3.0)/(2.0*(rhoe(i,2,Ndisc,Gmel,ppres,hent,xtit,ep
sm,hlq&
&))**2.0)+Gmel(i,2)*g*dbble(i)*dz*cos(the&
&ta)+epsm(i,2)*propev(ppres(i,2),T,hent(i,2),"VGP
")*propev(ppres(i,2),T,hent&
&(i,2),"VFP
")/rhom(i,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&

&*v_gj(xtit(i,2),Gmel(i,2),ppres(i,2),hlq(i,2),epsm(i,2),Dh,theta)&
&*propev(ppres(i,2),T,hent(i,2),"HGF ")
end if
if (hlq(i-1,2)<propev(ppres(i-1,2),T,hent(i-1,2),"HFP
")) then
T=propev(ppres(i-1,2),T,hlq(i-1,2),"TPHL")
val2=((Gmel(i-1,2))**3.0)/(2.0*(rhoe(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,2)*g*dbble(i-1)*dz&
&*cos(theta)+epsm(i-1,2)*propev(ppres(i-
1,2),T,hent(i-1,2),"VGP ")*propev(ppr&
&es(i-1,2),T,hent(i-1,2),"VPTL")&
&/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,2),Dh,theta)&

```

```

&*(propev(ppres(i-1,2),T,hent(i-1,2),"HGP ") -hlq(i-
1,2))
      else
        val2=((Gmel(i-1,2))**3.0)/(2.0*(rhoe(i-
1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-1,2)*g*dbple(i-1)*dz&
&*cos(theta)+epsm(i-1,2)*propev(ppres(i-
1,2),T,hent(i-1,2),"VGP ")*propev(ppr&
&es(i-1,2),T,hent(i-1,2),"VFP ")&
&/rhom(i-1,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*v_gj(xtit(i-1,2),Gmel(i-1,2),ppres(i-1,2),hlq(i-
1,2),epsm(i-1,2),Dh,theta)&
&*propev(ppres(i-1,2),T,hent(i-1,2),"HGF ")
      end if
      if (hlq(i-2,2)<propev(ppres(i-2,2),T,hent(i-2,2),"HFP
")) then
        T=propev(ppres(i-2,2),T,hlq(i-2,2),"TPHL")
        val3=((Gmel(i-2,2))**3.0)/(2.0*(rhoe(i-
2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-2,2)*g*dbple(i-2)*dz&
&*cos(theta)+epsm(i-2,2)*propev(ppres(i-
2,2),T,hent(i-2,2),"VGP ")*propev(ppr&
&es(i-2,2),T,hent(i-2,2),"VPTL")&
&/rhom(i-2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*v_gj(xtit(i-2,2),Gmel(i-2,2),ppres(i-2,2),hlq(i-
2,2),epsm(i-2,2),Dh,theta)&
&*(propev(ppres(i-2,2),T,hent(i-2,2),"HGP ") -hlq(i-
2,2))
      else
        val3=((Gmel(i-2,2))**3.0)/(2.0*(rhoe(i-
2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq&
&))**2.0)+Gmel(i-2,2)*g*dbple(i-2)*dz&
&*cos(theta)+epsm(i-2,2)*propev(ppres(i-
2,2),T,hent(i-2,2),"VGP ")*propev(ppr&
&es(i-2,2),T,hent(i-2,2),"VFP ")&
&/rhom(i-2,2,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)&
&*v_gj(xtit(i-2,2),Gmel(i-2,2),ppres(i-2,2),hlq(i-
2,2),epsm(i-2,2),Dh,theta)&
&*propev(ppres(i-2,2),T,hent(i-2,2),"HGF ")
      end if
      vectce(i)=Pe/A*qws(i)*2.0*dz-(3.0*val1-4.0*val2+val3)&
&+rhom(i,1,Ndisc,Gmel,ppres,hent,xtit,epsm,hlq)*hent(i,1)*2.0*dz/dt&
&-(val5-val4)*2.0*dz/dt
      end if
    end do
  end subroutine initvectce

```

```

!*****
!*****!

```

```

!
!
!
!

```

Remplissage du vecteur de transfert

```

!                                     de chaleur à l'état statique (VECTTIG)
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale
!
! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
!Double precision :
!
! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! Ttig : temperature du crayon en K
!
! vecttig : vecteur du systeme Ax=b
!
! qwv : densite de puissance en W/m3
!
! qws : flux de puissance en W/m2
!
! drc : pas pour le combustible en m
!
! drg : pas pour la gaine en m
!
! Dh : diametre hydraulique en m
!
! Dgext : diametre exterieur de la gaine en m
!
!
!
!*****
*****!

subroutine
initvecttigstat(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,epsm,
hlq,Ttig,&
&vecttig,qws,qwv,drc,drg,Dh,Dgext)
    implicit none
    integer, intent (in) :: Ndisc,Mdisc,Mcdisc,Mgdisc

```

```

      double precision, dimension(Ndisc,2), intent(in)  ::
Gmel,ppres,hent,hlq
      double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
      double precision, dimension(Mdisc,Ndisc,2), intent(in)  :: Ttig
      double precision, dimension(Mdisc,Ndisc), intent(out) :: vecttig
      double precision, dimension(Ndisc), intent(in) :: qws,qwv
      double precision, intent(in):: drc,drg,Dh,Dgext
      double precision, external :: Twal, propzir
      double precision :: T
      integer :: i,k
      T=0.0
      do k=1,Ndisc
        do i=1,Mdisc
          if (i<Mcdisc .AND. i>1) then
            vecttig(i,k)=qwv(k)*drc**2.0
          else if (i==Mdisc-1) then

T=Twal(Gmel(k,1),ppres(k,1),hlq(k,1),xtit(k,1),qws(k),Dh,Dgext)
            vecttig(i,k)=-2.0*drg*qws(k)/propzir(T,"k")
          else if (i==Mdisc) then

T=Twal(Gmel(k,1),ppres(k,1),hlq(k,1),xtit(k,1),qws(k),Dh,Dgext)
            vecttig(i,k)=T
          else
            vecttig(i,k)=0.0
          end if
        end do
      end do
end subroutine initvecttigstat

```

```

!*****
!*****!
!
!
!           Remplissage du vecteur de transfert
!
!           de chaleur à l'état dynamique (VECTTIG)
!
!
!
!
!Fait par: Majid FASSI FEHRI (31/12/2007)
!
!Entier:
!
! Ndisc : discretisation axiale du canal
!
! Mdisc : discretisation radiale
!
! Mcdisc : discretisation du combustible
!
! Mgdisc : discretisation de la gaine
!
!Double precision :
!

```

```

! Gmel : valeurs du flux massique en kg/m2/s
!
! ppres : valeurs de la pression en Pa
!
! hent : valeurs de l'enthalpie en J/kg
!
! xtit : titre de l'ecoulement
!
! epsm : taux de vide
!
! hlq : enthalpie du liquide J/kg
!
! Ttig : temperature du crayon en K
!
! vecttig : vecteur du systeme Ax=b
!
! qwv : densite de puissance en W/m3
!
! qws : flux de puissance en W/m2
!
! drc : pas pour le combustible en m
!
! drg : pas pour la gaine en m
!
! dt : pas de temps en s
!
! Dh : diametre hydraulique en m
!
! Dgext : diametre exterieur de la gaine en m
!
!
!
!*****
!*****!

```

```

subroutine
initvecttig(Ndisc,Mdisc,Mcdisc,Mgdisc,Gmel,ppres,hent,xtit,epsm,hlq,
Ttig,&
&vecttig,qws,qwv,drc,drg,dt,Dh,Dgext)
  implicit none
  integer, intent (in) :: Ndisc,Mdisc,Mcdisc,Mgdisc
  double precision, dimension(Ndisc,2), intent(in) ::
Gmel,ppres,hent,hlq
  double precision, dimension(Ndisc,2), intent(inout) ::
xtit,epsm
  double precision, dimension(Mdisc,Ndisc,2), intent(in) :: Ttig
  double precision, dimension(Mdisc,Ndisc), intent(out) :: vecttig
  double precision, dimension(Ndisc), intent(in) :: qws,qwv
  double precision, intent(in):: drc,drg,dt,Dh,Dgext
  double precision, external :: propzir, propur, Twal
  double precision :: T
  integer :: i,k
  T=0.0
  do k=1,Ndisc
    do i=1,Mdisc
      if (i<Mcdisc .AND. i>1) then

```

```

vecttig(i,k)=qvw(k)*drc**2.0+propur(Ttig(i,k,1),"c")*propur(Ttig(i,k
,1),"r")*&
                &Ttig(i,k,1)*drc**2.0/dt
                else if (i>Mdisc .AND. i<Mdisc-1) then

vecttig(i,k)=propzir(Ttig(i+1,k,1),"c")*propzir(Ttig(i+1,k,1),"r")*T
tig(i+1,k&
                &,1)*drg**2.0/dt
                else if (i==Mdisc-1) then

T=Twal(Gmel(k,2),ppres(k,2),hlq(k,2),xtit(k,2),qws(k),Dh,Dgext)
                vecttig(i,k)=-2.0*drg*qws(k)/propzir(T,"k")
                else if (i==Mdisc) then

T=Twal(Gmel(k,2),ppres(k,2),hlq(k,2),xtit(k,2),qws(k),Dh,Dgext)
                vecttig(i,k)=T
                else
                vecttig(i,k)=0.0
                end if
            end do
        end do
end subroutine initvecttig

```

conver.f

```

* DECK CONVER
  SUBROUTINE CONVER(IPTH,NCHAN,NBUN,VAL,PREC)
*-----
*
*
*   this module calculate convergence
*
*
*   IMPLICIT NONE
*
*   INTEGER IPTH,NBUN,NCHAN,VAL
*   REAL PREC
*
*   REAL TCOOLI(NCHAN,NBUN),DCOOI(NCHAN,NBUN),
*   + TFUELI(NCHAN,NBUN),PWRI(NCHAN,NBUN)
*   REAL TCOOLF(NCHAN,NBUN),DCOOLF(NCHAN,NBUN),
*   + TFUELF(NCHAN,NBUN),PWRP(NCHAN,NBUN)
*
* * GENERAL VARIABLES
*
*   INTEGER I,J
*   REAL ERR
*
*   CALL LCMGET(IPTH,'B-TCOOL-F',TCOOLF)
*   CALL LCMGET(IPTH,'B-DCOOL-F',DCOOLF)
*   CALL LCMGET(IPTH,'B-TFUEL-F',TFUELF)
*   CALL LCMGET(IPTH,'B-PWR-F',PWRP)
*
*   CALL LCMGET(IPTH,'B-TCOOL-I',TCOOI)

```

```

CALL LCMGET(IPTH,'B-DCOOL-I',DCOOLI)
CALL LCMGET(IPTH,'B-TFUEL-I',TFUELI)
CALL LCMGET(IPTH,'B-PWR-I',PWRI)

DO 10 I=1,NCHAN
  DO 20 J=1,NBUN
    ERR=ABS((TCOOLF(I,J)-TCOOLI(I,J))/TCOOLF(I,J))
    IF(ERR.GT.PREC) THEN
      VAL=-1
    ENDIF
    ERR=ABS((DCOOLF(I,J)-DCOOLI(I,J))/DCOOLF(I,J))
    IF(ERR.GT.PREC) THEN
      VAL=-1
    ENDIF
    ERR=ABS((TFUELF(I,J)-TFUELI(I,J))/TFUELF(I,J))
    IF(ERR.GT.PREC) THEN
      VAL=-1
    ENDIF
    ERR=ABS((PWRF(I,J)-PWRI(I,J))/PWRF(I,J))
    IF(ERR.GT.PREC) THEN
      VAL=-1
    ENDIF
20  CONTINUE
10  CONTINUE

RETURN
END

```

ntrthcv.f

```

* DECK NTRTHCV
  SUBROUTINE NTRTHCV(NZ,NBUN,VECTA,VECTB)
*
*
*   CONVERT NBUN SIMPLE PRECISION VECTOR TO
*   NZ DOUBLE PRECISION VECTORS USING LINEAR INTERPOLATION
*
*   Author(s) : M. Fassi Fehri
*
*   NZ      : DOUBLE PRECISION VECTOR DIMENSION
*   NBUN    : SIMPLE PRECISION VECTOR DIMENSION
*   VECTA   : DOUBLE PRECISION VECTOR
*   VECTB   : SIMPLE PRECISION VECTOR
*
*   IMPLICIT NONE
*
*   VECTOR CONVERSION
*
*   INTEGER NZ,NBUN
*   DOUBLE PRECISION VECTA(NZ)
*   REAL VECTB(NBUN)
*
*   GENERAL VARIABLES

```



```

*
DOUBLE PRECISION TEMP
INTEGER I,K
DOUBLE PRECISION DZ,DB
DOUBLE PRECISION PTE,ORD

DZ=1.0/(DBLE(NZ)-1.0)
DB=1.0/DBLE(NBUN)
K=1

DO 10 I=0,(NZ-1)
  IF((DBLE(I)*DZ).LE.(DBLE(K)+1.0/2.0)*DB).AND.
  * ((DBLE(I)*DZ).GE.(DBLE(K)-1.0/2.0)*DB)) THEN
    PTE=DBLE(VECTB(K+1)-VECTB(K))/DB
    ORD=DBLE(VECTB(K)-(REAL(K)-1.0/2.0)*(VECTB(K+1)-VECTB(K)))
    VECTA(I+1)=PTE*DBLE(I)*DZ+ORD
  ELSEIF((DBLE(I)*DZ).GT.(DB*(DBLE(NBUN)-1.0/2.0))) THEN
    PTE=DBLE(VECTB(NBUN)/1000.0-VECTB(NBUN))/(DB/2.0)
    ORD=DBLE(VECTB(NBUN)-(REAL(NBUN)-1.0/2.0)*
  * (VECTB(NBUN)/1000.0-VECTB(NBUN))/(1.0/2.0))
    VECTA(I+1)=PTE*DBLE(I)*DZ+ORD
  ELSEIF((DBLE(I)*DZ).LT.(DBLE(DB)/2.0)) THEN
    PTE=DBLE(VECTB(1)-VECTB(1)/1000.0)/(DB/2.0)
    ORD=DBLE(VECTB(1))/1000.0
    VECTA(I+1)=PTE*DBLE(I)*DZ+ORD
  ELSE
    K=K+1
    PTE=DBLE(VECTB(K+1)-VECTB(K))/DB
    ORD=DBLE(VECTB(K)-(REAL(K)-1.0/2.0)*(VECTB(K+1)-VECTB(K)))
    VECTA(I+1)=PTE*DBLE(I)*DZ+ORD
  ENDIF
10 CONTINUE

RETURN
END

```

thadd.f

```

*DECK THADD
      SUBROUTINE THADD(NENTRY,HENTRY,IENTRY,JENTRY,KENTRY)
*-----
*
*
*Purpose:
*  this module add thermalhydraulic linked lists
*
*Copyright:
*  Copyright (C) 2002 Ecole Polytechnique de Montreal
*  This library is free software; you can redistribute it and/or
*  modify it under the terms of the GNU Lesser General Public
*  License as published by the Free Software Foundation; either
*  version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri
*

```

```

*Parameters: input/output
* NENTRY  number of linked lists or files used by the module
* HENTRY  character*12 name of each linked list or file
* IENTRY  =1 linked list; =2 xsm file; =3 sequential binary file;
*          =4 sequential ascii file
* JENTRY  =0 the linked list or file is created
*          =1 the linked list or file is open for modifications;
*          =2 the linked list or file is open in read-only mode
* KENTRY  =file unit number; =linked list address otherwise
*
*
*
*-----
---
      IMPLICIT NONE
*
*  PARAMETER VARIABLES
*
      INTEGER  NSTATE
      PARAMETER (NSTATE=40)
      INTEGER  NENTRY
      CHARACTER HENTRY(NENTRY)*12
      INTEGER  IENTRY(NENTRY), JENTRY(NENTRY), KENTRY(NENTRY)
      INTEGER  THPARAM(NSTATE)
      INTEGER  THPARAMT(NSTATE)

*
*  SIGNATURE INFORMATION
*
      CHARACTER*12 HSIGN
      INTEGER  ISGNTR(3)
*
*  GENERAL VARIABLES
*
      INTEGER  J

*
*  INPUT AND OUTPUT VALIDATION
*
      IF(NENTRY.LE.1) CALL XABORT('THADD: PARAMETERS EXPECTED.')

      IF((IENTRY(1).NE.1).AND.(IENTRY(1).NE.2).AND.
+ (IENTRY(2).NE.1).AND.(IENTRY(2).NE.2))
+ CALL XABORT('THADD: LINKED LIST OR XSM FILE EXPECTED AT
THERMO')

      IF(JENTRY(1).NE.1)
+ CALL XABORT('THADD: MODIFICATION MODE EXPECTED.')
      IF(JENTRY(2).NE.2)
+ CALL XABORT('THADD: READ ONLY MODE EXPECTED.')

      CALL LCMSIX(KENTRY(1),' ',0)
      CALL LCMGET(KENTRY(1),'SIGNATURE',ISGNTR)
      WRITE(HSIGN,'(3A4)') (ISGNTR(J),J=1,3)
      IF (HSIGN.NE.'L_THERMO')

```

```

+ CALL XABORT('THADD: L_THERMO OBJECT IS EXPECTED')
CALL LCMSIX(KENTRY(2),'-',0)
CALL LCMGET(KENTRY(2),'SIGNATURE',ISGNTR)
WRITE(HSIGN,'(3A4)') (ISGNTR(J),J=1,3)
IF (HSIGN.NE.'L_THERMO')
+ CALL XABORT('THADD: L_THERMO OBJECT IS EXPECTED')

*
*   THERMALHYDRALIC MATRIX DIMENSIONS
*
CALL LCMGET(KENTRY(1),'STATE-VECTOR',THPARAM)
THPARAM(6)=0
THPARAM(7)=0
CALL LCMPUT(KENTRY(1),'STATE-VECTOR',NSTATE,1,THPARAM)

CALL LCMGET(KENTRY(2),'STATE-VECTOR',THPARAMT)
*
*   ADDING LINKED LISTS
*
CALL
THADDM(KENTRY(1),THPARAM(1),THPARAM(2)+THPARAM(3),THPARAM(4),
+ THPARAM(5),KENTRY(2),THPARAMT(6),THPARAMT(7))

RETURN
END

```

thaddm.f

```

* DECK THADDM
      SUBROUTINE THADDM(IPTHO,NZ,NR,NCHAN,NBUN,IPTHI,IBEG,IFIN)
*-----
*
*   this module add thermalhydraulic matrix to linked lists
*Copyright:
* Copyright (C) 2002 Ecole Polytechnique de Montreal
* This library is free software; you can redistribute it and/or
* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri
*
*Parameters: input/output
*
*IPTHO : L_THERMO ADRESSE IN OUTPUT/INPUT
*IPTHI : L_THERMO ADRESSE IN INPUT
*NZ    : AXIAL DISCRETIZATION
*NR    : RADIAL DISCRETIZATION
*IBEG  : FIRST CHANNEL CALCULUS
*IFIN  : LAST CHANNEL CALCULUS
*NBUN  : NUMBER OF BUNDLES

```

*NCHAN : NUMBER OF CHANNELS

*-----

IMPLICIT NONE

INTEGER IPTHO,NZ,NR,NBUN,NCHAN,IPTHI,IBEG,IFIN

*
 * THERMALHYDRAULIC MATRICES
 *

DOUBLE PRECISION HENT(NZ,2),PPRES(NZ,2),GMEL(NZ,2),EPSM(NZ,2),
 + XTIT(NZ,2),HLQ(NZ,2),TTIG(NR,NZ,2),PWRD(NZ)
 DOUBLE PRECISION
 HENTT(NZ,2),PPREST(NZ,2),GMELT(NZ,2),EPSMT(NZ,2),
 + XTITT(NZ,2),HLQT(NZ,2),TTIGT(NR,NZ,2),PWRDT(NZ)

*
 * NEUTRONIC PARAMETERS AND MATRICES
 *

REAL TCOOL(NCHAN,NBUN),DCOOL(NCHAN,NBUN),
 + TFUEL(NCHAN,NBUN),PWR(NCHAN,NBUN)
 REAL TCOOLT(NCHAN,NBUN),DCOOLT(NCHAN,NBUN),
 + TFUELT(NCHAN,NBUN),PWRT(NCHAN,NBUN)

*
 * GENERAL VARIABLES
 *

INTEGER I,J,K
 CHARACTER*12 CHANAM

*
 * NEUTRONIC VALUES
 *

CALL LCMSIX(IPTHO,' ',0)
 CALL LCMSIX(IPTHI,' ',0)

CALL LCMGET(IPTHO,'B-TCOOL-F',TCOOL)
 CALL LCMGET(IPTHO,'B-DCOOL-F',DCOOL)
 CALL LCMGET(IPTHO,'B-TFUEL-F',TFUEL)
 CALL LCMGET(IPTHO,'B-PWR-F',PWR)
 CALL LCMGET(IPTHI,'B-TCOOL-F',TCOOLT)
 CALL LCMGET(IPTHI,'B-DCOOL-F',DCOOLT)
 CALL LCMGET(IPTHI,'B-TFUEL-F',TFUELT)
 CALL LCMGET(IPTHI,'B-PWR-F',PWRT)

DO 10 I=IBEG,IFIN

DO 20 J=1,NBUN
 TCOOL(I,J)=TCOOLT(I,J)
 DCOOL(I,J)=DCOOLT(I,J)
 TFUEL(I,J)=TFUELT(I,J)
 PWR(I,J)=PWRT(I,J)

20 CONTINUE

10 CONTINUE

CALL LCMPUT(IPTHO,'B-TCOOL-F',NCHAN*NBUN,2,TCOOL)
 CALL LCMPUT(IPTHO,'B-DCOOL-F',NCHAN*NBUN,2,DCOOL)
 CALL LCMPUT(IPTHO,'B-TFUEL-F',NCHAN*NBUN,2,TFUEL)

```

CALL LCMPUT(IPTHO,'B-PWR-F',NCHAN*NBUN,2,PWR)

CALL LCMGET(IPTHO,'B-TCOOL-I',TCOOL)
CALL LCMGET(IPTHO,'B-DCOOL-I',DCOOL)
CALL LCMGET(IPTHO,'B-TFUEL-I',TFUEL)
CALL LCMGET(IPTHO,'B-PWR-I',PWR)
CALL LCMGET(IPTHI,'B-TCOOL-I',TCOOLT)
CALL LCMGET(IPTHI,'B-DCOOL-I',DCOOLT)
CALL LCMGET(IPTHI,'B-TFUEL-I',TFUEL)
CALL LCMGET(IPTHI,'B-PWR-I',PWRT)
DO 30 I=IBEG,IFIN
  DO 40 J=1,NBUN
    TCOOL(I,J)=TCOOLT(I,J)
    DCOOL(I,J)=DCOOLT(I,J)
    TFUEL(I,J)=TFUEL(I,J)
    PWR(I,J)=PWRT(I,J)
40  CONTINUE
30  CONTINUE
CALL LCMPUT(IPTHO,'B-TCOOL-I',NCHAN*NBUN,2,TCOOL)
CALL LCMPUT(IPTHO,'B-DCOOL-I',NCHAN*NBUN,2,DCOOL)
CALL LCMPUT(IPTHO,'B-TFUEL-I',NCHAN*NBUN,2,TFUEL)
CALL LCMPUT(IPTHO,'B-PWR-I',NCHAN*NBUN,2,PWR)

*
*  THERMALHYDRAULIC VALUES
*

DO 100 I=IBEG,IFIN

WRITE(CHANAM,'(A7,I5)') 'CHANNEL',I
CALL LCMSIX(IPTHO,CHANAM,1)
CALL LCMSIX(IPTHI,CHANAM,1)

CALL LCMGET(IPTHO,'MIX-ENTH',HENT)
CALL LCMGET(IPTHO,'PRESSURE',PPRES)
CALL LCMGET(IPTHO,'MASS-FLUX',GMEL)
CALL LCMGET(IPTHO,'VOID-FRAC',EPSM)
CALL LCMGET(IPTHO,'FLOW-QLTY',XTIT)
CALL LCMGET(IPTHO,'LIQ-ENTH',HLQ)
CALL LCMGET(IPTHO,'PIN-TEMP',TTIG)
CALL LCMGET(IPTHO,'PWR-DSTY',PWRD)

CALL LCMGET(IPTHI,'MIX-ENTH',HENTT)
CALL LCMGET(IPTHI,'PRESSURE',PPREST)
CALL LCMGET(IPTHI,'MASS-FLUX',GMELT)
CALL LCMGET(IPTHI,'VOID-FRAC',EPSMT)
CALL LCMGET(IPTHI,'FLOW-QLTY',XTITT)
CALL LCMGET(IPTHI,'LIQ-ENTH',HLQT)
CALL LCMGET(IPTHI,'PIN-TEMP',TTIGT)
CALL LCMGET(IPTHI,'PWR-DSTY',PWRDT)

DO 200 J=1,NZ
  HENT(J,1)=HENTT(J,1)
  HENT(J,2)=HENTT(J,2)
  PPRES(J,1)=PPREST(J,1)
  PPRES(J,2)=PPREST(J,2)

```

```

      GMEL(J,1)=GMELT(J,1)
      GMEL(J,2)=GMELT(J,2)
      EPSM(J,1)=EPSMT(J,1)
      EPSM(J,2)=EPSMT(J,2)
      XTIT(J,1)=XTITT(J,1)
      XTIT(J,2)=XTITT(J,2)
      HLQ(J,1)=HLQT(J,1)
      HLQ(J,2)=HLQT(J,2)
      PWRD(J)=PWRDT(J)
      DO 300 K=1,NR
         TTIG(K,J,1)=TTIGT(K,J,1)
         TTIG(K,J,2)=TTIGT(K,J,2)
300    CONTINUE
200    CONTINUE

      CALL LCMPUT(IPTHO,'MIX-ENTH',NZ*2*2,4,HENT)
      CALL LCMPUT(IPTHO,'PRESSURE',NZ*2*2,4,PPRES)
      CALL LCMPUT(IPTHO,'MASS-FLUX',NZ*2*2,4,GMEL)
      CALL LCMPUT(IPTHO,'VOID-FRAC',NZ*2*2,4,EPSM)
      CALL LCMPUT(IPTHO,'FLOW-QLTY',NZ*2*2,4,XTIT)
      CALL LCMPUT(IPTHO,'LIQ-ENTH',NZ*2*2,4,HLQ)
      CALL LCMPUT(IPTHO,'PIN-TEMP',NZ*NR*2*2,4,TTIG)
      CALL LCMPUT(IPTHO,'PWR-DSTY',NZ*2,4,PWRD)

      CALL LCMSIX(IPTHI,' ',0)
      CALL LCMSIX(IPTHO,' ',0)

100    CONTINUE

      RETURN
      END

```

thcalc.f

```

* DECK THCALC
      SUBROUTINE THCALC(IPTH,NZ,NFUEL,NCLAD,IPNTR,NCHAN,NBUN,VAL,
+ IPRNT,IBEG,IFIN)
*-----
*
*      this module calculate thermalhydraulic properties for channels
*Copyright:
* Copyright (C) 2002 Ecole Polytechnique de Montreal
* This library is free software; you can redistribute it and/or
* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri
*
*Parameters: input/output
*
*IPTH   : L THERMO ADDRESS
*NZ     : AXIAL DISCRETIZATION
*NFUEL  : FUEL DISCRETIZATION

```

```

*NCLAD : CLAD DISCRETIZATION
*IPNTR : L_MAP ADRESS
*NBUN : NUMBER OF BUNDLES
*NCHAN : NUMBER OF CHANNELS
*VAL : =1 STATIC CALCULUS
*      =2 KINTETIC CALCULUS
*IPRNT : PRINTING
*IBEG : FIRST CHANNEL CALCULUS
*IFIN : LAST CHANNEL CALCULUS
*-----
*--

      IMPLICIT NONE

      INTEGER
      IPTH,NZ,NFUEL,NCLAD,IPNTR,NBUN,NCHAN,VAL,IPRNT,IBEG,IFIN
      INTEGER NTHDISC,NTHGEOM,NTHCOND,NTHEVO,NTIMEP
      PARAMETER (NTHDISC=3)
      PARAMETER (NTHGEOM=7)
      PARAMETER (NTHCOND=5)
      PARAMETER (NTHEVO=3)
      PARAMETER (NTIMEP=1)
      DOUBLE PRECISION
      THGEOM(NTHGEOM),THCOND(NTHCOND),THEVO(NTHEVO),
      + TIMEP(NTIMEP)
*
* THERMALHYDRAULIC MATRICES
*
      DOUBLE PRECISION HENT(NZ,2),PPRES(NZ,2),GMEL(NZ,2),EPSM(NZ,2),
      + XTIT(NZ,2),HLQ(NZ,2),TTIG((NFUEL+NCLAD),NZ,2),PWRD(NZ),
      + TCAL(NZ),RCAL(NZ),TCOMB(NZ)
*
* NEUTRONIC PARAMETERS AND MATRICES
*
      REAL TCOOL(NCHAN,NBUN),DCOOL(NCHAN,NBUN),
      + TFUEL(NCHAN,NBUN),PWR(NCHAN,NBUN)
      INTEGER REFSCH(NCHAN)
*
* GENERAL VARIABLES
*
      INTEGER I,J
      CHARACTER*12 CHANAM
      REAL TMPTC(NBUN),TMPTF(NBUN),TMPRC(NBUN),TMPPWR(NBUN)
*
* Loading thermalhydraulic datas
*
      CALL LCMGET(IPTH,'B-TCOOL-F',TCOOL)
      CALL LCMGET(IPTH,'B-DCOOL-F',DCOOL)
      CALL LCMGET(IPTH,'B-TFUEL-F',TFUEL)
      CALL LCMGET(IPTH,'B-PWR-F',PWR)

      CALL LCMPUT(IPTH,'B-TCOOL-I',NCHAN*NBUN,2,TCOOL)
      CALL LCMPUT(IPTH,'B-DCOOL-I',NCHAN*NBUN,2,DCOOL)
      CALL LCMPUT(IPTH,'B-TFUEL-I',NCHAN*NBUN,2,TFUEL)
      CALL LCMPUT(IPTH,'B-PWR-I',NCHAN*NBUN,2,PWR)

      CALL LCMGET(IPTH,'BOUNDCOND',THCOND)

```

```

CALL LCMGET(IPTH,'GEOMETRY',THGEOM)
CALL LCMGET(IPTH,'EVOLUTION',THEVO)
CALL LCMGET(IPTH,'TIMEPARAM',TIMEP)
CALL LCMGET(IPNTR,'REFUEL-SCH',REFSCH)
CALL LCMGET(IPNTR,'POWER-BUN',PWR)

DO 10 I=IBEG,IFIN
  IF(IPRNT.GT.0) THEN
    WRITE(*,*) 'THERMO: CHANNEL=',I
  ENDIF
  IF(REFSCH(I).GT.0) THEN
    DO 50 J=1,NBUN
      TMPPWR(J)=PWR(I,J)*REAL(NBUN)/37.0*1000.0*0.945
50    CONTINUE
    ELSE
      DO 60 J=1,NBUN
        TMPPWR(NBUN-J+1)=PWR(I,J)*REAL(NBUN)/37.0*1000.0*0.945
60    CONTINUE
    ENDIF
    CALL NTRTHCV(NZ,NBUN,PWRD,TMPPWR)
    WRITE(CHANAM,'(A7,I5)') 'CHANNEL',I
    CALL LCMSIX(IPTH,CHANAM,1)
    CALL LCMGET(IPTH,'MIX-ENTH',HENT)
    CALL LCMGET(IPTH,'PRESSURE',PPRES)
    CALL LCMGET(IPTH,'MASS-FLUX',GMEL)
    CALL LCMGET(IPTH,'VOID-FRAC',EPSM)
    CALL LCMGET(IPTH,'FLOW-QLTY',XTIT)
    CALL LCMGET(IPTH,'LIQ-ENTH',HLQ)
    CALL LCMGET(IPTH,'PIN-TEMP',TTIG)

    CALL ARTHUR(NZ,NFUEL,NCLAD,NFUEL+NCLAD,TIMEP(1),
*   HENT,PPRES, GMEL, EPSM, XTIT, HLQ,TTIG,
*   PWRD,THCOND(1),THCOND(5),THCOND(2),THCOND(3),THCOND(4),
*   THEVO(1),THEVO(2),THEVO(3),THGEOM(6),THGEOM(2), THGEOM(3),
*   THGEOM(4), THGEOM(5), THGEOM(1), THGEOM(7),TCAL,RCAL,TCOMB,
*   VAL,IPRNT)

    CALL LCMPUT(IPTH,'MIX-ENTH',NZ*2*2,4,HENT)
    CALL LCMPUT(IPTH,'PRESSURE',NZ*2*2,4,PPRES)
    CALL LCMPUT(IPTH,'MASS-FLUX',NZ*2*2,4,GMEL)
    CALL LCMPUT(IPTH,'VOID-FRAC',NZ*2*2,4,EPSM)
    CALL LCMPUT(IPTH,'FLOW-QLTY',NZ*2*2,4,XTIT)
    CALL LCMPUT(IPTH,'LIQ-ENTH',NZ*2*2,4,HLQ)
    CALL LCMPUT(IPTH,'PIN-TEMP',NZ*(NFUEL+NCLAD)*2*2,4,TTIG)
    CALL LCMPUT(IPTH,'PWR-DSTY',NZ*2,4,PWRD)
    CALL LCMSIX(IPTH,' ',0)

    CALL THNTRCV(NZ,NBUN,TCAL,TMPTC)
    CALL THNTRCV(NZ,NBUN,RCAL,TMPRC)
    CALL THNTRCV(NZ,NBUN,TCOMB,TMPTF)
    IF(REFSCH(I).GT.0) THEN
      DO 30 J=1,NBUN
        TCOOL(I,J)=TMPTC(J)
        DCOOL(I,J)=TMPRC(J)/1000.0*1.125
        TFUEL(I,J)=TMPTF(J)
30    CONTINUE
    ELSE

```



```

DO 40 J=1,NBUN
  TCOOL(I,NBUN-J+1)=TMPTC(J)
  DCOOL(I,NBUN-J+1)=TMPRC(J)/1000.0*1.125
  TFUEL(I,NBUN-J+1)=TMPTF(J)
40  CONTINUE
ENDIF

10  CONTINUE

CALL LCMPUT(IPNTR,'B-TCOOL',NCHAN*NBUN,2,TCOOL)
CALL LCMPUT(IPNTR,'B-DCOOL',NCHAN*NBUN,2,DCOOL)
CALL LCMPUT(IPNTR,'B-TFUEL',NCHAN*NBUN,2,TFUEL)

CALL LCMPUT(IPTH,'B-TCOOL-F',NCHAN*NBUN,2,TCOOL)
CALL LCMPUT(IPTH,'B-DCOOL-F',NCHAN*NBUN,2,DCOOL)
CALL LCMPUT(IPTH,'B-TFUEL-F',NCHAN*NBUN,2,TFUEL)
CALL LCMPUT(IPTH,'B-PWR-F',NCHAN*NBUN,2,PWR)

RETURN
END

```

thermo.f

```

*DECK THERMO
  SUBROUTINE THERMO(NENTRY,HENTRY,IENTRY,JENTRY,KENTRY)
*
*
*-----
*
*
*Purpose:
*  this module offers coupling between DONJON and ARTHUR
*
*Copyright:
* Copyright (C) 2002 Ecole Polytechnique de Montreal
* This library is free software; you can redistribute it and/or
* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri, R. Chambon
*
*Parameters: input/output
* NENTRY  number of linked lists or files used by the module
* HENTRY  character*12 name of each linked list or file
* IENTRY  =1 linked list; =2 xsm file; =3 sequential binary file;
*         =4 sequential ascii file
* JENTRY  =0 the linked list or file is created
*         =1 the linked list or file is open for modifications;
*         =2 the linked list or file is open in read-only mode
* KENTRY  =file unit number; =linked list address otherwise
*
*
*

```

```

*-----
*
*
*      IMPLICIT NONE
*      INTEGER  NSTATE
*      PARAMETER (NSTATE=40)
*      INTEGER  NENTRY
*      CHARACTER HENTRY(NENTRY)*12
*      INTEGER  IENTRY(NENTRY),JENTRY(NENTRY),KENTRY(NENTRY)
*      INTEGER  THPARAM(NSTATE)

*      REDGET TYPES
*      INTEGER ITYP,NITMA
*      REAL  FLOTT
*      DOUBLE PRECISION DFLOTT
*      CHARACTER*12 TEXT

*      L_THERMO STRUCTURE
*
*      INTEGER NTHPAR
*      PARAMETER (NTHPAR=3)

*
*      SIGNATURE INFORMATION
*
*      CHARACTER*12 HSIGN
*      INTEGER ISGNTR(3)

*
*      GENERAL VARIABLES
*
*      INTEGER I,IPRNT,STA,KIN
*      REAL  PREC
*      PARAMETER(STA=1)
*      PARAMETER(KIN=2)
*      INTEGER IBEG,IFIN

*      IPRNT=0

*
*      INPUT AND OUTPUT VALIDATION
*
*      IF(NENTRY.LE.1) CALL XABORT('THERMO: PARAMETERS EXPECTED.')
*      IF(((IENTRY(1).NE.1).AND.(IENTRY(1).NE.2)) .AND.
*      + ((IENTRY(2).NE.1).AND.(IENTRY(2).NE.2)))
*      + CALL XABORT('THERMO: LINKED LIST OR XSM FILE EXPECTED AT
THERMO')
*      IF(((JENTRY(1).NE.0).AND.(JENTRY(1).NE.1)) .AND.
(JENTRY(2).NE.1))
*      + CALL XABORT('THERMO: CREATE OR MODIFICATION MODE EXPECTED.')

*
*      SIGNATURE CREATION
*
*      IF (JENTRY(1).EQ.0) THEN
*      HSIGN='L_THERMO'
*      READ(HSIGN,'(3A4)') (ISGNTR(I),I=1,3)
*      CALL LCMPUT(KENTRY(1),'SIGNATURE',3,3,ISGNTR)
*      CALL LCMSIX(KENTRY(2),' ',0)

```

```

      CALL LCMGET(KENTRY(2),'SIGNATURE',ISGNTR)
      WRITE(HSIGN,'(3A4)') (ISGNTR(I),I=1,3)
      IF (HSIGN.NE.'L_MAP')
+      CALL XABORT('THERMO: L_MAP OBJECT IS EXPECTED')
      CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
      IF(ITYP.NE.3)
+      CALL XABORT('THERMO: CHARACTER DATA EXPECTED')
*      INITIALIZATION AND PARAMETERS FOR CANDU THERMALHYDRAULIC
      IF(TEXT.EQ.'DEFCANDU') THEN
        CALL THINIT(NENTRY,HENTRY,IENTRY,JENTRY,KENTRY)
      ELSE
        CALL XABORT('THERMO: DEFINITION EXPECTED ')
      ENDIF
    ENDIF
  *
  *      SIGNATURE MODIFICATION
  *
      CALL LCMSIX(KENTRY(1),' ',0)
      CALL LCMGET(KENTRY(1),'SIGNATURE',ISGNTR)
      WRITE(HSIGN,'(3A4)') (ISGNTR(I),I=1,3)
      IF (HSIGN.NE.'L_THERMO')
+      CALL XABORT('THERMO: L_THERMO OBJECT IS EXPECTED')
      CALL LCMGET(KENTRY(2),'SIGNATURE',ISGNTR)
      WRITE(HSIGN,'(3A4)') (ISGNTR(I),I=1,3)
      IF (HSIGN.NE.'L_MAP')
+      CALL XABORT('THERMO: L_MAP OBJECT IS EXPECTED')
10      CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
      IF(ITYP.NE.3)
+      CALL XABORT('THERMO: CHARACTER DATA EXPECTED')

      IF(TEXT.EQ.'THSTATC') THEN
        CALL LCMGET(KENTRY(1),'STATE-VECTOR',THPARAM)
        CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
        IF(ITYP.NE.3)
+      CALL XABORT('THERMO: CHARACTER DATA EXPECTED')
        IF(TEXT.EQ.'SINGLP') THEN
          IBEG=1
          IFIN=THPARAM(4)
        ELSEIF(TEXT.EQ.'MULTIP') THEN
          CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
          IF(ITYP.NE.1)
+      CALL XABORT('THERMO: INTEGER DATA EXPECTED')
          IBEG=NITMA
          CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
          IF(ITYP.NE.1)
+      CALL XABORT('THERMO: INTEGER DATA EXPECTED')
          IFIN=NITMA
          THPARAM(6)=IBEG
          THPARAM(7)=IFIN
          CALL LCMPUT(KENTRY(1),'STATE-VECTOR',NSTATE,1,THPARAM)
        ELSE
          CALL XABORT('THERMO: CALCULUS TYPE EXPECTED')
        ENDIF
        CALL THCALC(KENTRY(1),THPARAM(1),THPARAM(2),THPARAM(3),
+      KENTRY(2),THPARAM(4),THPARAM(5),STA,IPRNT,IBEG,IFIN)
        ELSEIF(TEXT.EQ.'THKINEC') THEN
          CALL LCMGET(KENTRY(1),'STATE-VECTOR',THPARAM)

```

```

      CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
      IF(ITYP.NE.3)
+     CALL XABORT('THERMO: CHARACTER DATA EXPECTED')
      IF(TEXT.EQ.'SINGLP') THEN
        IBEG=1
        IFIN=THPARAM(4)
      ELSEIF(TEXT.EQ.'MULTIP') THEN
        CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
        IF(ITYP.NE.1)
+     CALL XABORT('THERMO: INTEGER DATA EXPECTED')
        IBEG=NITMA
        CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
        IF(ITYP.NE.1)
+     CALL XABORT('THERMO: INTEGER DATA EXPECTED')
        IFIN=NITMA
        THPARAM(6)=IBEG
        THPARAM(7)=IFIN
        CALL LCMPUT(KENTRY(1),'STATE-VECTOR',NSTATE,1,THPARAM)
      ELSE
        CALL XABORT('THERMO: CALCULUS TYPE EXPECTED')
      ENDIF
      CALL THCALC(KENTRY(1),THPARAM(1),THPARAM(2),THPARAM(3),
+     KENTRY(2),THPARAM(4),THPARAM(5),KIN,IPRNT,IBEG,IFIN)
      ELSEIF(TEXT.EQ.'THEVOL') THEN
        CALL LCMGET(KENTRY(1),'STATE-VECTOR',THPARAM)
        CALL THEVOL(THPARAM(4),THPARAM(1),THPARAM(2)+THPARAM(3),
+     KENTRY(1))
      ELSEIF(TEXT.EQ.'CONVER') THEN
        CALL LCMGET(KENTRY(1),'STATE-VECTOR',THPARAM)
        CALL REDGET(ITYP,NITMA,PREC,TEXT,DFLOTT)
        IF(ITYP.NE.2)
+     CALL XABORT('THERMO: REAL DATA EXPECTED')
        NITMA=1
        CALL CONVER(KENTRY(1),THPARAM(4),THPARAM(5),NITMA,PREC)
        CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
        IF(ITYP.NE.-1)
+     CALL XABORT('THERMO: INTEGER DATA EXPECTED')
        ITYP=1
        CALL REDPUT(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
      ELSEIF(TEXT.EQ.'EDIT') THEN
        CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOTT)
        IF(ITYP.NE.1)
+     CALL XABORT('THERMO: INTEGER DATA EXPECTED')
        IPRNT=NITMA
      ELSEIF(TEXT.EQ.';') THEN
        GOTO 100
      ELSE
        CALL XABORT('THERMO: OPTION EXPECTED')
      ENDIF
      GOTO 10
100  CONTINUE
      RETURN
      END

```

thevol.f

```

* DECK THEVOL
      SUBROUTINE THEVOL (NCHAN,NZ,NR,IPTH)
*-----
*
*
*Purpose:
*  this module is used to pass to the next time step
*
*Copyright:
* Copyright (C) 2002 Ecole Polytechnique de Montreal
* This library is free software; you can redistribute it and/or
* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri
*
*Parameters: input/output
*IPTH  : L_THERMO ADRESS
*NZ    : NUMBER OF DISCRETIZATION POINTS FOR THE CHANNEL
*NR    : NUMBER OF DISCRETIZATION POINTS FOR THE FUEL AND CLAD
*NCHAN : NUMBER OF CHANNELS
*
*
*-----
*
      IMPLICIT NONE
      INTEGER NZ, NR, NCHAN, I, J, K, IPTH
      DOUBLE PRECISION GMEL(NZ,2),PPRES(NZ,2),HENT(NZ,2),XTIT(NZ,2),
* EPSM(NZ,2),HLQ(NZ,2),TTIG(NR,NZ,2)
      CHARACTER CHANAM*12
      INTEGER NTHCOND,NTHEVO,NTIMEP
      PARAMETER (NTHCOND=5)
      PARAMETER (NTHEVO=3)
      PARAMETER (NTIMEP=1)
      DOUBLE PRECISION THCOND(NTHCOND),THEVO(NTHEVO),TIMEP(NTIMEP)

      CALL LCMGET(IPTH,'BOUNDCOND',THCOND)
      CALL LCMGET(IPTH,'EVOLUTION',THEVO)
      CALL LCMGET(IPTH,'TIMEPARAM',TIMEP)
      THCOND(1)=THCOND(1)+TIMEP(1)*THEVO(1)
      THCOND(2)=THCOND(2)+TIMEP(1)*THEVO(2)
      THCOND(3)=THCOND(3)+TIMEP(1)*THEVO(3)
      CALL LCMPUT(IPTH,'BOUNDCOND',2*NTHCOND,4,THCOND)

      DO 100 K=1,NCHAN
        WRITE(CHANAM,'(A7,I5)') 'CHANNEL',K
        CALL LCMSIX(IPTH,CHANAM,1)
        CALL LCMGET(IPTH,'MIX-ENTH',HENT)
        CALL LCMGET(IPTH,'PRESSURE',PPRES)
        CALL LCMGET(IPTH,'MASS-FLUX',GMEL)
        CALL LCMGET(IPTH,'VOID-FRAC',EPSM)
        CALL LCMGET(IPTH,'FLOW-QLTY',XTIT)
        CALL LCMGET(IPTH,'LIQ-ENTH',HLQ)
        CALL LCMGET(IPTH,'PIN-TEMP',TTIG)
      
```

```

      DO 10 I=1,NZ
        GMEL(I,1)=GMEL(I,2)
        HENT(I,1)=HENT(I,2)
        HLQ(I,1)=HLQ(I,2)
        PPRES(I,1)=PPRES(I,2)
        EPSM(I,1)=EPSM(I,2)
        XTIT(I,1)=XTIT(I,2)
      DO 20 J=1,NR
        TTIG(J,I,1)=TTIG(J,I,2)
20      CONTINUE
10      CONTINUE
      CALL LCMPUT(IPTH,'MIX-ENTH',NZ*2*2,4,HENT)
      CALL LCMPUT(IPTH,'PRESSURE',NZ*2*2,4,PPRES)
      CALL LCMPUT(IPTH,'MASS-FLUX',NZ*2*2,4,GMEL)
      CALL LCMPUT(IPTH,'VOID-FRAC',NZ*2*2,4,EPSM)
      CALL LCMPUT(IPTH,'FLOW-QLTY',NZ*2*2,4,XTIT)
      CALL LCMPUT(IPTH,'LIQ-ENTH',NZ*2*2,4,HLQ)
      CALL LCMPUT(IPTH,'PIN-TEMP',NZ*NR*2*2,4,TTIG)
      CALL LCMSIX(IPTH,' ',0)
100     CONTINUE
      RETURN
      END

```

thmtx.f

```

* DECK THIMTX
      SUBROUTINE THIMTX(IPTH,NZ,NFUEL,NCLAD,IPNTR,NCHAN,NBUN)
*-----
*-----
*
*Purpose:
*  this module initialize matrices needed for CANDU
thermalhydraulic
*  calculations with CHEVALIER and give thermalhydraulic states for
*  core calculations
*
*Copyright:
* Copyright (C) 2002 Ecole Polytechnique de Montreal
* This library is free software; you can redistribute it and/or
* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri
*
*Parameters: input/output
*IPTH  : L_THERMO ADRESS
*NZ    : NUMBER OF DISCRETIZATION POINTS FOR THE CHANNEL
*NFUEL : NUMBER OF DISCRETIZATION POINTS FOR THE FUEL
*NCLAD : NUMBER OF DISCRETIZATION POINTS FOR THE CLAD
*IPNTR : L_MAP ADRESS
*NBUN  : NUMBER OF BUNDLES
*NCHAN : NUMBER OF CHANNELS
*
*

```

```

*-----
*
*      IMPLICIT NONE
*      INTEGER IPTH,NZ,NFUEL,NCLAD,IPNTR,NBUN,NCHAN
*
*      THERMALHYDRAULIC MATRICES
*
*      DOUBLE PRECISION HENT(NZ,2),PPRES(NZ,2),GMEL(NZ,2),EPSM(NZ,2),
+ XTIT(NZ,2),HLQ(NZ,2),TTIG((NFUEL+NCLAD),NZ,2),PWRD(NZ),
+ TCAL(NZ),RCAL(NZ),TCOMB(NZ)
*      INTEGER NTHGEOM,NTHCOND
*      PARAMETER (NTHGEOM=7)
*      PARAMETER (NTHCOND=5)
*      DOUBLE PRECISION THGEOM(NTHGEOM), THCOND(NTHCOND)
*
*      NEUTRONIC PARAMETERS AND MATRICES
*
*      REAL TCOOL(NCHAN,NBUN),DCOOL(NCHAN,NBUN),
+ TFUEL(NCHAN,NBUN),PWR(NCHAN,NBUN)
*      INTEGER REFSCH(NCHAN)
*
*      GENERAL VARIABLES
*
*      INTEGER I,J
*      CHARACTER*12 CHANAM
*      REAL TMPTC(NBUN),TMPTF(NBUN),TMPRC(NBUN),TMPPWR(NBUN)
*      DOUBLE PRECISION DZERO,DVZERO(NZ)
*      INTEGER IZERO
*      PARAMETER (DZERO=0.0)
*      PARAMETER (IZERO=0)
*
*      DO 100 I=1,NZ
*          DVZERO(I)=DZERO
100  CONTINUE
*
*      Thermalhydraulic conversion and matrices storage
*
*      CALL LCMGET(IPTH,'BOUNDCOND',THCOND)
*      CALL LCMGET(IPTH,'GEOMETRY',THGEOM)
*      CALL ARTHUR(NZ,NFUEL,NCLAD,NCLAD+NFUEL,DZERO,
+ HENT,PPRES,GMEL,EPSM,XTIT,HLQ,TTIG,DVZERO,
*
*      THCOND(1),THCOND(5),THCOND(2),THCOND(3),THCOND(4),
*
*      DZERO,DZERO,DZERO,THGEOM(6),THGEOM(2),THGEOM(3),
*
*      THGEOM(4),THGEOM(5),THGEOM(1),THGEOM(7),TCAL,RCAL,
*      TCOMB,IZERO,IZERO)
*      CALL LCMPUT(IPTH,'BOUNDCOND',2*NTHCOND,4,THCOND)
*      CALL LCMGET(IPNTR,'REFUEL-SCH',REFSCH)
*      CALL LCMGET(IPNTR,'POWER-BUN',PWR)
*      DO 10 I=1,NCHAN
*          IF(REFSCH(I).GT.0) THEN
*              DO 50 J=1,NBUN
*                  TMPPWR(J)=PWR(I,J)*REAL(NBUN)/37.0*1000.0
50  CONTINUE

```

```

ELSE
  DO 60 J=1,NBUN
    TMPPWR(NBUN-J+1)=PWR(I,J)*REAL(NBUN)/37.0*1000.0
60    CONTINUE
  ENDIF
  CALL NTRTHCV(NZ,NBUN,PWRD,TMPPWR)
  WRITE(CHANAM,'(A7,I5)') 'CHANNEL',I
  CALL LCMSIX(IPTH,CHANAM,1)
  CALL LCMPUT(IPTH,'MIX-ENTH',NZ*2*2,4,HENT)
  CALL LCMPUT(IPTH,'PRESSURE',NZ*2*2,4,PPRES)
  CALL LCMPUT(IPTH,'MASS-FLUX',NZ*2*2,4,GMEL)
  CALL LCMPUT(IPTH,'VOID-FRAC',NZ*2*2,4,EPSM)
  CALL LCMPUT(IPTH,'FLOW-QLTY',NZ*2*2,4,XTIT)
  CALL LCMPUT(IPTH,'LIQ-ENTH',NZ*2*2,4,HLQ)
  CALL LCMPUT(IPTH,'PIN-TEMP',NZ*(NFUEL+NCLAD)*2*2,4,TTIG)
  CALL LCMPUT(IPTH,'PWR-DSTY',NZ*2,4,PWRD)
  CALL LCMSIX(IPTH,' ',0)
10  CONTINUE
  *
  *      Neutronic conversion and matrices storage
  *
  DO 20 I=1,NCHAN
    CALL THNTRCV(NZ,NBUN,TCAL,TMPTC)
    CALL THNTRCV(NZ,NBUN,RCAL,TMPRC)
    CALL THNTRCV(NZ,NBUN,TCOMB,TMPTF)
    IF(REFSCH(I).GT.0) THEN
      DO 30 J=1,NBUN
        TCOOL(I,J)=TMPTC(J)
        DCOOL(I,J)=TMPRC(J)/1000.0*1.125
        TFUEL(I,J)=TMPTF(J)
30      CONTINUE
      ELSE
        DO 40 J=1,NBUN
          TCOOL(I,NBUN-J+1)=TMPTC(J)
          DCOOL(I,NBUN-J+1)=TMPRC(J)/1000.0*1.125
          TFUEL(I,NBUN-J+1)=TMPTF(J)
40      CONTINUE
        ENDIF
      20  CONTINUE

    CALL LCMPUT(IPNTR,'B-TCOOL',NCHAN*NBUN,2,TCOOL)
    CALL LCMPUT(IPNTR,'B-DCOOL',NCHAN*NBUN,2,DCOOL)
    CALL LCMPUT(IPNTR,'B-TFUEL',NCHAN*NBUN,2,TFUEL)

    CALL LCMPUT(IPTH,'B-TCOOL-I',NCHAN*NBUN,2,TCOOL)
    CALL LCMPUT(IPTH,'B-DCOOL-I',NCHAN*NBUN,2,DCOOL)
    CALL LCMPUT(IPTH,'B-TFUEL-I',NCHAN*NBUN,2,TFUEL)
    CALL LCMPUT(IPTH,'B-PWR-I',NCHAN*NBUN,2,PWR)

    CALL LCMPUT(IPTH,'B-TCOOL-F',NCHAN*NBUN,2,TCOOL)
    CALL LCMPUT(IPTH,'B-DCOOL-F',NCHAN*NBUN,2,DCOOL)
    CALL LCMPUT(IPTH,'B-TFUEL-F',NCHAN*NBUN,2,TFUEL)
    CALL LCMPUT(IPTH,'B-PWR-F',NCHAN*NBUN,2,PWR)

    RETURN
  END

```


thinit.f

```

* DECK THINIT
  SUBROUTINE THINIT(NENTRY,HENTRY,IENTRY,JENTRY,KENTRY)

-----
----
*
*Purpose:
*   this module read parameters for CANDU thermalhydraulic
*   calculations with CHEVALIER
*
*Copyright:
*   Copyright (C) 2002 Ecole Polytechnique de Montreal
*   This library is free software; you can redistribute it and/or
*   modify it under the terms of the GNU Lesser General Public
*   License as published by the Free Software Foundation; either
*   version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri, R. Chambon
*
*Parameters: input/output
* NENTRY  number of linked lists or files used by the module
* HENTRY  character*12 name of each linked list or file
* IENTRY  =1 linked list; =2 xsm file; =3 sequential binary file;
*         =4 sequential ascii file
* JENTRY  =0 the linked list or file is created
*         =1 the linked list or file is open for modifications;
*         =2 the linked list or file is open in read-only mode
* KENTRY  =file unit number; =linked list address otherwise
*
-----
----
*
*
*   IMPLICIT NONE
*   INTEGER  NSTATE
*   PARAMETER (NSTATE=40)
*   INTEGER NENTRY
*   CHARACTER HENTRY(NENTRY)*12
*   INTEGER  IENTRY(NENTRY),JENTRY(NENTRY),KENTRY(NENTRY)

* REDGET TYPES
*   INTEGER ITYP,NITMA
*   REAL FLOTT
*   DOUBLE PRECISION DFLOT
*   CHARACTER*12 TEXT

* General Variables
*   INTEGER  I

```

```

*
* L_THERMO VARIABLES
*
      INTEGER THPARAM(NSTATE)
      INTEGER NTHDISC,NTHGEOM,NTHCOND,NTHEVO,NTIMEP
      PARAMETER (NTHDISC=3)
      PARAMETER (NTHGEOM=7)
      PARAMETER (NTHCOND=5)
      PARAMETER (NTHEVO=3)
      PARAMETER (NTIMEP=1)
      DOUBLE PRECISION
THGEOM(NTHGEOM), THCOND(NTHCOND), THEVO(NTHEVO),
+ TIMEP(NTIMEP)

*
* L_MAP VARIABLES
*
      INTEGER NTRPARAM(NSTATE)

*
* READ INPUT DATA FOR THERMALHYDRAULIC
*
      DO 10
        CALL REDGET(ITYP,NITMA,FLOTT,TEXT,DFLOT)
        IF (ITYP.NE.3)
+ CALL XABORT('DEFCANDU: CHARACTER DATA EXPECTED')
        IF(TEXT.EQ.'DISCRET') THEN
          CALL XDISET(THPARAM,NSTATE,0)
          DO 20 I=1,NTHDISC
            CALL REDGET(ITYP,THPARAM(I),FLOTT,TEXT,DFLOT)
            IF (ITYP.NE.1)
+ CALL XABORT('DISCRET: INTEGER DATA EXPECTED')
20        CONTINUE
          ELSEIF(TEXT.EQ.'GEOM') THEN
            DO 30 I=1,NTHGEOM
              CALL REDGET(ITYP,NITMA,FLOTT,TEXT,THGEOM(I))
              IF (ITYP.NE.4)
+ CALL XABORT('GEOM: DOUBLE DATA EXPECTED')
30        CONTINUE
          CALL LCMPUT(KENTRY(1),'GEOMETRY',2*NTHGEOM,4,THGEOM)
          ELSEIF(TEXT.EQ.'THCOND') THEN
            DO 40 I=1,NTHCOND
              IF (I.LE.(NTHCOND-1)) THEN
                CALL REDGET(ITYP,NITMA,FLOTT,TEXT,THCOND(I))
                IF (ITYP.NE.4)
+ CALL XABORT('THCOND: DOUBLE DATA EXPECTED')
              ELSE
                THCOND(I)=0.0
              ENDIF
40        CONTINUE
          CALL LCMPUT(KENTRY(1),'BOUNDCOND',2*NTHCOND,4,THCOND)
          ELSEIF(TEXT.EQ.'EVOLC') THEN
            DO 50 I=1,NTHEVO
              CALL REDGET(ITYP,NITMA,FLOTT,TEXT,THEVO(I))
              IF (ITYP.NE.4)
+ CALL XABORT('EVOLC: REAL DATA EXPECTED')
50        CONTINUE

```

```

        CALL LCMPUT(KENTRY(1), 'EVOLUTION', 2*NTHEVO, 4, THEVO)
    ELSEIF(TEXT.EQ. 'TIMEPAR') THEN
        DO 60 I=1, NTIMEP
            CALL REDGET(ITYP, NITMA, FLOTT, TEXT, TIMEP(I))
            IF (ITYP.NE.4)
+       CALL XABORT('EVLGC: REAL DATA EXPECTED')
60      CONTINUE
        CALL LCMPUT(KENTRY(1), 'TIMEPARAM', 2*NTIMEP, 4, TIMEP)
    ELSEIF(TEXT.EQ. 'ENDDDEF') THEN
        GOTO 100
    ELSE
        CALL XABORT('DEFCANDU: WRONG OPTION')
    ENDIF
10     CONTINUE
100    CONTINUE
*
*   LOAD DATA FROM NEUTRONIC
*
        CALL LCMGET(KENTRY(2), 'STATE-VECTOR', NTRPARAM)
*   NUMBER OF CHANNELS
        THPARAM(4) = NTRPARAM(2)
*   NUMBER OF BUNDLES
        THPARAM(5) = NTRPARAM(1)
        CALL LCMPUT(KENTRY(1), 'STATE-VECTOR', NSTATE, 1, THPARAM)
*
*   CREATE MATRICES FOR THERMALHYDRAULIC AND UPDATE
*   NEUTRONIQUE MATRICES
*
        CALL THIMTX(KENTRY(1), THPARAM(1), THPARAM(2), THPARAM(3),
+       KENTRY(2), THPARAM(4), THPARAM(5))

        RETURN
        END

```

thntrcv.f

```

*   DECK THNTRCV
        SUBROUTINE THNTRCV(NZ, NBUN, VECTA, VECTB)
*
*
*   CONVERT NZ DOUBLE PRECISION VECTORS TO
*   NBUN SIMPLE PRECISION VECTOR USING USUAL MEAN
*
*   Author(s) : M. Fassi Fehri
*
*   NZ      : DOUBLE PRECISION VECTOR DIMENSION
*   NBUN    : SIMPLE PRECISION VECTOR DIMENSION
*   VECTA   : DOUBLE PRECISION VECTOR
*   VECTB   : SIMPLE PRECISION VECTOR
*
        IMPLICIT NONE
*
*   VECTOR CONVERSION
*
        INTEGER NZ, NBUN

```

```

      DOUBLE PRECISION VECTA(NZ)
      REAL VECTB(NBUN)
*
*  GENERAL VARIABLES
*
      DOUBLE PRECISION TEMP
      INTEGER I,J,K
      DOUBLE PRECISION DZ,DB

      DZ=1.0/(DBLE(NZ)-1.0)
      DB=1.0/DBLE(NBUN)

      K=1
      J=0
      TEMP=0.0

      DO 10 I=1,NZ-1

        J=J+1

        TEMP=TEMP+VECTA(I)

        IF((REAL(J)*DZ).GE.DB) THEN
          TEMP=TEMP/J
          VECTB(K)=REAL(TEMP)
          TEMP=0.0
          K=K+1
          J=0
        ENDIF

10    CONTINUE

      TEMP=TEMP/J
      VECTB(K)=REAL(TEMP)

      RETURN
      END

```

thupdt.f

```

*DECK THUPDT
      SUBROUTINE THUPDT(NENTRY,HENTRY,IENTRY,JENTRY,KENTRY)
*-----
*
*Purpose:
*  this module update thermalhydraulic properties in L_MAP
*
*Copyright:
* Copyright (C) 2002 Ecole Polytechnique de Montreal
* This library is free software; you can redistribute it and/or

```

```

* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version
*
*Author(s): M. Fassi Fehri
*
*Parameters: input/output
* NENTRY  number of linked lists or files used by the module
* HENTRY  character*12 name of each linked list or file
* IENTRY  =1 linked list; =2 xsm file; =3 sequential binary file;
*          =4 sequential ascii file
* JENTRY  =0 the linked list or file is created
*          =1 the linked list or file is open for modifications;
*          =2 the linked list or file is open in read-only mode
* KENTRY  =file unit number; =linked list address otherwise
*
*-----
---
      IMPLICIT NONE
*
*  PARAMETER VARIABLES
*
      INTEGER  NSTATE
      PARAMETER (NSTATE=40)
      INTEGER  NENTRY
      CHARACTER HENTRY(NENTRY)*12
      INTEGER  IENTRY(NENTRY), JENTRY(NENTRY), KENTRY(NENTRY)
      INTEGER  PARAM(NSTATE)
      INTEGER  IOPOS
      INTEGER          IBASE(1)
      REAL           RBASE
      COMMON          RBASE(1)
      EQUIVALENCE     (RBASE(1), IBASE(1))
*
*  SIGNATURE INFORMATION
*
      CHARACTER*12 HSIGN
      INTEGER  ISGNTR(3)
*
*  GENERAL VARIABLES
*
      INTEGER  J
*
*  INPUT AND OUTPUT VALIDATION
*
      IF(NENTRY.LE.1) CALL XABORT('THUPDT: PARAMETERS EXPECTED.')

      IF((IENTRY(1).NE.1).AND.(IENTRY(1).NE.2).AND.
+ (IENTRY(2).NE.1).AND.(IENTRY(2).NE.2))
+ CALL XABORT('THUPDT: LINKED LIST OR XSM FILE EXPECTED')

      IF(JENTRY(1).NE.1)
+ CALL XABORT('THUPDT: MODIFICATION MODE EXPECTED.')
      IF(JENTRY(2).NE.2)

```

```

+ CALL XABORT('THUPDT: READ ONLY MODE EXPECTED.')

CALL LCMSIX(KENTRY(1), ' ', 0)
CALL LCMGET(KENTRY(1), 'SIGNATURE', ISGNTR)
WRITE(HSIGN, '(3A4)') (ISGNTR(J), J=1, 3)
WRITE(*, *) HSIGN
IF (HSIGN.NE.'L_MAP')
+ CALL XABORT('THUPDT: L_MAP OBJECT IS EXPECTED')
CALL LCMSIX(KENTRY(2), ' ', 0)
CALL LCMGET(KENTRY(2), 'SIGNATURE', ISGNTR)
WRITE(HSIGN, '(3A4)') (ISGNTR(J), J=1, 3)
WRITE(*, *) HSIGN
IF (HSIGN.NE.'L_THERMO')
+ CALL XABORT('THUPDT: L_THERMO OBJECT IS EXPECTED')

*
*   THERMALHYDRAULIC MATRIX DIMENSIONS
*
CALL LCMGET(KENTRY(1), 'STATE-VECTOR', PARAM)

CALL SETARA(RBASE, PARAM(1)*PARAM(2), IOPOS)

CALL LCMGET(KENTRY(2), 'B-TCOOL-F', RBASE(IOPOS))
CALL LCMPUT(KENTRY(1), 'B-
TCOOL', PARAM(1)*PARAM(2), 2, RBASE(IOPOS))
CALL LCMGET(KENTRY(2), 'B-DCOOL-F', RBASE(IOPOS))
CALL LCMPUT(KENTRY(1), 'B-
DCOOL', PARAM(1)*PARAM(2), 2, RBASE(IOPOS))
CALL LCMGET(KENTRY(2), 'B-TFUEL-F', RBASE(IOPOS))
CALL LCMPUT(KENTRY(1), 'B-
TFUEL', PARAM(1)*PARAM(2), 2, RBASE(IOPOS))

CALL RLSARA(RBASE(IOPOS))

RETURN
END

```

ANNEXE 7 : Fichiers d'entrée x2m et c2m.

Calcul stationnaire de réacteur à quatre canaux :

```

!*****
!* Calcul DONJON pour un reacteur fictif a 4 canaux      **
!*   test pour developpement du module THERMO          **
!*                                                     **
!*****
!* R. Chambon (23/11/2007)                             **
!* M. Fassi Fehri (28/01/2007)                         **
!*****

MODULE   END: GREP: CRE: DELETE: FLXAXC: REFUEL: INIMAC: INIRES:
        POWER: FLUD: GEOD: USPLIT: XSCONS: TRIVAA: TRIVAT:
        REFRES: THERMO: AFM: ;

SEQ_ASCII fnat rnat neutrodata thermodata MACFME5 ;

LINKED_LIST NFUEL NREFL GEOM INDEX MACRO DEVICE PROCEDE RESEAU
            TAB GEOM2 SYSTEM ;
LINKED_LIST MACRO2 MACRES FLUX TRACK THER DBASE ;

!* Base de donnees AFM
!*****
DBASE := MACFME5 ;

!* Declaration des variables de neutronique
!*****
INTEGER Maxreg ;
REAL B1      := 7500.0      ;
REAL        Keff ;
INTEGER      Iter Edit := 0 0 ;
REAL        Precf := 1.E-6 ;
REAL        Pow := 2064.0 380.0 / 4.0 * ;
REAL        Eps1 := 1.0 ;
STRING      NamCpo1 NamCpo2 ;
EVALUATE NamCpo1 := "FULBU7114WGT" ;
EVALUATE NamCpo2 := "FULBU5135WGT" ;

!* Declaration des variables de thermohydraulique
!*****
INTEGER nz  nf  nc  := 120 16 4 ;
DOUBLE lz  df  tg  dc  dp  theta  rufc :=
    5.9436D0 1.12D-2 3.8D-4 1.304D-2 1.486D-2 1.570796320D0
3.0D-7 ;
DOUBLE mfin pout tinit tcini := 7.007879D3 1.1205836D7 5.3552D2
6.0D2 ;
DOUBLE evolg evolt evolv := 0.0D0 0.0D0 0.0D0 ;
DOUBLE dt := 1.0D-1 ;
INTEGER cv iterc ;
EVALUATE cv iterc := 0 1 ;

```

```

REAL prec := 1.E-4 ;

!* Parametres de controle de la simulation
!*****
EVALUATE Maxreg := 27700 ; ! nombre de regions dans la geometrie

!* Definition de la gemometrie a 2 zones
!*****
GEOM := GEOD: :: CAR3D 4 4 12
      EDIT 0

      X- ZERO   X+ ZERO
      Y- ZERO   Y+ ZERO
      Z- ZERO   Z+ ZERO

MIX

PLANE 1
  2 2 2 2
  2 1 1 2
  2 1 1 2
  2 2 2 2
PLANE 2 SAME 1
PLANE 3 SAME 1
PLANE 4 SAME 1
PLANE 5 SAME 1
PLANE 6 SAME 1
PLANE 7 SAME 1
PLANE 8 SAME 1
PLANE 9 SAME 1
PLANE 10 SAME 1
PLANE 11 SAME 1
PLANE 12 SAME 1

MESHX 68.525 97.100 125.675 154.250 182.825
MESHY 68.525 97.100 125.675 154.250 182.825
MESHZ 3.940 53.470 103.000 152.530 202.060 251.590 301.120
350.650
      400.180 449.710 499.240 548.770 598.30
;
GEOM2 INDEX := USPLIT: GEOM :: MAXR <<Maxreg>> ;

!* Liaison avec le fichier de composition
!*****
NFUEL := fnat ;
NREFL := rnat ;

!* Generation des melanges (sections efficaces macroscopiques)
!*****
MACRO := CRE: NREFL ::
      EDIT 0 NMIX 2
      READ
      COMPO NREFL MIX 2 'MIXTMOD 1' ENDMIX
;

```



```

!* Creation d'une table des prop. du combustible en fonction du
burnup
!*****
**
TAB := XSCONS: NFUEL ::
      EDIT 0
      READ
      COMPO NFUEL TYPE 'NATURAL' 'MIXTRXE' 1'
      BURNUP MICRO ALL INTRPL 100.0 ENDTYP ;

!* Creation et initialisation de RESEAU
!*****
RESEAU := INIRES: ::

      NBUND 12 NCHAN 4 NZONE 1

      ::: GEOD: CAR3D 4 4 12
      EDIT 0

      X- VOID      X+ VOID
      Y- VOID      Y+ VOID
      Z- VOID      Z+ VOID

      MIX
      PLANE 1
      0 0 0 0
      0 1 1 0
      0 1 1 0
      0 0 0 0
      PLANE 2 SAME 1
      PLANE 3 SAME 1
      PLANE 4 SAME 1
      PLANE 5 SAME 1
      PLANE 6 SAME 1
      PLANE 7 SAME 1
      PLANE 8 SAME 1
      PLANE 9 SAME 1
      PLANE 10 SAME 1
      PLANE 11 SAME 1
      PLANE 12 SAME 1

      MESHX 68.525 97.100 125.675 154.250 182.825
      MESHY 68.525 97.100 125.675 154.250 182.825
      MESHZ 3.940 53.470 103.000 152.530 202.060 251.590 301.120
350.650
      400.180 449.710 499.240 548.770 598.30 ;

      NXNAME '-' '1' '2' '-'
      NYNAME '-' 'A' 'B' '-'

      BURN-ZONE
      1 1
      1 1

      BURNUP-ZC <<B1>>

```

```

;

!* Lien entre le reseau et la geometrie
!*****
INDEX RESEAU := REFRES: INDEX RESEAU GEOM2 ;

!* Initialisation de la forme axiale du flux
!*****
RESEAU := FLXAXC: RESEAU MACRO :: AXIAL REFR COMP FLAT ;

!* Creation de TRACK
!*****
TRACK := TRIVAT: GEOM2 ::
    EDIT 0
    TITLE 'G2 3D '
    MAXR <<Maxreg>> MCFD 1 ;

!* Boucle de couplage
!*****

REPEAT

!* Boucle pour converger sur la forme axiale du flux
!*****
REPEAT

    EVALUATE Iter := Iter 1 + ;

    RESEAU := REFUEL: RESEAU TAB :: TAVGC NS 8
        NTYP 1 TYPE 1 'NATURAL' ;

!* Calcul des sections efficaces
!*****

MACRES := AFM: DBASE RESEAU ::
    MAP
    INFOR SISSAOUI_ENDFB5_CANDU6_FEBRUARY_98
    DNAME 2 <<NamCpol>> <<NamCpol>>
    EDIT 0
    REFT 1 <<NamCpol>> 3 <<NamCpol>> ;

MACRO2 := INIMAC: INDEX MACRO MACRES ;

!* Creation des matrices de systeme d'elements finis
!*****
SYSTEM := TRIVAA: MACRO2 TRACK :: EDIT 0 ;

!* Calcul du flux (probleme aux valeurs propres)

```

```

!*****
  FLUX := FLUD: SYSTEM TRACK :: EDIT 0 PREC <<Precf>> ;

!* Calcul de la forme de flux axial
!*****
  RESEAU := FLXAXC: RESEAU FLUX TRACK INDEX :: AXIAL COMP ;

!* Calcul des puissances (normalisees) de canal
!*****
  RESEAU := POWER: RESEAU MACRES :: EDIT <<Edit>> POWER <<Pow>> ;

!* Ecriture des variables de sorties de la procedure
!*****
  GREP: RESEAU :: GETVAL EPSF-AX 1 >>Eps1<< ;
  GREP: FLUX :: GETVAL K-EFFECTIVE 1 >>Keff<< ;
  MACRO2 MACRES SYSTEM FLUX := DELETE: MACRO2 MACRES SYSTEM FLUX ;

UNTIL Eps1 0.001 < Iter 10 = + ;

!* Initialisation de la geo. et des prop. thermohydrauliques
!*****
IF iterc 2 < THEN
  THER RESEAU := THERMO: RESEAU ::
    DEFCANDU
      DISCRET <<nz>> <<nf>> <<nc>>
      GEOM <<lz>> <<df>> <<tg>> <<dc>> <<dp>> <<theta>> <<rufc>>
      THCOND <<mfin>> <<pout>> <<tinit>> <<tcini>>
      EVOLC <<evolg>> <<evolt>> <<evolp>>
      TIMEPAR <<dt>>
    ENDDEF
  !* Calcul thermohydraulique
  !*****
  THSTATC SINGLP
;
ELSE
  !* Calcul thermohydraulique
  !*****
  THER RESEAU := THERMO: THER RESEAU ::
    THSTATC SINGLP
    CONVER <<prec>> >>cv<<
  ;
ENDIF ;

EVALUATE iterc := iterc 1 + ;

UNTIL cv 0 > iterc 100 = + ;

thermodata := THER ;
neutrodata := RESEAU ;

END: ;

QUIT .

```

Calcul transitoire de réacteur à quatre canaux :

```

!*****
!* Calcul DONJON pour un reacteur fictif a 4 canaux          **
!*   test pour developpement du module THERMO                **
!*                                                            **
!*****
!* R. Chambon (23/11/2007)                                    **
!* M. Fassi Fehri (2/02/2008)                                **
!*****

MODULE GREP: CRE: DELETE: FLXAXC: REFUEL: INIMAC: INIRES: ;
MODULE POWER: FLUD: GEOD: USPLIT: XSCONS: TRIVAA: TRIVAT: ;
MODULE REFRES: THERMO: AFM: TRIVAA: DMAC: IQS: END: ;

SEQ_ASCII fnat rnat ETAT1 ETAT2 ETAT3 ETAT4 macfme5 ;

LINKED_LIST NFUEL NREFL GEOM INDEX MACRO DEVICE PROCEDE RESEAU
            TAB GEOM2 SYSTEM FLUXP FLUXPI SYSTEMP ;
LINKED_LIST MACRO2 MACRO2P DMACRO MACRES FLUX TRACK THER DBASE ;

!* Base de donnees AFM
!*****
DBASE := macfme5 ;

!* Declaration des variables de neutronique
!*****
INTEGER Maxreg ;
INTEGER TPS ;
REAL B1      := 7500.0 ;
REAL        Keff ;
INTEGER      Iter Edit := 0 0 ;
REAL        Precf := 1.E-6 ;
REAL        Powi := 2064.0 380.0 / 4.0 * ;
REAL        Powf ;
REAL        Eps1 := 1.0 ;
STRING      NamCpo1 NamCpo2 ;
EVALUATE NamCpo1 := "FULBU7114WGT" ;
EVALUATE NamCpo2 := "FULBU5135WGT" ;

!* Declaration des variables de thermohydraulique
!*****
INTEGER nz  nf  nc  := 120 16 4 ;
DOUBLE lz  df  tg  dc  dp  theta rufc :=
    5.9436D0 1.12D-2 3.8D-4 1.304D-2 1.486D-2 1.570796320D0
3.0D-7 ;
DOUBLE mfin pout tinit tcini := 7.007879D3 1.1205836D7 5.3552D2
6.0D2 ;
DOUBLE evolg evolt evolp := -5.0D0 0.0D0 0.0D0 ;
DOUBLE dt := 1.0D-1 ;
INTEGER cv iterc := 0 1 ;
REAL prec := 1.E-4 ;

!* Parametres de controle de la simulation
!*****

```

```
EVALUATE Maxreg := 27700 ; ! nombre de regions dans la geometrie
```

```
!* Definition de la geometrie a 2 zones
```

```
!*****
```

```
GEOM := GEOD: :: CAR3D 4 4 12
```

```
EDIT 0
```

```
X- ZERO X+ ZERO
```

```
Y- ZERO Y+ ZERO
```

```
Z- ZERO Z+ ZERO
```

```
MIX
```

```
PLANE 1
```

```
2 2 2 2
```

```
2 1 1 2
```

```
2 1 1 2
```

```
2 2 2 2
```

```
PLANE 2 SAME 1
```

```
PLANE 3 SAME 1
```

```
PLANE 4 SAME 1
```

```
PLANE 5 SAME 1
```

```
PLANE 6 SAME 1
```

```
PLANE 7 SAME 1
```

```
PLANE 8 SAME 1
```

```
PLANE 9 SAME 1
```

```
PLANE 10 SAME 1
```

```
PLANE 11 SAME 1
```

```
PLANE 12 SAME 1
```

```
MESHX 68.525 97.100 125.675 154.250 182.825
```

```
MESHY 68.525 97.100 125.675 154.250 182.825
```

```
MESHZ 3.940 53.470 103.000 152.530 202.060 251.590 301.120
```

```
350.650
```

```
400.180 449.710 499.240 548.770 598.30
```

```
;
```

```
GEOM2 INDEX := USPLIT: GEOM :: MAXR <<Maxreg>> ;
```

```
!* Liaison avec le fichier de composition
```

```
!*****
```

```
NFUEL := fnat ;
```

```
NREFL := rnat ;
```

```
!* Generation des melanges (sections efficaces macroscopiques)
```

```
!*****
```

```
MACRO := CRE: NREFL ::
```

```
EDIT 0 NMIX 2
```

```
READ
```

```
COMPO NREFL MIX 2 'MIXTMOD 1' ENDMIX
```

```
;
```

```
!* Creation d'une table des prop. du combustible en fonction du  
burnup
```

```
!*****
```

```
**
```

```
TAB := XSCONS: NFUEL ::
```

```

EDIT 0
READ
COMPO NFUEL  TYPE 'NATURAL' 'MIXTRXE    1'
              BURNUP MICRO ALL INTRPL 100.0 ENDTYP ;

!* Creation et initialisation de RESEAU
!*****
RESEAU := INIRES: ::

NBUND  12  NCHAN  4  NZONE  1

::: GEOD:  CAR3D 4 4 12
EDIT 0

X- VOID      X+ VOID
Y- VOID      Y+ VOID
Z- VOID      Z+ VOID

MIX
PLANE 1
  0 0 0 0
  0 1 1 0
  0 1 1 0
  0 0 0 0
PLANE 2  SAME 1
PLANE 3  SAME 1
PLANE 4  SAME 1
PLANE 5  SAME 1
PLANE 6  SAME 1
PLANE 7  SAME 1
PLANE 8  SAME 1
PLANE 9  SAME 1
PLANE 10 SAME 1
PLANE 11 SAME 1
PLANE 12 SAME 1

MESHX  68.525 97.100 125.675 154.250 182.825
MESHY  68.525 97.100 125.675 154.250 182.825
MESHZ  3.940 53.470 103.000 152.530 202.060 251.590 301.120
350.650
      400.180 449.710 499.240 548.770 598.30 ;

NXNAME  '-' '1' '2' '-'
NYNAME  '-' 'A' 'B' '-'

BURN-ZONE
1 1
1 1

BURNUP-ZC  <<B1>>
;

!* Lien entre le reseau et la geometrie

```

```

!*****
INDEX RESEAU := REFRES: INDEX RESEAU GEOM2 ;

!* Initialisation de la forme axiale du flux
!*****
RESEAU := FLXAXC: RESEAU MACRO :: AXIAL REFR COMP FLAT ;

!* Creation de TRACK
!*****
TRACK := TRIVAT: GEOM2 ::
      EDIT 0
      TITLE 'G2 3D '
      MAXR <<Maxreg>> MCFD 1 ;

!* Boucle de couplage
!*****

REPEAT

      ECHO "ITERATION NUMERO: " iterc ;

!* Boucle pour converger sur la forme axiale du flux
!*****
REPEAT

      ECHO "CALCUL DE NEUTRONIQUE STATIONNAIRE" ;

      IF Iter 0 > THEN
        MACRO2 MACRES SYSTEM FLUX := DELETE: MACRO2 MACRES SYSTEM FLUX ;
      ENDIF ;

      EVALUATE Iter := Iter 1 + ;

      RESEAU := REFUEL: RESEAU TAB :: TAVGC NS 8
              NTYP 1 TYPE 1 'NATURAL' ;

!* Calcul des sections efficaces
!*****

MACRES := AFM: DBASE RESEAU ::
      MAP
      INFOR SISSAOUI_ENDFB5_CANDU6_FEBRUARY_98
      DNAME 2 <<NamCpo1>> <<NamCpo1>>
      EDIT 0
      REFT 1 <<NamCpo1>> 3 <<NamCpo1>> ;

```

```

MACRO2 := INIMAC: INDEX MACRO MACRES ;

!* Creation des matrices de systeme d'elements finis
!*****
SYSTEM := TRIVAA: MACRO2 TRACK :: EDIT 0 ;

!* Calcul du flux (probleme aux valeurs propres)
!*****
FLUX := FLUD: SYSTEM TRACK :: EDIT 0 PREC <<Precf>> ADJ ;

!* Calcul de la forme de flux axial
!*****
RESEAU := FLXAXC: RESEAU FLUX TRACK INDEX :: AXIAL COMP ;

!* Calcul des puissances (normalisees) de canal
!*****
RESEAU := POWER: RESEAU MACRES :: EDIT <<Edit>> POWER <<Powi>> ;

!* Ecriture des variables de sorties de la procedure
!*****
GREP: RESEAU :: GETVAL EPSF-AX 1 >>Eps1<< ;
GREP: FLUX :: GETVAL K-EFFECTIVE 1 >>Keff<< ;

UNTIL Eps1 0.001 < Iter 100 = + ;

!* Initialisation de la geo. et des prop. thermohydrauliques
!*****

IF iterc 2 < THEN

ECHO "CALCUL DE THERMOHYDRAULIQUE STATIONNAIRE 1" ;

THER RESEAU := THERMO: RESEAU ::
DEFCANDU
DISCRET <<nz>> <<nf>> <<nc>>
GEOM <<lz>> <<df>> <<tg>> <<dc>> <<dp>> <<theta>> <<rufc>>
THCOND <<mfin>> <<pout>> <<tinit>> <<tcini>>
EVOLC <<evolg>> <<evolt>> <<evolp>>
TIMEPAR <<dt>>
ENDDEF
THSTATC SINGLP ;
ELSE

!* Calcul thermohydraulique statique
!*****

ECHO "CALCUL DE THERMOHYDRAULIQUE STATIONNAIRE 2" ;
THER RESEAU := THERMO: THER RESEAU ::
THSTATC SINGLP

```



```

        CONVER <<prec>> >>cv<<
    ;

ENDIF ;

EVALUATE interc := interc 1 + ;

UNTIL cv 0 > interc 100 = + ;

MACRES := DELETE: MACRES ;

!* CINETIQUE
!*****

EVALUATE interc := 0 ;
EVALUATE cv := 0 ;
EVALUATE TPS := 1 ;

!* Boucle de temps
!*****
REPEAT

!* Boucle de convergence
!*****
REPEAT

    EVALUATE interc := interc 1 + ;

    ECHO " ++++++ " ;
    ECHO " COUPLAGE CINETIQUE AU TEMPS: " TPS ;
    ECHO " ITERATION: " interc ;
    ECHO " ++++++ " ;

    IF TPS 1 = THEN
        IF interc 1 > THEN
            MACRO2P DMACRO SYSTEMP FLUXP MACRES := DELETE:
            MACRO2P DMACRO SYSTEMP FLUXP MACRES ;
        ENDIF ;
    ELSE
        MACRO2P DMACRO SYSTEMP MACRES := DELETE:
        MACRO2P DMACRO SYSTEMP MACRES ;
    ENDIF ;

    ECHO "CALCUL DE THERMOHYDRAULIQUE CINETIQUE" ;
    THER RESEAU := THERMO: THER RESEAU ::
    THKINEC SINGLP
    CONVER <<prec>> >>cv<<
    ;

MACRES := AFM: DBASE RESEAU ::
MAP
INFOR SISSAOUI_ENDFB5_CANDU6_FEBRUARY_98
DNAME 2 <<NamCpol>> <<NamCpol>>

```

```

EDIT 0
REFT 1 <<NamCpol>> 3 <<NamCpol>> ;

MACRO2P := INIMAC: INDEX MACRO MACRES ;

DMACRO := DMAC: MACRO2P MACRO2 :: STEP 1 ;

SYSTEMP := TRIVAA: MACRO2 TRACK DMACRO :: PERT UNIT OVEL ;

ECHO "CALCUL DE NEUTRONIQUE CINETIQUE" ;

IF TPS 1 > THEN
  FLUXP := DELETE: FLUXP ;
  FLUXP := FLUXPI ;
ENDIF ;

IF TPS 1 = THEN
  ECHO "CALCUL DE NEUTRONIQUE CINETIQUE 1" ;
  FLUXP := IQS: FLUX SYSTEMP SYSTEM TRACK DMACRO MACRO2 GEOM2 ::
  EDIT 0
  DELN 6
  BETA 0.000250 0.001562 0.001505 0.002755 0.000830 0.000317
  DECR 0.0129 0.0311 0.134 0.331 1.26 3.21
  TETA VTET 1.0
  POWE <<Powi>>
  BEGIN PERT 1 STEP 0.1 END
  ;
ELSE
  ECHO "CALCUL DE NEUTRONIQUE CINETIQUE 2" ;
  FLUXP := IQS: FLUXP FLUX SYSTEMP SYSTEM TRACK DMACRO MACRO2 GEOM2
  ::
  EDIT 0
  TETA VTET 1.0
  POWE <<Powi>>
  TMAC
  BEGIN PERT 1 STEP 0.1 END
  ;
ENDIF ;

RESEAU := FLXAXC: RESEAU FLUXP TRACK INDEX :: FLUX-AV ;

GREP: FLUXP :: GETVAL POWER 1 >>Powf<< ;

RESEAU := POWER: RESEAU MACRES :: POWER <<Powf>> ;

UNTIL cv 0 > iterc 10 = + ;

ECHO "CALCUL D'EVOLUTION " ;

EVALUATE Powi := Powf ;

EVALUATE cv := 0 ;
EVALUATE iterc := 0 ;
EVALUATE TPS := TPS 1 + ;

```

```

MACRO2 SYSTEM := DELETE: MACRO2 SYSTEM ;
MACRO2 := MACRO2P ;
SYSTEM := TRIVAA: MACRO2 TRACK :: EDIT 0 ;

```

```

IF TPS 2 > THEN
  FLUXPI := DELETE: FLUXPI ;
ENDIF ;
FLUXPI := FLUXP ;

```

```

THER RESEAU := THERMO: THER RESEAU ::
  THEVOL
;

```

```

IF TPS 150 = THEN
  ETAT1 := RESEAU ;
ELSEIF TPS 300 = THEN
  ETAT2 := RESEAU ;
ELSEIF TPS 450 = THEN
  ETAT3 := RESEAU ;
ELSEIF TPS 600 = THEN
  ETAT4 := RESEAU ;
ENDIF ;

```

```

UNTIL TPS 600 > ;

```

```

END: ;

```

```

QUIT .

```

Calcul stationnaire de réacteur CANDU :

```

!*****
!* Test for THERMO module development in DONJON3 **
!* **
!* **
!* **
!*****
!* Calcul DONJON a l'equilibre du rechargement a deux (2) zones **
!* **
!* Determination du burnup moyen de sortie pour un coeur **
!* avec des grappes quelconques provenant de fnat **
!* **
!* COURS ENE6209 **
!* **
!* Options: **
!* 1- Rapport B1/B2 (B1 zone centrale, B2 zone peripherique) **
!* 2- Position des BC **
!*****

```

```

!* B. Dionne (03/04/2001) **
!* E. Varin (23/02/2004) **
!* R. Chambon (18/10/2007) **
!* M. Fassi Fehri (02/02/2008) **
!*****
PROCEDURE Pburncal Pcalflu ;

MODULE          END:   CRE:   FLXAXC: REFUEL: INIMAC:
                POWER: FLUNRM: THERMO: AFM: DELETE: ;

SEQ_ASCII fnat thermodata neutrodata MACFME5 ;

LINKED_LIST NFUEL G23D2 INDEX MACRO DEVICE PROCEDE RESEAU TAB ;
LINKED_LIST MACRO2 MACRES FLUX TRACK MACOLD THER DBASE ;

!* Base de donnees AFM
!*****
DBASE := MACFME5 ;

!* Declaration des variables neutronique
!*****
REAL Rapport BCinout ;
INTEGER Maxreg ;
REAL B1 B2 ;
STRING      Refl ;
REAL        Xfacc Xfacp ;
REAL        Keff ;
INTEGER      iter := 1 ;
REAL        Precf := 1.E-6 ;
STRING      NamCpo1 NamCpo2 ;
EVALUATE NamCpo1 := "FULBU7114WGT" ;
EVALUATE NamCpo2 := "FULBU5135WGT" ;

!* Declaration des variables de thermohydraulique
!*****
INTEGER nz  nf  nc  := 120 16 4 ;
DOUBLE lz  df  tg  dc  dp  theta  rufc :=
    5.9436D0 1.12D-2 3.8D-4 1.304D-2 1.486D-2 1.570796320D0
3.0D-7 ;
DOUBLE mfin pout tinit tcini := 7.007879D3 1.1205836D7 5.3552D2
6.0D2 ;
DOUBLE evolg evolt evolp := 0.0D0 0.0D0 0.0D0 ;
DOUBLE dt := 1.0D-1 ;
INTEGER cv iterc ;
EVALUATE cv iterc := -1 1 ;
REAL prec := 1.E-4 ;

!* Parametres de controle de la simulation
!*****
EVALUATE Rapport := 0.8 ; ! Rapport B1 / B2
EVALUATE BCinout := 1.0 ; ! BC entrees = 1.0, BC sorties = 0.0
EVALUATE Maxreg := 27700 ; ! nombre de regions dans la geometrie

!* Liaison avec le fichier de composition
!*****

```

```

NFUEL := fnat ;

!* Appel de la procedure principale
!*****
G23D2 INDEX MACRO DEVICE PROCEDE RESEAU TAB
      Pburncal NFUEL :: <<Rapport>> <<BCinout>> <<Maxreg>> ;

EVALUATE Refl := "REFL-SIGF" ;
EVALUATE Xfacp := 1. ;
EVALUATE Xfacc := 2. ;

!* Boucle de couplage
!*****

REPEAT

!* Calcul des sections efficaces
!*****

MACRES := AFM: DBASE RESEAU ::
MAP
  INFOR SISSAOUI_ENDFB5_CANDU6_FEBRUARY_98
  DNAME 2 <<NamCpol>> <<NamCpol>>
  EDIT 0
  REFT 1 <<NamCpol>> 3 <<NamCpol>> ;

MACRO2 := INIMAC: INDEX MACRO MACRES ;

!* Calcul du flux pour la repartition convergee du burnup
!*****
FLUX MACOLD TRACK := Pcalflu MACRO2 DEVICE G23D2 ::
  <<iter>> <<Maxreg>> <<Refl>> <<Xfacc>> <<Xfacp>> <<Precf>> ;

RESEAU := FLXAXC: RESEAU FLUX TRACK INDEX MACRES :: AXIAL COMP ;

!* Calcul de la repartition de la puissance dans le reacteur
!*****
RESEAU := POWER: RESEAU MACRES :: EDIT 0 POWER 2061.4 FSTH 0.946 ;

!* Calcul de la lecture aux 102 detecteurs vanadium
!*****
FLUX := FLUNRM: FLUX MACOLD TRACK :: EDIT 0 POWER 2061.4E+6 REF 0
;

!* Initialisation de la geo. et des prop. thermohydrauliques
!*****
IF iterc 2 < THEN
  THER RESEAU := THERMO: RESEAU ::
    DEFCANDU
    DISCRET <<nz>> <<nf>> <<nc>>
    GEOM <<lz>> <<df>> <<tg>> <<dc>> <<dp>> <<theta>> <<rufc>>
    THCOND <<mfin>> <<pout>> <<tinit>> <<tcini>>
    EVOLC <<evolg>> <<evolt>> <<evolp>>
    TIMEPAR <<dt>>
  ENDDEF

```

```

      THSTATC SINGLP
    ;
  ELSE
    !* Calcul thermohydraulique
    !*****
    THER RESEAU := THERMO: THER RESEAU ::
    THSTATC SINGLP
    CONVER <<prec>> >>cv<<
    ;
  ENDIF ;
  MACRO2 MACRES FLUX TRACK := DELETE: MACRO2 MACRES FLUX TRACK ;
  MACOLD := DELETE: MACOLD ;
  EVALUATE iterc := iterc 1 + ;

UNTIL cv 0 > iterc 100 = + ;

thermodata := THER ;
neutrodata := RESEAU ;

QUIT .

```

Calcul transitoire de réacteur CANDU :

```

!*****
!* Test for THERMO module development in DONJON3 **
!* **
!* **
!* **
!*****
!* Calcul DONJON a l'equilibre du rechargement a deux (2) zones **
!* **
!* Determination du burnup moyen de sortie pour un coeur **
!* avec des grappes quelconques provenant de fnat **
!* **
!* COURS ENE6209 **
!* **
!* Options: **
!* 1- Rapport B1/B2 (B1 zone centrale, B2 zone peripherique) **
!* 2- Position des BC **
!*****
!* B. Dionne (03/04/2001) **
!* E. Varin (23/02/2004) **
!* R. Chambon (18/10/2007) **
!* M. Fassi Fehri (02/02/2008) **
!*****
PROCEDURE Pburncal Pcalflu ;

MODULE      END:   CRE:   FLXAXC: REFUEL: INIMAC:
            POWER: THERMO: AFM: DELETE: NEWMAC:
            IQS: TRIVAA: DMAC: GREP: ;

SEQ_ASCII fnat thermodata neutrodata macfme5 ;

LINKED_LIST NFUEL G23D2 INDEX MACRO DEVICE PROCEDE RESEAU TAB ;

```

```
LINKED_LIST  MACRO2 MACRES FLUX TRACK MACOLD THER DBASE SYSTEM ;
LINKED_LIST  FLUXP FLUXPI SYSTEMP DMACRO MACOLDP ;
```

```
INTEGER TPS ;
```

```
!* Base de donnees AFM
```

```
!*****
```

```
DBASE := macfme5 ;
```

```
!* Declaration des variables neutronique
```

```
!*****
```

```
REAL Rapport BCinout ;
```

```
INTEGER Maxreg ;
```

```
REAL B1 B2 ;
```

```
STRING      Refl ;
```

```
REAL        Xfacc Xfaccp ;
```

```
REAL        Keff ;
```

```
INTEGER      iter := 1 ;
```

```
REAL        Precf := 1.E-6 ;
```

```
STRING      NamCpo1 NamCpo2 ;
```

```
EVALUATE NamCpo1 := "FULBU7114WGT" ;
```

```
EVALUATE NamCpo2 := "FULBU5135WGT" ;
```

```
REAL Powi Powf ;
```

```
EVALUATE Powi := 2061.4 ;
```

```
!* Declaration des variables de thermohydraulique
```

```
!*****
```

```
INTEGER nz  nf  nc  := 60 10 4 ;
```

```
DOUBLE lz  df  tg  dc  dp  theta  rufc :=
5.9436D0 1.12D-2 3.8D-4 1.304D-2 1.486D-2 1.570796320D0
3.0D-7 ;
```

```
DOUBLE mfin pout tinit tcini := 7.007879D3 1.1205836D7 5.3552D2
6.0D2 ;
```

```
DOUBLE evolg evolt evolp := -1.2D1 0.0D0 0.0D0 ;
```

```
DOUBLE dt := 1.0D-1 ;
```

```
INTEGER cv iterc ;
```

```
EVALUATE cv iterc := -1 1 ;
```

```
REAL prec := 1.E-4 ;
```

```
!* Parametres de controle de la simulation
```

```
!*****
```

```
EVALUATE Rapport := 0.8 ; ! Rapport B1 / B2
```

```
EVALUATE BCinout := 1.0 ; ! BC entrees = 1.0, BC sorties = 0.0
```

```
EVALUATE Maxreg := 27700 ; ! nombre de regions dans la geometrie
```

```
!* Liaison avec le fichier de composition
```

```
!*****
```

```
NFUEL := fnat ;
```

```
!* Appel de la procedure principale
```

```
!*****
```

```
G23D2 INDEX MACRO DEVICE PROCEDE RESEAU TAB
```

```
Pburncal NFUEL :: <<Rapport>> <<BCinout>> <<Maxreg>> ;
```

```
EVALUATE Refl := "REFL-SIGF" ;
```

```

EVALUATE Xfacp := 1. ;
EVALUATE Xfacc := 2. ;

!* Boucle de couplage
!*****

REPEAT

    IF interc 1 > THEN
        MACRO2 MACRES FLUX TRACK := DELETE: MACRO2 MACRES FLUX TRACK ;
        MACOLD SYSTEM := DELETE: MACOLD SYSTEM ;
    ENDIF ;

!* Calcul des sections efficaces
!*****

MACRES := AFM: DBASE RESEAU ::
MAP
    INFOR SISSAOUI_ENDFB5_CANDU6_FEBRUARY_98
    DNAME 2 <<NamCpol1>> <<NamCpol1>>
    EDIT 0
    REFT 1 <<NamCpol1>> 3 <<NamCpol1>> ;

MACRO2 := INIMAC: INDEX MACRO MACRES ;

!* Calcul du flux pour la repartition convergee du burnup
!*****
FLUX MACOLD TRACK SYSTEM := Pcalflu MACRO2 DEVICE G23D2 ::
    <<iter>> <<Maxreg>> <<Refl>> <<Xfacc>> <<Xfacp>> <<Precf>> ;

RESEAU := FLXAXC: RESEAU FLUX TRACK INDEX MACRES :: AXIAL COMP ;

!* Calcul de la repartition de la puissance dans le reacteur
!*****
RESEAU := POWER: RESEAU MACRES :: EDIT 0 POWER <<Powi>> FSTH 0.946
;

!* Initialisation de la geo. et des prop. thermohydrauliques
!*****
IF interc 2 < THEN
    THER RESEAU := THERMO: RESEAU ::
        DEFCANDU
            DISCRET <<nz>> <<nf>> <<nc>>
            GEOM <<lz>> <<df>> <<tg>> <<dc>> <<dp>> <<theta>> <<rufc>>
            THCOND <<mfin>> <<pout>> <<tinit>> <<tcini>>
            EVOLC <<evolg>> <<evolt>> <<evolp>>
            TIMEPAR <<dt>>
        ENDDEF
        THSTATC SINGLP
    ;
ELSE
!* Calcul thermohydraulique
!*****
    THER RESEAU := THERMO: THER RESEAU ::

```



```

THSTATC SINGLP
CONVER <<prec>> >>cv<<
;
ENDIF ;

```

```

EVALUATE iterc := iterc 1 + ;

```

```

UNTIL cv 0 > iterc 100 = + ;

```

```

MACRES MACRO2 := DELETE: MACRES MACRO2 ;

```

```

!* CINETIQUE

```

```

!*****

```

```

EVALUATE iterc := 0 ;
EVALUATE cv := 0 ;
EVALUATE TPS := 1 ;

```

```

!* Boucle de temps

```

```

!*****

```

```

REPEAT

```

```

!* Boucle de convergence

```

```

!*****

```

```

REPEAT

```

```

EVALUATE iterc := iterc 1 + ;

```

```

ECHO " ++++++ " ;
ECHO " COUPLAGE CINETIQUE AU TEMPS: " TPS ;
ECHO " ITERATION: " iterc ;
ECHO " ++++++ " ;

```

```

IF TPS 1 = THEN

```

```

IF iterc 1 > THEN

```

```

MACOLDP MACRO2 DMACRO SYSTEMP FLUXP MACRES := DELETE:

```

```

MACOLDP MACRO2 DMACRO SYSTEMP FLUXP MACRES ;

```

```

ENDIF ;

```

```

ELSE

```

```

MACOLDP MACRO2 DMACRO SYSTEMP MACRES := DELETE:

```

```

MACOLDP MACRO2 DMACRO SYSTEMP MACRES ;

```

```

ENDIF ;

```

```

THER RESEAU := THERMO: THER RESEAU ::

```

```

THKINEC SINGLP

```

```

CONVER <<prec>> >>cv<<

```

```

;

```

```

MACRES := AFM: DBASE RESEAU ::

```

```

MAP
INFOR SISSAOUI_ENDFB5_CANDU6_FEBRUARY_98
DNAME 2 <<NamCpol>> <<NamCpol>>
EDIT 0
REFT 1 <<NamCpol>> 3 <<NamCpol>> ;

MACRO2 := INIMAC: INDEX MACRO MACRES ;

MACOLDP := NEWMAC: DEVICE MACRO2 G23D2 ::
  EDIT 1 <<Ref1>>
  TYPE CONTROLLED XFAC <<Xfacc>>
  TYPE PERT XFAC <<Xfacp>> ;

DMACRO := DMAC: MACOLDP MACOLD :: STEP 1 ;

SYSTEMP := TRIVAA: MACOLD TRACK DMACRO :: PERT UNIT OVEL ;

IF TPS 1 > THEN
  FLUXP := DELETE: FLUXP ;
  FLUXP := FLUXPI ;
ENDIF ;

IF TPS 1 = THEN
  FLUXP := IQS: FLUX SYSTEMP SYSTEM TRACK DMACRO MACOLD G23D2 ::
  EDIT 0
  DELN 6
  BETA 0.000250 0.001562 0.001505 0.002755 0.000830 0.000317
  DECR 0.0129 0.0311 0.134 0.331 1.26 3.21
  TETA VTET 1.0
  POWE <<Powi>>
  BEGIN PERT 1 STEP 0.1 REAC END
;
ELSE
  FLUXP := IQS: FLUXP FLUX SYSTEMP SYSTEM TRACK DMACRO MACOLD G23D2
::
  EDIT 0
  TETA VTET 1.0
  POWE <<Powi>>
  TMAC
  BEGIN PERT 1 STEP 0.1 REAC END
;
ENDIF ;

RESEAU := FLXAXC: RESEAU FLUXP TRACK INDEX :: FLUX-AV ;

GREP: FLUXP :: GETVAL POWER 1 >>Powf<< ;

RESEAU := POWER: RESEAU MACRES :: POWER <<Powf>> ;

UNTIL cv 0 > iterc 100 = + ;

EVALUATE Powi := Powf ;

EVALUATE cv := 0 ;

```

```
EVALUATE iterc := 0 ;
EVALUATE TPS := TPS 1 + ;
```

```
MACOLD SYSTEM := DELETE: MACOLD SYSTEM ;
MACOLD := MACOLDP ;
SYSTEM := TRIVAA: MACOLD TRACK :: EDIT 0 ;
```

```
IF TPS 2 > THEN
  FLUXPI := DELETE: FLUXPI ;
ENDIF ;
FLUXPI := FLUXP ;
```

```
THER RESEAU := THERMO: THER RESEAU ::
  THEVOL
;
```

```
UNTIL TPS 5 > ;
```

```
thermodata := THER ;
neutrodata := RESEAU ;
```

```
END: ;
```

```
QUIT .
```

```
*
* PROCEDURE: Pcalflu
* USAGE:      Static flux calculation for Gentilly2 reactor
* AUTHOR:     E.Varin (96/02/01)
* MODIFIED:   E.Varin (98/01/23)
* MODIFIED:   M. Fassi Fehri (08/02/02)
* CALL:
*
* FLUX MACOLD TRACK := Pcalflu MACRO2 DEVICE G23D2 ::
*                      i Maxreg Refl Xfacc Xfacp Precf ; When i =
1
*
* FLUX MACOLD      := Pcalflu FLUX TRACK MACRO2 DEVICE G23D2 ::
*                      i Maxreg Refl Xfacc Xfacp Precf ; When i <>
1
*
* NOTES:
*
* Maxreg = maximum number of matrial regions in geometry
* Refl= { REFL-SIGF | NOSIGF }
* Xfacc= facteur de correction des delta sigma
*        de type CONTROLLED
* Xfacp= facteur de correction des delta sigma
*        de type PERT
* Pour un autre type de controleurs, Xfacc s'applique.
*
```

```

*          Precf = Flux precision
*          i = 1  new flux calculation
*          > 1  update flux
*
PARAMETER  FLUX MACOLD TRACK SYSTEM MACRO2 DEVICE G23D2 ::
          :: LINKED_LIST  FLUX MACOLD TRACK SYSTEM MACRO2 DEVICE G23D2  ;
;
LINKED_LIST MACRO3 ;
MODULE      TRIVAT: TRIVAA: NEWMAC: FLUD: FREE: END: READ: ;
INTEGER     i Maxreg ;
REAL        Xfacc Xfacp Precf ;
CHARACTER   Refl ;
!
!
:: >>i<< >>Maxreg<< >>Refl<< >>Xfacc<< >>Xfacp<< >>Precf<<  ;
!
IF i 1 = THEN
TRACK := TRIVAT: G23D2 ::
    EDIT 1
    TITLE 'G2 3D '
    MAXR <<Maxreg>> MCFD 1 ;
ENDIF ;

MACRO3 := NEWMAC: DEVICE MACRO2 G23D2 ::
    EDIT 1 <<Refl>>
    TYPE CONTROLLED XFAC <<Xfacc>>
    TYPE PERT XFAC <<Xfacp>> ;

SYSTEM := TRIVAA: MACRO3 TRACK :: EDIT 1 ;

IF i 1 = THEN
    FLUX := FLUD: SYSTEM TRACK :: EDIT 1 PREC <<Precf>> ADJ ;
ELSE
    FLUX := FLUD: FLUX SYSTEM TRACK :: EDIT 1 PREC <<Precf>> ADJ ;
ENDIF ;
MACOLD := MACRO3 ;
END: ;

(*)
*)
(*) PROCEDURE:  Pinidet
*)
(*) USAGE:      Detector descriptions for test new implementation
*)
(*) AUTHOR:      E.Varin (96/02/01)
*)
(*) CALL:
*)
(*)
*)
(*)          DETEC := Pinidet  ;
*)
(*)
*)
(*) NOTES:
*)

```

```

(*)
*)
(*)      type = flag for actual or old platinum detector positions
*)
(*)      = 1 new positions, = 0 old positions
*)
(*)
*)
(*)      Detector positions were given by Hydro-Quebec
*)
*)
(*)
*)
(*)
*)
PARAMETER      DETEC CPODETECT ::
  :: LINKED_LIST DETEC CPODETECT ; ;
MODULE          END: INIDET: READ: ;
STRING MixName := 'BCAINTRE  1' ;
STRING XSName1 := 'TOTAL      ' ;
STRING XSName2 := 'NUSIGF     ' ;

REAL    Xm1      Xp1      Ym1      Yp1      Zm1      Zp1      :=
        200.      201.      200.      201.      200.      201.      ;

DETEC := INIDET: CPODETECT ::
  EDIT    5  NGRP 2
  TYPE    RAW_TOTAL
  INFO    1  2  CPO  CPODETECT <<MixName>> <<XSName1>>
  NAME    WHATEVER1
  POSITION <<Xm1>> <<Xp1>> <<Ym1>> <<Yp1>> <<Zm1>> <<Zp1>>
  RESP    0.0  0.0
  ENDN
  TYPE    RAW_NUSIGF
  INFO    1  2  CPO  CPODETECT <<MixName>> <<XSName2>>
  NAME    WHATEVER2
  POSITION <<Xm1>> <<Xp1>> <<Ym1>> <<Yp1>> <<Zm1>> <<Zp1>>
  RESP    0.0  0.0
  ENDN
  ;

END: ;

!*****
****
!*
**
!* PROCEDURE:  Pinires
**
!* USAGE:      Fuel map description for Gentilly2 reactor
**
!* AUTHOR:      E.Varin (96/02/12)
**
!* REVISED:     W.SHEN (98/11/19)
**

```

```

!*          B. Dionne (01/03/04) (pour cours ENE6209)
**
!* CALL:
**
!*
**
!* RESEAU := Pinires :: B1 B2 ;
**
!*
**
!* NOTES:
**
!*          B1 = average exit burnup for zone # 1
**
!*          B2 = average exit burnup for zone # 2
**
!*
**
!*****
****
PARAMETER RESEAU ::
          ::: LINKED_LIST RESEAU ;
          ;

MODULE END: INIRES: ;

!* Declaration des variables
!*****
REAL B1 B2 ;

!* Lecture des arguments de la procedure
!*****
:: >>B1<< >>B2<< ;

!* Creation et initialisation de RESEAU
!*****
RESEAU := INIRES: ::

NBUND 12 NCHAN 380 NZONE 2

::: GEOD: CAR3D 26 26 12
EDIT 0

X- VOID      X+ VOID
Y- VOID      Y+ VOID
Z- VOID      Z+ VOID

MIX
PLANE      1

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0

```

```

0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0

```

```

0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

```

```

PLANE 2 SAME 1
PLANE 3 SAME 1
PLANE 4 SAME 1
PLANE 5 SAME 1
PLANE 6 SAME 1
PLANE 7 SAME 1
PLANE 8 SAME 1
PLANE 9 SAME 1
PLANE 10 SAME 1
PLANE 11 SAME 1
PLANE 12 SAME 1

```

```

MESHX 3.150 45.050 68.525 97.100 125.675 154.250 182.825 211.400
239.975 268.550 297.125 325.700 354.275 382.850 411.425
440.000 468.575 497.150 525.725 554.300 582.875 611.450
640.025 668.600 697.175 720.650 762.550
MESHY 3.150 45.050 68.525 97.100 125.675 154.250 182.825 211.400
239.975 268.550 297.125 325.700 354.275 382.850 411.425
440.000 468.575 497.150 525.725 554.300 582.875 611.450
640.025 668.600 697.175 720.650 762.550
MESHZ 3.940 53.470 103.000 152.530 202.060 251.590 301.120
350.650
400.180 449.710 499.240 548.770 598.30 ;

```

```

NXNAME '-' '-' '1' '2' '3' '4' '5' '6' '7' '8' '9' '10' '11'
'12' '13' '14' '15' '16' '17' '18' '19' '20' '21' '22'
'-' '-'

```

```

NYNAME '-' '-' 'A' 'B' 'C' 'D' 'E' 'F' 'G' 'H' 'J' 'K' 'L'
'M' 'N' 'O' 'P' 'Q' 'R' 'S' 'T' 'U' 'V' 'W' '-' '-'

```

```

BURN-ZONE

```

```

2 2 2 2 2 2
2 2 2 2 2 2

```

```

      2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
    2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
  2 2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2
    2 2 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2
      2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2
        2 2 2 1 1 1 1 1 1 1 1 1 1 2 2 2
          2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2
            2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2
              2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2
                2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2

```

```

2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2
2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2
2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2
  2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2
    2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2
      2 2 2 2 2 1 1 1 1 1 1 1 1 2 2 2 2
        2 2 2 2 2 2 1 1 1 1 1 2 2 2 2
          2 2 2 2 2 2 2 1 1 1 1 2 2 2 2
            2 2 2 2 2 2 2 2 1 1 1 2 2 2
              2 2 2 2 2 2 2 2 2 1 1 2 2
                2 2 2 2 2 2 2 2 2 2 1 2
                  2 2 2 2 2 2 2 2 2 2 2 1
                    2 2 2 2 2 2 2 2 2 2 2 2

```

BURNUP-ZC <<B1>> <<B2>>

;

END: ;

QUIT .

```

!*****
****
!*
**
!* PROCEDURE:  Pmacfix
**
!* USAGE:      Macrolib construction Gentilly2 reactor from DRAGON
**
!*              properties for fixed material (devices) and reflector
**
!* AUTHOR:      E.Varin (96/02/12)
**
!* Modified:    W. SHEN (97/10/10)
**
!*              B. Dionne (01/03/04) (pour cours ENE6209)
**
!* CALL:
**
!*
**
!* MACRO := Pmacfix ;
**
!*
**
**

```



```

!* NOTES:
**
!*
**
!*      All the sequential ASCII files are exported COMPO files
**
!*      and must have the names and directory hierarchy as used
**
!*      in the calling to CRE: module
**
!*
**
!*      Check also mixture numbers with reference to geometry
**
!*      construction and device description
**
!*
**
!*****
****

PARAMETER MACRO ::
      ::: LINKED_LIST MACRO ;
      ;

MODULE CRE: END: ;

LINKED_LIST NREFL ZCR ADJ ;

SEQ_ASCII rnat zcunat adjnat ::
      FILE 'rnat' 'zcunat' 'adjnat' ;

!* Declaration des variables
!*****
!* Numero de melanges associes aux devices
INTEGER  m23M m23E m23F m12M m12E m12F m01M m01E m01F :=
      17  18  19  20  21  22  23  24  25 ;
INTEGER  mRai mRao mRbi mRci mRco mRdi
      mNRai mNRao mNRbi mNRci mNRco mNRdi :=
      5   6   7   8   9  10
      11  12  13  14  15  16 ;

!* Importation des fichiers de compositions
!*****
ADJ      := adjnat ;
ZCR      := zcunat ;
NREFL    := rnat ;

!* Generation des melanges (sections efficaces macroscopiques)
!*****
MACRO := CRE: NREFL ADJ ZCR ::
      EDIT 0 NMIX 25
      READ
      COMPO NREFL MIX 2 'MIXTMOD 1' ENDMIX
!
      COMPO ADJ MIX <<mRai>> 'BCAINT 1' ENDMIX

```

```

MIX <<mNRai>> 'BCAINTRE 1' ENDMIX
MIX <<mRao>> 'BCAOUT 1' ENDMIX
MIX <<mNRao>> 'BCAOUTRE 1' ENDMIX
MIX <<mRbi>> 'BCBINT 1' ENDMIX
MIX <<mNRbi>> 'BCBINTRE 1' ENDMIX
MIX <<mRci>> 'BCCINT 1' ENDMIX
MIX <<mNRci>> 'BCCINTRE 1' ENDMIX
MIX <<mRco>> 'BCCOUT 1' ENDMIX
MIX <<mNRco>> 'BCCOUTRE 1' ENDMIX
MIX <<mRdi>> 'BCDINT 1' ENDMIX
MIX <<mNRdi>> 'BCDINTRE 1' ENDMIX
!
      COMPO ZCR MIX <<m23M>> 'NOBL32 1' ENDMIX
      MIX <<m23E>> 'BLIQE32 1' ENDMIX
      MIX <<m23F>> 'BLIQF32 1' ENDMIX
      MIX <<m12M>> 'NOBL21 1' ENDMIX
      MIX <<m12E>> 'BLIQE21 1' ENDMIX
      MIX <<m12F>> 'BLIQF21 1' ENDMIX
      MIX <<m01M>> 'NOBL10 1' ENDMIX
      MIX <<m01E>> 'BLIQE10 1' ENDMIX
      MIX <<m01F>> 'BLIQF10 1' ENDMIX
;

END: ;

QUIT .

!*****
****
!*
**
!* PROCEDURE: PcritAx
**
!* USAGE:      Iteration for axial flux shape convergence
**
!*              for Gentilly2
**
!* AUTHOR:      E. Varin (96/02/12)
**
!* Modified:    W. SHEN (97/10/10)
**
!*              B. Dionne (01/03/04) (pour cours ENE6209)
**
!* CALL:
**
!*
**
!* RESEAU := PcritB RESEAU TAB INDEX MACRO DEVICE G23D2 ::
**
!*      Maxreg Precf Pow Keff Eps1 ;
**
!*
**
!*
**
!* NOTES:
**

```

```

!*
**
!*      Maxreg = maximum number of matrial regions in geometry
**
!*      Precf  = Static flux calculation precision
**
!*      Pow    = total fission power
**
!*      B1,B2  = burnup in each zone
**
!*      Keff,Eps1  = output
**
!*
**
!*****
****
PARAMETER RESEAU TAB INDEX MACRO DEVICE G23D2 ::
      ::: LINKED_LIST RESEAU TAB INDEX MACRO DEVICE G23D2 ;
      ;

MODULE DELETE: INIRES: END: REFUEL: ;

PROCEDURE Pflax ;

!* Declaration et initialisation des variables
!*****
INTEGER Maxreg ;
INTEGER Iter Edit := 0 1 ;

REAL Precf B1 B2 Eps1 Keff Pow ;

!* Lecture des arguments de la procedure
!*****
:: >>Precf<< >>B1<< >>B2<< >>Maxreg<< >>Pow<< ;

!* Boucle pour converger sur la forme axiale du flux
!*****
REPEAT

      EVALUATE Iter := Iter 1 + ;

      RESEAU := INIRES: RESEAU :: BURNUP-ZC <<B1>> <<B2>> ;

      RESEAU := REFUEL: RESEAU TAB :: TAVGC NS 8 8
                        NTYP 1 TYPE 1 'NATURAL' ;

      RESEAU := Pflax RESEAU TAB INDEX MACRO DEVICE G23D2 ::
        <<Maxreg>> <<Precf>> <<Pow>> <<Edit>> >>Keff<< >>Eps1<< ;

      UNTIL Eps1 0.001 < Iter 10 = + ;

!* Impression de la carte des puissances de canal
!*****
EVALUATE Edit := 2 ;

```

```
RESEAU := Pflax RESEAU TAB INDEX MACRO DEVICE G23D2 ::
<<Maxreg>> <<Precf>> <<Pow>> <<Edit>> >>Keff<< >>Eps1<< ;
```

```
IF Eps1 0.001 < THEN
  ECHO "FORME AXIALE DU FLUX CONVERGEE" ;
  ECHO "-----" ;
  ECHO "EXIT BURNUP          = " B1 B2 ;
  ECHO "KEFF                  = " Keff ;
  ECHO "ERREUR FORME FLUX     = " Eps1 ;
  ECHO "NBRE ITERATIONS INTERNES = " Iter ;
  ECHO "-----" ;
ELSE
  ECHO "ITERATION MAXIMALE ATTEINTE" ;
  ECHO "FORME AXIALE DU FLUX NON CONVERGEE" ;
  ECHO "-----" ;
  ECHO "EXIT BURNUP          = " B1 B2 ;
  ECHO "KEFF                  = " Keff ;
  ECHO "ERREUR FORME FLUX     = " Eps1 ;
  ECHO "NBRE ITERATIONS INTERNES = " Iter ;
  ECHO "-----" ;
ENDIF ;
```

```
!* Ecriture des variables de sorties de la procedure
!*****
:: <<Keff>> <<Eps1>> ;
```

```
END: ;
```

```
QUIT .
```

```
!*
**
!* PROCEDURE:  PdevT
**
!* USAGE:      Device descriptions for Gentilly2 reactor
**
!* AUTHOR:      E.Varin (96/02/01)
**
!* CALL:
**
!*
**
!* DEVICE INDEX := PdevT INDEX G23D2 :: fmean
**
!*   m23M m12M m01M  mNai mNao mNbi mNci mNco mNdi mNobs ;
**
!*
**
!* NOTES:
**
!*
**
!*           Mixture number for device must be set and changed HERE
**
!*           fmean = LZC water level
**
```

```

!*
**
!*      Other integer values are mixture numbers of no
**
!*      device
**
!*
**
!*
**

PARAMETER  DEVICE INDEX G23D2 ::
           ::: LINKED_LIST DEVICE INDEX G23D2 ;
           ;

MODULE      END: INIDEV: ;

!*Declaration et initialisation des variables
!*****
!* MEAN FILLING FOR ZCRS **
REAL      fmean ;
INTEGER    m23M m12M m01M mNai mNao mNbi mNci mNco mNdi mNobs nlp ;

!* FILLINGS FOR ALL ZCRS: ***CHANGE AT YOUR CONVENIENCE ***      **
REAL f01      f02      f03      f04      f05      f06      f07
      f08      f09      f10      f11      f12      f13      f14      ;

!* DEFINE LATTICE PITCH **
REAL lp := 28.5750 ;

!* DEFINE X- X+ ADJUSTER POSITIONS **
REAL Am1      Ap1      Am2      Ap2      Am3      Ap3
      Am4      Ap4      Am5      Ap5      Am6      Ap6
      Am7      Ap7      :=
      197.1125 225.6875 254.2625 282.8375 311.4125 339.9875
      368.5625 397.1375 425.7125 454.2875 482.8625 511.4375
      540.0125 568.5875 ;

!* DEFINE X- X+ LZC POSITIONS **
REAL LmU12      LpU12      LmU34      LpU34      LmU56      LpU56      :=
      197.1125 225.6875 368.5625 397.1375 540.0125 568.5875 ;

!* DEFINE MIX NUMBERS FOR ZCR BY TYPE: 32 21 10 FOR FULL/EMPTY **
INTEGER m23E m23F m12E m12F m01E m01F :=
      18 19 21 22 24 25 ;

!* DEFINE MIX NUMBERS FOR BC BY TYPE **
INTEGER mRai mRao mRbi mRci mRco mRdi :=
      5 6 7 8 9 10 ;
INTEGER mRbs :=
      26 ;

!* Lecture des arguments de la procedure
!*****
:: >>fmean<< >>m23M<< >>m12M<< >>m01M<< >>mNai<< >>mNao<< >>mNbi<<
>>mNci<< >>mNco<< >>mNdi<< >>mNobs<< ;

```

```

!* FILLINGS FOR ALL ZCRS: ***CHANGE AT YOUR CONVENIENCE ***      **
EVALUATE f01  f02      f03      f04      f05      f06      f07
          f08      f09      f10      f11      f12      f13      f14      :=
          fmean    fmean    fmean    fmean    fmean    fmean    fmean
          fmean    fmean    fmean    fmean    fmean    fmean    fmean    ;

!* Y REFERENCE LENGTHS      FOR ZCR **
REAL  yzcL1      yzcL2      yzcL3      yzcL4      yzcL5      :=
      lp 8.504 * lp 8. * lp 7.489 * lp 7. * lp 7. * ;

!* Y+ REFERENCE COORDINATES FOR ZCR **
REAL  yzcP1      yzcP2      yzcP3      yzcP4      yzcP5      :=
      354.2750 582.8750 239.9750 440.0000 640.0250 ;

!* Y LENGHTS      FOR ALL ZCRS IN G2 **
REAL  L01      L02      L03      L04      L05      L06      L07
      L08      L09      L10      L11      L12      L13      L14 :=
      yzcL1      yzcL2      yzcL3      yzcL4      yzcL5      yzcL1      yzcL2
      yzcL1      yzcL2      yzcL3      yzcL4      yzcL5      yzcL1      yzcL2    ;

!* Y+ COORDINATES FOR ALL ZCRS IN G2 **
REAL  yP01      yP02      yP03      yP04      yP05      yP06      yP07
      yP08      yP09      yP10      yP11      yP12      yP13      yP14 :=
      yzcP1      yzcP2      yzcP3      yzcP4      yzcP5      yzcP1      yzcP2
      yzcP1      yzcP2      yzcP3      yzcP4      yzcP5      yzcP1      yzcP2    ;

!* Y- COORDINATES FOR ALL ZCRS IN G2 **
REAL  yM01 := yP01 L01 - ;
REAL  yM02 := yP02 L02 - ;
REAL  yM03 := yP03 L03 - ;
REAL  yM04 := yP04 L04 - ;
REAL  yM05 := yP05 L05 - ;
REAL  yM06 := yP06 L06 - ;
REAL  yM07 := yP07 L07 - ;
REAL  yM08 := yP08 L08 - ;
REAL  yM09 := yP09 L09 - ;
REAL  yM10 := yP10 L10 - ;
REAL  yM11 := yP11 L11 - ;
REAL  yM12 := yP12 L12 - ;
REAL  yM13 := yP13 L13 - ;
REAL  yM14 := yP14 L14 - ;

!* YO COORDINATES FOR ALL ZCRS IN G2: **
!* TO SEPARATE FILL AND EMPTY ZONES **

REAL  yO01 := yP01 f01 L01 * - ;
REAL  yO02 := yP02 f02 L02 * - ;
REAL  yO03 := yP03 f03 L03 * - ;
REAL  yO04 := yP04 f04 L04 * - ;
REAL  yO05 := yP05 f05 L05 * - ;
REAL  yO06 := yP06 f06 L06 * - ;
REAL  yO07 := yP07 f07 L07 * - ;
REAL  yO08 := yP08 f08 L08 * - ;
REAL  yO09 := yP09 f09 L09 * - ;
REAL  yO10 := yP10 f10 L10 * - ;

```

```

REAL yO11 := yP11 f11 L11 * - ;
REAL yO12 := yP12 f12 L12 * - ;
REAL yO13 := yP13 f13 L13 * - ;
REAL yO14 := yP14 f14 L14 * - ;

```

```
!* Initialisation des devices
```

```
!*****
```

```
DEVICE INDEX := INIDEV: INDEX G23D2 ::
```

```
NDEV 35 DELTA NDESC 67
```

```

DEVICE !* DATA #1**
  CONTROLLED ADJUSTER BC#1          !*TYPE FAMILY SEQUENCE
NUMBER**
  !*POSITON MAX**
    <<Am1>> <<Ap1>> 325.700 440.000 196.3600 245.8900
    Y 0.0 0.0 0.0          !*DIRECTION SPEED PARAMETERS
  **
    BC#1B          !*DESCRIPTION
#1**
  !*POSITION**
    <<Am1>> <<Ap1>> 325.700 440.000 196.3600 245.8900
    <<mRdi>> <<mNdi>>          !*MIX
  **
  ENDDEV
  DEVICE !*DATA #2**
    CONTROLLED ADJUSTER BC#2          !*TYPE FAMILY SEQUENCE
NUMBER**
  !*POSITON MAX**
    <<Am2>> <<Ap2>> 211.400 554.300 196.3600 245.8900
    Y 0.0 0.0 0.0          !*DIRECTION SPEED PARAMETERS
  **
    BC#2C          !*DESCRIPTION
#1**
  !*POSITION**
    <<Am2>> <<Ap2>> 211.400 297.125 196.3600 245.8900
    <<mRco>> <<mNco>>          !*MIX
  **
    BC#2B          !*DESCRIPTION
#2**
  !*POSITION**
    <<Am2>> <<Ap2>> 297.125 468.575 196.3600 245.8900
    <<mRci>> <<mNci>>          !*MIX
  **
    BC#2A          !*DESCRIPTION
#3**
  !*POSITION**
    <<Am2>> <<Ap2>> 468.575 554.300 196.3600 245.8900
    <<mRco>> <<mNco>>          !*MIX
  **
  ENDDEV
  DEVICE !*DATA #3**
    CONTROLLED ADJUSTER BC#3          !*TYPE FAMILY SEQUENCE
NUMBER**
  !*POSITON MAX**
    <<Am3>> <<Ap3>> 211.400 554.300 196.3600 245.8900
    Y 0.0 0.0 0.0          !*DIRECTION SPEED PARAMETERS **

```

```

BC#3C-B-A                                !*DESCRIPTION
#1**
!*POSITION**
  <<Am3>> <<Ap3>> 211.400 554.300 196.3600 245.8900
    <<mRbi>> <<mNbi>>                                !*MIX
**
ENDDEV
DEVICE !*DATA #4**
  CONTROLLED ADJUSTER BC#4                !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am4>> <<Ap4>> 211.400 554.300 196.3600 245.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#4C                                !*DESCRIPTION #1**
!*POSITION**
  <<Am4>> <<Ap4>> 211.400 297.125 196.3600 245.8900
    <<mRao>> <<mNao>>                                !*MIX **
    BC#4B                                !*DESCRIPTION #2**
!*POSITION**
  <<Am4>> <<Ap4>> 297.125 468.575 196.3600 245.8900
    <<mRai>> <<mNai>>                                !*MIX **
    BC#4A                                !*DESCRIPTION #3**
!*POSITION**
  <<Am4>> <<Ap4>> 468.575 554.300 196.3600 245.8900
    <<mRao>> <<mNao>>                                !*MIX **
ENDDEV
DEVICE !*DATA #5**
  CONTROLLED ADJUSTER BC#5                !*TYPE FAMILY SEQUENCE
NUMBER**
!*POSITON MAX**
  <<Am5>> <<Ap5>> 211.400 554.300 196.3600 245.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#5C-B-A                                !*DESCRIPTION #3**
!*POSITION**
  <<Am5>> <<Ap5>> 211.400 554.300 196.3600 245.8900
    <<mRbi>> <<mNbi>>                                !*MIX **
ENDDEV
DEVICE !*DATA #6**
  CONTROLLED ADJUSTER BC#6                !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am6>> <<Ap6>> 211.400 554.300 196.3600 245.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#6C                                !*DESCRIPTION #1**
!*POSITION**
  <<Am6>> <<Ap6>> 211.400 297.125 196.3600 245.8900
    <<mRco>> <<mNco>>                                !*MIX **
    BC#6B                                !*DESCRIPTION #1**
!*POSITION**
  <<Am6>> <<Ap6>> 297.125 468.575 196.3600 245.8900
    <<mRci>> <<mNci>>                                !*MIX **
    BC#6A                                !*DESCRIPTION #1**
!*POSITION**
  <<Am6>> <<Ap6>> 468.575 554.300 196.3600 245.8900
    <<mRco>> <<mNco>>                                !*MIX **
ENDDEV
DEVICE !*DATA #7**
  CONTROLLED ADJUSTER BC#7                !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am7>> <<Ap7>> 325.700 440.000 196.3600 245.8900

```



```

Y      0.0 0.0 0.0      !*DIRECTION SPEED PARAMETERS **
BC#7B      !*DESCRIPTION #1**
!*POSITION**
  <<Am7>> <<Ap7>> 325.700 440.000 196.3600 245.8900
  <<mRdi>> <<mNdi>>      !*MIX      **
ENDDEV
DEVICE !*DATA #8**
  CONTROLLED ADJUSTER BC#8      !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am1>> <<Ap1>> 325.700 440.000 276.3600 325.8900
  Y      0.0 0.0 0.0      !*DIRECTION SPEED PARAMETERS **
  BC#8B      !*DESCRIPTION #1**
!*POSITION**
  <<Am1>> <<Ap1>> 325.700 440.000 276.3600 325.8900
  <<mRdi>> <<mNdi>>      !*MIX      **
ENDDEV
DEVICE !*DATA #9**
  CONTROLLED ADJUSTER BC#9      !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am2>> <<Ap2>> 211.400 554.300 276.3600 325.8900
  Y      0.0 0.0 0.0      !*DIRECTION SPEED PARAMETERS **
  BC#9C      !*DESCRIPTION #1**
!*POSITION**
  <<Am2>> <<Ap2>> 211.400 297.125 276.3600 325.8900
  <<mRco>> <<mNco>>      !*MIX      **
  BC#9B      !*DESCRIPTION #2**
!*POSITION**
  <<Am2>> <<Ap2>> 297.125 468.575 276.3600 325.8900
  <<mRci>> <<mNci>>      !*MIX      **
  BC#9A      !*DESCRIPTION #3**
!*POSITION**
  <<Am2>> <<Ap2>> 468.575 554.300 276.3600 325.8900
  <<mRco>> <<mNco>>      !*MIX      **
ENDDEV
DEVICE !*DATA #10**
  CONTROLLED ADJUSTER BC#10      !*TYPE FAMILY SEQUENCE NUMBER**
  <<Am3>> <<Ap3>> 211.400 554.300 276.3600 325.8900
  Y      0.0 0.0 0.0      !*DIRECTION SPEED PARAMETERS **
  BC#10C-B-A      !*DESCRIPTION #1**
!*POSITION**
  <<Am3>> <<Ap3>> 211.400 554.300 276.3600 325.8900
  <<mRbi>> <<mNbi>>      !*MIX      **
ENDDEV
DEVICE !*DATA #11**
  CONTROLLED ADJUSTER BC#11      !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am4>> <<Ap4>> 211.400 554.300 276.3600 325.8900
  Y      0.0 0.0 0.0      !*DIRECTION SPEED PARAMETERS **
  BC#11C      !*DESCRIPTION #1**
!*POSITION**
  <<Am4>> <<Ap4>> 211.400 297.125 276.3600 325.8900
  <<mRao>> <<mNao>>      !*MIX      **
  BC#11B      !*DESCRIPTION #2**
!*POSITION**
  <<Am4>> <<Ap4>> 297.125 468.575 276.3600 325.8900
  <<mRai>> <<mNai>>      !*MIX      **
  BC#11A      !*DESCRIPTION #3**
!*POSITION**

```

```

    <<Am4>> <<Ap4>> 468.575 554.300 276.3600 325.8900
    <<mRao>> <<mNao>> !*MIX **
ENDDEV
DEVICE !*DATA #12**
    CONTROLLED ADJUSTER BC#12 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am5>> <<Ap5>> 211.400 554.300 276.3600 325.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#12C-B-A !*DESCRIPTION #1**
!*POSITION**
    <<Am5>> <<Ap5>> 211.400 554.300 276.3600 325.8900
    <<mRbi>> <<mNbi>> !*MIX **
ENDDEV
DEVICE !*DATA #13**
    CONTROLLED ADJUSTER BC#13 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am6>> <<Ap6>> 211.400 554.300 276.3600 325.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#13C !*DESCRIPTION #1**
!*POSITION**
    <<Am6>> <<Ap6>> 211.400 297.125 276.3600 325.8900
    <<mRco>> <<mNco>> !*MIX **
    BC#13B !*DESCRIPTION #1**
!*POSITION**
    <<Am6>> <<Ap6>> 297.125 468.575 276.3600 325.8900
    <<mRci>> <<mNci>> !*MIX **
    BC#13A !*DESCRIPTION #1**
!*POSITION**
    <<Am6>> <<Ap6>> 468.575 554.300 276.3600 325.8900
    <<mRco>> <<mNco>> !*MIX **
ENDDEV
DEVICE !*DATA #14**
    CONTROLLED ADJUSTER BC#14 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am7>> <<Ap7>> 325.700 440.000 276.3600 325.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#14B !*DESCRIPTION #1**
!*POSITION**
    <<Am7>> <<Ap7>> 325.700 440.000 276.3600 325.8900
    <<mRdi>> <<mNdi>> !*MIX **
ENDDEV
DEVICE !*DATA #15**
    CONTROLLED ADJUSTER BC#15 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am1>> <<Ap1>> 325.700 440.000 356.3600 405.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#15B !*DESCRIPTION #1**
!*POSITION**
    <<Am1>> <<Ap1>> 325.700 440.000 356.3600 405.8900
    <<mRdi>> <<mNdi>> !*MIX **
ENDDEV
DEVICE !*DATA #16**
    CONTROLLED ADJUSTER BC#16 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am2>> <<Ap2>> 211.400 554.300 356.3600 405.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#16C !*DESCRIPTION #1**
!*POSITION**

```

```

    <<Am2>> <<Ap2>> 211.400 297.125 356.3600 405.8900
    <<mRco>> <<mNco>> !*MIX **
    BC#16B !*DESCRIPTION #2**
!*POSITION**
    <<Am2>> <<Ap2>> 297.125 468.575 356.3600 405.8900
    <<mRci>> <<mNci>> !*MIX **
    BC#16A !*DESCRIPTION #3**
!*POSITION**
    <<Am2>> <<Ap2>> 468.575 554.300 356.3600 405.8900
    <<mRco>> <<mNco>> !*MIX **
ENDDEV
DEVICE !*DATA #17**
CONTROLLED ADJUSTER BC#17 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am3>> <<Ap3>> 211.400 554.300 356.3600 405.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#17C-B-A !*DESCRIPTION #1**
!*POSITION**
    <<Am3>> <<Ap3>> 211.400 554.300 356.3600 405.8900
    <<mRbi>> <<mNbi>> !*MIX **
ENDDEV
DEVICE !*DATA #18**
CONTROLLED ADJUSTER BC#18 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am4>> <<Ap4>> 211.400 554.300 356.3600 405.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#18C !*DESCRIPTION #1**
!*POSITION**
    <<Am4>> <<Ap4>> 211.400 297.125 356.3600 405.8900
    <<mRao>> <<mNao>> !*MIX **
    BC#18B !*DESCRIPTION #2**
!*POSITION**
    <<Am4>> <<Ap4>> 297.125 468.575 356.3600 405.8900
    <<mRai>> <<mNai>> !*MIX **
    BC#18A !*DESCRIPTION #3**
!*POSITION**
    <<Am4>> <<Ap4>> 468.575 554.300 356.3600 405.8900
    <<mRao>> <<mNao>> !*MIX **
ENDDEV
DEVICE !*DATA #19**
CONTROLLED ADJUSTER BC#19 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am5>> <<Ap5>> 211.400 554.300 356.3600 405.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#19C-B-A !*DESCRIPTION #1**
!*POSITION**
    <<Am5>> <<Ap5>> 211.400 554.300 356.3600 405.8900
    <<mRbi>> <<mNbi>> !*MIX **
ENDDEV
DEVICE !*DATA #20**
CONTROLLED ADJUSTER BC#20 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<Am6>> <<Ap6>> 211.400 554.300 356.3600 405.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    BC#20C !*DESCRIPTION #1**
!*POSITION**
    <<Am6>> <<Ap6>> 211.400 297.125 356.3600 405.8900
    <<mRco>> <<mNco>> !*MIX **

```

```

BC#20B                                     !*DESCRIPTION #2**
!*POSITION**
  <<Am6>> <<Ap6>> 297.125 468.575 356.3600 405.8900
  <<mRci>> <<mNci>>                                     !*MIX **
BC#20A                                     !*DESCRIPTION #3**
!*POSITION**
  <<Am6>> <<Ap6>> 468.575 554.300 356.3600 405.8900
  <<mRco>> <<mNco>>                                     !*MIX **
ENDDEV
DEVICE !*DATA #21**
  CONTROLLED ADJUSTER BC#21 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<Am7>> <<Ap7>> 325.700 440.000 356.3600 405.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  BC#21B                                     !*DESCRIPTION #1**
!*POSITION**
  <<Am7>> <<Ap7>> 325.700 440.000 356.3600 405.8900
  <<mRdi>> <<mNdi>>                                     !*MIX **
ENDDEV

!* LIQUID ZONE CONTROLLERS AT VARIOUS LEVELS **
!* ZCU-1 **

DEVICE !*DATA #22**
  CONTROLLED LZC CONTL#1 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU12>> <<LpU12>> <<yM01>> <<yP01>> 146.3600 195.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#1                                     !*DESCRIPTION #1**
!*POSITION**
  <<LmU12>> <<LpU12>> <<yM01>> <<yO01>> 146.3600 195.8900
  <<m12E>> <<m12M>>                                     !*MIX **
  FULL#1                                     !*DESCRIPTION #2**
!*POSITION**
  <<LmU12>> <<LpU12>> <<yO01>> <<yP01>> 146.3600 195.8900
  <<m12F>> <<m12M>>                                     !*MIX **
ENDDEV
DEVICE !*DATA #23**
  CONTROLLED LZC CONTL#2 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU12>> <<LpU12>> <<yM02>> <<yP02>> 146.3600 195.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#2                                     !*DESCRIPTION #1**
!*POSITION**
  <<LmU12>> <<LpU12>> <<yM02>> <<yO02>> 146.3600 195.8900
  <<m01E>> <<m01M>>                                     !*MIX **
  FULL#2                                     !*DESCRIPTION #2**
!*POSITION**
  <<LmU12>> <<LpU12>> <<yO02>> <<yP02>> 146.3600 195.8900
  <<m01F>> <<m01M>>                                     !*MIX **
ENDDEV
DEVICE !*DATA #24**
  CONTROLLED LZC CONTL#3 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU34>> <<LpU34>> <<yM03>> <<yP03>> 146.3600 195.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **

```

```

EMPTY#3
!*POSITION**
  <<LmU34>> <<LpU34>> <<yM03>> <<yO03>> 146.3600 195.8900
  <<m23E>> <<m23M>> !*MIX **
  FULL#3 !*DESCRIPTION #2**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yO03>> <<yP03>> 146.3600 195.8900
  <<m23F>> <<m23M>> !*MIX **
ENDDEV
DEVICE !*DATA #25**
  CONTROLLED LZC CONTL#4 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU34>> <<LpU34>> <<yM04>> <<yP04>> 146.3600 195.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#4 !*DESCRIPTION #1**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yM04>> <<yO04>> 146.3600 195.8900
  <<m12E>> <<m12M>> !*MIX **
  FULL#4 !*DESCRIPTION #2**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yO04>> <<yP04>> 146.3600 195.8900
  <<m12F>> <<m12M>> !*MIX **
ENDDEV
DEVICE !*DATA #26**
  CONTROLLED LZC CONTL#5 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU34>> <<LpU34>> <<yM05>> <<yP05>> 152.535 202.065
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#5 !*DESCRIPTION #1**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yM05>> <<yO05>> 152.535 202.065
  <<m01E>> <<m01M>> !*MIX **
  FULL#5 !*DESCRIPTION #2**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yO05>> <<yP05>> 152.535 202.065
  <<m01F>> <<m01M>> !*MIX **
ENDDEV
DEVICE !*DATA #27**
  CONTROLLED LZC CONTL#6 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU56>> <<LpU56>> <<yM06>> <<yP06>> 146.3600 195.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#6 !*DESCRIPTION #1**
!*POSITION**
  <<LmU56>> <<LpU56>> <<yM06>> <<yO06>> 146.3600 195.8900
  <<m12E>> <<m12M>> !*MIX **
  FULL#6 !*DESCRIPTION #2**
!*POSITION**
  <<LmU56>> <<LpU56>> <<yO06>> <<yP06>> 146.3600 195.8900
  <<m12F>> <<m12M>> !*MIX **
ENDDEV
DEVICE !*DATA #28**
  CONTROLLED LZC CONTL#7 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU56>> <<LpU56>> <<yM07>> <<yP07>> 146.3600 195.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#7 !*DESCRIPTION #1**
!*POSITION**

```

```

    <<LmU56>> <<LpU56>> <<yM07>> <<yO07>> 146.3600 195.8900
    <<m01E>> <<m01M>> !*MIX **
    FULL#7 !*DESCRIPTION #2**
!*POSITION**
    <<LmU56>> <<LpU56>> <<yO07>> <<yP07>> 146.3600 195.8900
    <<m01F>> <<m01M>> !*MIX **
ENDDEV
DEVICE !*DATA #29**
    CONTROLLED LZC CONTL#8 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<LmU12>> <<LpU12>> <<yM08>> <<yP08>> 406.3600 455.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    EMPTY#8 !*DESCRIPTION #1**
!*POSITION**
    <<LmU12>> <<LpU12>> <<yM08>> <<yO08>> 406.3600 455.8900
    <<m12E>> <<m12M>> !*MIX **
    FULL#8 !*DESCRIPTION #2**
!*POSITION**
    <<LmU12>> <<LpU12>> <<yO08>> <<yP08>> 406.3600 455.8900
    <<m12F>> <<m12M>> !*MIX **
ENDDEV
DEVICE !*DATA #30**
    CONTROLLED LZC CONTL#9 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<LmU12>> <<LpU12>> <<yM09>> <<yP09>> 406.3600 455.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    EMPTY#9 !*DESCRIPTION #1**
!*POSITION**
    <<LmU12>> <<LpU12>> <<yM09>> <<yO09>> 406.3600 455.8900
    <<m01E>> <<m01M>> !*MIX **
    FULL#9 !*DESCRIPTION #2**
!*POSITION**
    <<LmU12>> <<LpU12>> <<yO09>> <<yP09>> 406.3600 455.8900
    <<m01F>> <<m01M>> !*MIX **
ENDDEV
DEVICE !*DATA #31**
    CONTROLLED LZC CONTL#10 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<LmU34>> <<LpU34>> <<yM10>> <<yP10>> 406.3600 455.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    EMPTY#10 !*DESCRIPTION #1**
!*POSITION**
    <<LmU34>> <<LpU34>> <<yM10>> <<yO10>> 406.3600 455.8900
    <<m23E>> <<m23M>> !*MIX **
    FULL#10 !*DESCRIPTION #2**
!*POSITION**
    <<LmU34>> <<LpU34>> <<yO10>> <<yP10>> 406.3600 455.8900
    <<m23F>> <<m23M>> !*MIX **
ENDDEV
DEVICE !*DATA #32**
    CONTROLLED LZC CONTL#11 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
    <<LmU34>> <<LpU34>> <<yM11>> <<yP11>> 406.3600 455.8900
    Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
    EMPTY#11 !*DESCRIPTION #1**
!*POSITION**
    <<LmU34>> <<LpU34>> <<yM11>> <<yO11>> 406.3600 455.8900
    <<m12E>> <<m12M>> !*MIX **

```

```

FULL#11                                !*DESCRIPTION #2**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yO11>> <<yP11>> 406.3600 455.8900
  <<m12F>> <<m12M>> !*MIX **
ENDDEV
DEVICE !*DATA #33**
  CONTROLLED LZC CONTL#12 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU34>> <<LpU34>> <<yM12>> <<yP12>> 406.3600 455.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#12 !*DESCRIPTION #1**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yM12>> <<yO12>> 406.3600 455.8900
  <<m01E>> <<m01M>> !*MIX **
  FULL#12 !*DESCRIPTION #2**
!*POSITION**
  <<LmU34>> <<LpU34>> <<yO12>> <<yP12>> 406.3600 455.8900
  <<m01F>> <<m01M>> !*MIX **
ENDDEV
DEVICE !*DATA #34**
  CONTROLLED LZC CONTL#13 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU56>> <<LpU56>> <<yM13>> <<yP13>> 406.3600 455.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#13 !*DESCRIPTION #1**
!*POSITION**
  <<LmU56>> <<LpU56>> <<yM13>> <<yO13>> 406.3600 455.8900
  <<m12E>> <<m12M>> !*MIX **
  FULL#13 !*DESCRIPTION #2**
!*POSITION**
  <<LmU56>> <<LpU56>> <<yO13>> <<yP13>> 406.3600 455.8900
  <<m12F>> <<m12M>> !*MIX **
ENDDEV
DEVICE !*DATA #35**
  CONTROLLED LZC CONTL#14 !*TYPE FAMILY SEQUENCE NUMBER**
!*POSITON MAX**
  <<LmU56>> <<LpU56>> <<yM14>> <<yP14>> 406.3600 455.8900
  Y 0.0 0.0 0.0 !*DIRECTION SPEED PARAMETERS **
  EMPTY#14 !*DESCRIPTION #1**
!*POSITION**
  <<LmU56>> <<LpU56>> <<yM14>> <<yO14>> 406.3600 455.8900
  <<m01E>> <<m01M>> !*MIX **
  FULL#14 !*DESCRIPTION #2**
!*POSITION**
  <<LmU56>> <<LpU56>> <<yO14>> <<yP14>> 406.3600 455.8900
  <<m01F>> <<m01M>> !*MIX **
ENDDEV
;

```

END: ;

QUIT .

```

!*****
****
!*
**
!* PROCEDURE:  Pflax
**
!* USAGE:      Flux calculation for Gentilly2 for axial flux shape
**
!*              convergence or criticality on exit burnup (TAVG COMP)
**
!* AUTHOR:     E.Varin (96/02/12)
**
!*              B. Dionne (01/03/04) (pour cours ENE6209)
**
!* CALL:
**
!*
**
!* RESEAU := Pflax RESEAU TAB INDEX MACRO DEVICE G23D2 ::
**
!*      <<Maxreg>> <<Precf>> <<Pow>> >>Eps1<< >>Keff<< ;
**
!*
**
!* NOTES:
**
!*
**
!*      Maxreg = maximum number of matrial regions in geometry
**
!*      Precf= Static flux calculation precision
**
!*      Pow  = total fission power
**
!*      Eps1 = error on flux form
**
!*      Keff = effective mutltiplication factor
**
!*
**
!*****
****
PARAMETER RESEAU TAB INDEX MACRO DEVICE G23D2 ::
      ::: LINKED_LIST RESEAU TAB INDEX MACRO DEVICE G23D2 ;
      ;

MODULE TRIVAT: TRIVAA: NEWMAC: FLUD: END: GREP:
      INIMAC: CRE: REFUEL: FLXAXC: POWER: ;

LINKED_LIST MACRES MACRO2 TRACK MACRO3 SYSTEM FLUX      ;

!* Declaration des variables
!*****
REAL Eps1 Precf Keff Pow ;
INTEGER Maxreg Edit ;

!* Lecture des arguments de la procedure

```



```

!*****
:: >>Maxreg<< >>Precf<< >>Pow<< >>Edit<< ;

!* Proprietes en fonction du burnup actuel
!*****
MACRES := CRE: TAB RESEAU ::
      EDIT 0
      READ
      TABLE TAB MIX 1 'NATURAL'
              BURNUP TAVGC ENDMIX ;

MACRO2 := INIMAC: INDEX MACRO MACRES ;

MACRO3 := NEWMAC: DEVICE MACRO2 G23D2 ::
      EDIT 0 NOSIGF
      TYPE CONTROLLED XFAC 2.0
      TYPE PERT XFAC 2.0
      ;

!* Creation du fichier de tracking
!*****
TRACK := TRIVAT: G23D2 ::
      EDIT 0
      TITLE 'G2 3D '
      MAXR <<Maxreg>> MCFD 1 ;

!* Creation des matrices de systeme d'elements finis
!*****
SYSTEM := TRIVAA: MACRO3 TRACK :: EDIT 0 ;

!* Calcul du flux (probleme aux valeurs propres)
!*****
FLUX := FLUD: SYSTEM TRACK :: EDIT 0 PREC <<Precf>> ;

!* Calcul de la forme de flux axial
!*****
RESEAU := FLXAXC: RESEAU FLUX TRACK INDEX :: AXIAL COMP ;

!* Calcul des puissances (normalisees) de canal
!*****
RESEAU := POWER: RESEAU MACRES :: EDIT <<Edit>> POWER <<Pow>> TCOMB
;

!* Ecriture des variables de sorties de la procedure
!*****
GREP: RESEAU :: GETVAL EPSF-AX 1 >>Eps1<< ;
GREP: FLUX :: GETVAL K-EFFECTIVE 1 >>Keff<< ;

:: <<Keff>> <<Eps1>> ;

END: ;

QUIT .

```

```

!*****
****
!*
**
!* PROCEDURE:  PgeoG2
**
!* USAGE:      Geometry construction for Gentilly2 reactor
**
!* AUTHOR:     E.Varin (96/02/01)
**
!*             B. Dionne (01/03/04) (pour cours ENE6209)
**
!* CALL:
**
!*
**
!*  GEOM2 INDEX := PgeoG2  ;
**
!*
**
!*
**
!* NOTES:
**
!*
**
!*
**
!*
**
!*      G2 reactor is a CAR3D geometry of 26*26*12 cells
**
!*      It is the Coarse definition, for fine geometry,
**
!*      There are mesh-splitting in X direction only
**
!*
**
!*      The boundary conditions are VOID or CYLI, for cylindric
**
!*      correction. By default, Only TWO mixture numbers are
**
!*      defined :  # 1 for the fuel and # 2 for the reflector
**
!*
**
!*      For any modification, you have to copy the file
**
!*      and modify it manually
**
!*
**
!*
**
!*****
****
PARAMETER GEOM2 INDEX ::
::: LINKED_LIST GEOM2 INDEX ;

```

[illegible]

```

0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0
0 0 0 0 0 2 2 2 2 2 1 1 1 1 1 1 1 1 2 2 2 2 0 0 0 0 0
0 0 0 0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 0 0 0 0 0
0 0 0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 0 0 0 0
0 0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 0 0 0
0 0 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0 0 0
0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 0
0 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0
2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2
2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2
2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2
2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2
2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2
2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2
2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2
0 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0
0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0
0 0 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0 0
0 0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0 0
0 0 0 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 0 0 0
0 0 0 0 2 2 2 1 1 1 1 1 1 1 1 1 1 2 2 2 0 0 0 0
0 0 0 0 0 2 2 2 2 2 1 1 1 1 1 2 2 2 2 2 0 0 0 0 0
0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0
0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0

```

```

PLANE 2 SAME 1
PLANE 4 SAME 3
PLANE 5 SAME 3
PLANE 6 SAME 3
PLANE 7 SAME 3
PLANE 8 SAME 3
PLANE 9 SAME 3
PLANE 10 SAME 3
PLANE 11 SAME 1
PLANE 12 SAME 1

```

```

MESHX 3.150 45.050 68.525 97.100 125.675 154.250 182.825 211.400
239.975 268.550 297.125 325.700 354.275 382.850 411.425
440.000 468.575 497.150 525.725 554.300 582.875 611.450
640.025 668.600 697.175 720.650 762.550
MESHY 3.150 45.050 68.525 97.100 125.675 154.250 182.825 211.400
239.975 268.550 297.125 325.700 354.275 382.850 411.425
440.000 468.575 497.150 525.725 554.300 582.875 611.450
640.025 668.600 697.175 720.650 762.550
MESHZ 3.940 53.470 103.000 152.530 202.060 251.590 301.120
350.650
400.180 449.710 499.240 548.770 598.30
;

```

```

GEOM2 INDEX := USPLIT: GEOM :: MAXR <<Maxreg>> ;

```

```

END: ;

```

QUIT .