

UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN SCIENCES DE L'ÉNERGIE ET DES MATÉRIAUX
OFFERT EN EXTENSION PAR L'INSTITUT NATIONAL DE LA
RECHERCHE SCIENTIFIQUE

PAR
ÉTIENNE DAUPHINAIS-RIVARD

SIMULATION NUMÉRIQUE ET VALIDATION EXPÉRIMENTALE D'UN
MODÈLE DE DÉTENTE DE GAZ RÉEL À HAUTE PRESSION

JUIN 2007

Université du Québec à Trois-Rivières

Service de la bibliothèque

Avertissement

L'auteur de ce mémoire ou de cette thèse a autorisé l'Université du Québec à Trois-Rivières à diffuser, à des fins non lucratives, une copie de son mémoire ou de sa thèse.

Cette diffusion n'entraîne pas une renonciation de la part de l'auteur à ses droits de propriété intellectuelle, incluant le droit d'auteur, sur ce mémoire ou cette thèse. Notamment, la reproduction ou la publication de la totalité ou d'une partie importante de ce mémoire ou de cette thèse requiert son autorisation.

Résumé

Le présent ouvrage porte sur la modélisation d'une détente d'hydrogène à haute pression hors d'un réservoir muni d'un conduit à section constante. Nous utilisons un modèle à paramètres localisés pour le réservoir et un modèle uni-dimensionnel en régime permanent pour le conduit.

Les technologies de l'hydrogène sont de plus en plus regardées comme des technologies d'avenir. Bien sûr l'application de ces technologies se doit d'être sécuritaire, d'où le besoin d'évaluer les risques associés, entre autres, à l'utilisation de la haute pression. Cette évaluation se fait, dans la plupart des cas, par simulation. Notre modèle s'attaque à l'aspect de la simulation visant à déterminer les propriétés et la vitesse du fluide à la sortie de la fuite.

Les principales hypothèses de la modélisation sont : un écoulement isentropique au niveau du réservoir et un écoulement en régime permanent avec friction au niveau du conduit. On tient également compte des effets de gaz réel par le biais de la base de données expérimentales REFPROP du NIST.

Afin de valider le modèle, nous avons comparé les résultats obtenus à des données expérimentales. Bien que les courbes expérimentales et simulées de température présentent un écart important, les courbes de débit massique et de pression montrent une excellente concordance. Par conséquent, nous croyons que la concordance entre les simulations numériques et les résultats expérimentaux est satisfaisante pour la plupart des besoins d'ingénierie. Il faut également considérer le fait que les courbes expérimentales de température sont influencées par la méthode de mesure rudimentaire utilisée.

Remerciements

Je tiens d'abord à remercier MM. Yves Dubé et Pierre Bénard, pour leur grande contribution au présent travail de recherche et pour leur attitude chaleureuse et amicale. Également M. Eric W. Lemmon du «National Institute of Standards and Technology (NIST)» pour son support technique au niveau de la programmation.

Ce projet a été rendu possible grâce à la participation financière du réseau de centres d'excellence Auto21 et d'Hydro-Québec.

Une mention spéciale va également aux parents et amis m'ayant supporté tout au long de ce projet.

Table des matières

Résumé	iii
Remerciements	v
Table des figures	xii
Liste des tableaux	xiii
Liste des symboles	xvi
Introduction	1
1 Objectifs	3
2 Modélisation	5
2.1 Réservoir	5
2.1.1 Hypothèses	5
2.1.2 Volume de contrôle	6
2.1.3 Équation de continuité	6
2.1.4 Écoulement isentropique quasi-unidimensionnel en régime per-	
manent	7
2.1.5 Premier principe de la thermodynamique	9
2.1.6 Transfert de chaleur	10
2.1.7 Intégration numérique	11
2.2 Conduit	14
2.2.1 Hypothèses	14
2.2.2 Volume de contrôle	15
2.2.3 Équation de continuité	15
2.2.4 Premier principe de la thermodynamique	16
2.2.5 Conservation de la quantité de mouvement	17

2.2.6	Coefficient de friction de Darcy	19
2.2.7	Intégration numérique	20
2.2.8	Effet du transfert de chaleur	21
2.3	Combinaison du conduit et du réservoir	23
2.4	Influence de l'équation d'état	24
3	Validation	29
3.1	Montage	29
3.1.1	Instrumentation	29
3.1.2	Acquisition	31
3.2	Traitement des résultats	36
3.2.1	Temps zéro	36
3.2.2	Interpolation	36
3.2.3	Calcul du débit massique par différentiation numérique	36
4	Comparaison	41
4.1	Powertech	41
4.1.1	Compatibilité de la simulation et des résultats expérimentaux	41
4.1.2	Résultats	43
4.2	IRH	46
4.2.1	3,175mm × 1cm	46
4.2.2	1,5875mm × 1cm	46
4.2.3	3,175mm × 150cm	49
4.2.4	1,5875mm × 152cm	49
4.3	Discussion	54
4.3.1	Explication de l'écart en température	54
4.3.2	Précision du modèle	57
	Conclusion	59
	A Graphiques supplémentaires	61
	B Listing	77
	Bibliographie	107

Table des figures

2.1	Volume de contrôle du réservoir	6
2.2	Ordinogramme du calcul des propriétés à la sortie en fonction des propriétés dans le réservoir	8
2.3	Cylindre creux	10
2.4	Sphere creuse	11
2.5	Élément d'intégration pour le conduit	15
2.6	Bilan de force sur l'élément d'intégration du conduit	18
2.7	Identification du col et de la sortie	23
2.8	Influence de l'équation d'état utilisée pour une fuite d'azote	24
2.9	Influence de l'équation d'état utilisée pour une fuite d'hydrogène	25
2.10	Pressions et températures en utilisant REFPROP pour un réservoir de 150L ayant un orifice de sortie de 6mm de diamètre à une pression de 700bar.	25
2.11	Vitesse à la sortie en utilisant REFPROP pour un réservoir de 150L ayant un orifice de sortie de 6mm de diamètre à une pression de 700bar.	26
2.12	Nombre de Mach à la sortie en utilisant REFPROP pour un réservoir de 150L ayant un orifice de sortie de 6mm de diamètre à une pression de 700bar.	26
3.1	Vue d'ensemble du montage expérimental	30
3.2	Thermocouple	31
3.3	Boucle de lecture de la pression	33
3.4	Initialisation du matériel d'acquisition de la température	34
3.5	Boucle de lecture de la température	35
3.6	Température en fonction du temps	38
3.7	Masse volumique calculée selon l'algorithme en fonction du temps	39
3.8	Exemple de calcul du débit massique à l'aide d'un algorithme de dérivation numérique simple	39

4.1	Montage expérimental de la firme Powertech	42
4.2	Instruments près de la sortie du conduit	42
4.3	Pression statique dans le conduit selon la position	43
4.4	Powertech : pression expérimentale et simulée	44
4.5	Powertech : température expérimentale et simulée	44
4.6	Powertech : débit massique expérimental et simulé	45
4.7	Powertech : débit massique dans la partie de la courbe au-delà du point d'inflexion suivant le sommet	45
4.8	Sortie du réservoir	47
4.9	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur totale de 10cm	47
4.10	Température expérimentale et simulée pour un conduit ayant un dia- mètre externe de 3,175mm et une canalisation d'une longueur totale de 10cm	48
4.11	Débit massique expérimental et simulé pour un conduit ayant un dia- mètre externe de 3,175mm et une canalisation d'une longueur totale de 10cm	48
4.12	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 1cm . . .	49
4.13	Température expérimentale et simulée pour un conduit ayant un dia- mètre externe de 1,5875mm et une canalisation d'une longueur de 1cm	50
4.14	Débit massique expérimental et simulé pour un conduit ayant un dia- mètre externe de 1,5875mm et une canalisation d'une longueur de 1cm	50
4.15	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 150cm . .	51
4.16	Température expérimentale et simulée pour un conduit ayant un dia- mètre externe de 3,175mm et une canalisation d'une longueur de 150cm	51
4.17	Débit massique expérimental et simulé pour un conduit ayant un dia- mètre externe de 3,175mm et une canalisation d'une longueur de 150cm	52
4.18	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 152cm . .	52
4.19	Température expérimentale et simulée pour un conduit ayant un dia- mètre externe de 1,5875mm et une canalisation d'une longueur de 152cm	53
4.20	Débit massique expérimental et simulé pour un conduit ayant un dia- mètre externe de 1,5875mm et une canalisation d'une longueur de 152cm	53
4.21	Schéma de l'instrumentation du réservoir	54

4.22	Élément d'intégration du modèle d'ailette-tige	55
4.23	Courbe corrigée de la température	56
A.1	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 30cm . . .	62
A.2	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 30cm	63
A.3	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 30cm	63
A.4	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 60cm . . .	64
A.5	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 60cm	64
A.6	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 60cm	65
A.7	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 90cm . . .	65
A.8	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 90cm	66
A.9	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 90cm	66
A.10	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 120cm . .	67
A.11	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 120cm	67
A.12	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 120cm	68
A.13	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 25cm . .	69
A.14	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 25cm	70
A.15	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 25cm	70
A.16	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 50cm . .	71

A.17	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 50cm	71
A.18	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 50cm	72
A.19	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 80cm . .	72
A.20	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 80cm	73
A.21	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 80cm	73
A.22	Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 110cm . .	74
A.23	Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 110cm	74
A.24	Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 110cm	75

Liste des tableaux

2.1	Paramètres de la simulation	22
2.2	Effet du transfert de chaleur sur la longueur maximale du conduit . .	22
3.1	Paramètres et caractéristiques du capteur de pression	30
3.2	Résultats avant traitement	37
3.3	Résultats après traitement	37
4.1	Paramètres de l'équation de l'ailette-tige	56

Liste des Symboles

A	Section, Aire
c	Vitesse du son
C_p	Chaleur massique à pression constante
D	Diamètre
f	Coefficient de friction de Darcy
F	Force
g	Accélération gravitationnelle
h	Enthalpie, coefficient de convection
k	Rapport des chaleurs massiques
k_c	Coefficient de conduction thermique
L_c	Longueur du cylindre
m	Masse
M	Nombre de Mach
\dot{m}	Débit massique
N_c	Nombre de cylindres
p	Pression
\dot{Q}	Taux d'échange de chaleur
r	Rayon
Re	Nombre de Reynolds
s	Entropie
t	Temps

T	Température
u	Énergie interne
U	Conductivité globale
V	Vitesse
v_s	Vitesse moyenne du fluide
x	Position dans le conduit
Z	Hauteur

Lettres grecques

ε	Rugosité relative
μ	Viscosité
ρ	Masse volumique
τ_f	Résistance à l'écoulement en cisaillement
Ψ	Volume du réservoir

Indices

amb	Ambiant
atm	Atmosphérique
c	Au col
cyl	Cylindre
e	À l'entrée, externe
h	Hydraulique
i	Interne
s	À la sortie
$s.c.$	Surface de contrôle
sph	Sphère
t	À l'arrêt (à l'intérieur du réservoir)
$v.c.$	Volume de contrôle

Introduction

Avec une demande en énergie à l'échelle mondiale sans cesse grandissante, le besoin d'un moyen efficace d'emmagasiner cette énergie devient de plus en plus criant. Les solutions basées sur l'utilisation de l'hydrogène comme vecteur énergétique présentent des avantages de taille. Elles ont notamment le potentiel d'être complètement respectueuses de l'environnement, ce qui les rend très attrayantes. Comme l'optimisation du stockage de l'hydrogène reste un défi technologique, l'utilisation de la haute pression est, à l'heure actuelle, très répandue. L'application des technologies de haute pression à l'hydrogène étant relativement nouvelle, nous devons développer des outils permettant d'évaluer les risques associés aux fuites d'hydrogène à haute pression afin d'assurer la sécurité des utilisateurs. Les logiciels de simulations constituent une alternative peu coûteuse aux essais expérimentaux. Le présent ouvrage porte sur la modélisation d'une détente d'hydrogène à haute pression hors d'un réservoir muni d'un conduit à section constante à l'aide d'un modèle à zéro dimensions. Le programme basé sur ce modèle permet de calculer les propriétés thermodynamiques du fluide à l'intérieur du réservoir et à la sortie ainsi que la vitesse locale. Le particularité la plus innovante de nos travaux est sans doute l'utilisation de la base de données expérimentales «Reference Fluid Thermodynamic and Transport Properties (REFPROP)» du «National Institute of Standards and Technology (NIST)» pour calculer les propriétés thermodynamiques du fluide et ainsi tenir compte de l'effet d'un gaz réel. Nous avons d'ailleurs inclus, dans cet ouvrage, une comparaison permettant d'apprécier l'effet sur le calcul de la relation thermodynamique utilisée.

Le phénomène ayant lieu au niveau du réservoir est modélisé par un écoulement isentropique et celui du conduit par un écoulement en régime permanent avec friction.

En dernier lieu, comme nous avons trouvé très peu de données expérimentales sur le sujet, nous avons procédé à une série d'essais sur un réservoir réel afin de comparer aux mesures les résultats obtenus lors des simulations.

Le modèle développé s'adresse à la partie de la simulation d'une fuite d'hydrogène qui vise à déterminer les propriétés de la fuite à la sortie. Il n'est donc pas question

de la dispersion du fluide dans l'air après l'orifice, cela constitue un tout autre sujet où l'on utilise des méthodes passablement différentes.

Chapitre 1

Objectifs

La finalité du travail réalisé est de développer des standards de sécurité pour l'hydrogène. L'un des aspects de la sécurité est d'être en mesure de déterminer si une fuite d'hydrogène donnera lieu à des conditions favorables à une détonation ou une déflagration. Pour ce faire, nous devons d'abord connaître les propriétés thermodynamiques et la vitesse du fluide à la sortie de la fuite et nous devons ensuite simuler la dispersion du gaz dans l'environnement. La plupart des travaux réalisés sur la détente de l'hydrogène à haute pression portent sur la dispersion. Les travaux de Angers et al. [1] en sont un bon exemple. Certains autres auteurs comme Gallego et al. [2] s'attardent au phénomène ayant lieu dans le réservoir, mais utilisent des outils avancés d'analyse tri-dimensionnelle. Or, nous souhaitons conserver un modèle simple à paramètres localisés.

En somme, les programmes de simulation et les données expérimentales sur les propriétés du fluide à la fuite sont plus rares. Mohamed et Paraschivoiu [3] se sont attardés au phénomène ainsi que Jo et al. [4]. Les travaux de Jo et al. visent toutefois à estimer le débit massique rapidement afin de pouvoir réagir en cas d'accident. Notre objectif est différent, nous souhaitons obtenir le plus d'information possible pour des besoins de simulation. L'ouvrage de Mohamed et Paraschivoiu [3] va exactement dans le sens que nous souhaitons donner à notre travail. Nous allons pousser plus avant cette analyse en validant le modèle avec des résultats expérimentaux et en utilisant le «Reference Fluid Thermodynamic and Transport Properties Database» (REFPROP) du «National Institute of Standards and Technology» (NIST) pour le calcul des relations thermodynamiques.

Chapitre 2

Modélisation

La modélisation peut se diviser en deux parties distinctes. D’abord, le réservoir et ensuite, le conduit. Pour la modélisation du réservoir, nous avons basé nos travaux sur ceux de K. Mohamed et M. Paraschivoiu [3]. Bien que le modèle soit essentiellement le même, il permet en plus de décrire l’écoulement au-delà du moment où le régime passe en mode subsonique et d’utiliser la base données expérimentales du NIST plutôt qu’une équation afin de considérer l’effet du gaz réel. Pour le conduit, nous avons développé un modèle à partir des équations générales de la mécanique des fluides compressibles, toujours en utilisant le REFPROP du NIST.

2.1 Réservoir

2.1.1 Hypothèses

Le modèle du réservoir se veut simple et à zéro-dimension, par opposition à un modèle numérique tenant compte de la géométrie du réservoir. Voici les principales hypothèses sur lesquelles le modèle se base :

- Les propriétés du fluide dans le réservoir sont uniformes ;
- L’échange de chaleur entre le réservoir et l’air ambiant est considéré au moyen d’un modèle d’échange de chaleur en régime permanent, le caractère transitoire de l’échange de chaleur est donc négligé ;
- Pour des pressions suffisamment élevées, la vitesse du fluide à la sortie est égale à la vitesse du son locale. Une fois que la pression d’arrêt (pression que le fluide exercerait s’il était forcé de s’arrêter) chute sous le niveau critique (pression à laquelle la vitesse du fluide à la sortie passe de sonique à sub-sonique), la vitesse du fluide à la sortie devient subsonique et la pression au même endroit est égale

- à la pression atmosphérique ou toute pression arrière utilisée pour la simulation ;
- La détente de l’hydrogène à partir de son état d’arrêt dans le réservoir jusqu’à son état critique à la sortie s’effectue dans une petite région située près de l’orifice et est modélisée comme une écoulement isentropique quasi-unidimensionnel ;
- L’hydrogène demeure en tout temps en phase gazeuse ;
- Le modèle peut tenir compte de l’effet des gaz réels en utilisant le REFPROP du NIST.

2.1.2 Volume de contrôle

Par souci de cohérence, nous allons développer le modèle utilisé à partir des équations générales. Les hypothèses posées précédemment nous permettent d’utiliser le volume de contrôle illustré à la fin suivante et d’utiliser les équations de la thermodynamique de l’ouvrage de Van Wylen [5].

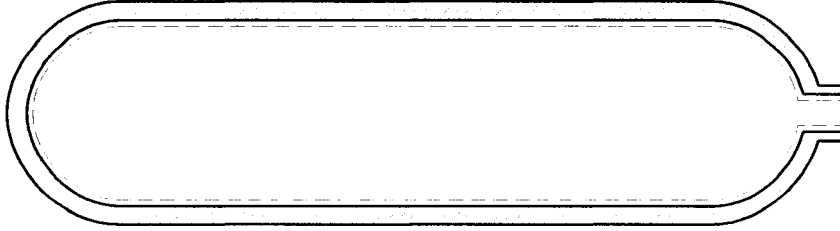


FIG. 2.1 – Volume de contrôle du réservoir

2.1.3 Équation de continuité

Pour un intervalle de temps dt , si nous avons de la masse qui entre à l’intérieur d’un volume de contrôle et de la masse qui en sort, alors, en vertu du principe de conservation de la masse, l’équation de continuité s’énonce comme suit

$$\frac{dm_{v.c.}}{dt} + \sum \dot{m}_s - \sum \dot{m}_e = 0 \quad (2.1)$$

Nous n’avons pas de masse entrante et seulement un débit sortant. L’équation devient donc

$$\frac{dm_{v.c.}}{dt} = -\dot{m}_s \quad (2.2)$$

Puisque la masse dans le volume de contrôle s'exprime comme $m_{v.c.} = \Psi \rho_t$, que le volume du réservoir est constant et que $\dot{m}_s = \rho_s V_s A_s$, on peut dire que

$$\frac{d\rho_t}{dt} = \frac{-\rho_s V_s A_s}{\Psi} \quad (2.3)$$

La vitesse quand à elle est définie par $V = Mc$. Donc,

$$\frac{d\rho_t}{dt} = \frac{-\rho_s M_s c_s A_s}{\Psi} \quad (2.4)$$

Il s'agit maintenant d'évaluer le nombre de Mach à la sortie.

2.1.4 Écoulement isentropique quasi-unidimensionnel en régime permanent

Pour ce faire, nous utilisons une méthode itérative tirée de l'ouvrage de Zucrow [6]. La figure 2.2 illustre cette méthode. La première étape consiste à évaluer l'entropie du système à partir de la pression et de la température du fluide à l'arrêt (à l'intérieur du réservoir) à l'aide de le REFPROP du NIST. Puisque nous considérons que l'écoulement est isentropique, nous connaissons l'entropie à la sortie.

$$s_t = s_s = s \quad (2.5)$$

En posant une seconde variable thermodynamique à la sortie, nous pouvons déterminer l'état thermodynamique du fluide au même endroit. La variable que nous avons choisi de poser est la pression. Ensuite, à l'aide de la définition de l'enthalpie d'arrêt, nous pouvons déterminer la vitesse du fluide à la sortie en fonction de la pression posée.

$$h_t = h_s + \frac{V_s^2}{2} = h_s + \frac{(M_s c_s)^2}{2} \quad (2.6)$$

Comme le système à l'étude ne comporte que des sections convergentes, la vitesse du fluide à la sortie ne pourra jamais dépasser Mach 1, en raison du phénomène d'étranglement des gaz. Notre objectif consiste maintenant à trouver la valeur de la pression pour laquelle le nombre de Mach est égal à 1. Pour ce faire nous prenons, comme première approximation, la valeur de la pression critique pour un gaz parfait. On peut évaluer cette dernière à l'aide de l'équation suivante.

$$\frac{p_s}{p_t} = \left(1 + \frac{1}{2} (k-1) M^2 \right)^{\frac{k}{k-1}} \quad (2.7)$$

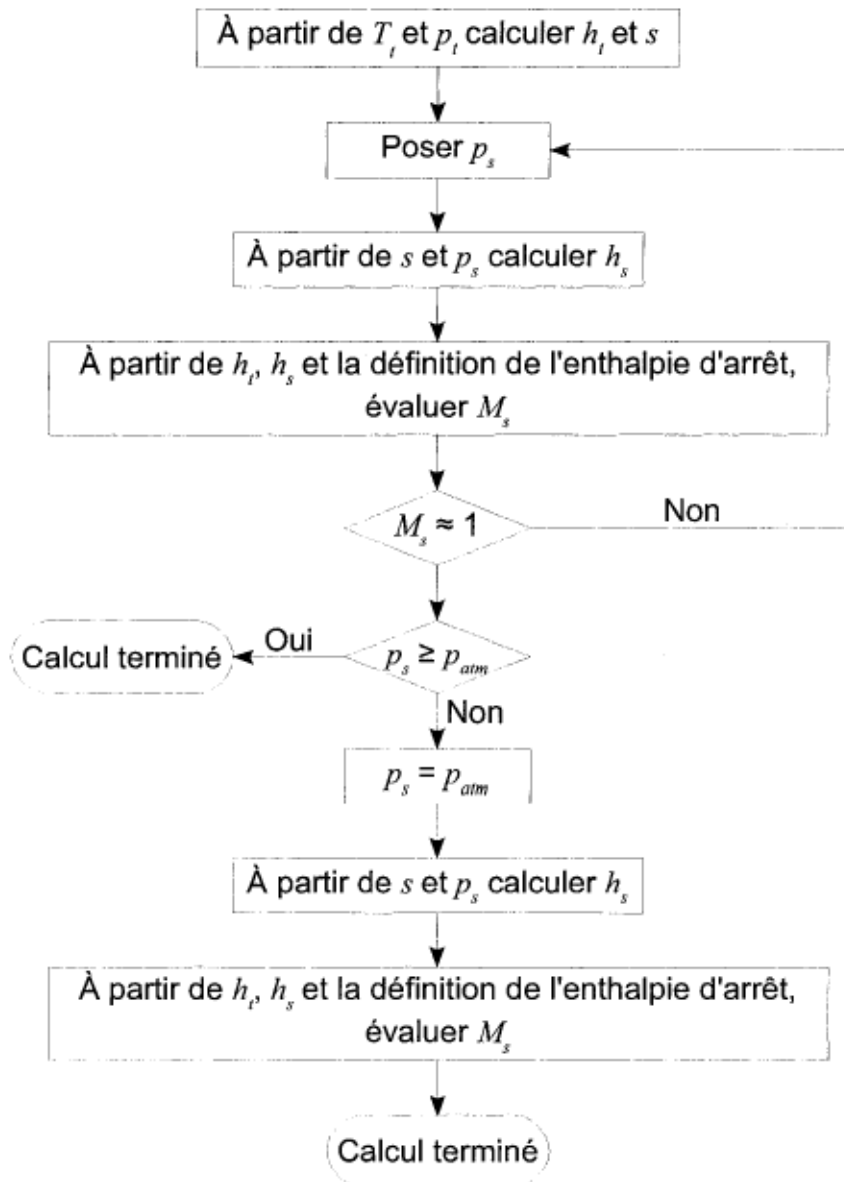


FIG. 2.2 – Ordinogramme du calcul des propriétés à la sortie en fonction des propriétés dans le réservoir

Pour l'itération, nous utilisons un algorithme simple basé sur l'interpolation linéaire. Lorsque la pression est identifiée, nous comparons sa valeur à la pression atmosphérique. Si la pression à la sortie est supérieur à la pression atmosphérique, alors le régime est bel et bien sonique et les valeurs obtenues sont valides. Si toutefois, la pression obtenue est inférieure à la pression atmosphérique, alors nous savons que le régime est passé en mode subsonique. Il est également raisonnable de penser que la pression à la sortie est égale à la pression atmosphérique. La valeur pression étant connue, il est possible déterminer le nombre de Mach et les autres variables thermodynamiques.

Nous avons donc un première équation différentielle qui comporte une variable indépendante, le temps, et deux variables dépendantes, deux des variables thermodynamiques. La seconde équation du système provient du premier principe de la thermodynamique.

2.1.5 Premier principe de la thermodynamique

Toujours d'après Van Wylen [5], le premier principe de la thermodynamique pour un volume de contrôle s'énonce comme suit

$$\dot{Q}_{v.c.} + \sum \dot{m}_e \left(h_e + \frac{V_e^2}{2} + gZ_e \right) = \frac{dE_{v.c.}}{dt} + \sum \dot{m}_s \left(h_s + \frac{V_s^2}{2} + gZ_s \right) + W_{v.c.} \quad (2.8)$$

Or, nous n'avons pas de débit sortant ni de travail sur le volume de contrôle et nous négligeons les effets de la gravité, ce qui donne

$$\frac{dE_{v.c.}}{dt} = \dot{Q}_{v.c.} - \dot{m} \left(h_s + \frac{V_s^2}{2} \right) \quad (2.9)$$

En substituant les définitions de l'enthalpie d'arrêt et de l'énergie interne

$$\frac{d(m_{v.c.} u_t)}{dt} = \dot{Q}_{v.c.} - \dot{m} h_t \quad (2.10)$$

Par analyse,

$$\begin{aligned} \frac{dm_{v.c.}}{dt} u_t + m_{v.c.} \frac{du_t}{dt} &= \dot{Q}_{v.c.} - \dot{m} h_t \\ -\dot{m} u_t + m_{v.c.} \frac{du_t}{dt} &= \dot{Q}_{v.c.} - \dot{m} h_t \\ \frac{du_t}{dt} &= \frac{\dot{Q}_{v.c.} + \dot{m} (u_t - h_t)}{m_{v.c.}} \end{aligned} \quad (2.11)$$

Voilà donc la seconde équation permettant d'obtenir un système d'équations différentielles cohérent. On peut résoudre ce système par un algorithme d'intégration numérique de type Runge-Kutta. Nous verrons dans la prochaine section que la valeur de $\dot{Q}_{v.c.}$ ne dépend que de la température du fluide et de la géométrie du réservoir.

2.1.6 Transfert de chaleur

Pour le terme $\dot{Q}_{v.c.}$ de l'équation 2.11, nous nous sommes inspirés de modèles classiques qui figurent dans l'ouvrage de Kreith [7] sur le transfert de chaleur. Nous avons assimilé notre réservoir à un cylindre à bouts sphériques. L'échange de chaleur se fait donc à l'interface d'une sphère pleine (deux demi-sphères) et d'un cylindre.

L'équation générale du transfert de chaleur en régime permanent est

$$\dot{Q} = U \cdot A \cdot \Delta T$$

Pour le cylindre représenté à la figure 2.3, on peut décrire le transfert de chaleur en

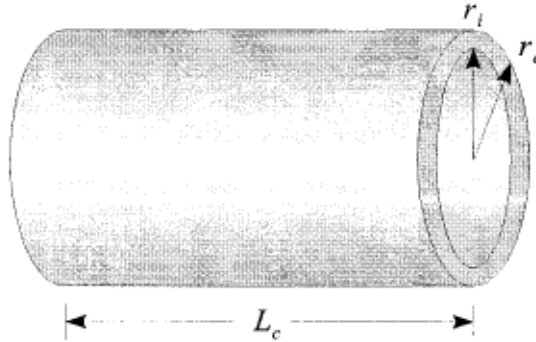


FIG. 2.3 – Cylindre creux

régime permanent à l'aide de l'équation tirée du modèle de Kreith [7]

$$\dot{Q}_{cyl} = \frac{T_{amb} - T}{\frac{1}{2} \frac{1}{h_i \pi r_i L_c} + \frac{1}{2} \frac{\ln \frac{r_e}{r_i}}{\pi k_c L_c} + \frac{1}{2} \frac{1}{h_e \pi r_e L_c}} \quad (2.12)$$

Pour la partie sphérique du réservoir illustrée à la figure 2.4, nous avons l'équation 2.13, également tirée de l'ouvrage de Kreith [7]

$$\dot{Q}_{sph} = \frac{T_{amb} - T}{\frac{1}{4} \frac{1}{h_i \pi r_i} + \frac{1}{4} \frac{\left(\frac{1}{r_i} - \frac{1}{r_e}\right)}{\pi k_c} + \frac{1}{4} \frac{1}{h_e \pi r_e}} \quad (2.13)$$

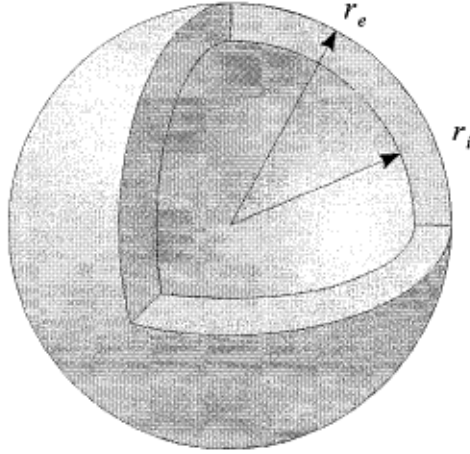


FIG. 2.4 – Sphere creuse

Pour obtenir l'équation qui s'applique au réservoir en entier, il s'agit d'additionner les équations 2.12 et 2.13. Nous négligeons également la résistance thermique de l'interface entre le gaz et la paroi intérieure du réservoir parce qu'il s'agit d'un milieu fermé et que ce paramètre serait extrêmement difficile à déterminer. Nous allons plutôt jouer sur d'autres paramètres afin d'ajuster notre modèle à une situation particulière. L'équation obtenue par analyse est donc la suivante

$$\dot{Q}_{v.c.} = N_c (T_{amb} - T_t) \left(\frac{2\pi r_e L_c}{\frac{r_e \ln\left(\frac{r_e}{r_i}\right)}{\frac{1}{k_c}} + \frac{1}{h_e}} + \frac{1}{\frac{1}{4} \frac{1}{h_e \pi r_e^2} + \frac{1}{4} \frac{r_e - r_i}{\pi k_c r_e r_i}} \right) \quad (2.14)$$

Nous avons également ajouté à l'équation la variable N_{cyl} . Cette variable permet d'ajuster l'échange de chaleur pour le cas où l'on simule une fuite sur plusieurs cylindres branchés en parallèle. La surface d'échange est alors beaucoup plus importante. Pour une simulation «normale» cette variable conserve la valeur 1. Il faut également noter que cette équation ne dépend que de la température, les autres paramètres étant des constantes. Cela permet de conserver un système à deux équations, deux variables dépendantes et une seule variable indépendante.

2.1.7 Intégration numérique

Nous utilisons la méthode de Runge-Kutta-Fehlberg. Cette méthode consiste principalement à calculer en parallèle la solution d'ordre 4 et la solution d'ordre 5. À partir de ces deux solutions, il est possible d'évaluer l'erreur due à l'intégration numérique.

Dans notre cas cette erreur sera utilisée par un algorithme de contrôle adaptatif du pas d'intégration. Donc, nous avons un système de la forme

$$\frac{dx}{dt} = f(t, x, y)$$

$$\frac{dy}{dt} = g(t, x, y)$$

Pour un tel système, la solution d'ordre 4 est

$$\hat{x}_{n+1} = x + \left(\frac{25}{216}q_1 + \frac{1408}{2565}q_3 + \frac{2197}{4104}q_4 - \frac{1}{5}q_5 \right)$$

$$\hat{y}_{n+1} = y + \left(\frac{25}{216}r_1 + \frac{1408}{2565}r_3 + \frac{2197}{4104}r_4 - \frac{1}{5}r_5 \right)$$

et la solution d'ordre 5 est

$$x_{n+1} = x + \left(\frac{16}{135}q_1 + \frac{6656}{12825}q_3 + \frac{28561}{56430}q_4 - \frac{9}{50}q_5 + \frac{2}{55}q_6 \right)$$

$$y_{n+1} = y + \left(\frac{16}{135}r_1 + \frac{6656}{12825}r_3 + \frac{28561}{56430}r_4 - \frac{9}{50}r_5 + \frac{2}{55}r_6 \right)$$

Où

$$q_1 = h \cdot f(t, x, y)$$

$$r_1 = h \cdot g(t, x, y)$$

$$q_2 = h \cdot f\left(t + \frac{1}{4}h, x + \frac{1}{4}q_1, y + \frac{1}{4}r_1\right)$$

$$r_2 = h \cdot g\left(t + \frac{1}{4}h, x + \frac{1}{4}q_1, y + \frac{1}{4}r_1\right)$$

$$q_3 = h \cdot f\left(t + \frac{3}{8}h, x + \frac{3}{32}q_1 + \frac{9}{32}q_2, y + \frac{3}{32}r_1 + \frac{9}{32}r_2\right)$$

$$r_3 = h \cdot g\left(t + \frac{3}{8}h, x + \frac{3}{32}q_1 + \frac{9}{32}q_2, y + \frac{3}{32}r_1 + \frac{9}{32}r_2\right)$$

$$q_4 = h \cdot f\left(t + \frac{12}{13}h, x + \frac{1932}{2197}q_1 - \frac{7200}{2197}q_2 + \frac{7296}{2197}q_3, y + \frac{1932}{2197}r_1 - \frac{7200}{2197}r_2 + \frac{7296}{2197}r_3\right)$$

$$r_4 = h \cdot g\left(t + \frac{12}{13}h, x + \frac{1932}{2197}q_1 - \frac{7200}{2197}q_2 + \frac{7296}{2197}q_3, y + \frac{1932}{2197}r_1 - \frac{7200}{2197}r_2 + \frac{7296}{2197}r_3\right)$$

$$q_5 = h \cdot f \left(t + h, x + \frac{439}{216}q_1 - 8q_2 + \frac{3680}{513}q_3 - \frac{845}{4104}q_4, \right. \\ \left. y + \frac{439}{216}r_1 - 8r_2 + \frac{3680}{513}r_3 - \frac{845}{4104}r_4 \right)$$

$$r_5 = h \cdot g \left(t + h, x + \frac{439}{216}q_1 - 8q_2 + \frac{3680}{513}q_3 - \frac{845}{4104}q_4, \right. \\ \left. y + \frac{439}{216}r_1 - 8r_2 + \frac{3680}{513}r_3 - \frac{845}{4104}r_4 \right)$$

$$q_6 = h \cdot f \left(t + \frac{1}{2}h, x - \frac{8}{27}q_1 + 2q_2 - \frac{3544}{2565}q_3 + \frac{1859}{4104}q_4 - \frac{11}{40}q_5, \right. \\ \left. y - \frac{8}{27}r_1 + 2r_2 - \frac{3544}{2565}r_3 + \frac{1859}{4104}r_4 - \frac{11}{40}r_5 \right)$$

$$r_6 = h \cdot g \left(t + \frac{1}{2}h, x - \frac{8}{27}q_1 + 2q_2 - \frac{3544}{2565}q_3 + \frac{1859}{4104}q_4 - \frac{11}{40}q_5, \right. \\ \left. y - \frac{8}{27}r_1 + 2r_2 - \frac{3544}{2565}r_3 + \frac{1859}{4104}r_4 - \frac{11}{40}r_5 \right)$$

Les erreurs absolues et relatives sont respectivement évaluées de la façon suivante

$$E_x = |\hat{x} - x|$$

$$E_y = |\hat{y} - y|$$

$$R_x = \frac{E_x}{x}$$

$$R_y = \frac{E_y}{y}$$

Afin d'optimiser le temps de calcul, nous avons décidé d'implanter une routine de contrôle adaptatif du pas d'intégration plutôt que d'utiliser un pas d'intégration constant. Il s'agit d'abord de spécifier une erreur désirée. Cette erreur est ensuite comparée avec l'erreur observée la plus élevée. Si l'erreur observée excède de plus de

10%, alors on ajuste le pas d'intégration de la façon suivante

$$h_{new} = h_{old} S \left(\frac{E}{D} \right)^{\frac{-1}{q}} \quad (2.15)$$

où q est l'ordre de la méthode d'intégration (e.g. $q = 1$ pour Runge-Kutta-Fehlberg), S est un facteur de sécurité, E est l'erreur relative la plus élevée et D est l'erreur désirée. Si toutefois, l'erreur observée est 50% plus faible que l'erreur désirée, alors le pas sera augmenté de la façon suivante

$$h_{new} = h_{old} S \left(\frac{E}{D} \right)^{\frac{-1}{q+1}} \quad (2.16)$$

De plus, pour éviter des oscillations trop importantes ou une perte de contrôle du pas d'intégration, le coefficient global d'ajustement du pas est limité à l'intervalle $\left[\frac{1}{5}, 5 \right]$.

2.2 Conduit

Lors de la modélisation du conduit, nous nous sommes demandé, en premier lieu, si l'approximation d'un écoulement en régime permanent dans le conduit est juste. Nous avons pris la décision de premièrement faire cette hypothèse, de vérifier si elle donne de bons résultats et finalement, dans le cas où l'hypothèse s'avérerait trop simpliste, d'évaluer ce qui peut être fait pour améliorer la fidélité du modèle. Cette section traite donc de la modélisation du conduit sous l'hypothèse d'un régime permanent.

2.2.1 Hypothèses

Voici une liste des hypothèses utilisées pour la modélisation du conduit :

- Il s'agit d'un écoulement unidimensionnel, les propriétés du fluide à chaque section du conduit sont donc uniformes;
- L'échange de chaleur entre le conduit et l'air ambiant est considéré au moyen d'un modèle d'échange de chaleur en régime permanent, le caractère transitoire de l'échange de chaleur est donc négligé;
- Pour des pressions suffisamment élevées, la vitesse du fluide à la sortie est égale à la vitesse du son au même endroit. Une fois que la pression d'arrêt chute sous le niveau critique, la vitesse du fluide à la sortie devient subsonique et la pression au même endroit est égale à la pression atmosphérique ou toute pression arrière utilisée pour la simulation;

- L'hydrogène demeure en tout temps en phase gazeuse ;
- Le modèle peut tenir compte de l'effet des gaz réels en utilisant le REFPROP du NIST.

2.2.2 Volume de contrôle

Ces hypothèses permettent d'utiliser le volume de contrôle représenté à la figure 2.5 pour le développement du modèle. Par l'application sur ce volume de contrôle des

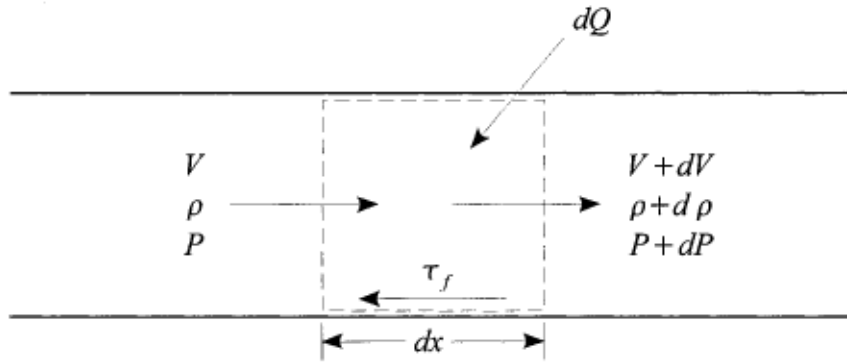


FIG. 2.5 – Élément d'intégration pour le conduit

bilans de masse, d'énergie et de quantité de mouvement, nous arriverons à développer un système d'équations différentielles ordinaires cohérent permettant de décrire le phénomène.

2.2.3 Équation de continuité

Le principe de continuité ou de conservation de la masse veut, pour un tel volume de contrôle, que le débit entrant soit égal au débit sortant. Or le débit peut s'exprimer de la façon suivante.

$$\dot{m} = \rho V A \quad (2.17)$$

Donc, si on se reporte à la figure 2.5, on peut écrire

$$\rho V A = (\rho + d\rho) (V + dV) A$$

Par analyse,

$$\rho V = \rho V + \rho dV + V d\rho + d\rho dV$$

$$\rho dV = -V d\rho$$

$$\frac{dV}{V} + \frac{d\rho}{\rho} = 0 \quad (2.18)$$

Pour un gaz réel, $\rho(P, T)$. Ce qui veut dire que

$$d\rho = \left(\frac{\partial \rho}{\partial P} \right)_T dP + \left(\frac{\partial \rho}{\partial T} \right)_P dT$$

En substituant cette dernière équation dans l'équation 2.18 et en divisant par dx ,

$$\left(\frac{\partial \rho}{\partial P} \right)_T \frac{dP}{dx} + \left(\frac{\partial \rho}{\partial T} \right)_P \frac{dT}{dx} + \frac{\rho}{V} \frac{dV}{dx} = 0 \quad (2.19)$$

Les dérivées partielles de l'équation 2.19 peuvent être considérées comme des variables thermodynamiques d'état. L'équation ne comporte donc que des variables thermodynamiques à l'exception de la position et de la vitesse. Puisqu'il est possible de déterminer toutes les variables thermodynamiques d'un état donné à partir de deux d'entre-elles, nous avons une équation à une variable indépendante et trois variables dépendantes. Nous aurons besoin d'autres équations afin d'obtenir un système que l'on peut résoudre. Ces équations seront développées dans les prochaines sections à l'aide du premier principe de la thermodynamique et du principe de conservation de la quantité de mouvement.

2.2.4 Premier principe de la thermodynamique

Pour le volume de contrôle illustré à la figure 2.5, c'est-à-dire un écoulement en régime permanent et sans travail, le premier principe de la thermodynamique se réduit à l'équation suivante.

$$\dot{Q}_{v.c.} + \dot{m} \left(h + \frac{V^2}{2} \right) = \dot{m} \left(h + dh + \frac{(V + dV)^2}{2} \right)$$

Par analyse seulement, nous pouvons réduire l'équation encore davantage

$$dh + V dV = \frac{\dot{Q}_{v.c.}}{\dot{m}} \quad (2.20)$$

Comme

$$dh = \left(\frac{\partial h}{\partial P} \right)_T dP + \left(\frac{\partial h}{\partial T} \right)_P dT$$

et que $\left(\frac{\partial h}{\partial T}\right)_P$ est la définition de C_p , l'équation 2.20 devient

$$\left(\frac{\partial h}{\partial P}\right)_T dP + \left(\frac{\partial h}{\partial T}\right)_P dT + VdV = \frac{\dot{Q}_{v.c.}}{\dot{m}}$$

Le transfert de chaleur est évalué de la même façon qu'à la section 2.1.6. En substituant

$$\dot{Q}_{v.c.} = \frac{T_{amb} - T}{\frac{1}{2} \frac{1}{h_i \pi r_i} + \frac{1}{2} \frac{\ln \frac{r_a}{r_i}}{\pi k_c} + \frac{1}{2} \frac{1}{h_e \pi r_e}} dx$$

et en divisant par dx nous avons

$$\left(\frac{\partial h}{\partial P}\right)_T \frac{dP}{dx} + C_p \frac{dT}{dx} + V \frac{dV}{dx} = \frac{1}{\dot{m}} \frac{T_{amb} - T}{\frac{1}{2} \frac{1}{h_i \pi r_i} + \frac{1}{2} \frac{\ln \frac{r_a}{r_i}}{\pi k_c} + \frac{1}{2} \frac{1}{h_e \pi r_e}} \quad (2.21)$$

Voilà la deuxième équation du système d'équations différentielles du conduit. Nous allons maintenant appliquer le principe de conservation de la quantité de mouvement afin d'obtenir la troisième et dernière équation.

2.2.5 Conservation de la quantité de mouvement

L'application du principe de conservation du momentum sur un volume de contrôle tel que celui que l'on utilise est un problème bien connu. On l'utilise en fait pour développer les équations de l'écoulement de type «Fanno Line Flow». On retrouve cet exercice entre autres dans l'ouvrage de John [8]. Pour l'écoulement de type «Fanno Line Flow» on ne s'intéresse qu'à la friction entre le fluide et la paroi du conduit, l'écoulement est adiabatique. Il s'agit en fait du contraire de l'écoulement de type «Rayleigh Line Flow» qui ne tient compte que de l'échange de chaleur et non de la friction.

Notre objectif consiste à développer un modèle tenant compte à la fois et de l'échange de chaleur entre le conduit et l'air ambiant et de la friction. Il est intéressant de noter qu'il est possible d'appliquer le premier principe de la thermodynamique de la même façon qu'on l'applique pour le «Rayleigh Line Flow» puisque la friction n'intervient pas dans ce bilan. De même, il est possible d'appliquer le principe de conservation de la quantité de mouvement de la même façon qu'on l'applique pour «Fanno Line Flow» puisque l'échange de chaleur n'intervient pas dans ce bilan.

Pour un écoulement en régime permanent, le principe de conservation de la quan-

tité de mouvement s'énonce de la façon suivante :

$$\sum F_x = \iiint_{s.c.} V (\rho V dA)$$

En vertu du principe de conservation de la quantité de mouvement et en se référant à la figure 2.5, le bilan des forces en jeu peu s'énoncer comme ceci

$$PA - (P + dP) A - \tau_f A_{s.c.} = (\rho AV) (V + dV) - (\rho AV) V$$

Ce qui se réduit à

$$-AdP - \tau_f A_{s.c.} = \rho AV dV \quad (2.22)$$

Le diamètre hydraulique d'un conduit se définit comme

$$D_h = \frac{4A}{\text{périmètre}}$$

Pour un conduit circulaire, $D_h = D$. En substituant dans l'équation 2.22,

$$-AdP - \tau_f \frac{4A}{D} dx = \rho AV dV \quad (2.23)$$

Nous allons maintenant exprimer τ_f en termes de pression et surface. Comme le

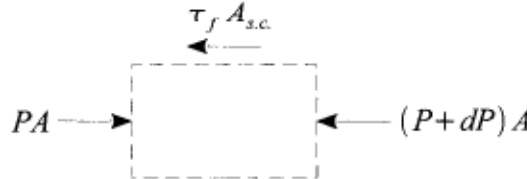


FIG. 2.6 – Bilan de force sur l'élément d'intégration du conduit

régime est permanent, le bilan de force représenté à la figure 2.6 s'énonce comme suit :

$$PA - \tau_f A_{s.c.} - (P + dP) A = 0$$

$$PA - \tau_f \frac{4A}{D} dx - PA - dPA = 0$$

$$\tau_f = -\frac{dP}{dx} \frac{D}{4}$$

Le coefficient de friction de Darcy-Weisbach est un nombre adimensionnel intervenant dans le calcul des écoulements internes. Il établit la proportionnalité entre la vitesse

moyenne de l'écoulement et le gradient de pression. On le définit comme

$$f = -\frac{dP}{dx} \frac{D_h}{\frac{1}{2}\rho V^2}$$

Donc,

$$\tau_f = \frac{f\rho V^2}{8}$$

En substituant dans l'équation 2.23,

$$-AdP - \frac{1}{2} \frac{f\rho AV^2}{D} dx = \rho AV dV$$

et finalement, par analyse,

$$\frac{1}{\rho V} \frac{dP}{dx} + \frac{dV}{dx} = -\frac{1}{2} \frac{Vf}{D} \quad (2.24)$$

Afin de pouvoir utiliser l'équation 2.24, nous devons être en mesure d'évaluer le coefficient de friction (coefficient de Darcy). Il existe plusieurs façons d'évaluer ce coefficient, la section suivante traite de ce sujet.

2.2.6 Coefficient de friction de Darcy

Nous avons tiré les différentes équations permettant d'évaluer f du livre de Zucrow et Hoffman [6]. On compte au total cinq équations. Nous avons d'abord l'équation de von Karman.

$$\frac{1}{\sqrt{f}} + 2 \log_{10} \left(18.6 \frac{1}{\text{Re} \sqrt{f}} \right) = 1.74$$

Cette équation est appropriée dans le cas de tubes lisses. L'équation de Colebrook quant à elle s'applique aux écoulements turbulents.

$$\frac{1}{\sqrt{f}} = -2 \log_{10} \left(0.2702702703 \frac{\varepsilon}{D} + 2.51 \frac{\sqrt{f}}{\text{Re}} \right)$$

L'équation de Prandtl s'applique aux conduits rugueux.

$$\frac{1}{\sqrt{f}} = 1.74 + 2 \log_{10} \left(\frac{\varepsilon}{D} \right)$$

Les deux premières équations ne se solutionnent que par des méthodes numériques itératives, ce qui requiert un temps de calcul considérable. Dans le cas où le conduit n'est pas rugueux et où l'on souhaite accélérer l'algorithme, les deux dernières équations

peuvent constituer une solution. D'abord l'équation de Génereaux :

$$f = 4 (0.04 \text{Re}^{-0.16})$$

qui se veut une équation générale avec une précision raisonnable. Sinon, on peut utiliser un coefficient de friction constant, dans ce cas, l'auteur recommande d'utiliser la valeur suivante

$$f = 0.02$$

Bien sûr l'utilisation de cette constante est limitée. Il s'agit alors de consulter le diagramme de Moody, que l'on peut retrouver dans l'ouvrage de Zucrow et Hoffman [6], afin de déterminer s'il est juste de le faire dans un cas en particulier. Pour ce qui nous concerne, nous allons nous en tenir aux autres équations.

Seulement à titre de rappel, le nombre de Reynolds, pour un conduit circulaire, est évalué de cette façon :

$$\text{Re} = \frac{\rho v_s D}{\mu}$$

Nous avons maintenant tous les éléments du système d'équations différentielles ordinaires du conduit. La forme actuelle des équations, toutefois, se prête mal à une intégration numérique de type Runge-Kutta. La prochaine section explique comment cette adaptation a été effectuée.

2.2.7 Intégration numérique

Afin de pouvoir résoudre un système d'équations différentielles ordinaires par la méthode de Runge-Kutta-Fehlberg, les équations doivent être de la forme

$$\frac{dx}{dt} = f(t, x, y)$$

Or, plus d'un terme des équations 2.19, 2.21 et 2.24 comprennent des dérivées. Il est toutefois possible de rendre les dérivées explicites par la méthode d'élimination de Gauss-Jordan. Une fois que les dérivées sont exprimées de façon explicite, nous devons spécifier des valeurs initiales pour les variables dépendantes et indépendantes et nous pouvons ensuite procéder à l'intégration.

L'intervalle d'intégration était évident dans le cas du réservoir, on intègre jusqu'à ce que le réservoir soit vide. Pour le conduit on souhaite intégrer jusqu'à la sortie. Or on sait, comme dans le cas du réservoir, que la condition à la sortie sera une des

conditions suivantes :

1. $M_s = 1$

2. $P_s = P_{atm}$

Il s'agit donc d'intégrer jusqu'à ce que l'une de ces deux conditions se produise. On obtient alors une longueur de conduit pour des conditions données à l'entrées.

2.2.8 Effet du transfert de chaleur

Les premières simulations réalisées avec le modèle de conduit incluant à la fois l'effet de la friction et du transfert de chaleur nous ont laissé croire que l'effet du transfert de chaleur est négligeable. Nous allons comparer, dans cette section, les résultats engendrés par les modèles avec et sans échange de chaleur au niveau du conduit.

Lorsqu'on néglige le transfert de chaleur, les équations issues du principe de conservation de la quantité de mouvement et de l'équation de continuité restent inchangées, seule l'équation provenant du premier principe de la thermodynamique sera modifiée pour devenir

$$\left(\frac{\partial h}{\partial P}\right)_T \frac{dP}{dx} + C_p \frac{dT}{dx} + V \frac{dV}{dx} = 0 \quad (2.25)$$

Nous allons évaluer, pour chacun des cas, la longueur maximale de conduit possible pour des conditions données à l'entrée de ce dernier. La liste qui suit donne la valeur assignée à chacun des paramètres.

Le REFPROP du NIST utilisé pour tenir compte de l'effet des gaz réels comporte des données pour plusieurs fluides, le fluide utilisé ici est l'hydrogène.

Bien que la simulation tienne compte de l'effet des gaz réels, certaines approximations de gaz parfait sont faites à l'intérieur du programme afin d'accélérer la convergence des méthodes numériques associées au calculs impliquant des gaz réels. Voilà pourquoi nous retrouvons, dans la liste des paramètres, des constantes de gaz parfaits.

La table suivante résume les résultats du calcul de la longueur de conduit avec et sans transfert de chaleur. À la lumière de ce résultat, nous avons pris la décision de négliger le transfert de chaleur dans le modèle du conduit puisque l'impact de ce dernier sur le résultat n'est pas suffisant.

<i>Paramètre</i>	<i>Valeur</i>	
Pression atmosphérique P_{atm}	101353.0	Pa
Température ambiante T_{amb}	300	K
Coefficient de convection interne h_i	100	W/(m ² · K)
Coefficient de convection externe h_e	10	W/(m ² · K)
Coefficient de conduction k_c	15	W/(m ² · K)
Section du conduit A_c	3.17×10^{-5}	m ²
Épaisseur de la paroi du conduit e_c	0.000889	m
Diamètre interne du conduit D_i	$2\sqrt{\frac{A_c}{\pi}}$	
Diamètre externe du conduit D_e	$D_i + 2M_c$	
Longueur du conduit L_c	7	m
Coefficient de friction de Darcy f	0.015	
Temps initial t_0	0	s
Pression d'arrêt initiale P_{t0}	34.5×10^6	Pa
Température d'arrêt initiale T_{t0}	T_{amb}	
Pas d'intégration initial pour le conduit h_0	$\frac{1}{100}L_c$	
Précision désirée pour l'intégration du conduit D_c	0.00000001	
Constante des gaz parfaits (g.p.) R	4124.18	J/(kg · K)
Chaleur massique à pression cte. pour un g.p. C_{p0}	14209.1	J/(kg · K)
Chaleur massique à volume ct. pour un g.p. C_{v0}	10084.9	J/(kg · K)
Rapport des chaleurs massiques k	1.409	

TAB. 2.1 – Paramètres de la simulation

<i>Transfert de chaleur</i>	<i>Longueur du conduit (m)</i>
Considéré	7.80035707183858
Négligé	7.80036090143392
Différence	0.00000382959534

TAB. 2.2 – Effet du transfert de chaleur sur la longueur maximale du conduit

2.3 Combinaison du conduit et du réservoir

Nous avons maintenant un modèle pour le conduit et un modèle pour le réservoir. Nous devons combiner ces deux modèles afin d'obtenir un modèle pour le système complet.

La «combinaison» des deux modèles consiste principalement à déterminer les conditions au col. Les seules informations dont nous disposons en tout temps sont

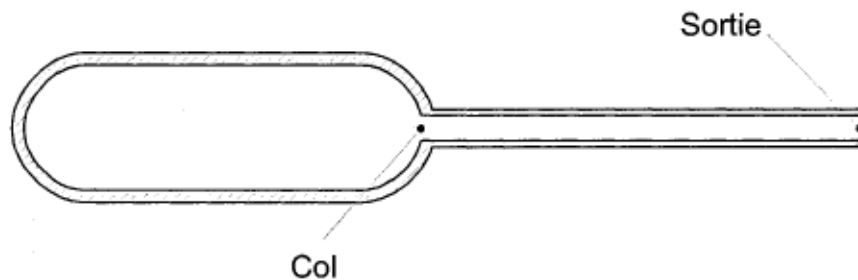


FIG. 2.7 – Identification du col et de la sortie

- la température d'arrêt,
- la pression d'arrêt,
- le nombre de Mach à la sortie ou la pression à la sortie (selon le résultat du calcul).

La stratégie utilisée pour déterminer les conditions au col est la suivante. Pour chaque pas de temps lors de l'intégration du système d'EDO du réservoir, nous posons une variable thermodynamique au col. En l'occurrence, la pression. Comme nous l'avons expliqué à la section 2.1.4, le fait de poser cette variable nous permet de déterminer les autres variables thermodynamiques ainsi que la vitesse du fluide. Ces valeurs servent ensuite de données initiales pour l'intégration du système d'EDO du conduit. Une fois que l'intégration le long du conduit est terminée, nous obtenons la longueur du conduit correspondant aux valeurs initiales. On compare alors cette longueur à la longueur réelle du conduit. On répète le processus en posant chaque fois une nouvelle pression au niveau du col, jusqu'à ce que la longueur obtenue corresponde à la longueur réelle au degré de précision voulu. Les itérations sur la pression se font à l'aide d'une méthode d'itération par extrapolation linéaire.

2.4 Influence de l'équation d'état

Avant de passer à la partie expérimentale, nous aimerions donner au lecteur une idée de la magnitude de l'effet de l'équation d'état sur les résultats. Bien qu'il ne fasse aucun doute que l'utilisation du REFPROP du NIST améliore la précision des résultats, nous avons réalisé quelques simulations utilisant des relations thermodynamiques différentes ainsi que différents fluides. La figure 2.8 présente l'écart entre les différentes équations d'état identifiées sur le graphique pour une fuite d'azote et la figure 2.9 répète l'exercice pour une fuite d'hydrogène. On peut voir que l'écart est

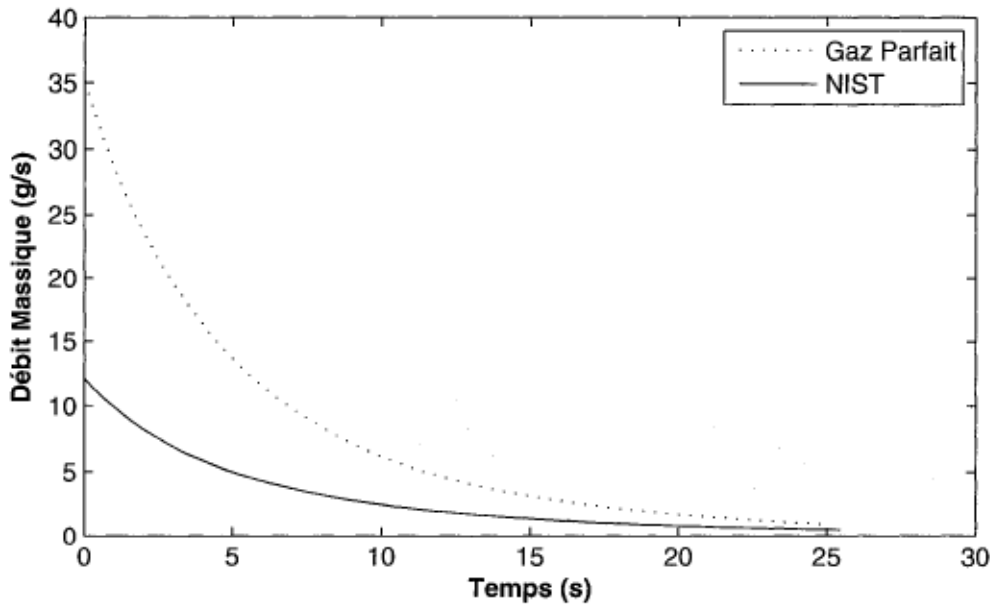


FIG. 2.8 – Influence de l'équation d'état utilisée pour une fuite d'azote

plus faible dans le cas de l'hydrogène puisque le comportement de ce gaz est très près de celui d'un gaz parfait, mais tout de même non négligeable. La différence moyenne entre le débit engendré par l'équation de Beattie-Bridgeman et celui du REFPROP est de 11,5% tandis que l'écart maximal est 13,7%.

Les figures 2.10, 2.11 et 2.12 illustrent le reste des données que peut fournir le logiciel lors des simulations. Évidemment, aux endroits où nous connaissons la pression et la température, il est également possible de déterminer toutes les autres propriétés thermodynamiques telles que l'entropie, l'enthalpie, l'énergie interne, etc.

Dans le but de valider le modèle décrit dans ce chapitre, nous avons procédé à une série d'essais. Le chapitre qui suit traite des méthodes employées lors de ces essais. Nous allons ensuite comparer les résultats des simulations avec les résultats

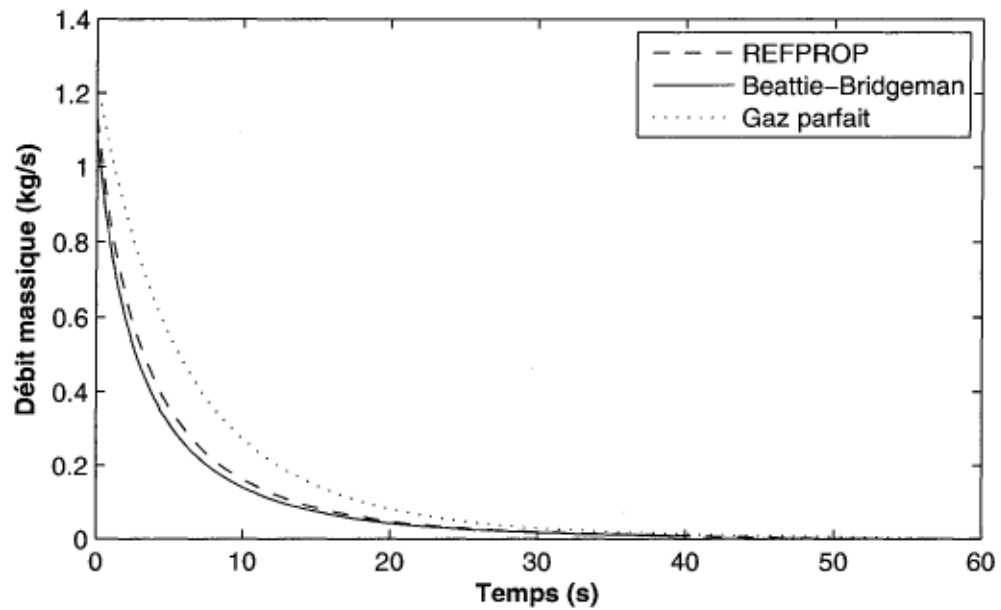


FIG. 2.9 – Influence de l'équation d'état utilisée pour une fuite d'hydrogène

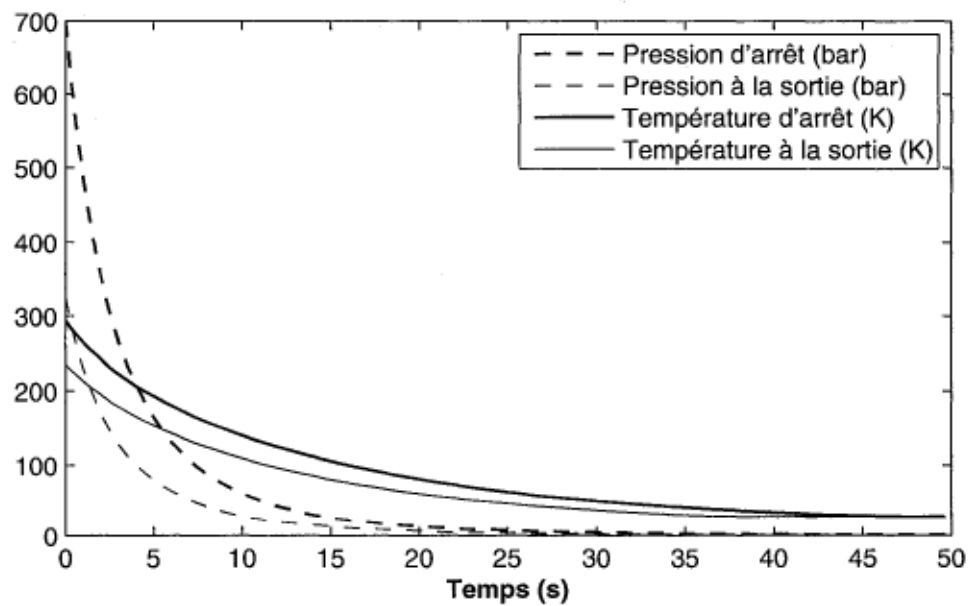


FIG. 2.10 – Pressions et températures en utilisant REFPROP pour un réservoir de 150L ayant un orifice de sortie de 6mm de diamètre à une pression de 700bar.

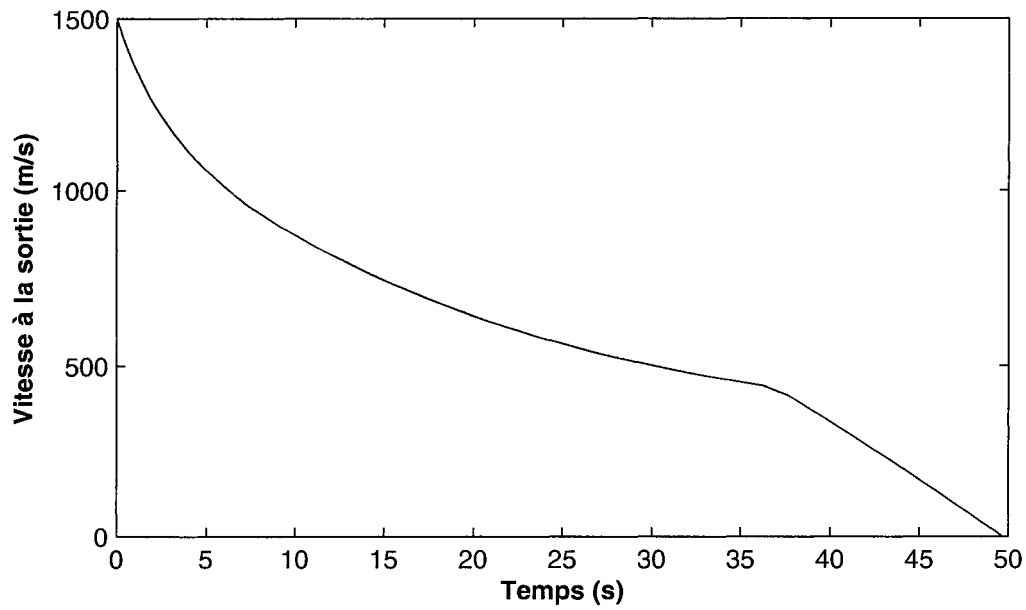


FIG. 2.11 – Vitesse à la sortie en utilisant REFPROP pour un réservoir de 150L ayant un orifice de sortie de 6mm de diamètre à une pression de 700bar.

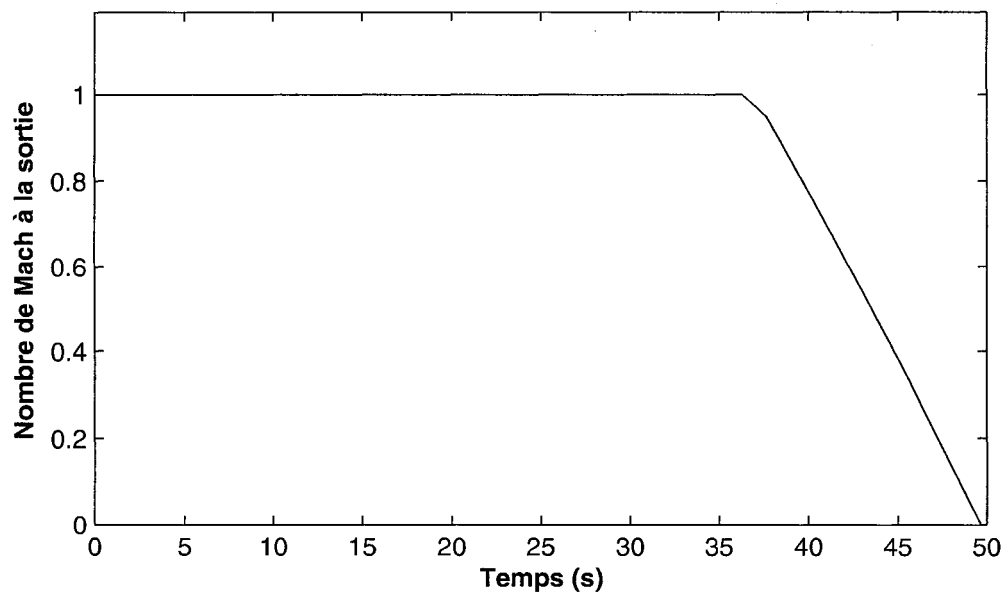


FIG. 2.12 – Nombre de Mach à la sortie en utilisant REFPROP pour un réservoir de 150L ayant un orifice de sortie de 6mm de diamètre à une pression de 700bar.

expérimentaux.

Chapitre 3

Validation

Afin de déterminer à quel point le modèle développé est fidèle à la réalité, nous avons choisi de réaliser un montage et de procéder à une série d'essais. Voici d'abord une description du montage utilisé.

3.1 Montage

La figure 3.1 donne une vue d'ensemble du montage utilisé. Il s'agit principalement d'un réservoir d'azote de taille industrielle standard servant à mettre sous pression un plus petit réservoir instrumenté pour nos besoins et d'un ordinateur pour l'acquisition de données. Puisque les expériences devaient être faites à l'intérieur, nous avons choisi d'utiliser l'azote plutôt que l'hydrogène comme gaz pour des considérations de sécurité. Commençons par discuter des instruments utilisés.

3.1.1 Instrumentation

Les instruments utilisés sont :

- Un capteur de pression de haute précision de marque Paroscientific, modèle 740.
- Un thermocouple de type T à fils très fins.

Les données techniques et le manuel d'utilisation du capteur sont disponibles sur le site web de la compagnie (www.paroscientific.com). Le capteur de pression est doté d'une mémoire interne permettant d'ajuster plusieurs paramètres d'acquisition. Bien sûr, le comportement de l'appareil dépend de ces paramètres. La table 3.1 présente une liste des paramètres utilisés et les caractéristiques engendrées par ces paramètres. Le thermocouple, quant à lui, a été fabriqué à partir de fil de constantan et de cuivre de 3 millièmes de pouce de diamètre. Nous avons inséré les fils dans un tube d'acier

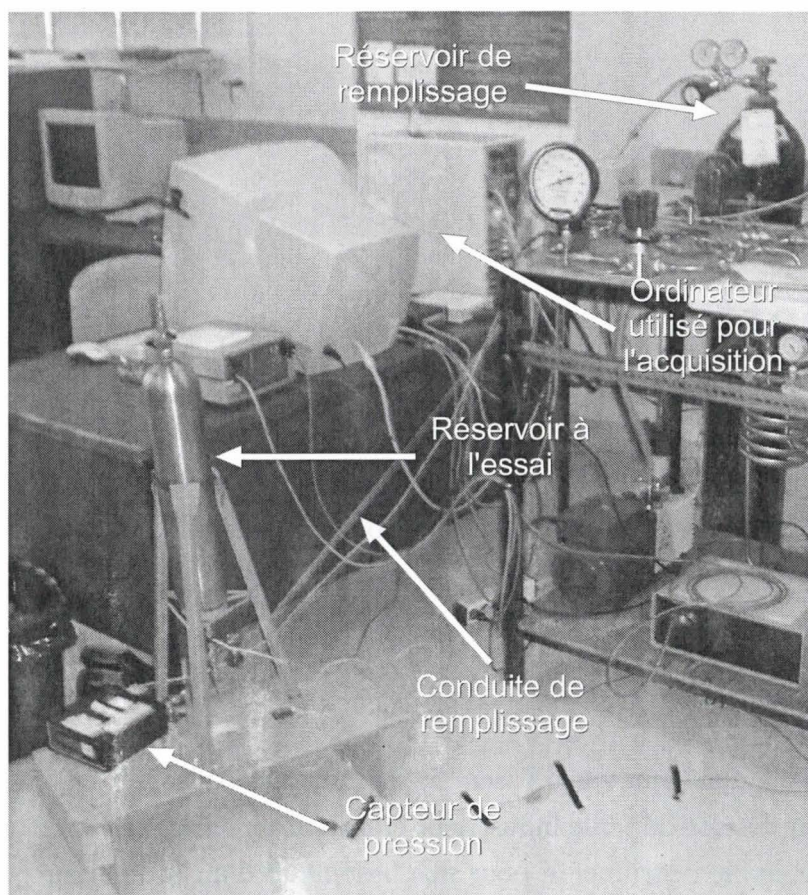


FIG. 3.1 – Vue d'ensemble du montage expérimental

<i>Paramètres / caractéristiques</i>	<i>Valeur</i>
Mode d'échantillonnage	Lecture et envoi simple
Plage de pression mesurable	0 à 41,3 MPa
Résolution	10 ppm
Précision	0,004 MPa
Taux d'échantillonnage	8 éch./s
Unités de pression	MPa
Taux de transfert («Baud rate»)	19 200

TAB. 3.1 – Paramètres et caractéristiques du capteur de pression

inoxydable de $\frac{1}{8}$ de pouce de diamètre de marque Swagelok afin de donner une certaine rigidité à l'instrument et ainsi s'assurer que la jonction du thermocouple n'entre pas en contact avec la paroi du cylindre. L'extrémité du tube allant dans le réservoir a ensuite été scellée à l'aide d'une résine afin de préserver l'étanchéité du système. Nous avons pris soin, lors de cette étape, de ne pas couvrir la jonction du thermocouple de résine afin de ne pas affecter le temps de réponse de celui-ci. Nous tenions à avoir un fil nu dans le fluide.

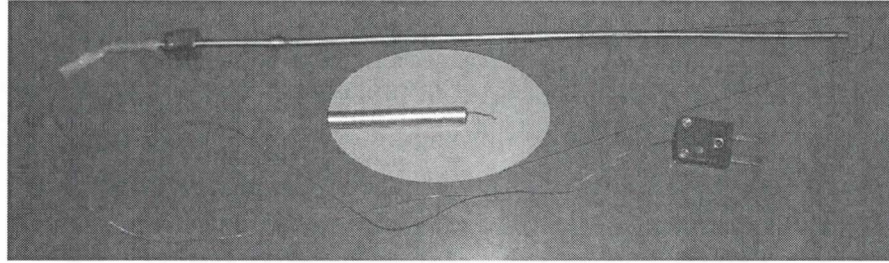


FIG. 3.2 – Thermocouple

3.1.2 Acquisition

L'acquisition a été réalisée à l'aide d'un PC muni du logiciel LabVIEW. Le capteur de pression se connecte au PC par un port série de type RS-232 et le thermocouple se connecte au même PC via un bornier et une carte d'acquisition de marque «National Instruments», modèle NI4351. La précision, selon le fabricant, devrait se trouver au dessus de 1°C. Nous allons nous attarder ici au fonctionnement du logiciel LabVIEW afin de pouvoir ensuite expliquer le traitement des données obtenues.

Le programme consiste essentiellement en deux boucles (une pour la pression et une pour la température) indépendantes s'exécutant en parallèle tout au long de l'essai. Aucune vitesse d'exécution n'est spécifiée pour les boucles. Elles tournent donc aussi rapidement que le processeur du PC peut le faire.

La boucle de lecture de la pression est représentée à la figure 3.3, telle que vue à l'intérieur de l'environnement de programmation LabVIEW. Lorsqu'on lance le programme, la boucle s'exécute et ne s'arrête que lorsque l'utilisateur met fin à l'exécution du programme. À l'intérieur de la boucle, on note d'abord le temps. On relève ensuite la pression à l'intérieur du réservoir et on note le temps une seconde fois. On envoie ensuite ces trois informations sur une seule et même ligne du fichier de résultats.

Pour ce qui est de la boucle de température illustrée à la figure 3.5, le processus est légèrement plus complexe. On doit d'abord initialiser le matériel d'acquisition. Cette

procédure d'initialisation que l'on peut voir à la figure 3.4 se connecte à la boucle de lecture de la température par la gauche et a été tirée d'un exemple de programme du manufacturier, National Instruments. L'initialisation ne se fait qu'une seule fois et on entre ensuite à l'intérieur de la boucle. Le matériel fonctionne de la façon suivante : il prend des lectures de façon continue et les envoie au PC seulement lorsque l'utilisateur le demande. Or, la vitesse d'exécution de la boucle est suffisante pour qu'il n'y ait qu'une seule lecture disponible à la fois. Donc, la première étape consiste à vérifier si des lectures sont disponibles. Ensuite, qu'il y ait une lecture disponible ou non, on note le temps. Si aucune lecture n'est disponible, alors on note le temps pour un usage ultérieur. Dans le cas contraire, on prend la lecture, on note le temps et on envoie au fichier la lecture, le temps qui vient d'être relevé, mais également le temps de l'itération précédente afin de déterminer entre quels moments les lectures ont été prises et ainsi obtenir l'incertitude sur le temps.

Le résultat final de l'acquisition est donc deux fichiers textes où les valeurs sont séparées par des caractères de tabulation. Chaque fichier compte trois colonnes. La première contient le temps avant lequel la lecture n'a pu être prise, la seconde contient le temps après lequel la lecture a été prise et la dernière contient la lecture de pression ou de température. À ce stade, nous n'avons que des résultats bruts qui doivent être raffinés afin de les représenter sous forme de graphique.

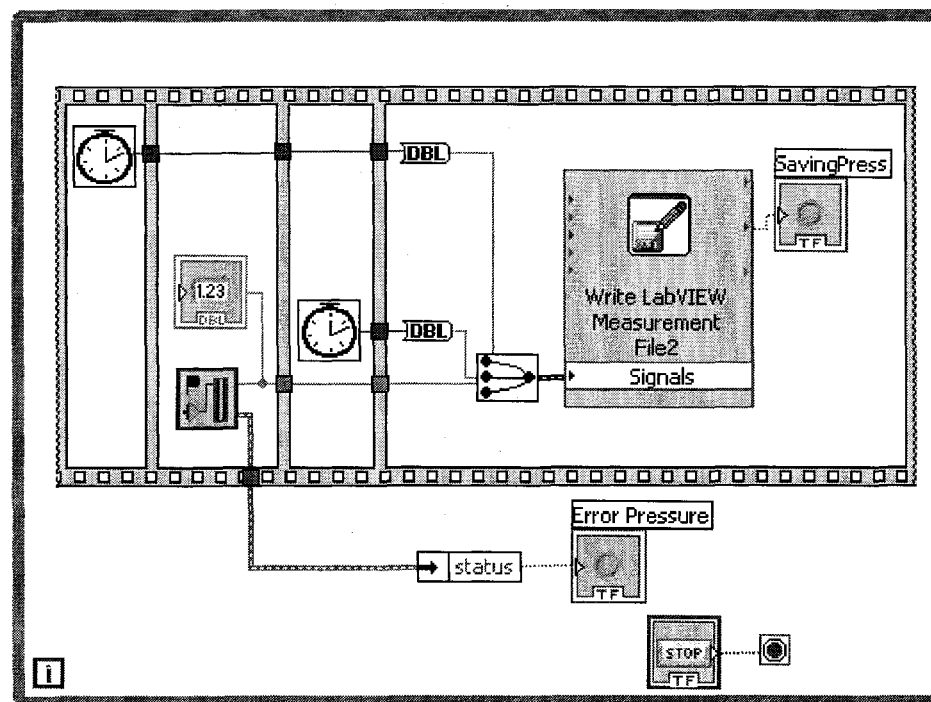


FIG. 3.3 – Boucle de lecture de la pression

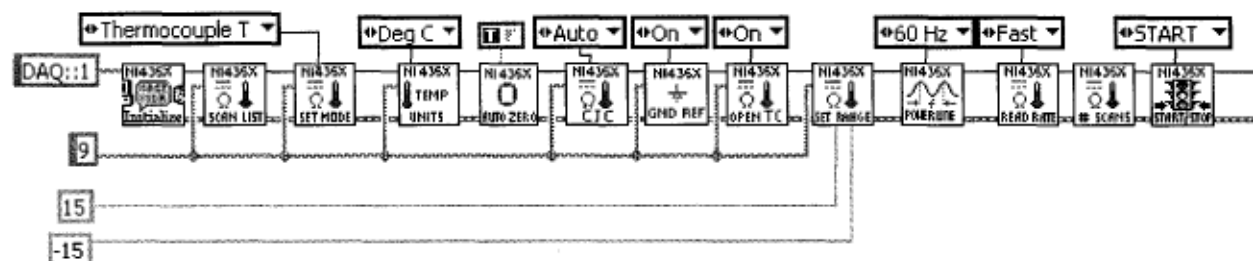


FIG. 3.4 – Initialisation du matériel d'acquisition de la température

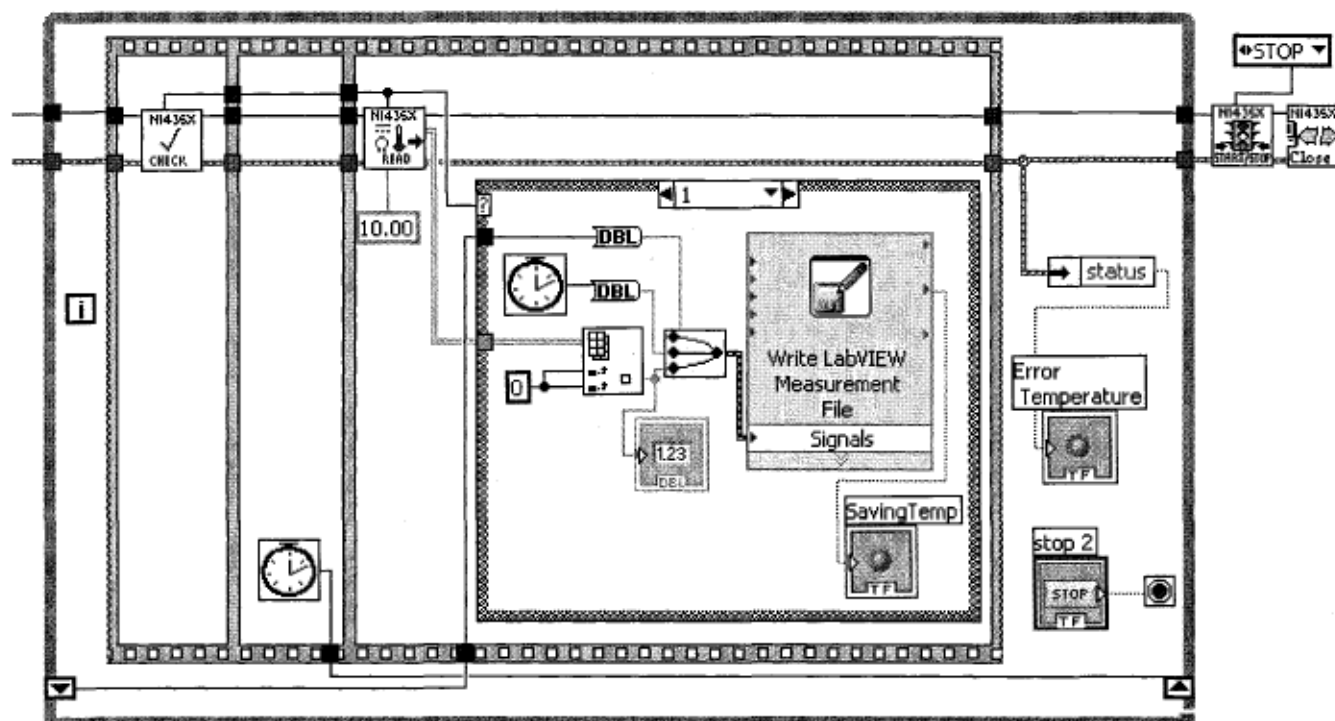


FIG. 3.5 – Boucle de lecture de la température

3.2 Traitement des résultats

3.2.1 Temps zéro

Nous devons apporter plusieurs changements aux résultats bruts avant de pouvoir les représenter sous forme de graphique. Le premier d'entre eux concerne le temps. Nous devons, pour deux raisons, identifier à quel moment précis l'écoulement a commencé. Premièrement, nous devons, pour des raisons pratiques, démarrer l'acquisition avant d'ouvrir la valve du réservoir. De ce seul fait, nous sommes certains qu'il y aura un décalage dans les résultats. La seconde raison est due au fait que le programme d'acquisition envoie le temps au fichier de résultats sous la forme d'un nombre de millisecondes à 8 chiffres et que la valeur du premier de ces temps est aléatoire.

3.2.2 Interpolation

En plus d'identifier le «temps zéro», nous devons calculer, à partir de la pression et de la température, les autres propriétés thermodynamiques. Pour ce faire, nous devons disposer de couples de points pression-température. Or, les taux d'échantillonnage de la pression et de la température ne sont pas les mêmes, ce qui veut dire qu'aucune température n'a été prise au même moment qu'une pression et vice-versa. Nous avons contourné ce problème de façon mathématique en effectuant une interpolation sur les résultats. Nous voulions garder tous les points de pression puisqu'ils sont plus nombreux. Nous avons fait donc fait l'interpolation sur la température. Pour chaque couple temps-pression, nous avons trouvé la température par interpolation. Cette opération a été faite sur Microsoft Excel à l'aide d'un ajout contenant les fonctions d'interpolation nommé «XlXtrFun». La fonction utilisée fonctionne avec un algorithme d'interpolation de type parabolique. Les tables 3.2 et 3.3 donnent un aperçu des données avant et après traitement. Les incertitudes sur les données sont respectivement de 1°C, 0,004 MPa et 0,5 kg/m³ pour la température, la pression et la masse volumique.

On peut noter que le «rééchantillonnage» des résultats par interpolation a permis d'entrer des couples de points dans le REFPROP du NIST et d'obtenir la masse volumique.

3.2.3 Calcul du débit massique par différentiation numérique

Maintenant que nous connaissons, pour n'importe quel temps donné, la masse volumique du fluide dans le réservoir et le volume du réservoir, il est possible de déterminer la masse de fluide contenu dans le cylindre. Pour ce faire, il suffit de

t_1 (ms)	t_2 (ms)	<i>Pression</i> (MPa)	t_1 (ms)	t_2 (ms)	<i>Temperature</i> (K)
17018600	17018780	9.903	17018620	17018620	294
17018800	17018980	9.903	17018930	17018930	294
17018990	17019180	9.903	17019240	17019240	294
17019190	17019370	9.903	17019550	17019550	294
17019390	17019570	9.904	17019860	17019860	294
17019580	17019770	9.904	17020170	17020170	294
17019780	17019960	9.863	17020480	17020480	292
17019980	17020160	9.506	17020790	17020790	289
17020180	17020360	9.164	17021100	17021100	287
17020370	17020550	8.838	17021410	17021410	285
17020570	17020750	8.530	17021720	17021720	280
17020760	17020940	8.237	17022030	17022030	277
17020960	17021150	7.949	17022340	17022340	274

TAB. 3.2 – Résultats avant traitement

<i>Temps</i> (s)	<i>Pression</i> (MPa)	<i>Température</i> (K)	<i>Masse Volumique</i> (kg/m ³)
0	9.904	293	114
0.195	9.863	294	113
0.395	9.506	294	109
0.595	9.164	294	105
0.785	8.838	292	102
0.985	8.530	290	100
1.175	8.237	289	97
1.38	7.949	288	94
1.58	7.681	286	91
1.775	7.426	284	89

TAB. 3.3 – Résultats après traitement

multiplier, à chaque point, la masse volumique et le volume du réservoir. Pour obtenir le débit massique, nous devons dériver la fonction de la masse à l'intérieur du réservoir par rapport au temps. Or, comme en témoigne la figure 3.6, les courbes formées par les données recueillies lors de l'expérimentation comportent du «bruit». Cela

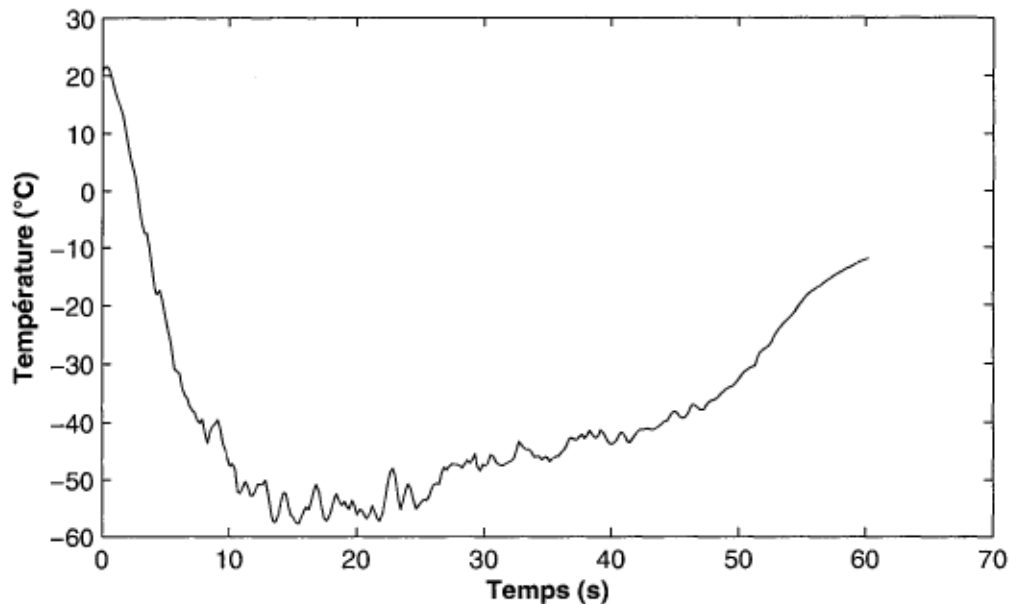


FIG. 3.6 – Température en fonction du temps

rend le processus de dérivation numérique délicat puisque les algorithmes simples de différentiation numérique sont très sensibles au bruit et peuvent donner de très mauvais résultats. Il s'agit donc de «lisser» les courbes obtenues. Le problème de bruit concerne davantage les courbes de température que de pression. Favorablement, le seul fait de calculer la courbe de masse volumique (figure 3.7) à partir des deux autres propriétés introduit déjà un certain lissage. Comme on peut le voir sur la figure 3.8 le résultat de la dérivée numérique réalisée à l'aide d'un algorithme simple est acceptable, grâce au lissage introduit par le calcul de la masse volumique. Nous aurions pu faire appel à des algorithmes plus évolués, mais nous préférons plutôt préserver l'intégrité des résultats et ensuite voir si la courbe de simulation passe par les différents points de la courbe expérimentale.

Dans la prochaine section, nous allons comparer les résultats des simulations avec les données obtenues expérimentalement.

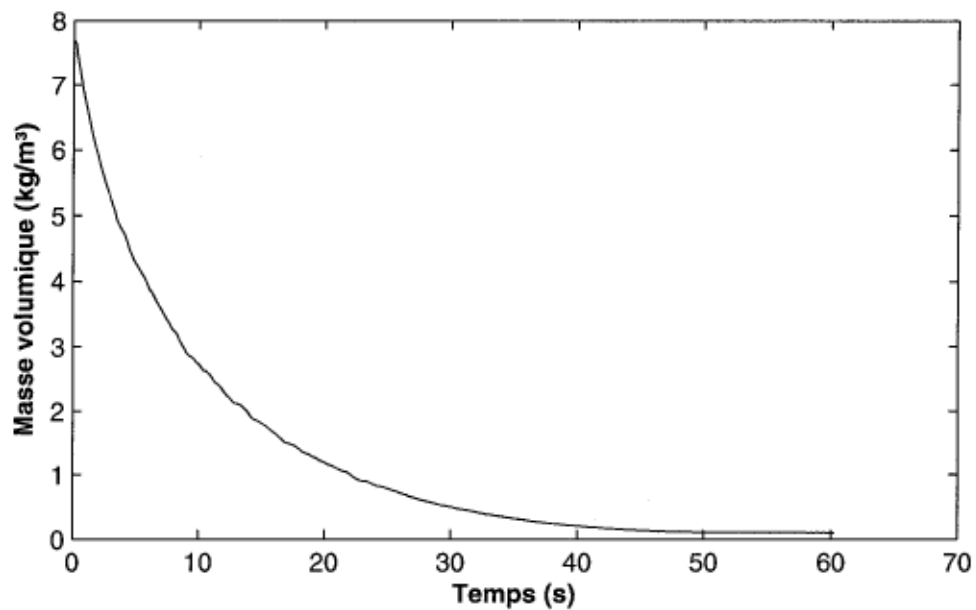


FIG. 3.7 – Masse volumique calculée selon l'algorithme en fonction du temps

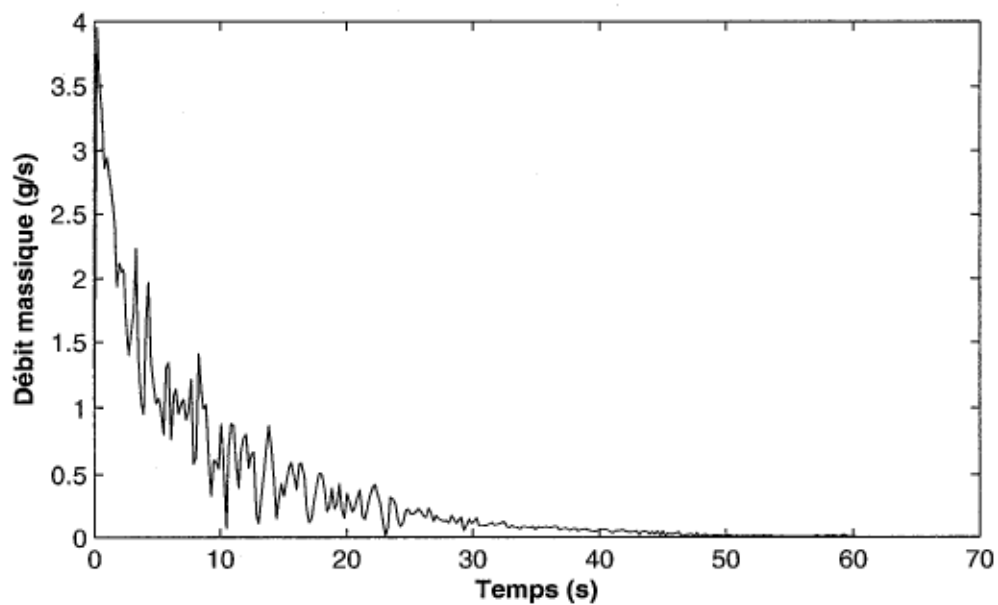


FIG. 3.8 – Exemple de calcul du débit massique à l'aide d'un algorithme de dérivation numérique simple

Chapitre 4

Comparaison

Dans cette section, nous allons comparer les résultats obtenus par simulation avec les données expérimentales disponibles. Nous avons deux sources de données expérimentales. La première est le rapport de l'expérience réalisée par la firme Powertech et la seconde vient, bien sûr, de l'expérience que nous avons réalisé nous-mêmes à l'IRH.

4.1 Powertech

4.1.1 Compatibilité de la simulation et des résultats expérimentaux

Comme on peut le voir sur la figure 4.1, on a réalisé une expérience sur un réservoir avec conduit. Le gaz contenu à l'intérieur du cylindre est de l'hydrogène. On a souhaité mesurer les propriétés du fluide en deux endroits : dans le réservoir et à la sortie du conduit. La figure 4.2 montre l'instrumentation à la sortie du conduit. On peut voir que les instruments ne sont pas directement à la sortie, mais bien légèrement avant. Il est important de noter, pour les besoins de la comparaison, que les propriétés du fluide, à un moment donné dans le temps lors de l'expérience, varient très rapidement selon la position où l'on se trouve dans le conduit vers la fin de celui-ci. En d'autres termes, il y a une chute de pression très importante à la fin du conduit (la température varie elle aussi). La figure 4.3 illustre ce phénomène. Alors, comme il est très difficile de déterminer précisément où se trouvent les instruments, nous avons choisi de n'effectuer la comparaison qu'avec les données provenant des capteurs à l'intérieur du réservoir. Pour ce qui est du débit massique, il ne varie pas selon la position, nous allons donc le considérer également.

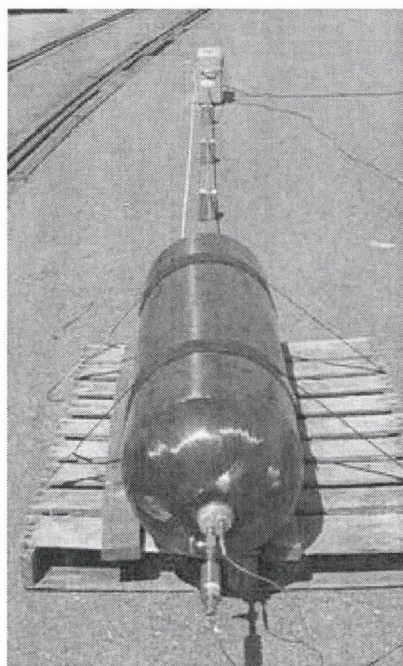


FIG. 4.1 – Montage expérimental de la firme Powertech

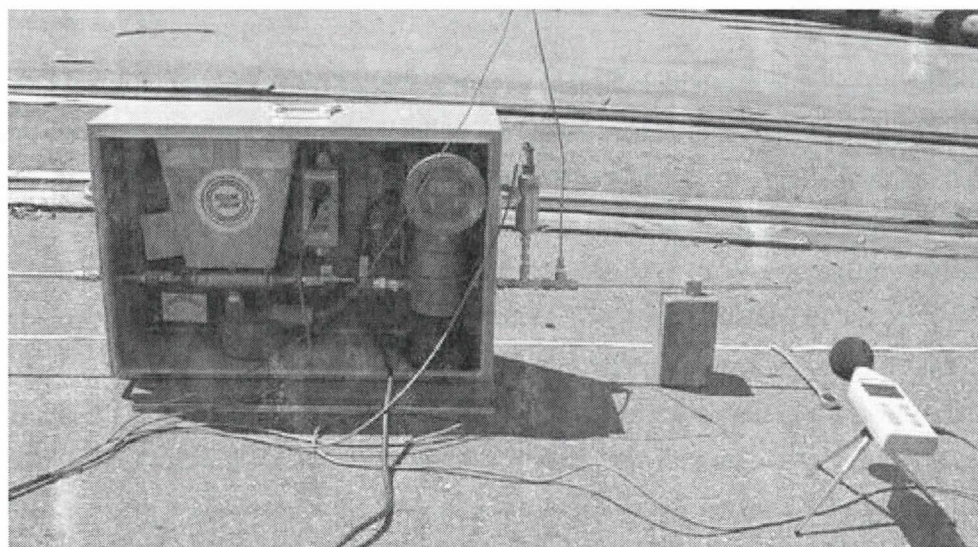


FIG. 4.2 – Instruments près de la sortie du conduit

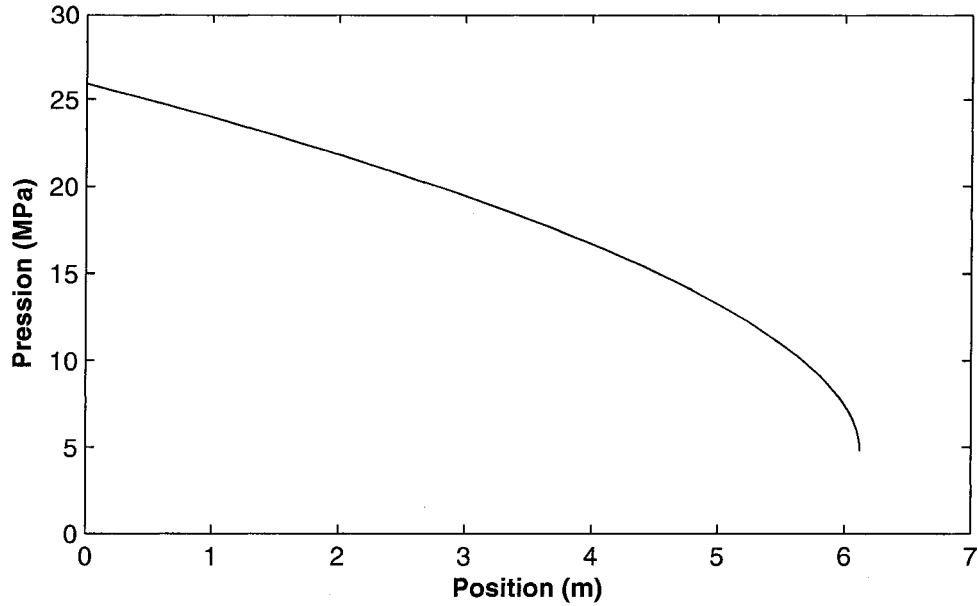


FIG. 4.3 – Pression statique dans le conduit selon la position

4.1.2 Résultats

Les figures 4.4, 4.5 et 4.6 montrent respectivement les courbes expérimentales et simulées pour la pression, la température et le débit massique. On remarque tout de suite une différence très nette entre la simulation et les résultats expérimentaux. Cette différence est due à un phénomène que nous n'avions pas anticipé. Nous avons modélisé le conduit sous l'hypothèse d'un écoulement en régime permanent. Or, il s'avère que pour des conduits ayant un rapport $\frac{L}{D}$ très élevé, comme dans le cas de Powertech, le caractère transitoire du phénomène est non-négligeable. Nous pouvons imaginer qu'au début de l'expérience, le débit massique est fortement restreint par l'inertie du fluide à l'arrêt dans le conduit. Le débit augmente donc graduellement à mesure que la pression arrive à vaincre l'inertie du fluide, jusqu'à ce qu'il atteigne une valeur maximale. Il redescend ensuite suivant la chute de pression dans le réservoir.

Lorsque l'on examine de plus près la courbe expérimentale du débit massique, nous sommes portés à croire que l'écoulement dans le conduit s'approche d'un écoulement en régime permanent après le point d'inflexion suivant le sommet. Cette partie de la courbe ressemble en effet aux courbes que les simulations peuvent donner. Nous nous demandons également à quel point le coefficient de friction utilisé dans le conduit pouvait influencer la précision des résultats. Nous avons donc procédé à plusieurs simulations en utilisant différentes équations de calcul du coefficient de friction [6]. La figure 4.7 montre que le phénomène transitoire ayant lieu dans le conduit reste non-

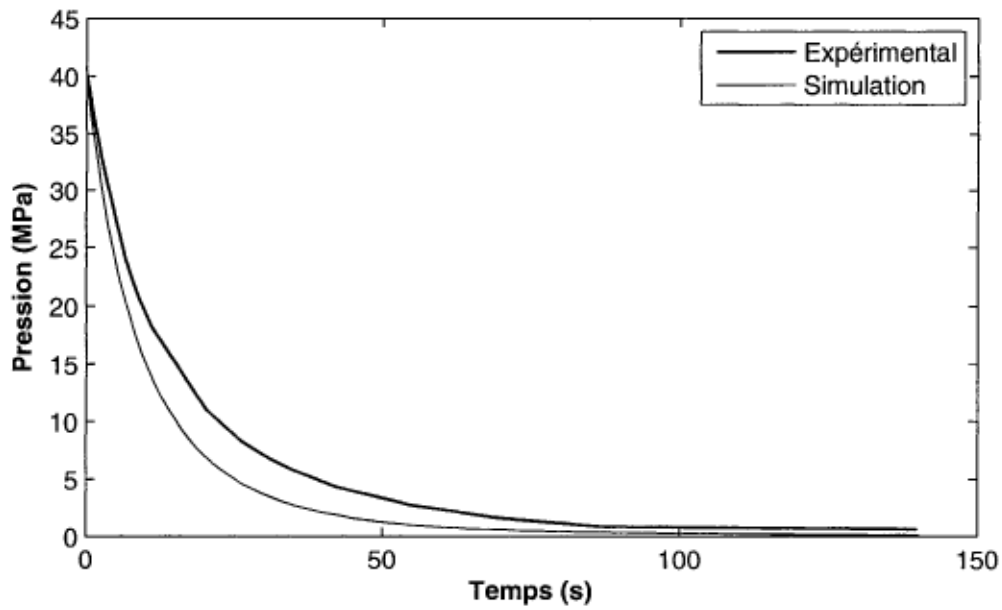


FIG. 4.4 – Powertech : pression expérimentale et simulée

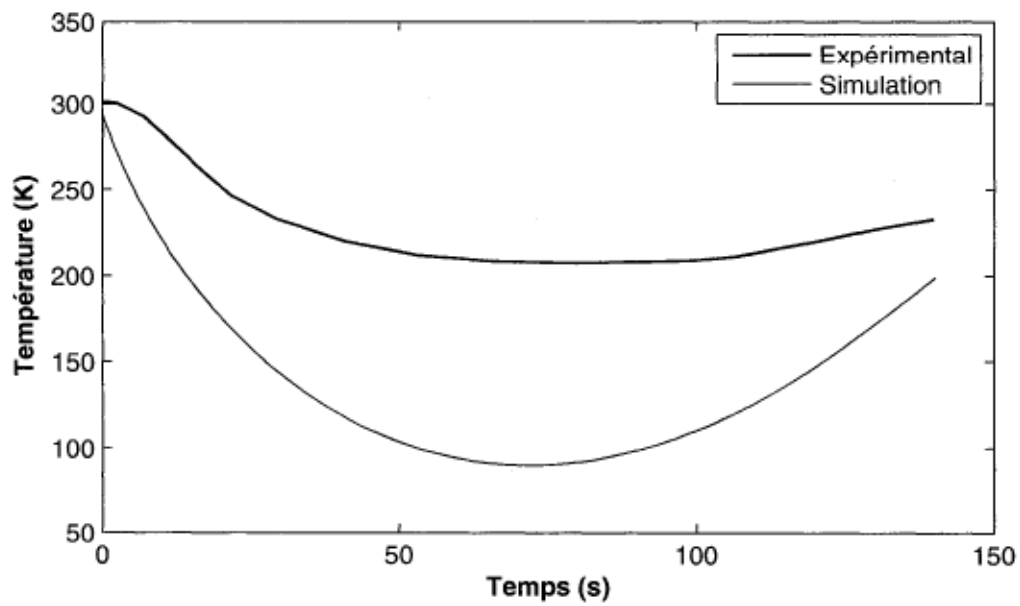


FIG. 4.5 – Powertech : température expérimentale et simulée

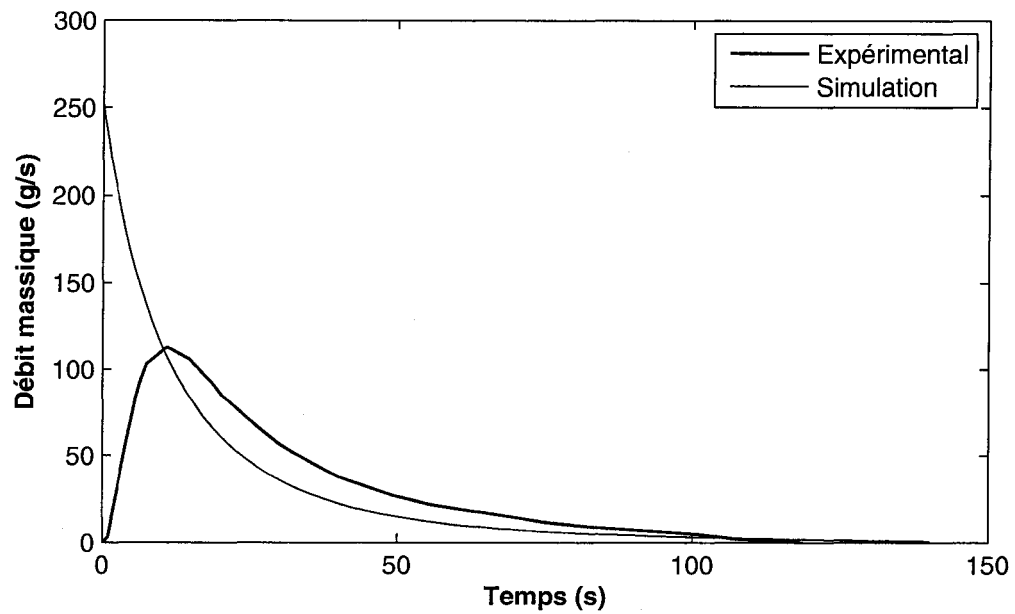


FIG. 4.6 – Powertech : débit massique expérimental et simulé

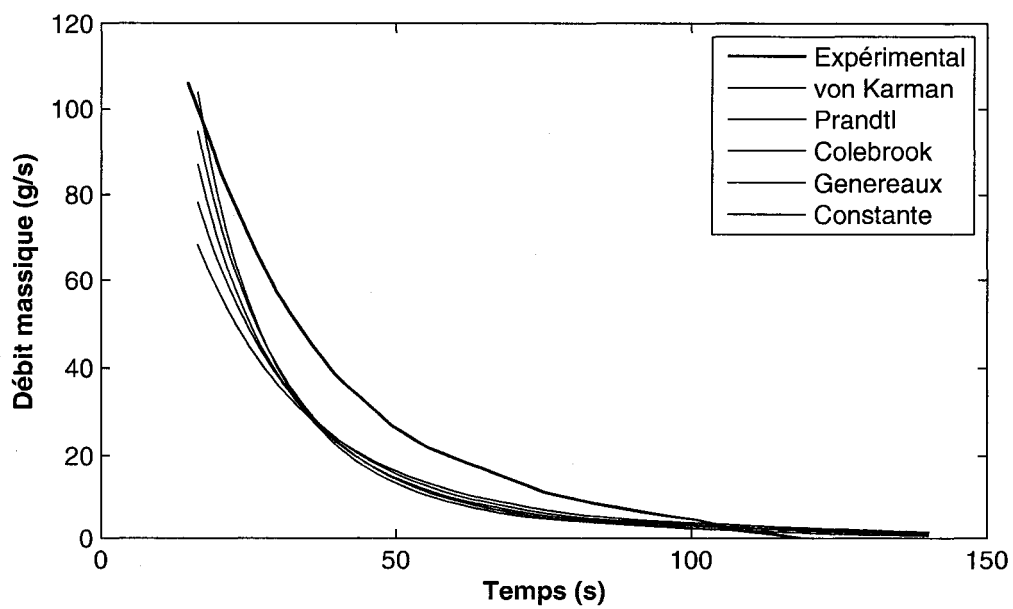


FIG. 4.7 – Powertech : débit massique dans la partie de la courbe au-delà du point d'inflexion suivant le sommet

négligeable tout au long de l'expérience. Notre programme de simulation n'arrive donc pas à reproduire ce comportement, peu importe l'équation utilisée pour le coefficient de friction.

Notre modèle serait donc inapproprié pour des conduits très longs. Dans la prochaine section nous allons voir ce qu'il en est des conduits plus courts et d'un réservoir sans conduit.

4.2 IRH

Comme nous n'avons pas trouvé, dans la littérature, de données expérimentales permettant de valider notre modèle, nous avons décidé de réaliser nos propres expériences. Nous avons testé un réservoir avec des conduits ayant différentes longueurs et différents diamètres externes.

4.2.1 3,175mm \times 1cm

Cette section traite de l'essai réalisé avec un conduit de marque Swagelok d'une longueur de 1cm ayant un diamètre extérieur de 3,175mm et une épaisseur de mur de 0,7112mm. Ce qui représente une aire de sortie de 2.412mm². La pression et la température initiales sont respectivement de 9,9043MPa et de 293.842K.

Pour des raisons pratiques évidentes, la réalisation de l'expérience nécessite une vanne à la sortie du réservoir (figure 4.8). Or, en ajoutant les raccords nécessaires à l'installation de la vanne et de la section de conduit, on se retrouve avec une canalisation d'une longueur de 10cm plutôt que de 1cm. Nous allons d'ailleurs utiliser cette valeur comme paramètre de simulation. La canalisation comporte également plusieurs variations abruptes de diamètre interne. De telles sections convergentes et ensuite divergentes peuvent engendrer des ondes de choc et augmenter la friction, ce qui nuit à l'écoulement. Comme nous en avons discuté à la section 2.2.6, il existe plusieurs équations pour calculer le coefficient de friction de Darcy. Nous allons utiliser l'équation de Prandtl, puisqu'elle donne lieu aux coefficients de friction les plus élevés.

Les figures 4.9, 4.10 et 4.11 montrent les graphiques de comparaison de la pression, de la température et du débit massique.

4.2.2 1,5875mm \times 1cm

Pour la simulation d'un conduit d'un seizième de pouce de diamètre externe et d'un centimètre de longueur, nous avons utilisé un paramètre de longueur de 1cm



FIG. 4.8 – Sortie du réservoir

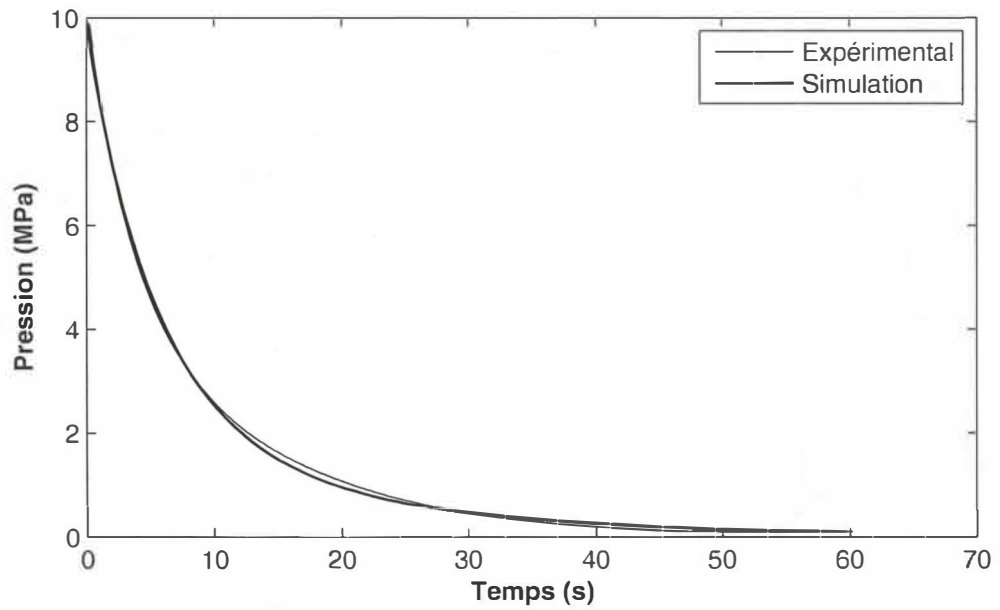


FIG. 4.9 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur totale de 10cm

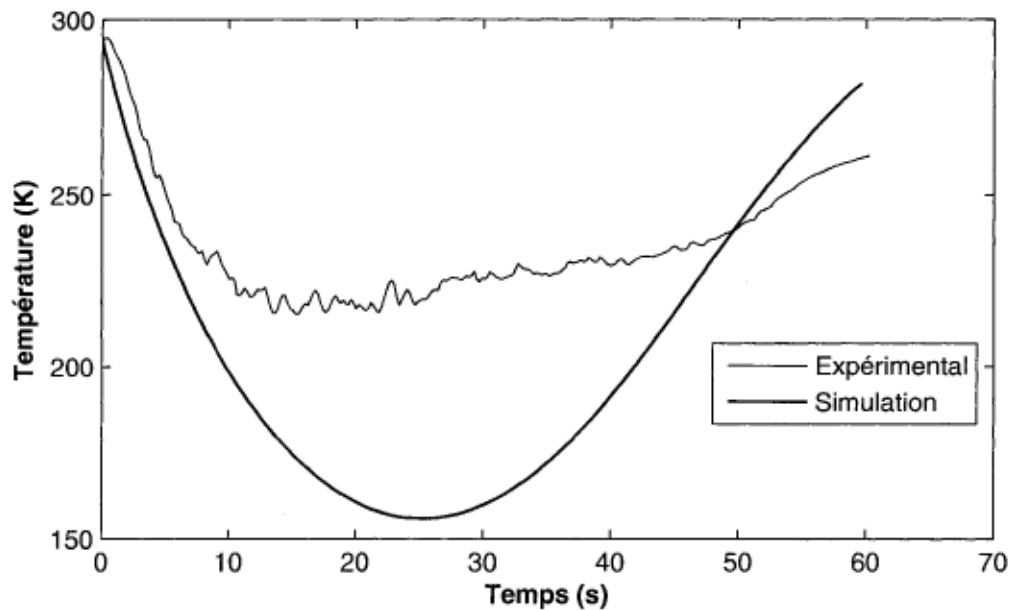


FIG. 4.10 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur totale de 10cm

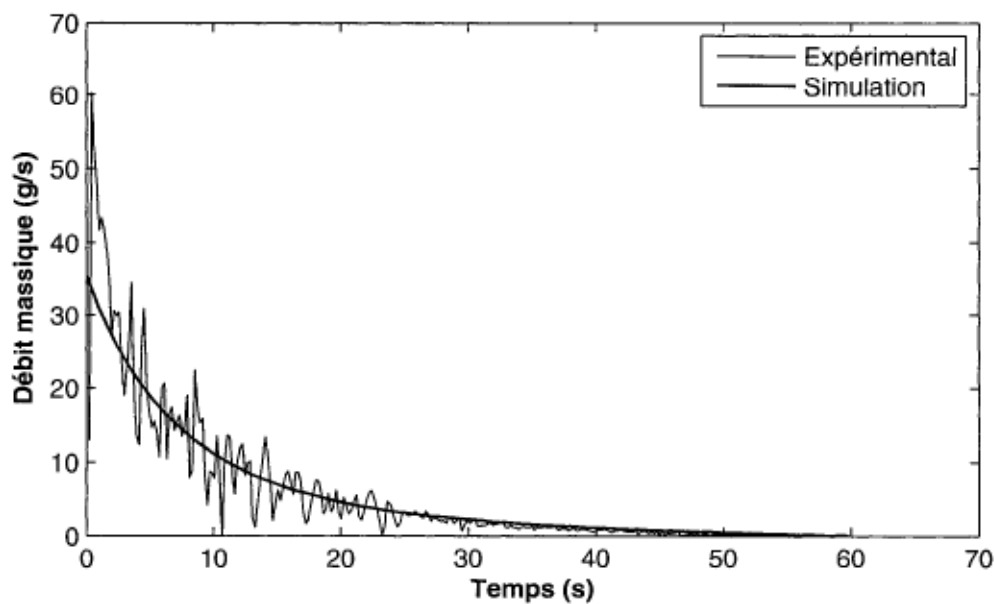


FIG. 4.11 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur totale de 10cm

également puisque le diamètre interne de la canalisation est très grand devant celui du conduit. Celle-ci a donc un effet très restreint. Les variations de diamètre à l'intérieur de la canalisation jouent également un rôle moins important. Le facteur limitant est vraiment le diamètre de la section de conduit. Nous avons donc utilisé, pour calculer le coefficient de friction, la loi de von Karman pour les conduits lisses. Les figures 4.12, 4.13 et 4.14 comparent les résultats expérimentaux à la simulation.

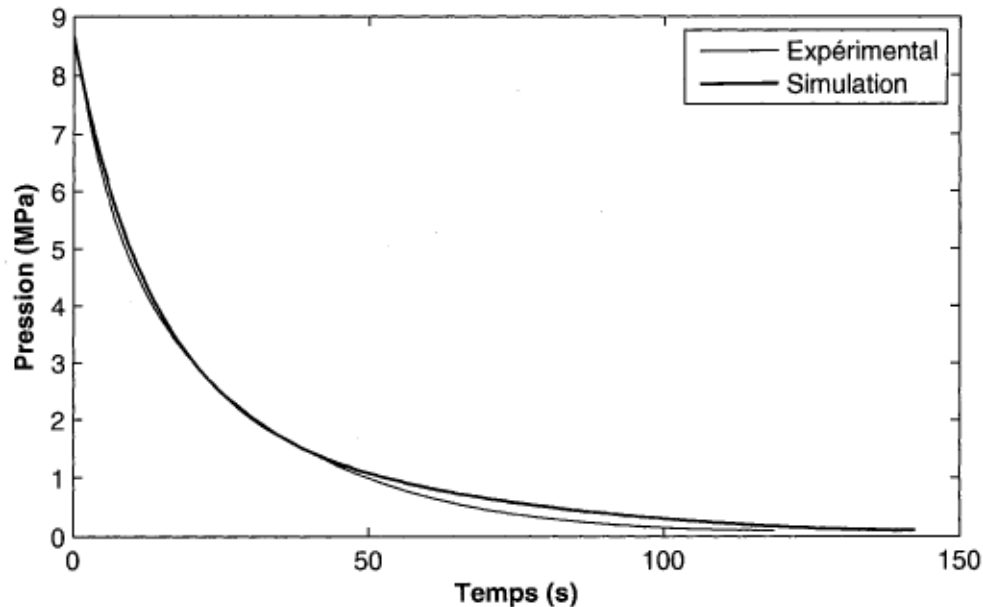


FIG. 4.12 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 1cm

4.2.3 3,175mm × 150cm

Pour cette simulation, nous avons utilisé un paramètre de longueur de 159cm, ce qui correspond à la longueur de la section de conduit additionnée de la canalisation. La loi utilisée pour le frottement est également la loi de von Karman. Les figures 4.15, 4.16 et 4.17 présentent les résultats.

4.2.4 1,5875mm × 152cm

Pour cette simulation, nous utilisons la loi de von Karman ainsi qu'une longueur de conduit de 152cm. Les résultats se trouvent dans les figures 4.18, 4.19 et 4.20.

Nous n'avons présenté, dans cette section, que quatre des 12 essais réalisés. Le reste des résultats présentés sous forme de graphique, figure dans l'annexe A.

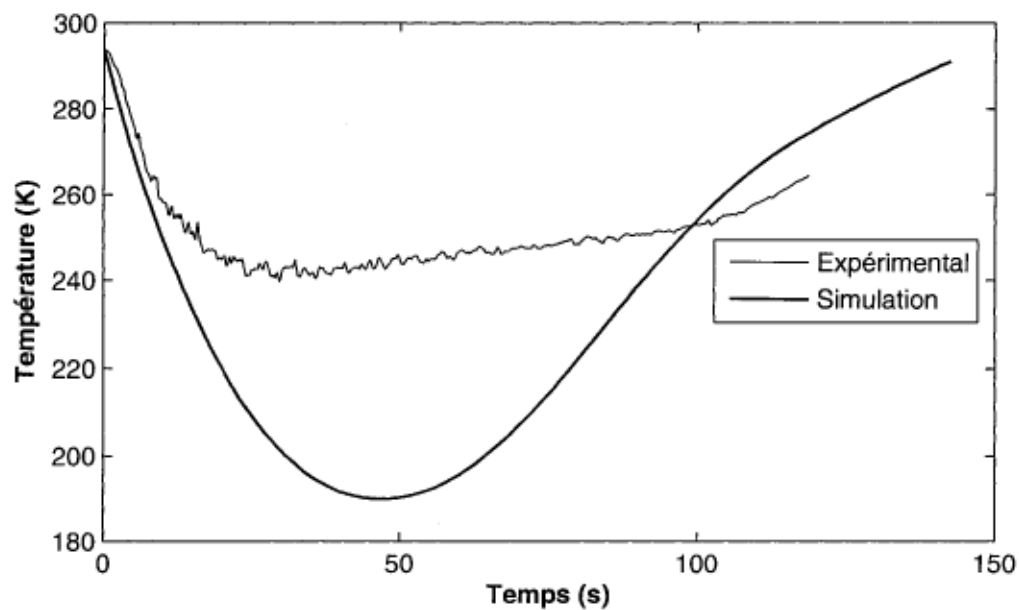


FIG. 4.13 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 1cm

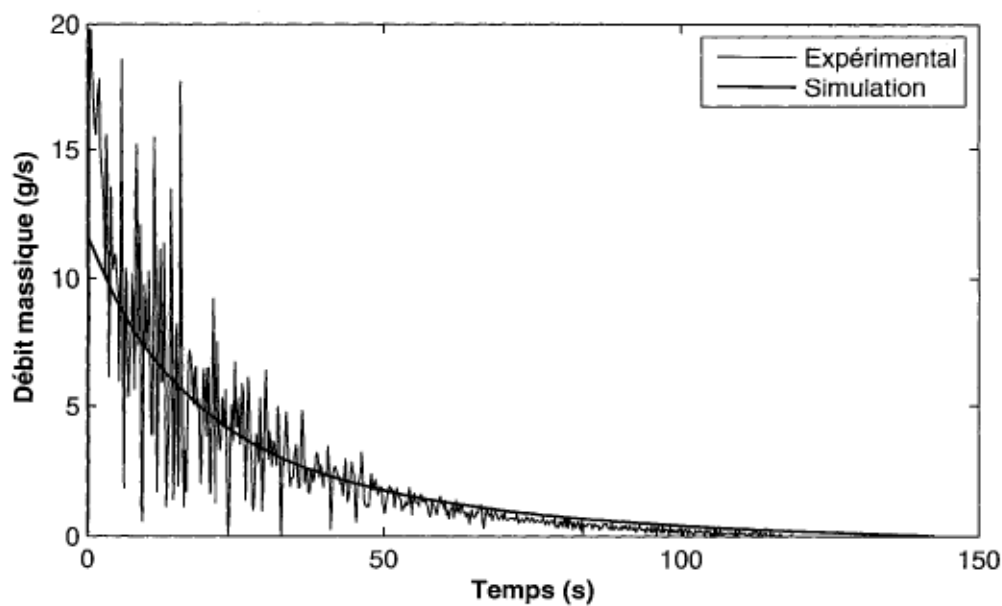


FIG. 4.14 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 1cm

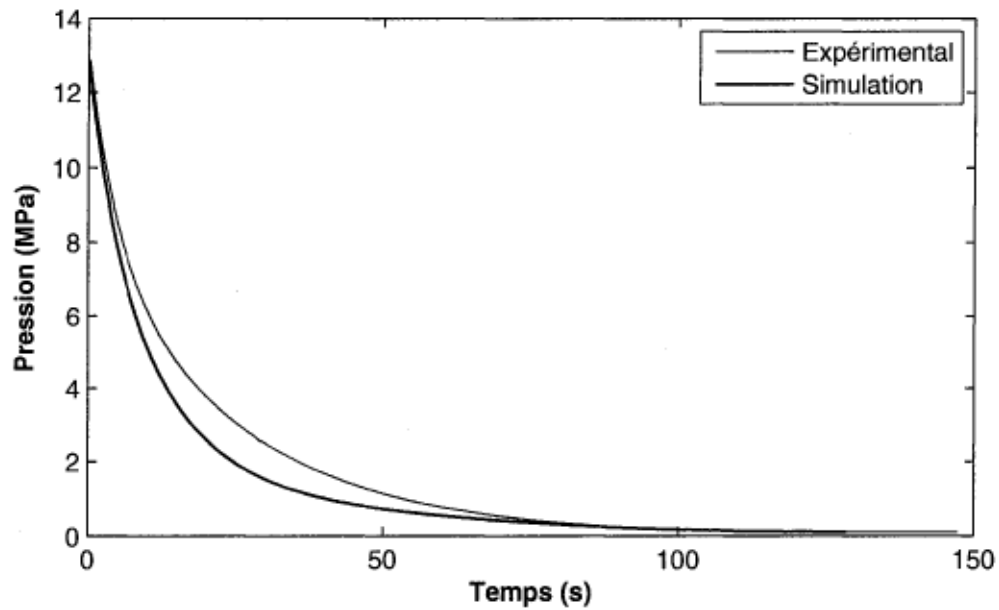


FIG. 4.15 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 150cm

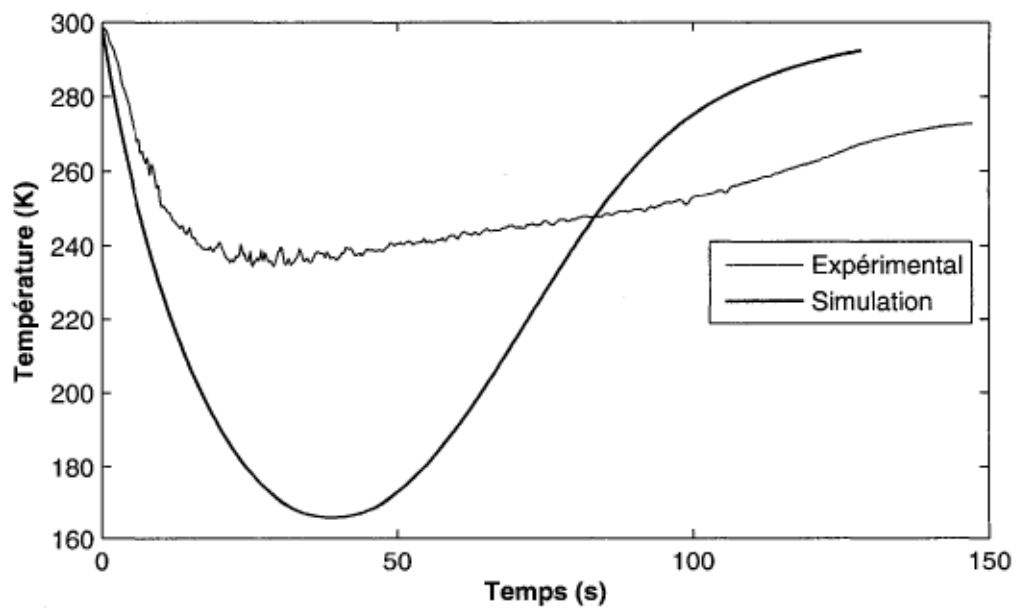


FIG. 4.16 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 150cm

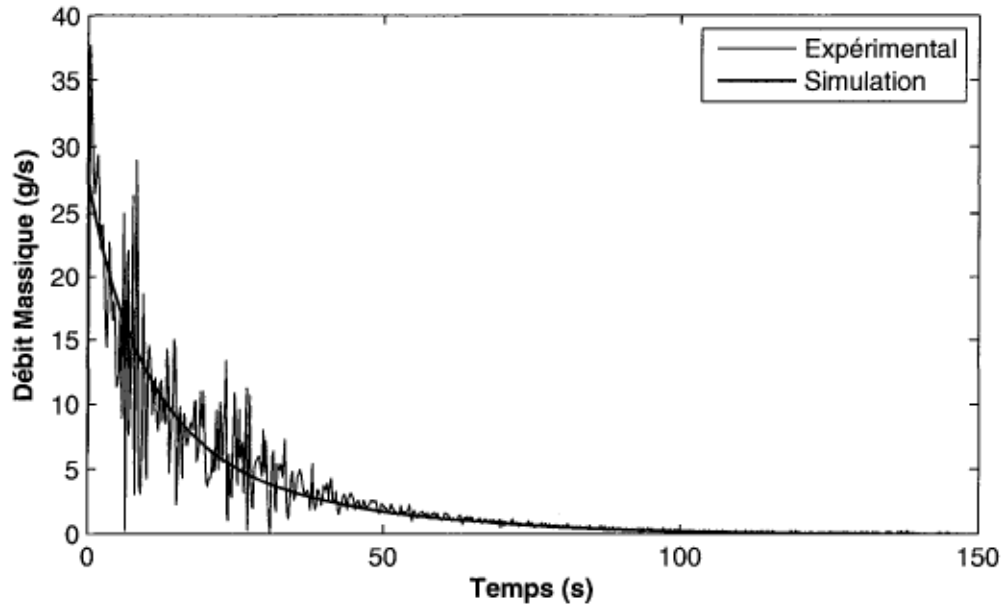


FIG. 4.17 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 150cm

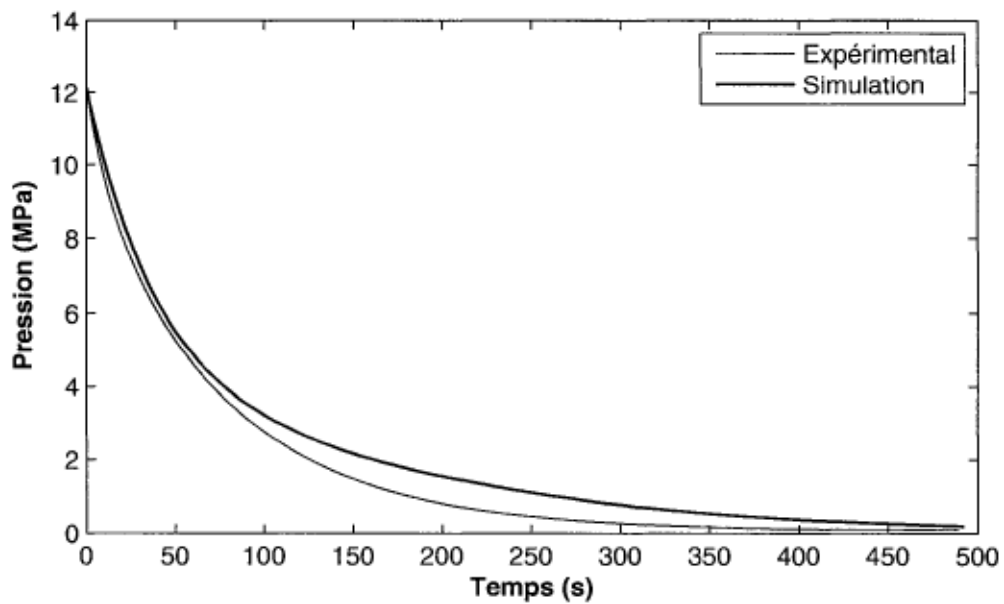


FIG. 4.18 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 152cm

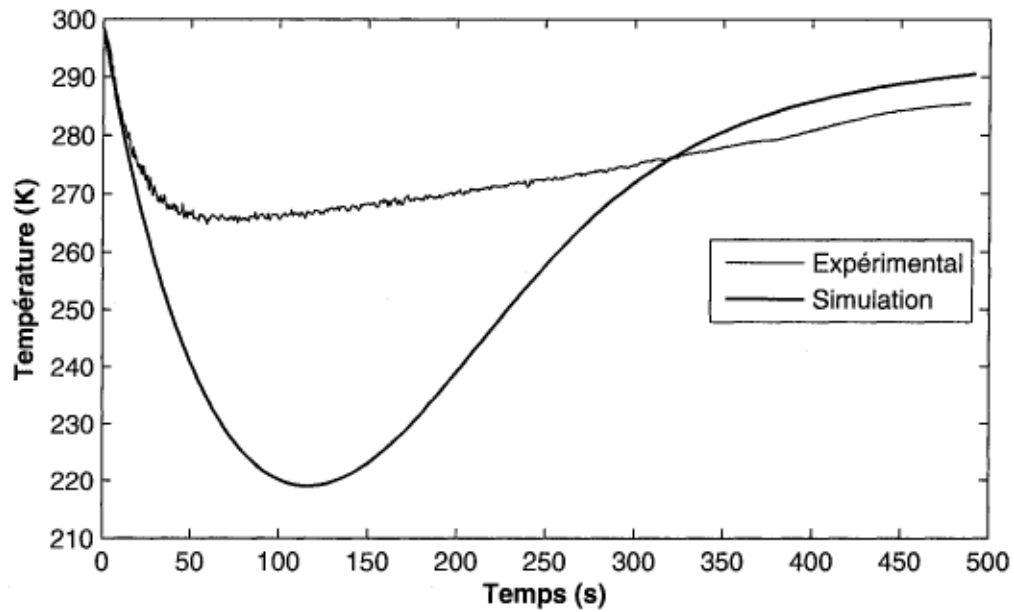


FIG. 4.19 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 152cm

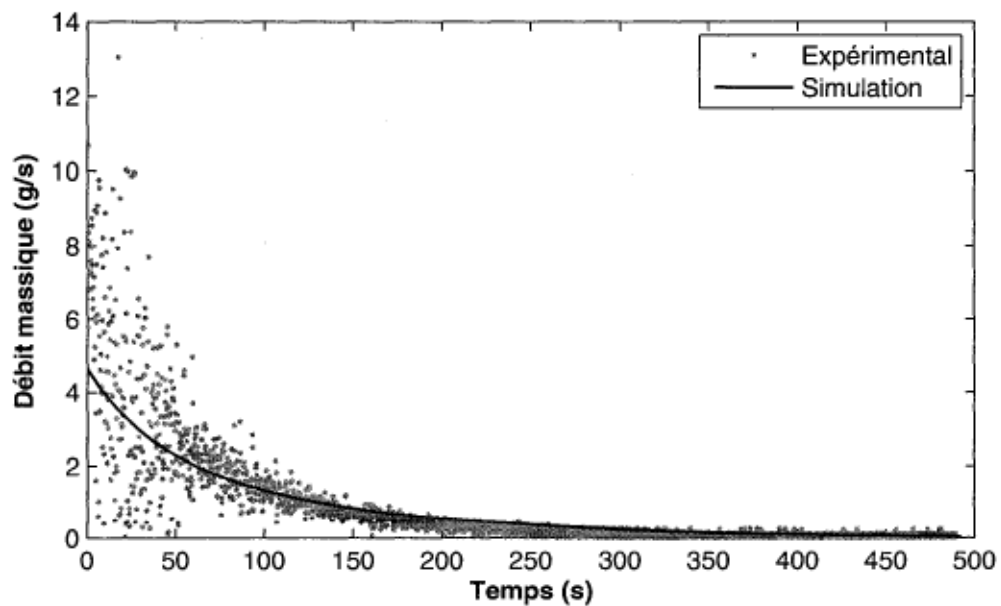


FIG. 4.20 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 152cm

4.3 Discussion

Dans cette section, nous nous attarderons principalement à l'écart en température observé entre les mesures et les données issues des simulations. Nous aborderons également l'aspect de la précision du modèle.

4.3.1 Explication de l'écart en température

On remarque tout de suite une forte différence entre les courbes expérimentales et simulées de température. Nous croyons que cela est principalement dû à la méthode de mesure. La figure 4.21 illustre le montage utilisé lors des essais. On peut y voir que les fils du thermocouple, qui n'ont presque aucune rigidité, sont supportés par une gaine d'acier inoxydable. Il est donc réaliste d'imaginer qu'il s'opère, lorsque le réservoir se refroidit, un transfert de chaleur par conduction du milieu ambiant à la gaine qui se transforme en quelque sorte en une ailette en forme de tige. Le thermocouple mesurerait donc davantage la température du bout de cette ailette que la température du fluide dans lequel elle baigne.

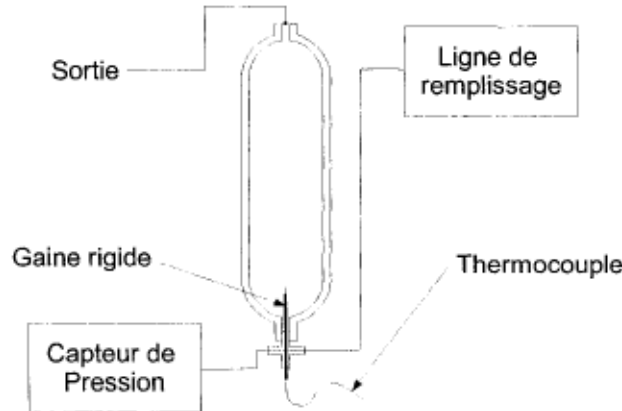


FIG. 4.21 – Schéma de l'instrumentation du réservoir

Afin de vérifier cette hypothèse, nous avons eu recours à un modèle stationnaire d'échange de chaleur pour une ailette-tige tiré de l'ouvrage de Kreith [7]. Les équations du modèle s'élaborent principalement à partir de ce qui est illustré à la figure 4.22.

Un bilan d'énergie permet d'obtenir l'équation suivante

$$\frac{d^2T(x)}{dx^2} - \frac{\bar{h}_c P}{kA} [T(x) - T_\infty] = 0 \quad (4.1)$$

où x est la position le long de la tige, $T(x)$ la température à un endroit donné de la

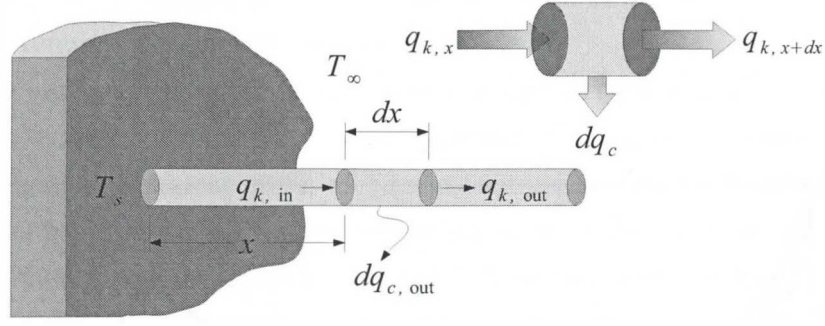


FIG. 4.22 – Élément d'intégration du modèle d'ailette-tige

tige, T_∞ la température du fluide environnant, \bar{h}_c le coefficient de convection, P le périmètre de la tige, k la conductivité thermique et A l'aire d'une section de la tige.

En définissant $\theta(x) = [T(x) - T_\infty]$ et $m^2 = \frac{h_c P}{kA}$, on obtient

$$\frac{d^2\theta}{dx^2} - m^2\theta = 0 \quad (4.2)$$

La solution générale de cette équation est

$$\theta(x) = C_1 e^{mx} + C_2 e^{-mx} \quad (4.3)$$

Les constantes C_1 et C_2 dépendent des conditions frontières utilisées pour résoudre le problème. Dans le cas où température de la base de l'ailette est égale à la température du mur et où l'autre extrémité de l'ailette est isolée, la distribution de température s'exprime par cette équation

$$\frac{\theta}{\theta_s} = \frac{\cosh m(L-x)}{\cosh mL} \quad (4.4)$$

où $\theta_s = T_s - T_\infty$ et T_s est la température à la surface du mur.

Donc l'expression permettant d'évaluer la température au bout de la tige serait la suivante

$$T(L) = \frac{1}{\cosh mL} (T_s - T_\infty) + T_\infty \quad (4.5)$$

Il est raisonnable de penser que la chaleur se transmette très rapidement du mur à l'ailette. Cet hypothèse nous permet de négliger les effets transitoires et d'appliquer l'équation à chaque instant de la simulation afin d'obtenir la courbe de température simulée et corrigée apparaissant à la figure 4.23.

Le tableau 4.1 présente les valeurs utilisées dans l'équation 4.5. On considère

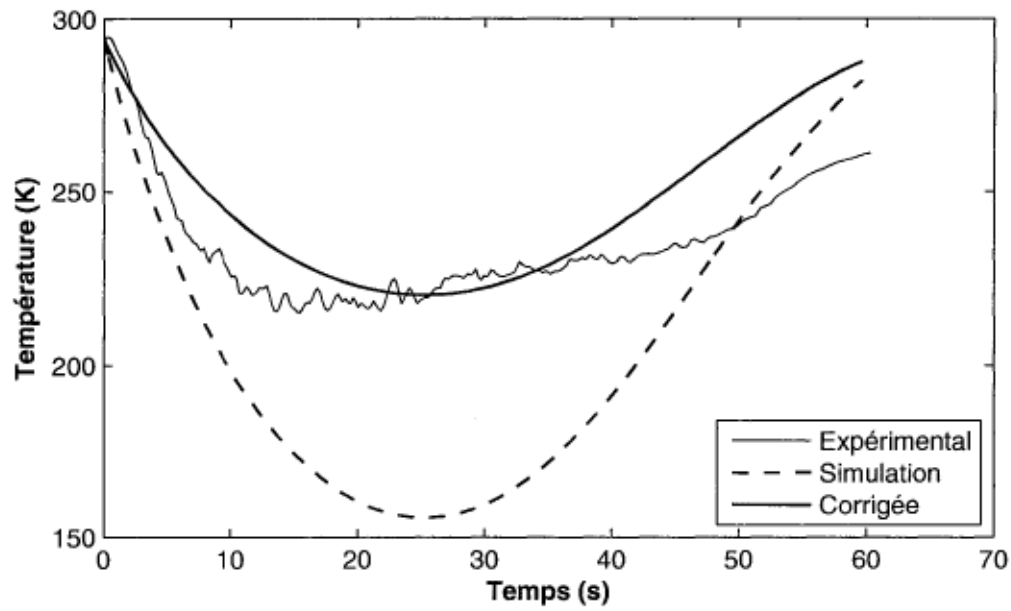


FIG. 4.23 – Courbe corrigée de la température

<i>Paramètre</i>	<i>Valeur</i>
Périmètre P	0,01 m
Aire d'une section de l'ailette A	$7,917 * 10^{-6}$ m ²
Conductivité thermique k	16,2 $\frac{W}{mK}$
Coefficient de convection \bar{h}_c	10 $\frac{W}{m^2K}$

TAB. 4.1 – Paramètres de l'équation de l'ailette-tige

également que T_s est égale à la température ambiante et que T_∞ est égale en chaque instant à la température d'arrêt simulée du fluide.

On note, sur le graphique, que la courbe corrigée se trouve au dessus de la courbe expérimentale. Ceci concorde avec notre hypothèse de négliger les effets transitoires. Comme l'échange de chaleur réel n'est pas instantané, le bout de l'ailette se réchauffe plus lentement en réalité. Un autre facteur pouvant expliquer ce résultat est que la jonction du thermocouple excède légèrement le bout de l'ailette et baigne à nu dans le fluide, elle est donc nécessairement plus froide. Finalement, notre modèle ne tient pas compte du fait que le réservoir, donc le mur se lequel l'ailette est fixée, se refroidit avec le temps, ce qui influence également la température ressentie par le thermocouple.

Nous croyons toutefois que notre hypothèse de transfert de chaleur entre la paroi du réservoir et le thermocouple explique très bien l'écart en température observé puisque l'ordre de grandeur de la courbe corrigée correspond à celui de la courbe expérimentale.

Afin de réduire cette source d'erreur au minimum ou de l'enrayer, il faudrait qu'une longueur plus importante de thermocouple baigne dans le fluide dont on veut mesurer la température. En effet, le thermocouple se comportant comme un ailette-tige, si la longueur est suffisamment longue pour qu'elle apparaisse infinie devant le diamètre, la température au bout du thermocouple sera celle du fluide dans lequel il est plongé. Il faudra toutefois s'assurer que le fil du thermocouple ne touche pas à la paroi. Une solution à ce problème serait de fixer le thermocouple le long de la surface extérieure d'une tige rigide faite d'un matériau isolant.

4.3.2 Précision du modèle

Nous avons observé que la précision du modèle est moins bonne dans le cas de conduits longs, mais pas autant que le laisse croire la comparaison avec les données expérimentales de Powertech. L'essai réalisé avec un conduit ayant un diamètre extérieur de $\varnothing 1,5875\text{mm}$ et une longueur de 152cm ne montre pas un effet très prononcé d'augmentation du débit au début de l'essai. Et pourtant, la longueur relative du conduit est encore plus élevée que celle du conduit utilisé par Powertech.

Conclusion

Bien que les résultats des simulations présentent un écart en température important en comparaison avec les données expérimentales, nous avons montré que cet écart est dû bien davantage à la méthode de mesure qu'au modèle lui-même. L'application du REFPROP du NIST à un programme tel que le nôtre ainsi que l'addition d'un conduit au réservoir sont sans précédent. Nous avons pu constater que l'utilisation du REFPROP du NIST améliore considérablement, de l'ordre de 10% à 15%, les données obtenues par simulation.

Il n'en demeure pas moins que le modèle n'est pas totalement fidèle à la réalité. Cet écart est dû aux différentes hypothèses simplificatrices utilisées. Notamment, le modèle de réservoir à zéro dimensions, la modélisation du conduit en régime permanent et le modèle de transfert de chaleur en régime permanent. Toutefois, l'objectif ultime du programme développé qu'est celui du développement de standards de sécurité, relève de l'ingénierie. Or, un des principes sur lesquels repose cette discipline consiste à négliger certains aspects d'un problème afin de le rendre soluble. Dans cette optique, une utilisation prudente par un individu expérimenté du programme développé peut contribuer à l'atteinte de cet objectif.

Afin d'améliorer encore davantage la précision du programme, il serait possible de tenir compte du caractère transitoire du phénomène ayant lieu dans le conduit. Toutefois nous obtiendrions un modèle à deux variables indépendantes, ce qui est considérablement plus difficile à solutionner. Il faudrait en fait faire appel à une grille de résolution par différence finie et appliquer le modèle de réservoir comme condition frontière. Cela, ainsi que le phénomène inverse, le remplissage d'un réservoir, pourrait faire l'objet de futurs travaux de recherche.

Annexe A

Graphiques supplémentaires

Cette annexe contient les graphiques tracés à partir des données expérimentales et simulées correspondant aux différents essais réalisés. Ces figures n'ont pas été incluses dans le texte principal par souci de clarté.

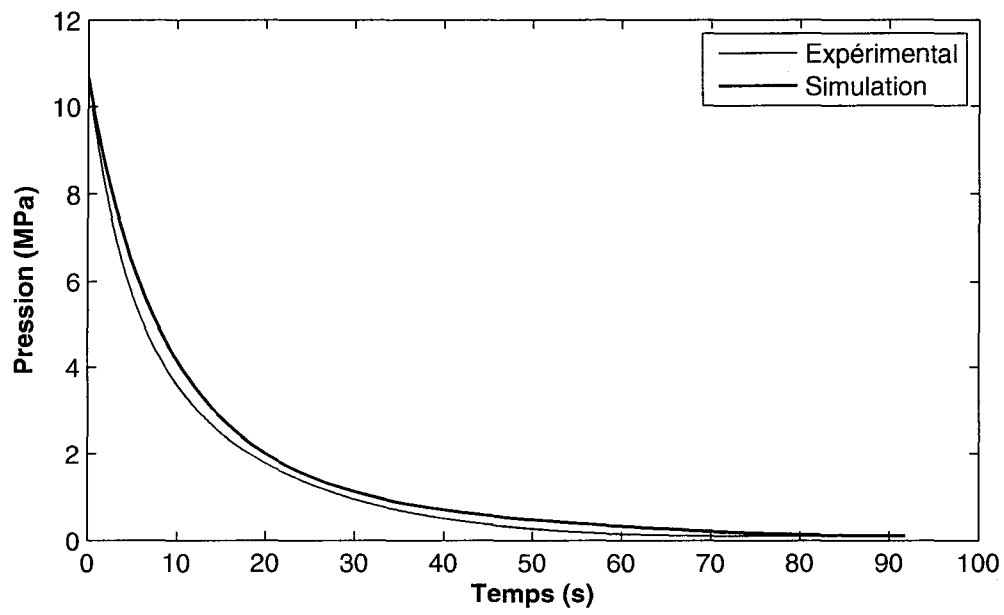


FIG. A.1 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 30cm

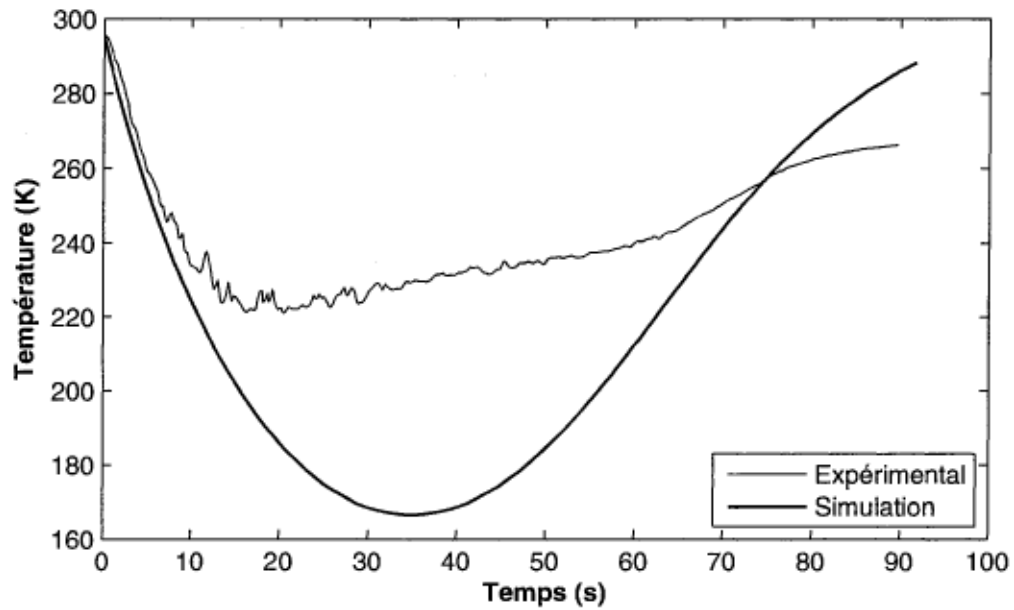


FIG. A.2 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 30cm

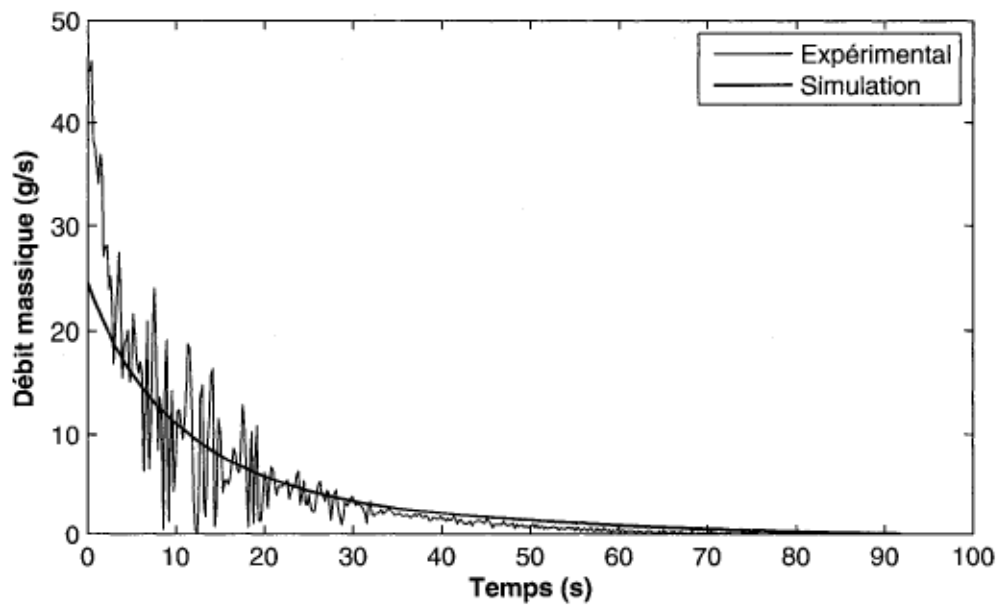


FIG. A.3 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 30cm

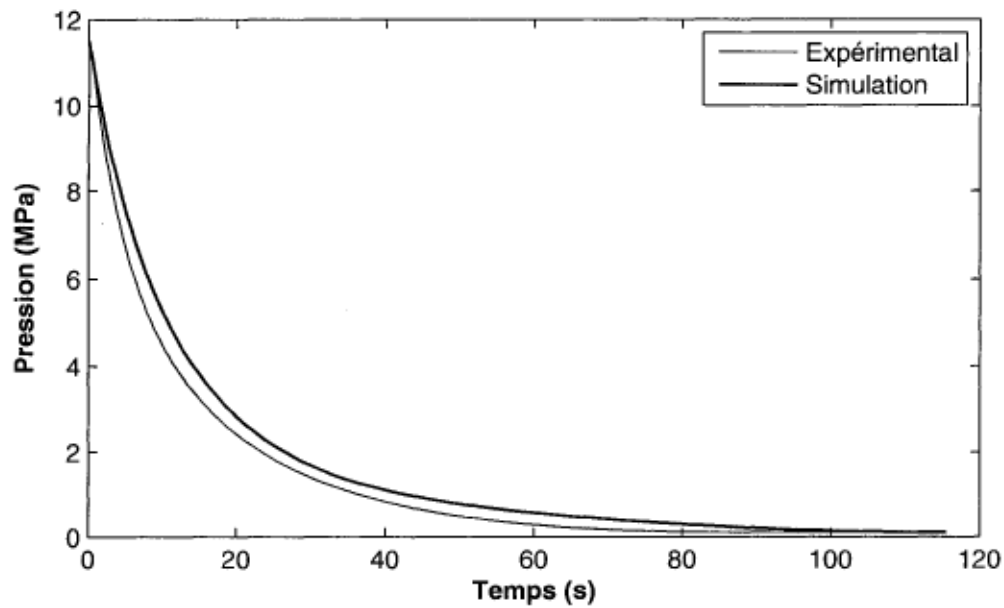


FIG. A.4 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 60cm

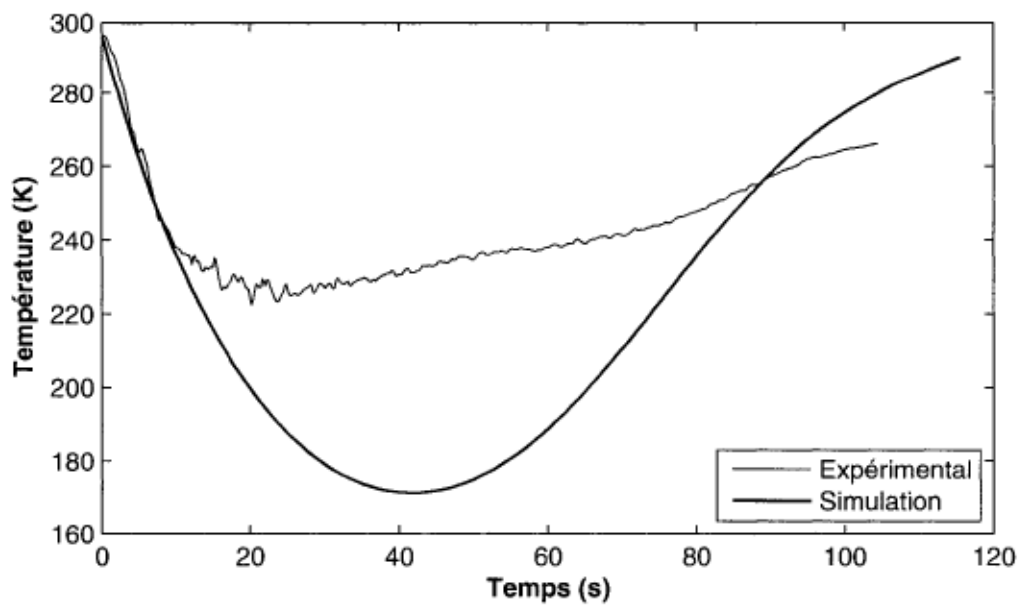


FIG. A.5 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 60cm

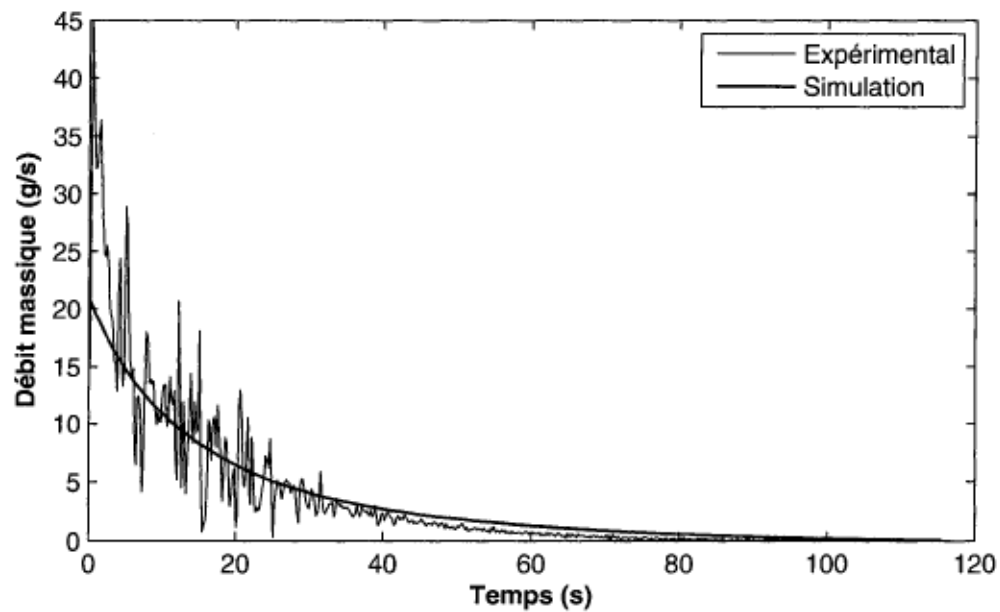


FIG. A.6 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 60cm

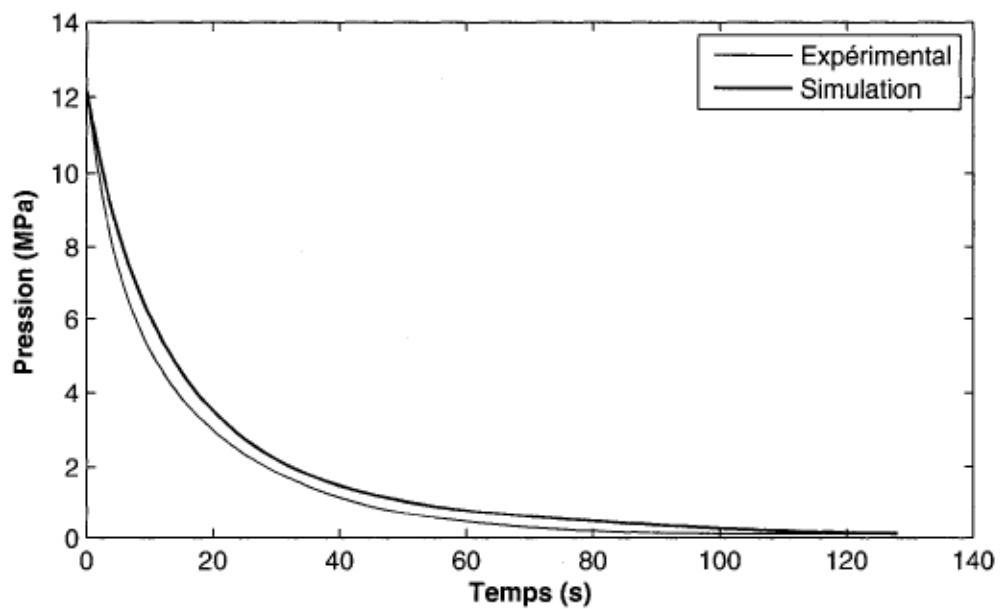


FIG. A.7 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 90cm

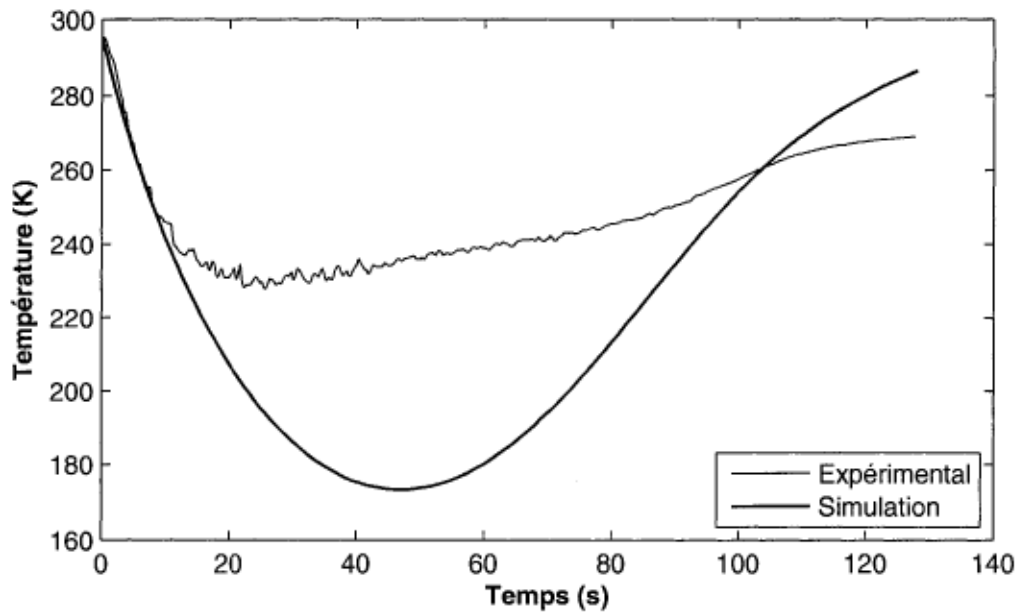


FIG. A.8 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 90cm

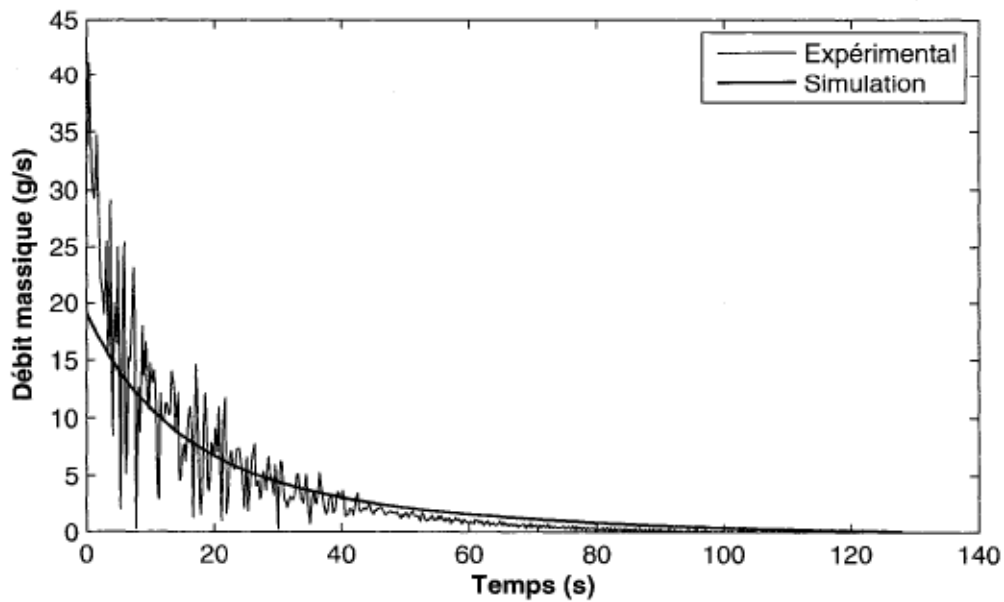


FIG. A.9 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d’une longueur de 90cm

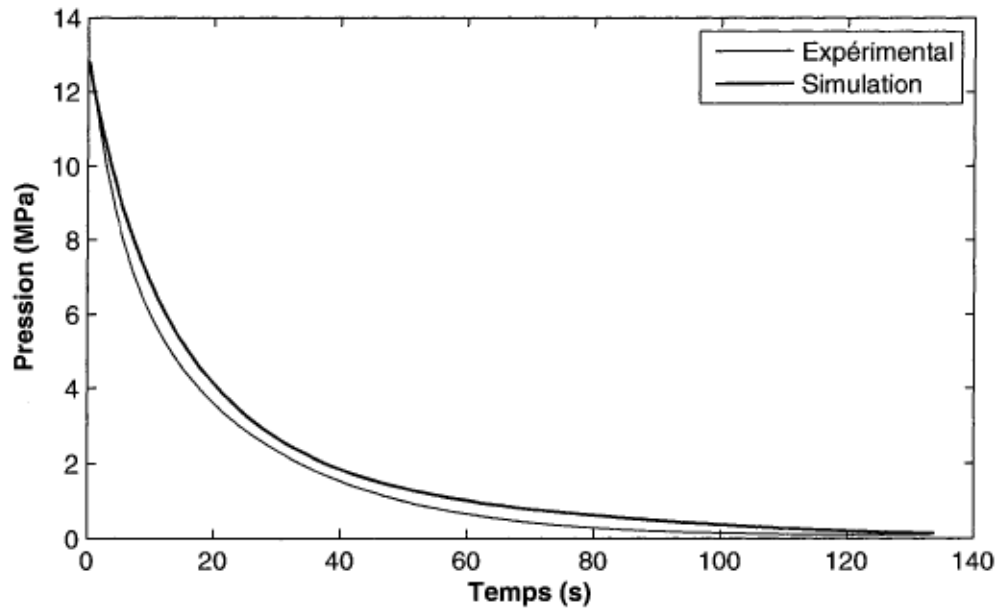


FIG. A.10 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 120cm

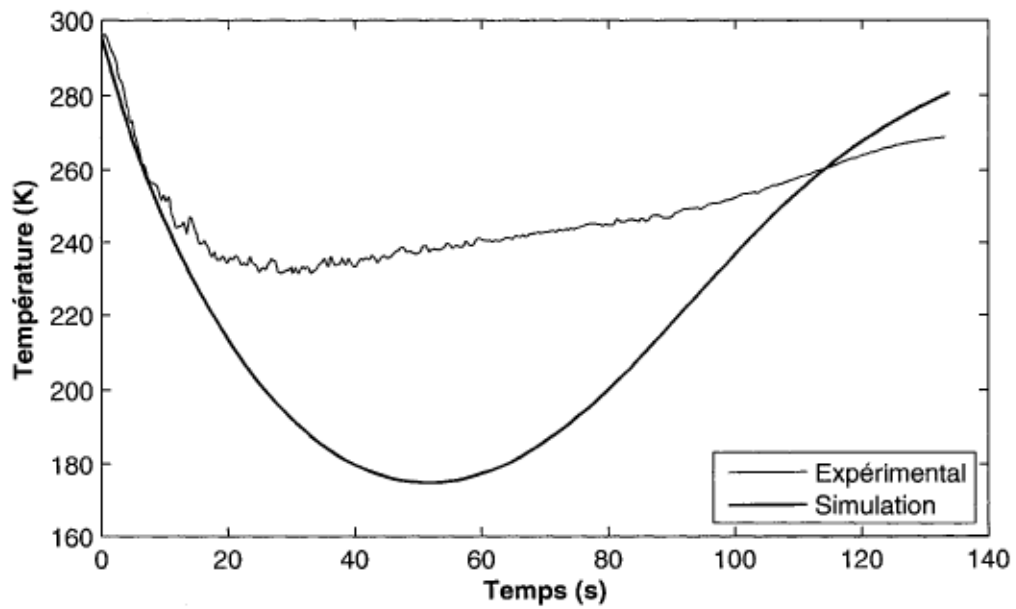


FIG. A.11 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 120cm

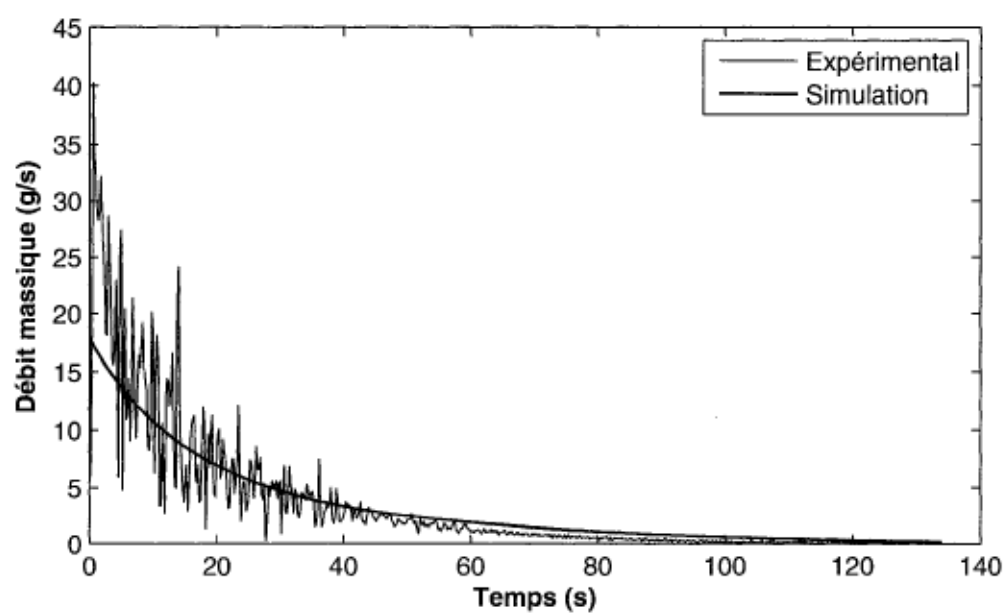


FIG. A.12 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 3,175mm et une canalisation d'une longueur de 120cm

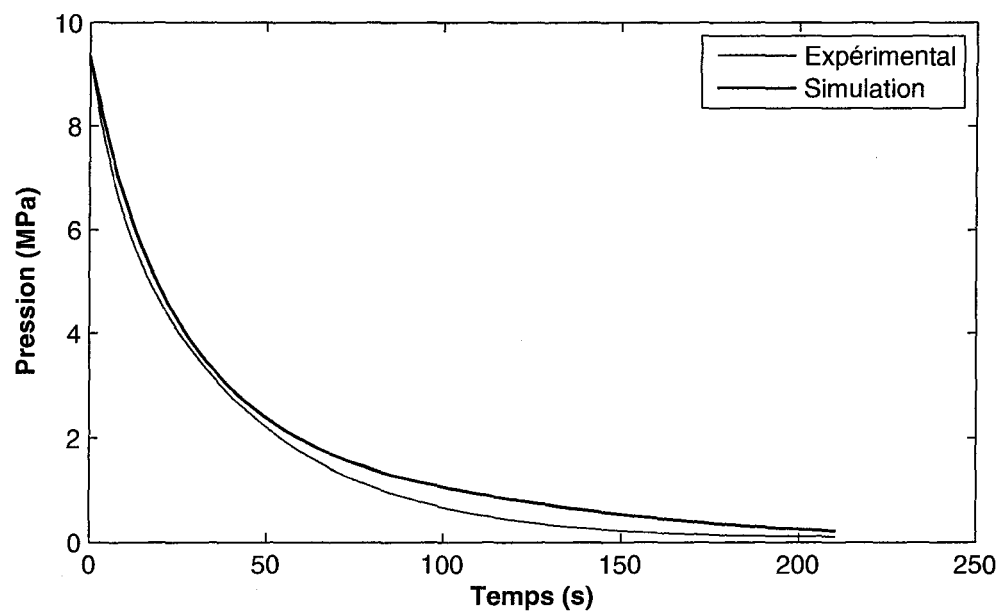


FIG. A.13 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 25cm

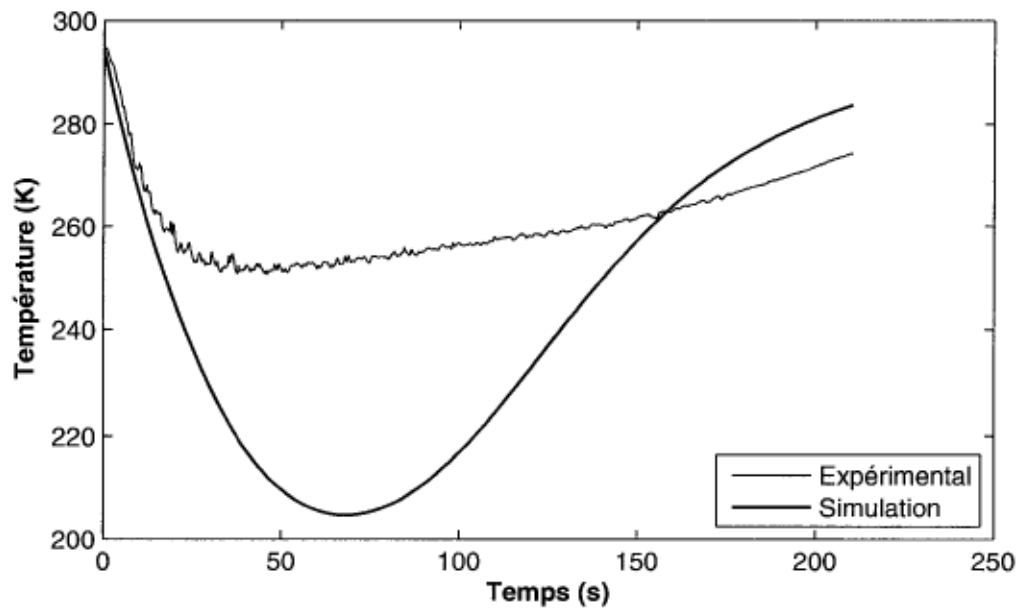


FIG. A.14 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 25cm

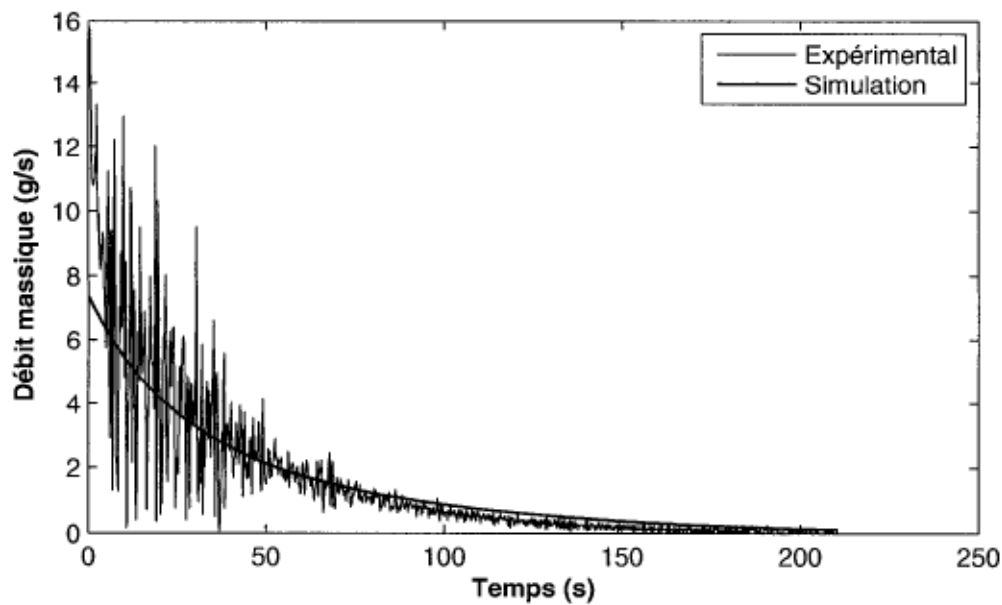


FIG. A.15 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 25cm

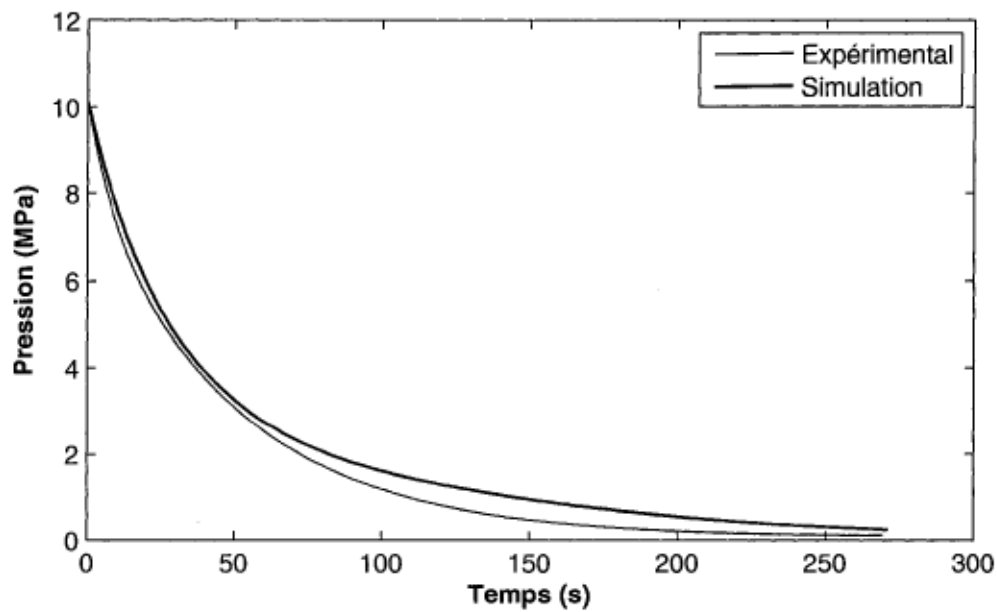


FIG. A.16 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 50cm

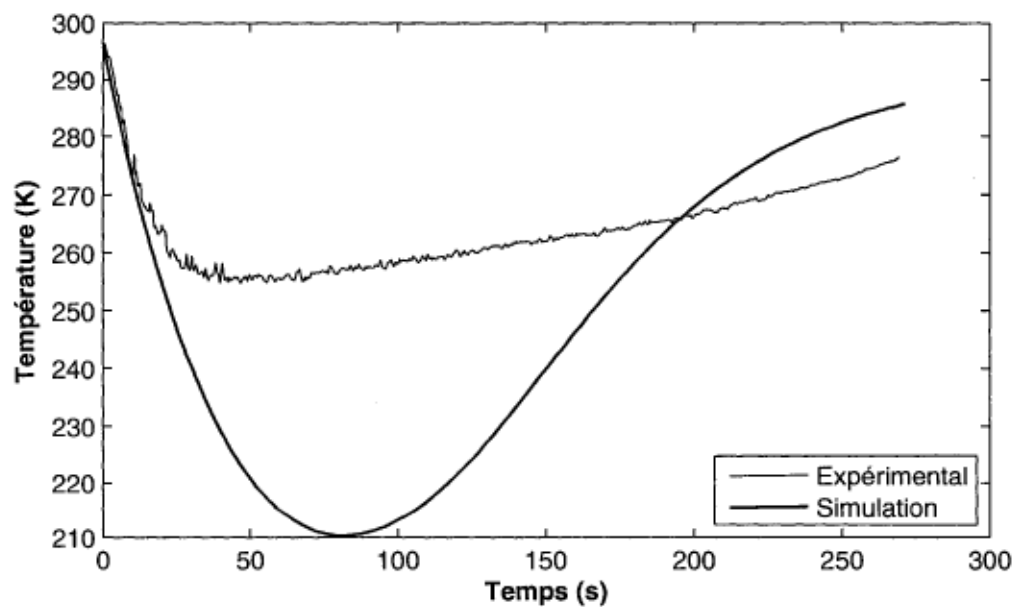


FIG. A.17 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 50cm

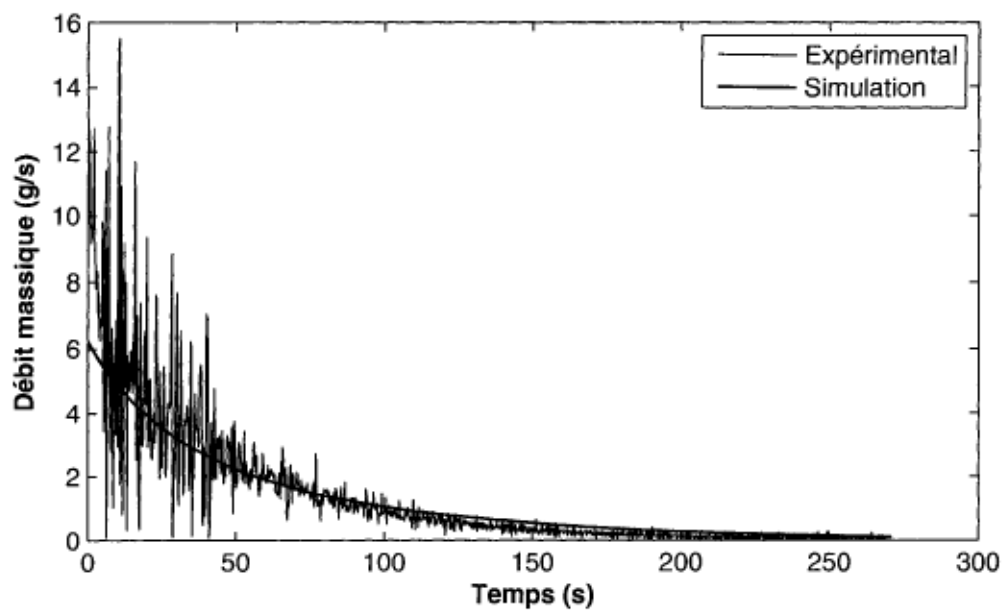


FIG. A.18 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 50cm

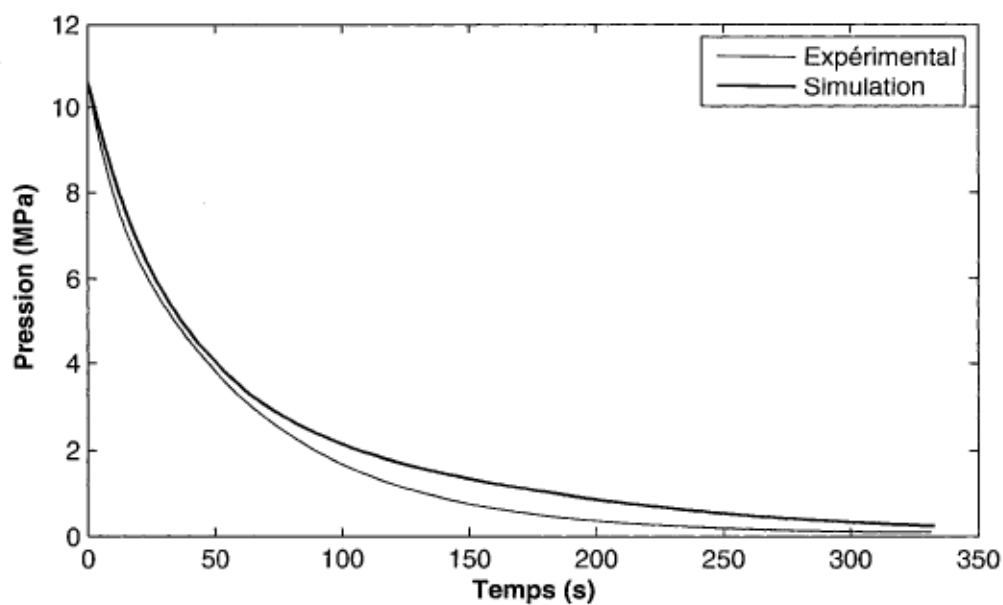


FIG. A.19 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 80cm

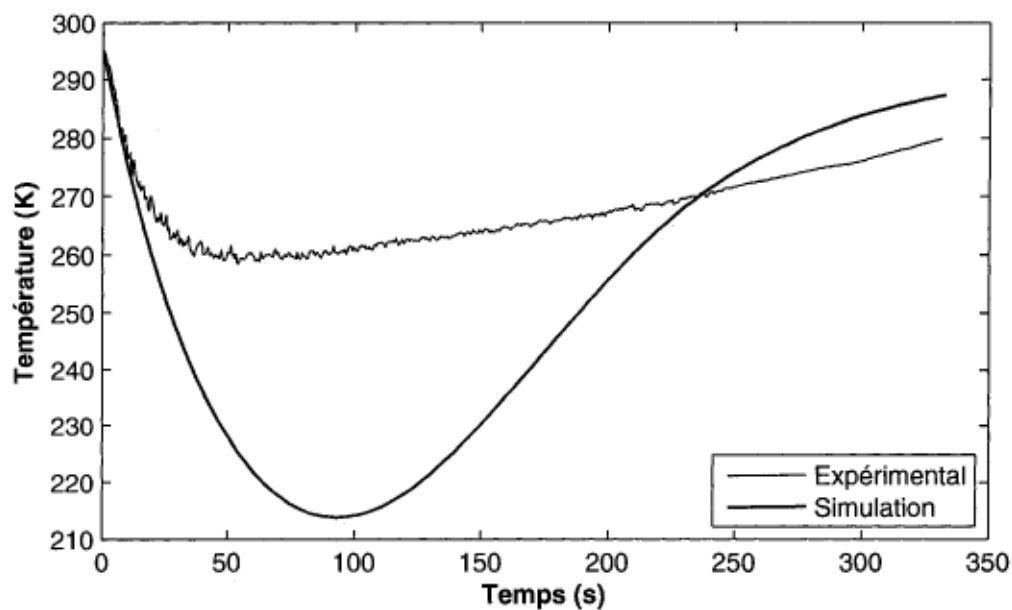


FIG. A.20 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 80cm

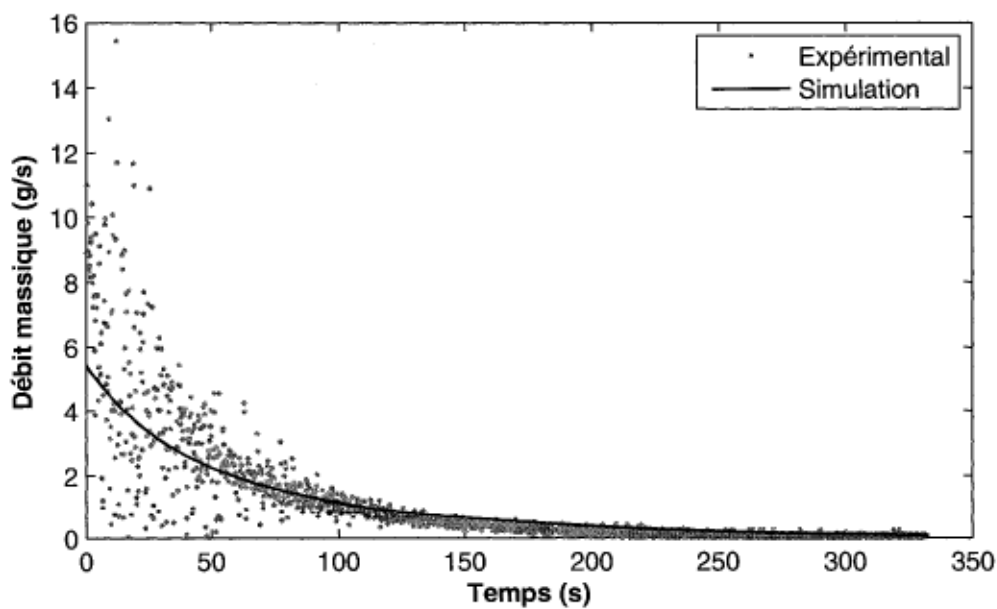


FIG. A.21 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d’une longueur de 80cm

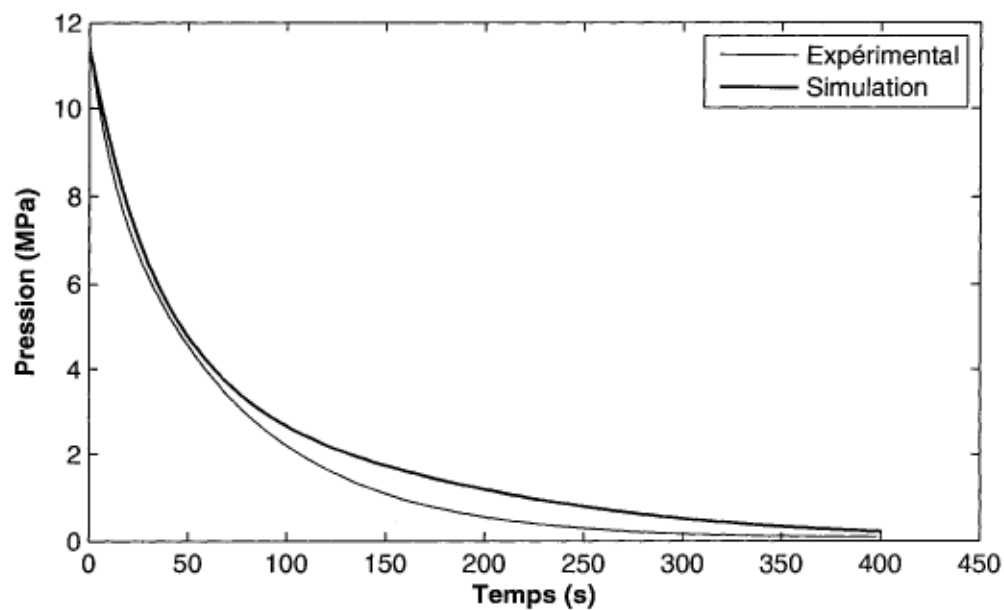


FIG. A.22 – Pression expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 110cm

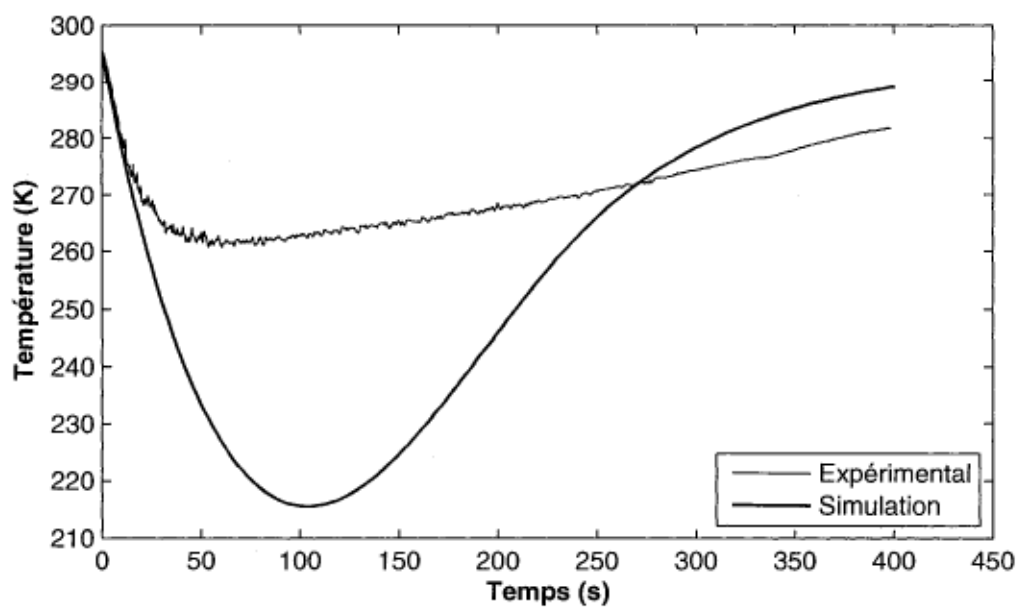


FIG. A.23 – Température expérimentale et simulée pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 110cm

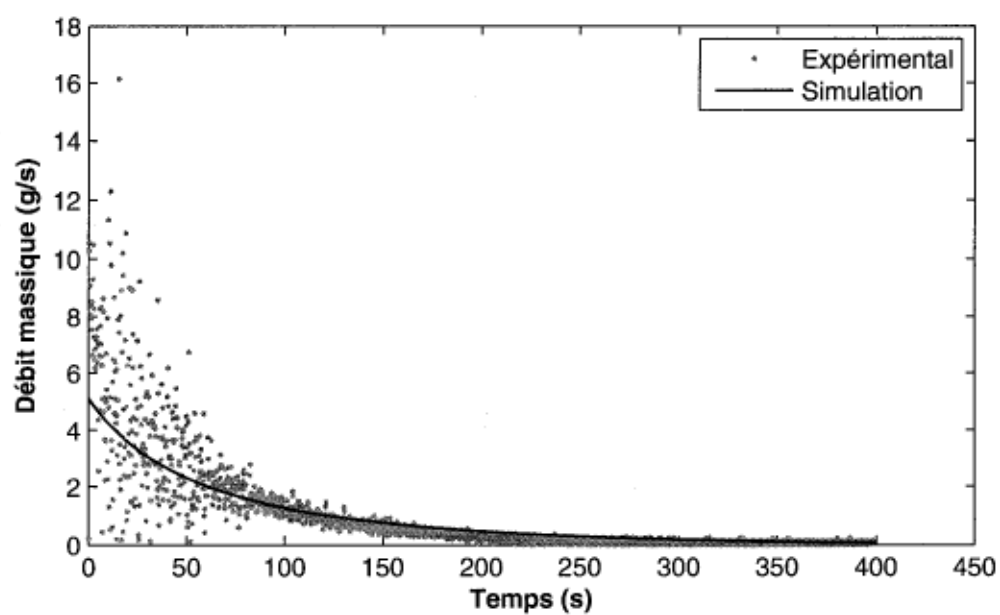


FIG. A.24 – Débit massique expérimental et simulé pour un conduit ayant un diamètre externe de 1,5875mm et une canalisation d'une longueur de 110cm

Annexe B

Listing

Cette annexe contient le code source complet du programme de simulation écrit en langage «Visual Basic .NET».

Module Conduit

```

Dim TooFar As Boolean
Function Lmax(ByVal Pe As Double, ByVal Te As Double, ByVal Ve As Double, _
Optional ByRef Ps As Double = 0, Optional ByRef Ts As Double = 0, _
Optional ByRef Vs As Double = 0) As Double
    'Cette fonction calcule la longueur de conduit requise pour atteindre M=1 ou P=Patm
    'en fonction des propriétés et de la vitesse à l'entrée. Elle calcule également
    'les propriétés et la vitesse du fluide à la sortie du conduit
    Static TimeUsed As Long = 0
    TimeUsed = TimeUsed + 1
    Dim cntr As Double = 0
    Dim assertion As Boolean = True
    Dim x, P, T, V, h, ER, M As Double
    Dim lastx, lastP, lastT, lastV, lasth, lastM As Double
    Dim upperx As Double
    Dim h0 As Double = Param.h0conduit
    Dim busted As Boolean = False
    'Contrôle des données
    If Pe <= Param.Patm Then
        System.Diagnostics.Debugger.Break()
    End If
    'Intégration
    'Initialisation des variables d'intégration
    x = 0
    P = Pe
    T = Te
    V = Ve
    h = h0
    M = Ve / SoSPT(Pe, Te)
    If M >= 1 Then
        Lmax = 0
    ElseIf M <= 0 Then
        MsgBox("Conduit de longueur infinie")
    Else
        Output.Header(IntLmax, "x", "P", "T", "V", "M", "h", "ER")
        Do
            'Mise en mémoire des dernières valeurs
            lastx = x
            lastP = P
            lastT = T
            lastV = V
            lasth = h
            lastM = M
            Output.Save(IntLmax, x, P, T, V, M, h, ER)
            ER = Param.Dconduit
            RKF453.RKF453(AddressOf dPdx, AddressOf dTdx, AddressOf dVdx, x, P, T, V, h,
ER)
            M = V / SoSPT(P, T)
            cntr = cntr + 1
        Loop Until TooFar = True Or M > 1 Or M < lastM Or P < Patm
        TooFar = False
        upperx = x
        lasth = lasth / 2
        'À ce stade, l'intégration a dépassé le point où M=1 ou P=Patm
        'Nous allons maintenant rechercher l'endroit où M=1 ou P=Patm à
        'l'aide d'une méthode de bisection
        Output.Header(Param.BisLmax, "x", "P", "M")
        Do
            'Mise en mémoire des dernières valeurs
            x = lastx
            P = lastP
            T = lastT
            V = lastV
            h = lasth
            ER = Param.Dconduit
            RKF453.RKF453(AddressOf dPdx, AddressOf dTdx, AddressOf dVdx, x, P, T, V, h,
ER)
            M = V / SoSPT(P, T)
            If P = 0 Then System.Diagnostics.Debugger.Break()

```

```

        Output.Save(Param.BisLmax, x, P, M)
        'Déterminer si on a dépassé le point où P=Patm ou M=1
        If x > lastx And M < lastM Then
            TooFar = True
        ElseIf x < lastx And M > lastM Then
            TooFar = True
        ElseIf M > 1 Then
            TooFar = True
        ElseIf P < Param.Patm Then
            TooFar = True
        Else
            TooFar = False
        End If
        If TooFar = True Then
            lasth = lasth / 2
            upperx = x
        Else
            lastx = x
            lastP = P
            lastT = T
            lastV = V
            lastM = M
            lasth = (x + upperx) / 2 - x
        End If
        'Condition de précision pour sortir de la boucle
        If TooFar = False And System.Math.Abs(M - 1) / 1 < 1 / 10000 And M <= 1 Then ✓
Exit Do
        If TooFar = False And System.Math.Abs(P - Patm) / Patm < Param.Dconduit Then ✓
Exit Do
        cntr = cntr + 1
        If cntr > 500 Then System.Diagnostics.Debugger.Break()
        Loop 'Until (x - lastx) / x < 1 / 100000
        'Affectation des valeurs finales aux paramètres de la fonction
        Lmax = x
        Ps = P
        Ts = T
        Vs = V
    End If
End Function
Function dPdx(ByVal x As Double, ByVal P As Double, ByVal T As Double, ByVal V As Double) ✓
    As Double
        Dim err As Boolean
        'Fonction de la dérivée de la pression selon la position dans le conduit
        Static cntr As Long
        'Contrôle des valeurs d'entrée
        If Param.lowerT < T And T < Param.upperT And Param.lowerP < P And P < Param.upperP ✓
Then
            Dim rho, u, h, s, cv, cp, SoS, beta, dDdT, dDdP As Double
            NIST.TPFLSHSI(T, P, rho, u, h, s, cv, cp, SoS, err)
            NIST.THERM2SI(T, rho, P, u, h, s, cv, cp, SoS, beta, dDdT, dDdP, err)
            'Reynolds number calculation
            Dim vol As Double = 1 / rho
            Dim dhdp_T As Double = vol * (1 - T * beta)
            Dim f As Double = Conduit.f(P, T, V, Param.epsilon, Param.ID, Param.equation, ✓
err)
            dPdx = 1 / 2 * V ^ 2 * f * rho * (-rho * cp + dDdT * V ^ 2) / (Param.ID * (rho * ✓
cp - rho * cp * dDdP * V ^ 2 - dDdT * V ^ 2 + dDdT * V ^ 2 * dhdp_T * rho))
        Else
            'Si les valeurs d'entrée sont invalides, nous avons sûrement dépassé
            'le point où M=1 ou P=Patm, et nous le signalons à Lmax.
            TooFar = True
            Exit Function
        End If
        If err = True Then TooFar = True
        cntr = cntr + 1
End Function
Function dTdx(ByVal x As Double, ByVal P As Double, ByVal T As Double, ByVal V As Double) ✓
    As Double
        'Fonction de la dérivée de la température selon la position dans le conduit

```

```

Static cntr As Long
Dim err As Boolean
If Param.lowerT < T And T < Param.upperT And Param.lowerP < P And P < Param.upperP ✓
Then
    Dim rho, u, h, s, cv, cp, SoS, beta, dDdT, dDdP As Double
    NIST.TPFLSHSI(T, P, rho, u, h, s, cv, cp, SoS, err)
    NIST.THERM2SI(T, rho, P, u, h, s, cv, cp, SoS, beta, dDdT, dDdP, err)
    'Reynolds number calculation
    Dim vol As Double = 1 / rho
    Dim dhdP_T As Double = vol * (1 - T * beta)
    Dim f As Double = Conduit.f(P, T, V, Param.epsilon, Param.ID, Param.equation, ✓
err)
    dTdx = -1 / 2 * V ^ 2 * f * rho * (-dhdP_T * rho + dDdP * V ^ 2) / (Param.ID * ✓
(rho * cp - rho * cp * dDdP * V ^ 2 - dDdT * V ^ 2 + dDdT * V ^ 2 * dhdP_T * rho))
    Else
        TooFar = True
        Exit Function
    End If
    If err = True Then TooFar = True
    cntr = cntr + 1
End Function
Function dVdx(ByVal x As Double, ByVal P As Double, ByVal T As Double, ByVal V As Double) ✓
As Double
    'Fonction de la dérivée de la vitesse selon la position dans le conduit
    Dim err As Boolean
    If Param.lowerT < T And T < Param.upperT And Param.lowerP < P And P < Param.upperP ✓
Then
        Dim rho, u, h, s, cv, cp, SoS, beta, dDdT, dDdP As Double
        NIST.TPFLSHSI(T, P, rho, u, h, s, cv, cp, SoS, err)
        NIST.THERM2SI(T, rho, P, u, h, s, cv, cp, SoS, beta, dDdT, dDdP, err)
        Dim vol As Double = 1 / rho
        Dim dhdP_T As Double = vol * (1 - T * beta)
        Dim f As Double = Conduit.f(P, T, V, Param.epsilon, Param.ID, Param.equation, ✓
err)
        dVdx = -1 / 2 * V ^ 3 * f * rho * (-dDdP * cp + dDdT * dhdP_T) / (Param.ID * (rho ✓
* cp - rho * cp * dDdP * V ^ 2 - dDdT * V ^ 2 + dDdT * V ^ 2 * dhdP_T * rho))
        Else
            TooFar = True
            Exit Function
        End If
        If err = True Then TooFar = True
    End Function
Function f(ByVal P As Double, ByVal T As Double, ByVal V As Double, ByVal epsilon As ✓
Double, ByVal ID As Double, _
ByVal equation As String, ByRef err As Boolean) As Double
    'Calcul du nombre de Reynolds
    Dim Re As Double
    If Param.lowerT < T And T < Param.upperT And Param.lowerP < P And P < Param.upperP ✓
Then
        'Reynolds number calculation
        Dim rho, u, h, s, cv, cp, SoS As Double
        NIST.TPFLSHSI(T, P, rho, u, h, s, cv, cp, SoS, err)
        Dim mu As Double = NIST.mu(T, rho, err)
        Re = rho * V * Param.ID / mu
    Else
        TooFar = True
        Exit Function
    End If
    If Re < 6 * 10 ^ 2 Or Re > 10 ^ 8 Then
        TooFar = True
        Exit Function
    End If
    'Calcul du coefficient de friction de Darcy
    If equation = "vonKarman" Then
        'von Karman (Ultimate Smoothness)
        'Il s'agit d'une équation transcendante que l'on doit résoudre par méthode ✓
numérique
        'Nous avons choisi la méthode de la sécante
        Dim x0, x1, x2 As Double

```

```

Dim fx0, fx1, fx2 As Double
x0 = sqrt(0.16 * 1 / (Re ^ 0.16))
x1 = sqrt(4 * 0.005)
fx0 = 1 / (x0) - 1.74 + 2 * ln(18.6 * 1 / (Re * x0)) / ln(10)
fx1 = 1 / (x1) - 1.74 + 2 * ln(18.6 * 1 / (Re * x1)) / ln(10)
If fx0 < fx1 Then
    Dim xi, fxi As Double
    xi = x0
    x0 = x1
    x1 = xi
    fxi = fx0
    fx0 = fx1
    fx1 = fxi
End If
Do
    x2 = x1 - fx1 * (x0 - x1) / (fx0 - fx1)
    fx2 = 1 / (x2) - 1.74 + 2 * ln(18.6 * 1 / (Re * x2)) / ln(10)
    x0 = x1
    fx0 = fx1
    x1 = x2
    fx1 = fx2
Loop Until System.Math.Abs(fx2) < 0.00001
f = x2 ^ 2
ElseIf equation = "Prandtl" Then
    'Prandtl (wholly rough)
    f = 1 / ((1.74 + 2 * ln(epsilon / Param.ID) / ln(10)) ^ 2)
ElseIf equation = "Colebrook" Then
    'Colebrook (turbulent flow)
    'Il s'agit d'une équation transcendante que l'on doit résoudre par méthode
numérique ✓
    'Nous avons choisi la méthode de la sécante
    Dim cntr As Integer = 0
    Dim eps_D As Double = epsilon / ID
    Dim x0, x1, x2 As Double
    Dim fx0, fx1, fx2 As Double
    x0 = sqrt(0.16 * 1 / (Re ^ 0.16))
    x1 = sqrt(4 * 0.005)
    fx0 = 1 / (x0) + 2 * ln(0.2702702703 * epsilon / Param.ID + 2.51 * x0 / Re) / ln ✓
(10)
    fx1 = 1 / (x1) + 2 * ln(0.2702702703 * epsilon / Param.ID + 2.51 * x1 / Re) / ln ✓
(10)
    If fx0 < fx1 Then
        Dim xi, fxi As Double
        xi = x0
        x0 = x1
        x1 = xi
        fxi = fx0
        fx0 = fx1
        fx1 = fxi
    End If
    Do
        x2 = x1 - fx1 * (x0 - x1) / (fx0 - fx1)
        fx2 = 1 / (x2) + 2 * ln(0.2702702703 * epsilon / Param.ID + 2.51 * x2 / Re) / ✓
ln(10)
        x0 = x1
        fx0 = fx1
        x1 = x2
        fx1 = fx2
        cntr = cntr + 1
        If cntr >= 500 Then System.Diagnostics.Debugger.Break()
    Loop Until System.Math.Abs(fx2) < 0.0001
    If x2 <= 0 Then
        System.Diagnostics.Debugger.Break()
    Else
        f = x2 ^ 2
    End If
ElseIf equation = "Genereaux" Then
    'Genereaux (4·10^3 < f < 20·10^6)
    f = 4 * (0.04 * Re ^ (-0.16))

```

```
        ElseIf equation = "constant" Then
            f = Param.mdf
        End If
    End Function
Function SoSPT(ByVal P As Double, ByVal T As Double) As Double
    'Fonction calculant la vitesse du son dans un fluide réel à partir de la pression
    'et de la température
    If Param.lowerT < T And T < Param.uppertT And Param.lowerP < P And P < Param.upperP
    Then
        Dim rho, u, h, s, cv, cp, w As Double
        Dim err As Boolean
        NIST.TPFLSHSI(T, P, rho, u, h, s, cv, cp, w, err)
        If err = True Then TooFar = True
        SoSPT = w
    Else
        TooFar = True
        Exit Function
    End If
End Function
Function sqrt(ByVal x As Double) As Double
    'fonction servant simplement à diminuer la quantité de code requis pour évaluer une
    racine
    sqrt = System.Math.Sqrt(x)
End Function
Function ln(ByVal x As Double) As Double
    'fonction servant simplement à diminuer la quantité de code requis pour évaluer un
    'logarithme naturel
    ln = System.Math.Log(x)
End Function
End Module
```


'Ce module comporte des fonctions propres à l'écoulement de Fanno.
 'On y considère donc un conduit à section constante, un gaz parfait et
 'aucun échange de chaleur.
 'On utilise ce module, dans ce programme, pour obtenir des premières
 'approximations permettant d'accélérer la convergence des méthodes
 'itératives.
 'Plusieurs fonctions de ce module correspondent aux équations derrière
 'les tables de calcul pour un écoulement de Fanno que l'on peut retrouver
 'dans les volumes classiques de mécanique des fluides compressibles.

Module Fanno

```
Function Mc(ByVal pt As Double, ByVal k As Double, _
  ByVal Pb As Double, ByVal D As Double, _
  ByVal f As Double, ByVal L As Double, ByVal FTOL As Double) As Double
  'La fonction "Mc" calcule le nombre de Mach à l'entrée du conduit en
  'fonction de la pression d'arrêt à l'entrée du conduit, du rapport
  'des chaleurs massiques, de la pression arrière (normalement la pression
  'atmosphérique), du diamètre du conduit, du coefficient de friction et de
  'la longueur du conduit.
  Dim M1 As Double
  Dim fL_D As Double
  Dim P1 As Double, Pcr As Double
  'Calculate M1 for choked flow
  fL_D = f * L / D
  M1 = MfLmax_D(fL_D, k)
  'Determine if the flow is choked
  'Calculate pressure at the throat
  P1 = Isentropic.p_p0(M1, k) * pt
  'Calculate critical pressure
  Pcr = P1 / Fanno.p_pcr(M1, k)
  'Compare critical pressure and back pressure
  If Pb <= Pcr Then
    'The flow is choked
    Mc = M1
  Else
    'Secant method
    Dim x0, x1, x2 As Double
    Dim fx0, fx1, fx2 As Double
    Dim ER As Double
    x0 = M1 * 0.9
    fx0 = Fanno.P2M1(x0, pt, f, L, D, k) - Pb
    x1 = M1
    fx1 = Fanno.P2M1(x1, pt, f, L, D, k) - Pb
    If fx0 < fx1 Then
      Dim xi, fxi As Double
      xi = x0
      x0 = x1
      x1 = xi
      fxi = fx0
      fx0 = fx1
      fx1 = fxi
    End If
    Do
      x2 = x1 - fx1 * (x0 - x1) / (fx0 - fx1)
      fx2 = Fanno.P2M1(x2, pt, f, L, D, k) - Pb
      x0 = x1
      fx0 = fx1
      x1 = x2
      fx1 = fx2
      ER = System.Math.Abs(fx2 / Pb)
    Loop Until ER < 0.00001
    Mc = x2
  'End secant method
  End If
End Function

Function p_pcr(ByVal M As Double, ByVal k As Double) As Double
  'Cette fonction calcule le ratio de la pression statique sur la pression
  'critique à partir du nombre de Mach et du rapport des chaleurs massiques.
  p_pcr = ((k + 1) / (2 + (k - 1) * M ^ 2)) ^ (1 / 2) / M
End Function
```

```

Function T_Tcr(ByVal M As Double, ByVal k As Double) As Double
    'Cette fonction calcule le ratio de la température statique sur la température
    'critique à partir du nombre de Mach et du rapport des chaleurs massiques.
    T_Tcr = (k + 1) / (2 + (k - 1) * M ^ 2)
End Function
Function rho_rhocr(ByVal M As Double, ByVal k As Double) As Double
    'Cette fonction calcule le ratio de la masse volumique statique sur la
    'masse volumique critique à partir du nombre de Mach et du rapport des
    'chaleurs massiques.
    rho_rhocr = System.Math.Sqrt(2) * System.Math.Sqrt((1 + 1 / 2 * (k + 1) * M ^ 2) / (
(k + 1) * M ^ 2))
End Function
Function V_Vcr(ByVal M As Double, ByVal k As Double) As Double
    'Cette fonction calcule le ratio du volume massique statique sur le
    'volume massique critique à partir du nombre de Mach et du rapport des
    'chaleurs massiques.
    V_Vcr = 1 / 2 * System.Math.Sqrt(2) * System.Math.Sqrt((k + 1) * M ^ 2 / (1 + 1 / 2 *
(k - 1) * M ^ 2))
End Function
Function fLmax_D(ByVal M As Double, ByVal k As Double) As Double
    'Cette fonction calcule la longueur relative maximale du conduit à partir du
    'nombre de Mach et du rapport des chaleurs massiques.
    fLmax_D = (1 - M ^ 2) / (M ^ 2 * k) + 1 / 2 * (k + 1) * System.Math.Log((k + 1) * M ^
2 / (2 + (k - 1) * M ^ 2)) / k
End Function
Function P2M1(ByVal M1 As Double, ByVal P0 As Double, ByVal f As Double, ByVal L As
Double, ByVal D As Double, ByVal k As Double) As Double
    'Cette fonction calcule la pression à la sortie dans le cas où l'écoulement est
    subsonique
    Dim fLmax_D2 As Double
    Dim M2 As Double
    Dim P1 As Double
    P1 = Isentropic.p_p0(M1, k) * P0
    fLmax_D2 = Fanno.fLmax_D(M1, k) - f * L / D
    M2 = Fanno.MfLmax_D(fLmax_D2, k)
    P2M1 = Fanno.p_pcr(M2, k) / Fanno.p_pcr(M1, k) * P1
End Function
Function MfLmax_D(ByVal fLmax_D As Double, ByVal k As Double) As Double
    'Cette fonction calcule le nombre de Mach correspondant à une longueur de conduit
    'relative donnée
    'Bisection Method
    Dim FTOL As Double = 0.0001
    Dim ML As Double, MM As Double, MR As Double
    'Define initial domain
    ML = FTOL
    MR = 1
    'Start loop
    Do While System.Math.Abs(MR - ML) > FTOL
        'Calculate midpoint of domain
        MM = (MR + ML) / 2
        'Find f(MM)
        If (Fanno.fLmax_D(ML, k) - fLmax_D) * (Fanno.fLmax_D(MM, k) - fLmax_D) > 0 Then
            'Throw away left half
            ML = MM
        Else
            'Throw away right half
            MR = MM
        End If
    Loop
    MfLmax_D = (MR + ML) / 2
End Function
End Module

```

Module GazParfait

```
'Ce module comporte des fonctions de calcul des propriétés thermodynamiques d'un gaz parfait
'Tous les paramètres doivent être en unités SI
Function rho(ByVal P As Double, ByVal T As Double, ByVal R As Double) As Double
    'Cette fonction calcule la densité à partir de la pression, de la température et de la
    'constante spécifique du gaz parfait.
    rho = P / (R * T)
End Function
Function u(ByVal T As Double, ByVal Cv0 As Double) As Double
    'Cette fonction calcule l'énergie interne à partir de la température et de la chaleur
    'massique à volume constant pour un gaz parfait.
    u = Cv0 * T
End Function
Function T(ByVal u As Double, ByVal Cv0 As Double) As Double
    'Cette fonction calcule la température à partir de l'énergie interne et de la chaleur
    'massique à volume constant pour un gaz parfait.
    T = u / Cv0
End Function
Function P(ByVal rho As Double, ByVal u As Double, ByVal Cv0 As Double) As Double
    'Cette fonction calcule la pression à partir de la densité, de l'énergie interne et de
    'la chaleur massique à volume constant pour un gaz parfait.
    Dim T As Double = u / Cv0
    P = rho * Param.R * T
End Function
Function h(ByVal u As Double, ByVal Cp0 As Double, ByVal Cv0 As Double) As Double
    'Cette fonction calcule l'enthalpie à partir de l'énergie interne et des chaleurs
    'massiques à volume et pression constants.
    Dim T As Double = u / Cv0
    h = Cp0 * T
End Function
Function SoS(ByVal k As Double, ByVal R As Double, ByVal T As Double) As Double
    'Cette fonction calcule la vitesse du son dans un gaz parfait à partir du rapport des
    'chaleurs massiques, de la constante des gaz parfaits et de la température.
    SoS = System.Math.Sqrt(k * R * T)
End Function
End Module
```

```

Module Isentropic
    'Ce module comporte des fonctions de calcul pour l'écoulement isentropique d'un gaz parfait en
    'régime permanent.
    Function p_p0(ByVal M As Double, ByVal k As Double) As Double
        'Cette fonction calcule le rapport de la pression statique sur la pression d'arrêt
        'pour l'écoulement isentropique d'un gaz parfait en régime permanent.
        
$$p\_p0 = (1 + 1 / 2 * (k - 1) * M ^ 2) ^ (k / (1 - k))$$

    End Function
    Function T_T0(ByVal M As Double, ByVal k As Double) As Double
        'Cette fonction calcule le rapport de la température statique sur la température d'arrêt
        'pour l'écoulement isentropique d'un gaz parfait en régime permanent.
        
$$T\_T0 = 1 / (1 + 1 / 2 * (k - 1) * M ^ 2)$$

    End Function
    Function rho_rho0(ByVal M As Double, ByVal k As Double) As Double
        'Cette fonction calcule le rapport de la masse volumique statique sur la masse volumique
        'd'arrêt pour l'écoulement isentropique d'un gaz parfait en régime permanent.
        
$$\rho\_rho0 = (1 + 1 / 2 * (k - 1) * M ^ 2) ^ (1 / (1 - k))$$

    End Function
    Function Mp_p0(ByVal p_p0 As Double, ByVal k As Double) As Double
        'Cette fonction calcule le nombre Mach correspondant à un rapport de la pression statique
        'sur la pression d'arrêt pour l'écoulement isentropique d'un gaz parfait en régime permanent.
        If p_p0 >= 1 Then
            Mp_p0 = 0
        Else
            
$$Mp\_p0 = \text{System.Math.Sqrt}((p\_p0 ^ ((1 - k) / k) - 1) / (1 / 2 * k - 1 / 2))$$

        End If
    End Function
End Module

```

Module Main

'Ce module est le module principal à partir duquel sont lancés l'ensemble des autres modules

Sub Main()

'Initialisation de la base de données du NIST

NIST.init()

'Écriture à la console de la description du programme

Console.WriteLine("Hydrogen release through steady one-dimensional duct with friction")

" + ControlChars.NewLine + "using nist data for thermodynamic relations")

'Calcul des propriétés initiales

Dim t0, Pt0, Tt0, rhot0, ut0, ht0, st0, cvt0, cpt0, SoSt0 As Double

Dim err As Boolean

t0 = Param.t0

Pt0 = Param.Pt0

Tt0 = Param.Tt0

NIST.TPFLSHSI(Tt0, Pt0, rhot0, ut0, ht0, st0, cvt0, cpt0, SoSt0, err)

'Écriture des en-têtes dans les fichiers de sortie

Output.Header(Param.Darcy, "Friction")

Output.Header(Param.Reservoir, "t", "rhot", "ut", "h", "ER", "Pt", "Tt", "Ps", "Ts",

"Vs", "Ms", "MFR")

'Déterminer le pas initial pour l'intégration du conduit

Dim mfr As Double = Reservoir.mfr(rhot0, ut0)

Param.h0reservoir = (rhot0 * Param.Psi) / mfr * 0.01

'Lancement de l'intégration numérique du système d'EDO du réservoir

RKF452.RKF452(AddressOf Reservoir.drhotdt, AddressOf Reservoir.dutdt,

t0, rhot0, ut0, Param.h0reservoir, Param.Dconduit, AddressOf Reservoir.RKF452Step)

End Sub

End Module

Option Strict Off
Option Explicit On
Module NIST

'Pour plus d'informations sur ce module, consulter la documentation du NIST

```
Public Declare Sub SETUPdll Lib "Refprop" (ByRef i As Integer, ByVal hfld As String,
ByVal hfmix As String, ByVal hrf As String, ByRef ierr As Integer, ByVal herr As String,
ByRef ln1 As Integer, ByRef ln2 As Integer, ByRef ln3 As Integer, ByRef ln4 As Integer)
Public Declare Sub SETREFdll Lib "Refprop" (ByVal hrf As String, ByRef ixflag As Integer,
ByRef x0 As Double, ByRef h0 As Double, ByRef s0 As Double, ByRef t0 As Double, ByRef p0
As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln1 As Integer, ByRef ln2
As Integer)
Public Declare Sub SETMODdll Lib "Refprop" (ByRef i As Integer, ByVal htype As String,
ByVal hmix As String, ByVal hcomp As String, ByRef ierr As Integer, ByVal herr As String,
ByRef ln1 As Integer, ByRef ln2 As Integer, ByRef ln3 As Integer, ByRef ln4 As Integer)
Public Declare Sub TPRHODll Lib "Refprop" (ByRef t As Double, ByRef p As Double, ByRef x
As Double, ByRef j As Integer, ByRef i As Integer, ByRef d As Double, ByRef ierr As
Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub THERM2dll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x
As Double, ByRef p As Double, ByRef e As Double, ByRef h As Double, ByRef s As Double,
ByRef cv As Double, ByRef cp As Double, ByRef w As Double, ByRef z As Double, ByRef hjt
As Double, ByRef aH As Double, ByRef g As Double, ByRef xkappa As Double, ByRef beta As
Double, ByRef dPdD As Double, ByRef d2PdD2 As Double, ByRef xPdT As Double, ByRef dDdT As
Double, ByRef dDdP As Double, ByRef spare1 As Double, ByRef spare2 As Double, ByRef
spare3 As Double, ByRef spare4 As Double)
Public Declare Sub THERMDll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x
As Double, ByRef p As Double, ByRef e As Double, ByRef h As Double, ByRef s As Double,
ByRef cv As Double, ByRef cp As Double, ByRef w As Double, ByRef hjt As Double)
Public Declare Sub ENTROdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x
As Double, ByRef s As Double)
Public Declare Sub ENTHALdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x
As Double, ByRef h As Double)
Public Declare Sub CVCPdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x
As Double, ByRef cv As Double, ByRef cp As Double)
Public Declare Sub PRESSdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x
As Double, ByRef p As Double)
Public Declare Sub AGdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x As
Double, ByRef a As Double, ByRef g As Double)

Public Declare Sub DPDDdll Lib "Refprop" (ByRef t As Double, ByRef rho As Double, ByRef x
As Double, ByRef dPdD As Double)
Public Declare Sub DPDD2dll Lib "Refprop" (ByRef t As Double, ByRef rho As Double, ByRef
x As Double, ByRef d2PdD2 As Double)
Public Declare Sub DPDTdll Lib "Refprop" (ByRef t As Double, ByRef rho As Double, ByRef x
As Double, ByRef dPdT As Double)
Public Declare Sub DDDPdll Lib "Refprop" (ByRef t As Double, ByRef rho As Double, ByRef x
As Double, ByRef dDdP As Double)
Public Declare Sub DDDTdll Lib "Refprop" (ByRef t As Double, ByRef rho As Double, ByRef x
As Double, ByRef dDdT As Double)
Public Declare Sub DHDTdll Lib "Refprop" (ByRef t As Double, ByRef rho As Double, ByRef x
As Double, ByRef dHdT As Double)

Public Declare Sub SATTdll Lib "Refprop" (ByRef t As Double, ByRef x As Double, ByRef i
As Integer, ByRef p As Double, ByRef D1 As Double, ByRef Dv As Double, ByRef xliq As
Double, ByRef xvap As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As
Integer)
Public Declare Sub SATPdll Lib "Refprop" (ByRef p As Double, ByRef x As Double, ByRef i
As Integer, ByRef t As Double, ByRef D1 As Double, ByRef Dv As Double, ByRef xliq As
Double, ByRef xvap As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As
Integer)
Public Declare Sub SATDdll Lib "Refprop" (ByRef d As Double, ByRef x As Double, ByRef kph
As Integer, ByRef kr As Integer, ByRef t As Double, ByRef p As Double, ByRef D1 As
Double, ByRef Dv As Double, ByRef xliq As Double, ByRef xvap As Double, ByRef ierr As
Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub SATHdll Lib "Refprop" (ByRef h As Double, ByRef x As Double, ByRef kph
As Integer, ByRef nroot As Integer, ByRef k1 As Integer, ByRef t1 As Double, ByRef p1 As
Double, ByRef d1 As Double, ByRef k2 As Integer, ByRef t2 As Double, ByRef p2 As Double,
ByRef d2 As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub SATSdll Lib "Refprop" (ByRef s As Double, ByRef x As Double, ByRef kph
```

[illegible]

```
Public Declare Sub TQFLSHdll Lib "Refprop" (ByRef t As Double, ByRef q As Double, ByRef x As Double, ByRef kq As Integer, ByRef p As Double, ByRef d As Double, ByRef Dl As Double, ByRef Dv As Double, ByRef xliq As Double, ByRef xvap As Double, ByRef e As Double, ByRef h As Double, ByRef s As Double, ByRef cv As Double, ByRef cp As Double, ByRef w As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub PQFLSHdll Lib "Refprop" (ByRef p As Double, ByRef q As Double, ByRef x As Double, ByRef kq As Integer, ByRef t As Double, ByRef d As Double, ByRef Dl As Double, ByRef Dv As Double, ByRef xliq As Double, ByRef xvap As Double, ByRef e As Double, ByRef h As Double, ByRef s As Double, ByRef cv As Double, ByRef cp As Double, ByRef w As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub ABFLdll Lib "Refprop" (ByRef a As Double, ByRef b As Double, ByRef x As Double, ByRef i As Integer, ByVal ab As String, ByRef dmin As Double, ByRef dmax As Double, ByRef t As Double, ByRef p As Double, ByRef d As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln1 As Integer, ByRef ln2 As Integer)
Public Declare Sub ABFL2dll Lib "Refprop" (ByRef a As Double, ByRef b As Double, ByRef x As Double, ByRef kq As Integer, ByRef ksat As Integer, ByVal ab As String, ByRef tbub As Double, ByRef tdew As Double, ByRef pbub As Double, ByRef pdew As Double, ByRef Dlbub As Double, ByRef Dvdew As Double, ByRef ybub As Double, ByRef xdew As Double, ByRef t As Double, ByRef p As Double, ByRef Dl As Double, ByRef Dv As Double, ByRef x As Double, ByRef y As Double, ByRef q As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer, ByRef ln2 As Integer)
Public Declare Sub DBFL2dll Lib "Refprop" (ByRef d As Double, ByRef b As Double, ByRef x As Double, ByRef i As Integer, ByVal ab As String, ByRef t As Double, ByRef p As Double, ByRef Dl As Double, ByRef Dv As Double, ByRef xliq As Double, ByRef xvap As Double, ByRef q As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer, ByRef ln2 As Integer)

Public Declare Sub CRITPd11 Lib "Refprop" (ByRef x As Double, ByRef tc As Double, ByRef pc As Double, ByRef dc As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub VIRBdll Lib "Refprop" (ByRef t As Double, ByRef x As Double, ByRef b As Double)
Public Declare Sub DBDTdll Lib "Refprop" (ByRef t As Double, ByRef x As Double, ByRef dbt As Double)
Public Declare Sub VIRCdll Lib "Refprop" (ByRef t As Double, ByRef x As Double, ByRef c As Double)
Public Declare Sub TRNPRPd11 Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x As Double, ByRef eta As Double, ByRef tcx As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub FGCTYdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x As Double, ByRef f As Double)
Public Declare Sub DIBLECd11 Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x As Double, ByRef de As Double)
Public Declare Sub SURFTdll Lib "Refprop" (ByRef t As Double, ByRef d As Double, ByRef x As Double, ByRef sigma As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub SURTENDdll Lib "Refprop" (ByRef t As Double, ByRef rhol As Double, ByRef rhov As Double, ByRef xl As Double, ByRef xv As Double, ByRef sigma As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub MELTTdll Lib "Refprop" (ByRef t As Double, ByRef x As Double, ByRef p As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub MELTH2Od11 Lib "Refprop" (ByRef t As Double, ByRef pl As Double, ByRef p2 As Double)
Public Declare Sub MELTPdll Lib "Refprop" (ByRef p As Double, ByRef x As Double, ByRef t As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub SUBLTd11 Lib "Refprop" (ByRef t As Double, ByRef x As Double, ByRef p As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)
Public Declare Sub SUBLPd11 Lib "Refprop" (ByRef p As Double, ByRef x As Double, ByRef t As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ln As Integer)

Public Declare Sub WMOLdll Lib "Refprop" (ByRef x As Double, ByRef wm As Double)
Public Declare Sub XMASSdll Lib "Refprop" (ByRef xmol As Double, ByRef xkg As Double, ByRef wmix As Double)
Public Declare Sub XMOEd11 Lib "Refprop" (ByRef xkg As Double, ByRef xmol As Double, ByRef wmix As Double)
Public Declare Sub QMASSdll Lib "Refprop" (ByRef qmol As Double, ByRef xl As Double, ByRef xv As Double, ByRef qkg As Double, ByRef xkg As Double, ByRef xvkg As Double)
```



```

ByRef wliq As Double, ByRef wvap As Double, ByRef ierr As Integer, ByVal herr As String, ✓
ByRef ln As Integer)
Public Declare Sub QMOLEdll Lib "Refprop" (ByRef qkg As Double, ByRef xkg As Double, ✓
ByRef xvk As Double, ByRef qmol As Double, ByRef xl As Double, ByRef xv As Double, ByRef ✓
wliq As Double, ByRef wvap As Double, ByRef ierr As Integer, ByVal herr As String, ByRef ✓
ln As Integer)
Public Declare Sub INFOdll Lib "Refprop" (ByRef icomp As Integer, ByRef wmm As Double, ✓
ByRef ttrp As Double, ByRef tnbpt As Double, ByRef tc As Double, ByRef pc As Double, ✓
ByRef dc As Double, ByRef Zc As Double, ByRef acf As Double, ByRef dip As Double, ByRef ✓
Rgas As Double)
Public Declare Sub LIMITXdll Lib "Refprop" (ByVal htyp As String, ByRef t As Double, ✓
ByRef d As Double, ByRef p As Double, ByRef x As Double, ByRef tmin As Double, ByRef tmax ✓
As Double, ByRef dmax As Double, ByRef pmax As Double, ByRef ierr As Integer, ByVal herr ✓
As String, ByRef ln1 As Integer, ByRef ln2 As Integer)
Public Declare Sub LIMITKdll Lib "Refprop" (ByVal htyp As String, ByRef icomp As Integer, ✓
ByRef t As Double, ByRef d As Double, ByRef p As Double, ByRef tmin As Double, ByRef ✓
tmax As Double, ByRef dmax As Double, ByRef pmax As Double, ByRef ierr As Integer, ByVal ✓
herr As String, ByRef ln1 As Integer, ByRef ln2 As Integer)

Public Declare Sub SETKTVdll Lib "Refprop" (ByRef icomp As Integer, ByRef jcomp As ✓
Integer, ByVal hmodij As String, ByRef fij As Double, ByVal hfmix As String, ByRef ierr ✓
As Integer, ByVal herr As String, ByRef ln1 As Integer, ByRef ln2 As Integer, ByRef ln3 ✓
As Integer)
Public Declare Sub GETKTVdll Lib "Refprop" (ByRef icomp As Integer, ByRef jcomp As ✓
Integer, ByVal hmodij As String, ByRef fij As Double, ByVal hfmix As String, ByVal hfij ✓
As String, ByVal hbinp As String, ByVal hmxrul As String, ByRef ln1 As Integer, ByRef ln2 ✓
As Integer, ByRef ln3 As Integer, ByRef ln4 As Integer, ByRef ln5 As Integer)
Public Declare Sub GETFIJdll Lib "Refprop" (ByVal hmodij As String, ByRef fij As Double, ✓
ByVal hfij As String, ByVal hmxrul As String, ByRef ln1 As Integer, ByRef ln2 As Integer, ✓
ByRef ln3 As Integer)

```

```

Const MaxComps As Short = 20

```

```

Dim hfmix As String
Dim htyp As String
Dim hrf As String
Dim htype As String
Dim hmix As String
Dim hcomp As String
Dim hfld As String
Dim nc As Integer
'UPGRADE_WARNING: Lower bound of array x was changed from 1 to 0. Click for more: 'ms- ✓
help://MS.VSCC.2003/commoner/redirect/redirect.htm?keyword="vbup1033"'
Dim x(MaxComps) As Double
'UPGRADE_WARNING: Lower bound of array xmm was changed from 1 to 0. Click for more: 'ms- ✓
help://MS.VSCC.2003/commoner/redirect/redirect.htm?keyword="vbup1033"'
Dim xmm(MaxComps) As Double
'UPGRADE_WARNING: Lower bound of array xliq was changed from 1 to 0. Click for more: 'ms- ✓
help://MS.VSCC.2003/commoner/redirect/redirect.htm?keyword="vbup1033"'
Dim xliq(MaxComps) As Double
'UPGRADE_WARNING: Lower bound of array xvap was changed from 1 to 0. Click for more: 'ms- ✓
help://MS.VSCC.2003/commoner/redirect/redirect.htm?keyword="vbup1033"'
Dim xvap(MaxComps) As Double
'UPGRADE_WARNING: Lower bound of array xv was changed from 1 to 0. Click for more: 'ms- ✓
help://MS.VSCC.2003/commoner/redirect/redirect.htm?keyword="vbup1033"'
Dim xv(MaxComps) As Double
Dim rhov, hv As Double
Dim nroot, kg, kr, ix As Integer
Dim q, dl, p, t, d, Dv, wm As Double
Dim cp, s, e, h, cv, w As Double
Dim spare3, spare1, dDdT, d2PdD2, beta, g, hjt, z, a, xkappa, dPdD, dPdT, dDdP, spare2, ✓
spare4 As Double
Dim tcx, de, b, c, eta, hvap As Double
Dim dmax, tmin, tmax, pmax As Double
Dim pc, tc, dc As Double
Dim hRef, Tref, pref, sRef As Double
'UPGRADE_WARNING: Lower bound of array f was changed from 1 to 0. Click for more: 'ms- ✓
help://MS.VSCC.2003/commoner/redirect/redirect.htm?keyword="vbup1033"'

```

```

Dim f(MaxComps) As Double
Dim sigma As Double
Dim k2, k1, k3 As Integer
Dim p1, t1, d1 As Double
Dim p2, t2, d2 As Double
Dim p3, t3, d3 As Double
Dim dip, Zc, ttrp, wmm, tnbpt, acf, Rgas As Double

Sub Example()
    Dim ierr As Double
    Dim herr As String = ""
    'Pure fluid:
    nc = 1
    hfld = "fluids\nitrogen.fld"

    'Mixture:
    nc = 3
    hfld = "fluids\nitrogen.fld|fluids\argon.fld|fluids{oxygen.fld|"
    x(1) = 0.7812
    x(2) = 0.0092
    x(3) = 0.2096

    'See fortran for inputs:
    'Call SETMODdll(nc, htype, hmix, hcomp, ierr, herr, 3&, 3&, 255&, 255&)

    hfmix = "fluids\hmx.bnc"
    hrf = "DEF"
    Call SETUPdll(nc, hfld, hfmix, hrf, ierr, herr, 10000, 255, 3, 255)

    'See fortran for inputs:
    'Call SETREFdll(hrf, ix, x(1), hRef, sRef, Tref, pref, ierr, herr, 3&, 255&)

    t = 100 'K
    p = 1000 'kPa

    'Get fluid info for component 1
    Call INFODll(1, wmm, ttrp, tnbpt, tc, pc, dc, Zc, acf, dip, Rgas)

    'Get mixture critical parameters:
    Call CRITPdll(x(1), tc, pc, dc, ierr, herr, 255)

    'Get molecular weight:
    Call WMOLDll(x(1), wm)

    'Get limits:
    htyp = "EOS"
    Call LIMITXdll(htyp, t, d, p, x(1), tmin, tmax, dmax, pmax, ierr, herr, 3, 255)

    'Get d from t,p if phase is known:
    'Fourth input: 1 - liquid, 2 - vapor
    'Fourth and fifth inputs must be LONG values! (include the &)
    Call TPRHODll(t, p, x(1), 1, 0, d, ierr, herr, 255)

    'Get d from t,p if phase is not known:
    Call TPFLSHdll(t, p, x(1), d, D1, Dv, xliq(1), xvap(1), q, e, h, s, cv, cp, w, ierr, ✓
    herr, 255)

    'Calculate properties given T and d:
    Call PRESSdll(t, d, x(1), p)
    Call THERM2dll(t, d, x(1), p, e, h, s, cv, cp, w, z, hjt, a, g, xkappa, beta, dPdD, ✓
    d2PdD2, dPdT, dDdT, dDdP, spare1, spare2, spare3, spare4)
    Call VIRBdll(t, x(1), b)
    Call VIRCdll(t, x(1), c)
    Call DIELECdll(t, d, x(1), de)
    Call FGCTYdll(t, d, x(1), f(1))
    Call SURTENDll(t, d, rhov, x(1), xv(1), sigma, ierr, herr, 255)
    Call TRNPRPdll(t, d, x(1), eta, tcx, ierr, herr, 255)

    'Saturation routines:

```

```

'Third input for Satt and Satp is: 1 - bubble point, 2 - dew point
Call SATTd11(t, x(1), 1, p, D1, Dv, xliq(1), xvap(1), ierr, herr, 255)
Call SATPd11(p, x(1), 1, t, D1, Dv, xliq(1), xvap(1), ierr, herr, 255)
'Call SATDd11(d, x(1), 1&, kr, t, p, D1, Dv, xliq(1), xvap(1), ierr, herr, 255&)
'Call SATHd11(h, x(1), 0&, nroot, k1, t, p1, d1, k2, t2, p2, d2, ierr, herr, 255&)
'Call SATSd11(s, x(1), 0&, nroot, k1, t1, p1, d1, k2, t2, p2, d2, k3, t3, p3, d3,
ierr, herr, 255&)

'Routines to calculate properties when T and d are not both known:
'Call DEFLSHd11(d, e, x(1), t, p, D1, Dv, xliq(1), xvap(1), q, h, s, cv, cp, w, ierr,
herr, 255&)
'Call DHFLSHd11(d, h, x(1), t, p, D1, Dv, xliq(1), xvap(1), q, e, s, cv, cp, w, ierr,
herr, 255&)
'Call DSFLSHd11(d, s, x(1), t, p, D1, Dv, xliq(1), xvap(1), q, e, h, cv, cp, w, ierr,
herr, 255&)
'Call PDEFLSHd11(p, d, x(1), t, D1, Dv, xliq(1), xvap(1), q, e, h, s, cv, cp, w, ierr,
herr, 255&)
'Call PEFLSHd11(p, e, x(1), t, d, D1, Dv, xliq(1), xvap(1), q, e, h, cv, cp, w, ierr,
herr, 255&)
'Call PHFLSHd11(p, h, x(1), t, d, D1, Dv, xliq(1), xvap(1), q, e, s, cv, cp, w, ierr,
herr, 255&)
'Call PQFLSHd11(p, q, x(1), 1&, t, d, D1, Dv, xliq(1), xvap(1), e, h, s, cv, cp, w,
ierr, herr, 255&)
'Call PSFLSHd11(p, s, x(1), t, d, D1, Dv, xliq(1), xvap(1), q, e, h, cv, cp, w, ierr,
herr, 255&)
'Call TDEFLSHd11(t, d, x(1), p, D1, Dv, xliq(1), xvap(1), q, e, h, s, cv, cp, w, ierr,
herr, 255&)
'Call TEFLSHd11(t, e, x(1), 1&, p, d, D1, Dv, xliq(1), xvap(1), q, h, s, cv, cp, w,
ierr, herr, 255&)
'Call THFLSHd11(t, h, x(1), 1&, p, d, D1, Dv, xliq(1), xvap(1), q, e, s, cv, cp, w,
ierr, herr, 255&)
'Call TQFLSHd11(t, q, x(1), 1&, p, d, D1, Dv, xliq(1), xvap(1), e, h, s, cv, cp, w,
ierr, herr, 255&)
'Call TSFLSHd11(t, s, x(1), 1&, p, d, D1, Dv, xliq(1), xvap(1), q, e, h, cv, cp, w,
ierr, herr, 255&)

'Melting and sublimation routines:
'Call MELTPd11(p, x(1), t, ierr, herr, 255&)
'Call MELTDd11(t, x(1), p, ierr, herr, 255&)
'Call SUBLTDd11(t, x(1), p, ierr, herr, 255&)
'Call SUBLPd11(p, x(1), t, ierr, herr, 255&)

'Convert mole fraction to/from mass fraction:
'Call XMASSd11(x(1), xmm(1), wm)
'Call XMOEd11(x(1), xmm(1), wm)
End Sub
Sub init()
'Initialisation pour une substance pure
Dim ierr As Double
Dim herr As String = ""
nc = 1
hfld = Param.FluidFile
hfmix = "fluids\hmx.bnc"
hrf = "DEF"
x(1) = 1
Call SETUPd11(nc, hfld, hfmix, hrf, ierr, herr, 10000, 255, 3, 255)
If ierr <> 0 Then System.Diagnostics.Debugger.Break()
End Sub
Public Sub TPFLSHSI(ByRef t As Double, ByRef p As Double, ByRef d As Double, _
ByRef e As Double, ByRef h As Double, ByRef s As Double, ByRef cv As Double, _
ByRef cp As Double, ByRef w As Double, ByRef err As Boolean)
Dim ierr As Double
Dim herr As String = ""
Dim MM As Double = NIST.MolarMass
'Conversion des unités SI en unités NIST
'La pression en Pa doit être transformée en kPa
p = p / 1000
Call TPFLSHd11(t, p, x(1), d, D1, Dv, xliq(1), xvap(1), q, e, h, s, cv, cp, w, ierr,
herr, 255&)

```

```

    If herr <> "" Or ierr <> 0 Then err = True Else err = False
    'Conversion des Unités NIST en unités SI
    'La pression en kPa doit être transformée en Pa
    p = p * 1000
    'Les densités en mol/L doit être transformées en kg/m³
    d = d * MM
    Dl = Dl * MM
    Dv = Dv * MM
    'Les J/mol doivent devenir des J/kg
    e = e / MM * 1000
    h = h / MM * 1000
    'Les J/(mol*K) doivent devenir des J/(kg*K)
    s = s / MM * 1000
    cv = cv / MM * 1000
    cp = cp / MM * 1000
End Sub
Public Sub PSFLSHSI(ByRef p As Double, ByRef s As Double, ByRef t As Double, _
ByRef d As Double, ByRef e As Double, ByRef h As Double, ByRef cv As Double, _
ByRef cp As Double, ByRef w As Double, ByRef err As Boolean)
    Dim MM As Double = NIST.MolarMass
    Dim ierr As Double
    Dim herr As String = ""
    'Conversion des unités SI en unités NIST
    'La pression en Pa doit être transformée en kPa
    p = p / 1000
    'L'entropie en J/(kg*K) doit être transformée en J/(mol*K)
    s = s / 1000 * MM
    Call PSFLSHdll(p, s, x(1), t, d, Dl, Dv, xliq(1), xvap(1), q, e, h, cv, cp, w, ierr, ✓
herr, 255&)
    If herr <> "" Or ierr <> 0 Then err = True Else err = False
    'Conversion des Unités NIST en unités SI
    'La pression en kPa doit être transformée en Pa
    p = p * 1000
    'Les densités en mol/L doit être transformées en kg/m³
    d = d * MM
    Dl = Dl * MM
    Dv = Dv * MM
    'Les J/mol doivent devenir des J/kg
    e = e / MM * 1000
    h = h / MM * 1000
    'Les J/(mol*K) doivent devenir des J/(kg*K)
    s = s / MM * 1000
    cv = cv / MM * 1000
    cp = cp / MM * 1000
End Sub
Public Sub DEFLSHSI(ByRef d As Double, ByRef e As Double, ByRef t As Double, _
ByRef p As Double, ByRef h As Double, ByRef s As Double, ByRef cv As Double, _
ByRef cp As Double, ByRef w As Double, ByRef err As Boolean)
    Dim ierr As Double
    Dim herr As String = ""
    Dim MM As Double = NIST.MolarMass
    'Conversion des unités SI en unités NIST
    'La densité en kg/m³ doit être transformée en mol/L
    d = d / MM
    'L'énergie interne en J/kg doit être transformée en J/mol
    e = e / 1000 * MM
    Call DEFLSHdll(d, e, x(1), t, p, Dl, Dv, xliq(1), xvap(1), q, h, s, cv, cp, w, ierr, ✓
herr, 255&)
    If herr <> "" Or ierr <> 0 Then err = True Else err = False
    'Conversion des Unités NIST en unités SI
    'La pression en kPa doit être transformée en Pa
    p = p * 1000
    'Les densités en mol/L doit être transformées en kg/m³
    d = d * MM
    Dl = Dl * MM
    Dv = Dv * MM
    'Les J/mol doivent devenir des J/kg
    e = e / MM * 1000
    h = h / MM * 1000

```

```

'Les J/(mol*K) doivent devenir des J/(kg*K)
s = s / MM * 1000
cv = cv / MM * 1000
cp = cp / MM * 1000
End Sub
Public Sub THERM2SI(ByRef t As Double, ByRef d As Double, ByRef p As Double, _
ByRef e As Double, ByRef h As Double, ByRef s As Double, ByRef cv As Double, _
ByRef cp As Double, ByRef w As Double, ByRef beta As Double, ByRef dDdT As Double, _
ByRef dDdP As Double, ByRef err As Boolean)
Dim MM As Double = NIST.MolarMass
Dim ierr As Double
Dim herr As String = ""
'La densité en kg/m³ doit être transformée en mol/L
d = d / MM
Call THERM2dll(t, d, x(1), p, e, h, s, cv, cp, w, z, hgt, a, g, xkappa, beta, dPdD, d2PdD2, dPdT, dDdT, dDdP, spare1, spare2, spare3, spare4)
If herr <> "" Or ierr <> 0 Then err = True Else err = False
'Conversion des Unités NIST en unités SI
'Les densités en mol/L doit être transformées en kg/m³
d = d * MM
'La pression en kPa doit être transformée en Pa
p = p * 1000
'Les J/mol doivent devenir des J/kg
e = e / MM * 1000
h = h / MM * 1000
'Les J/(mol*K) doivent devenir des J/(kg*K)
s = s / MM * 1000
cv = cv / MM * 1000
cp = cp / MM * 1000
'Les mol/L doivent être transformées en kg/m³
dDdT = dDdT * MM
'Les mol/(L*kPa) doivent être transformées en kg/(m³Pa)
dDdP = dDdP * MM / 1000
End Sub
Public Function mu(ByRef t As Double, ByRef d As Double, ByRef err As Boolean) As Double
Dim MM As Double = NIST.MolarMass
Dim ierr As Double
Dim herr As String = ""
'La densité en kg/m³ doit être transformée en mol/L
d = d / MM
Call TRNPRPdll(t, d, x(1), eta, tcx, ierr, herr, 255)
If herr <> "" Or ierr <> 0 Then err = True Else err = False
'Les densités en mol/L doit être transformées en kg/m³
d = d * MM
'La viscosité en µPa.s doit être transformée en Pa.s
mu = eta * 10 ^ (-6)
End Function
Public Function MolarMass() As Double
Dim wm As Double
WMOLdll(x(1), wm)
MolarMass = wm
End Function
End Module

```

Module Output

'Ce module comporte des routines servant à simplifier l'écriture dans les fichiers de sortie

Public Sub Header(ByVal Path As String, ByVal ParamArray variables() As String)
'Cette routine sert à écrire l'en-tête d'un fichier, elle écrase l'information
'contenue dans le fichier s'il existe déjà.

Dim out As New _
 System.IO.FileStream(Path, _
 System.IO.FileMode.Create, _
 System.IO.FileAccess.Write, _
 System.IO.FileShare.ReadWrite)

Dim Writer As New System.IO.StreamWriter(out)

Dim line As String = ""

Dim lmnt As String

For Each lmnt In variables

 If line = "" Then

 line = lmnt

 Else

 line = line + ControlChars.Tab + lmnt

 End If

Next

Writer.WriteLine(line)

Writer.Close()

out.Close()

End Sub

Public Sub Save(ByVal Path As String, ByVal ParamArray variables() As String)

'Cette routine sert à ajouter une ligne de données au fichier.

Dim out As New _
 System.IO.FileStream(Path, _
 System.IO.FileMode.Append, _
 System.IO.FileAccess.Write, _
 System.IO.FileShare.ReadWrite)

Dim Writer As New System.IO.StreamWriter(out)

Dim line As String = ""

Dim lmnt As String

For Each lmnt In variables

 If line = "" Then

 line = lmnt

 Else

 line = line + ControlChars.Tab + CStr(lmnt)

 End If

Next

Writer.WriteLine(line)

Writer.Close()

out.Close()

End Sub

End Module

```

Module Param
    'Condition atmosphériques
    Public ReadOnly Patm As Double = 101353 'Pa
    Public ReadOnly Tamb As Double = 294.15 'K
    'Conditions initiales
    Public ReadOnly t0 As Double = 0
    Public ReadOnly Pt0 As Double = 12.1276 * 10 ^ 6 'Pa
    Public ReadOnly Tt0 As Double = 298.4678182 'K
    Public ReadOnly Lc As Double = 1.52 'm
    'Propriétés du réservoir
    Public ReadOnly Psi As Double = 3 / 1000 'm^3
    Public ReadOnly ro As Double = 5 / 100 'm
    Public ReadOnly Paroi As Double = 0.476637044 / 100 'm
    Public ReadOnly Lt As Double = 16 * 2.54 / 100 'Longueur du réservoir (m)
    Public ReadOnly ri As Double = ro - Paroi 'm
    Public ReadOnly hi As Double = 20
    Public ReadOnly ho As Double = 7
    Public ReadOnly kc As Double = 15
    Public ReadOnly nbcyl As Double = 1
    'Propriétés du conduit
    Public ReadOnly OD As Double = (1 / 16) * 2.54 / 100 'm
    Public ReadOnly wall As Double = 0.014 * 2.54 / 100
    Public ReadOnly ID As Double = OD - 2 * wall 'm
    Public ReadOnly Ac As Double = System.Math.PI * (ID / 2) ^ 2 'm^2
    Public ReadOnly epsilon As Double = 8 * 10 ^ (-7)
    Public ReadOnly mdf As Double = 0.02 'Mean D'Arcy friction coefficient
    'NIST
    Public ReadOnly FluidFile As String = "fluids\nitrogen.fld"
    Public ReadOnly upperT As Double = 2000 'K
    Public ReadOnly lowerT As Double = 63.151 'K
    Public ReadOnly upperP As Double = 2200000.0 * 1000 'Pa
    Public ReadOnly lowerP As Double = 50000 'Pa
    'Intégration
    Public h0reservoir As Double
    Public ReadOnly h0conduit As Double = Lc * 0.01
    Public ReadOnly Dconduit As Double = 1 / 10 ^ 9
    Public ReadOnly FTOL As Double = 0.00000001
    'Fichiers de sortie
    Public ReadOnly Reservoir As String = "Reservoir.tab"
    Public ReadOnly OutDeltaL As String = "DeltaL.tab"
    Public ReadOnly IntLmax As String = "IntLmax.tab"
    Public ReadOnly BisLmax As String = "BisLmax.tab"
    Public ReadOnly SODIFIG As String = "SODIFIG.tab"
    Public ReadOnly Darcy As String = "Darcy.tab"
    'Constantes thermodynamiques
    Public ReadOnly k As Double = 1.4
    Public ReadOnly R As Double = 593.606774 'J/(kg*K)
    Public ReadOnly Cpo As Double = 1040 'J/(kg*K)
    Public ReadOnly Cvo As Double = Cpo / k 'J/(kg*K)
    'Équation utilisée pour calculer le coefficient de friction
    Public ReadOnly equation As String = "vonKarman"
    ' "vonKarman" "Prandtl" "Colebrook" "Genereaux" "constant"
End Module

```

Module Reservoir

```

Function drhotdt(ByVal t As Double, ByVal rhot As Double, ByVal ut As Double) As Double
    'Fonction de la dérivée de la masse volumique du fluide à l'intérieur du réservoir
    selon le temps
    drhotdt = -mfr(rhot, ut) / Param.Psi
End Function
Function dutdt(ByVal t As Double, ByVal rhot As Double, ByVal ut As Double) As Double
    'Fonction de la dérivée de l'énergie interne massique du fluide à l'intérieur du
    réservoir selon le temps
    Dim Tt, Pt, ht, s, cvt, cpt, ct As Double
    Dim err As Boolean
    NIST.DEFLSHSI(rhot, ut, Tt, Pt, ht, s, cvt, cpt, ct, err)
    Dim mvc, dQvcdt As Double
    mvc = rhot * Param.Psi
    dQvcdt = nbcyl * (Param.Tamb - Tt) * (2 * System.Math.PI * ro * Param.Lt / (ro *
    System.Math.Log(ro / ri) / k + 1 / ho) + 1 / (1 / 4 * 1 / (ho * System.Math.PI * ro ^ 2)
    + 1 / 4 * (ro - ri) / (System.Math.PI * kc * ro * ri)))
    dutdt = (dQvcdt + mfr(rhot, ut) * (ut - ht)) / mvc
End Function
Function mfr(ByVal rhot As Double, ByVal ut As Double, _
Optional ByRef Ps As Double = Nothing, Optional ByRef Ts As Double = Nothing, _
Optional ByRef Vs As Double = Nothing) As Double
    'Fonction de calcul du débit en fonction des propriétés d'arrêt
    'La fonction renvoie également les propriétés du fluide à la sortie
    Dim err As Boolean
    Static lastrhot, lastut, lastmfr, lastPs, LastTs, LastVs As Double
    Static lMc, llMc As Double
    Dim cntr As Integer = 0
    Dim L0, Ps0, Ts0, Vs0, M0, DeltaL0 As Double
    Dim L1, Ps1, Ts1, Vs1, M1, DeltaL1 As Double
    Dim L2, Ps2, Ts2, Vs2, M2, DeltaL2 As Double
    Dim Tt, Pt, ht, s, cvt, cpt, ct As Double
    Dim Mc, ER As Double
    If rhot = lastrhot And ut = lastut Then
        'Si les paramètres d'entrées sont les mêmes que lors de la dernière utilisation
        de la fonction,
        'alors la fonction renvoie le même résultat
        mfr = lastmfr
        Ps = lastPs
        Ts = LastTs
        Vs = LastVs
    Else
        'On doit procéder par itération sur le nombre de Mach au col dans conduit.Lmax
        afin de converger
        'vers la longueur de conduit réelle
        'Calcul des propriétés d'arrêt
        NIST.DEFLSHSI(rhot, ut, Tt, Pt, ht, s, cvt, cpt, ct, err)
        'Initialisation du fichier de sortie
        Output.Header(Param.OutDeltaL, "Pt", "Tt", "Mc", "L", "DeltaL", "ER")
        'Valeur initiales pour la résolution par interpolation linéaire
        If lMc = 0 Or llMc = 0 Then
            If lMc = 0 And llMc = 0 Then
                'Si aucune valeur précédente n'est disponible pour Mc, alors on utilise
                'Mc pour un écoulement de Fanno et un autre Mc calculé à l'aide d'un
                'coefficient constant
                M0 = Fanno.Mc(Pt, Param.k, Patm, Param.ID, Param.mdf, Lc, FTOL)
                DeltaL0 = DeltaL(Pt, Tt, M0, Ps0, Ts0, Vs0, L0)
                Output.Save(Param.OutDeltaL, Pt, Tt, M0, L0, DeltaL0)
                Console.WriteLine(ControlChars.Tab + ControlChars.Tab + "(Mc, DeltaL) = "
                + "(" + CStr(M2) + " , " + CStr(DeltaL0) + ")")
                M1 = M0 * 0.99
                'M1 = M0 * L0 / Param.Lc
                DeltaL1 = DeltaL(Pt, Tt, M1, Ps1, Ts1, Vs1, L1)
                Output.Save(Param.OutDeltaL, Pt, Tt, M1, L1, DeltaL1)
                Console.WriteLine(ControlChars.Tab + ControlChars.Tab + "(Mc, DeltaL) = "
                + "(" + CStr(M2) + " , " + CStr(DeltaL1) + ")")
            Else
                'Si un seul Mc précédent est disponible, alors on l'utilise avec le Mc
                pour

```



```

        'un écoulement de Fanno
        M0 = Fanno.Mc(Pt, Param.k, Patm, Param.ID, Param.mdf, Lc, FTOL)
        DeltaL0 = DeltaL(Pt, Tt, M0, Ps0, Ts0, Vs0, L0)
        Output.Save(Param.OutDeltaL, Pt, Tt, M0, L0, DeltaL0)
        Console.WriteLine(ControlChars.Tab + ControlChars.Tab + "(Mc, DeltaL) = " &
+ "(" + CStr(M2) + " , " + CStr(DeltaL0) + ")")
        M1 = lMc
        DeltaL1 = DeltaL(Pt, Tt, M1)
    End If
Else
    'Si deux Mc précédents sont disponibles, alors on les utilise comme première
    'approximation à la méthode de la sécante
    M0 = llMc
    DeltaL0 = DeltaL(Pt, Tt, M0)
    M1 = lMc
    DeltaL1 = DeltaL(Pt, Tt, M1)
End If
'Résolution d'équation par la méthode de la sécante
If System.Math.Abs(DeltaL0) < System.Math.Abs(DeltaL1) Then
    Dim Mi, DeltaLi As Double
    Mi = M0
    M0 = M1
    M1 = Mi
    DeltaLi = DeltaL0
    DeltaL0 = DeltaL1
    DeltaL1 = DeltaLi
End If
Do
    M2 = M1 - DeltaL1 * (M0 - M1) / (DeltaL0 - DeltaL1)
    If M2 > 1 Then System.Diagnostics.Debugger.Break()
    DeltaL2 = DeltaL(Pt, Tt, M2, Ps2, Ts2, Vs2, L2)
    ER = System.Math.Abs(DeltaL2 / Param.Lc)
    Output.Save(Param.OutDeltaL, Pt, Tt, M2, L2, DeltaL2, ER)
    Console.WriteLine(ControlChars.Tab + ControlChars.Tab + "(Mc, DeltaL) = " &
+ "(" + CStr(M2) + " , " + CStr(DeltaL2) + ")")
    M0 = M1
    DeltaL0 = DeltaL1
    M1 = M2
    DeltaL1 = DeltaL2
    cntr = cntr + 1
    If cntr > 50 Then System.Diagnostics.Debugger.Break()
Loop Until ER < 1 / 1000
Mc = M2
Ps = Ps2
Ts = Ts2
Vs = Vs2
mfr = rhoPT(Ps2, Ts2) * Vs2 * Param.Ac
llMc = lMc
lMc = Mc
End If
'Mise en mémoire des dernières valeurs pour le calcul suivant
lastrhot = rhot
lastut = ut
lastmfr = mfr
lastPs = Ps
LastTs = Ts
LastVs = Vs
Console.WriteLine(ControlChars.Tab + ControlChars.Tab + "Mass Flow Rate = " & CStr
(mfr))
End Function
Function DeltaL(ByVal Pt As Double, ByVal Tt As Double, ByVal Mc As Double, _
Optional ByRef Ps As Double = 0, Optional ByRef Ts As Double = 0, Optional ByRef Vs As
Double = 0, _
Optional ByRef L As Double = 0) As Double
    'Fonction de calcul de la différence entre la longueur de conduit correspondant à un
Mc donné
    'et la longueur de conduit réelle (param.Lc)
    Dim Pc, Tc, Vc As Double
    SODIFIG.PTV(Pt, Tt, Mc, Pc, Tc, Vc)

```

```

    L = Lmax(Pc, Tc, Vc, Ps, Ts, Vs)
    DeltaL = L - Param.Lc
End Function
Function rhoPT(ByVal P As Double, ByVal T As Double) As Double
    'Fonction de calcul de la masse volumique en fonction de la pression et de la
    température
    If T < Param.lowerT Or T > Param.upperT Or P < Param.lowerP Or P > Param.upperP _
    Then System.Diagnostics.Debugger.Break()
    Dim rho, u, h, s, cv, cp, w As Double
    Dim err As Boolean
    NIST.TPFLSHSI(T, P, rho, u, h, s, cv, cp, w, err)
    rhoPT = rho
End Function
Sub RKF452Step(ByVal t As Double, ByVal rhot As Double, ByVal ut As Double, _
ByVal ER As Double, ByVal h As Double, ByRef StopHere As Boolean)
    'Fonction s'exécutant après chaque boucle de RKF452
    'Post-traitement des résultats
    'Propriétés d'arrêt
    Dim Tt, Pt, ht, sr, cvt, cpt, ct As Double
    Dim Ts, Ps, rhos, us, hs, ss, cvs, cps, cs, Vs, Ms, MassFlowRate As Double
    Dim err As Boolean
    NIST.DEFLSHSI(rhot, ut, Tt, Pt, ht, sr, cvt, cpt, ct, err)
    'Propriétés à la sortie
    MassFlowRate = mfr(rhot, ut, Ps, Ts, Vs)
    NIST.TPFLSHSI(Ts, Ps, rhos, us, hs, ss, cvs, cps, cs, err)
    Ms = Vs / cs
    If t = 0 Then
        'Écriture de l'en-tête au fichier de sortie s'il s'agit du premier pas d
        'intégration
        Output.Header(Param.Reservoir, "t", "PtMPa", "Tt", "MFRgs", "rhot", "ut", "h",
        "ER", "Ps", "Ts", "Vs", "Ms")
    End If
    'Écriture des résultats au fichier de sortie et à la console
    Output.Save(Param.Reservoir, t, Pt / 10 ^ 6, Tt, MassFlowRate * 1000, rhot, ut, h, ER
    , Ps, Ts, Vs, Ms)
    Console.WriteLine("Mass flow rate at " + CStr(t) + "s: " + CStr(MassFlowRate * 1000)
    + "g/s")
    If Pt <= Param.Patm * 1.01 Then
        'Si la pression dans le réservoir est près de la pression arrière, alors on arrête
        l'intégration
        System.Diagnostics.Debugger.Break()
        StopHere = True
    End If
End Sub
End Module

```

Module RKF452

```

'Module d'intégration numérique par la méthode de Runge-Kutta-Fehlberg à 2 équations
Delegate Function ftxy(ByVal t As Double, ByVal x As Double, ByVal y As Double) As Double
Delegate Sub RKF452Step(ByVal t As Double, ByVal x As Double, ByVal y As Double, ByVal ER As Double, ByVal h As Double, ByRef StopHere As Boolean)
Sub RKF452(ByVal f As ftxy, ByVal g As ftxy, _
    ByVal t0 As Double, ByVal x0 As Double, ByVal y0 As Double, _
    ByVal h0 As Double, ByVal D As Double, ByVal RKF452step As RKF452Step)
    Dim k1x, k2x, k3x, k4x, k5x, k6x As Double
    Dim k1y, k2y, k3y, k4y, k5y, k6y As Double
    Dim t, h, xc, x, yc, y As Double
    Dim Ex, Ey As Double
    Dim ERx, ERy, ER As Double
    Dim StopHere As Boolean
    'Initialisation des variables et du pas
    t = t0
    x = x0
    y = y0
    h = h0
    RKF452step(t, x, y, ER, h, StopHere)
    Do
        k1x = h * f(t, x, y)
        Console.WriteLine(ControlChars.Tab + "k1x = " + CStr(k1x) + " @ " + CStr(t) + "s"✓
    )
        k1y = h * g(t, x, y)
        Console.WriteLine(ControlChars.Tab + "k1y = " + CStr(k1y) + " @ " + CStr(t) + "s"✓
    )

        k2x = h * f(t + 1 / 4 * h, x + 1 / 4 * k1x, y + 1 / 4 * k1y)
        Console.WriteLine(ControlChars.Tab + "k2x = " + CStr(k2x) + " @ " + CStr(t) + "s"✓
    )
        k2y = h * g(t + 1 / 4 * h, x + 1 / 4 * k1x, y + 1 / 4 * k1y)
        Console.WriteLine(ControlChars.Tab + "k2y = " + CStr(k2y) + " @ " + CStr(t) + "s"✓
    )

        k3x = h * f(t + 3 / 8 * h, x + 3 / 32 * k1x + 9 / 32 * k2x, y + 3 / 32 * k1y + 9 / 32 * k2y)
        Console.WriteLine(ControlChars.Tab + "k3x = " + CStr(k3x) + " @ " + CStr(t) + "s"✓
    )
        k3y = h * g(t + 3 / 8 * h, x + 3 / 32 * k1x + 9 / 32 * k2x, y + 3 / 32 * k1y + 9 / 32 * k2y)
        Console.WriteLine(ControlChars.Tab + "k3y = " + CStr(k3y) + " @ " + CStr(t) + "s"✓
    )

        k4x = h * f(t + 12 / 13 * h, x + 1932 / 2197 * k1x - 7200 / 2197 * k2x + 7296 / 2197 * k3x, y + 1932 / 2197 * k1y - 7200 / 2197 * k2y + 7296 / 2197 * k3y)
        Console.WriteLine(ControlChars.Tab + "k4x = " + CStr(k4x) + " @ " + CStr(t) + "s"✓
    )
        k4y = h * g(t + 12 / 13 * h, x + 1932 / 2197 * k1x - 7200 / 2197 * k2x + 7296 / 2197 * k3x, y + 1932 / 2197 * k1y - 7200 / 2197 * k2y + 7296 / 2197 * k3y)
        Console.WriteLine(ControlChars.Tab + "k4y = " + CStr(k4y) + " @ " + CStr(t) + "s"✓
    )

        k5x = h * f(t + h, x + 439 / 216 * k1x - 8 * k2x + 3680 / 513 * k3x - 845 / 4104 * k4x, y + 439 / 216 * k1y - 8 * k2y + 3680 / 513 * k3y - 845 / 4104 * k4y)
        Console.WriteLine(ControlChars.Tab + "k5x = " + CStr(k5x) + " @ " + CStr(t) + "s"✓
    )
        k5y = h * g(t + h, x + 439 / 216 * k1x - 8 * k2x + 3680 / 513 * k3x - 845 / 4104 * k4x, y + 439 / 216 * k1y - 8 * k2y + 3680 / 513 * k3y - 845 / 4104 * k4y)
        Console.WriteLine(ControlChars.Tab + "k5y = " + CStr(k5y) + " @ " + CStr(t) + "s"✓
    )

        k6x = h * f(t + 1 / 2 * h, x - 8 / 27 * k1x + 2 * k2x - 3544 / 2565 * k3x + 1859 / 4104 * k4x - 11 / 40 * k5x, y - 8 / 27 * k1y + 2 * k2y - 3544 / 2565 * k3y + 1859 / 4104 * k4y - 11 / 40 * k5y)
        Console.WriteLine(ControlChars.Tab + "k6x = " + CStr(k6x) + " @ " + CStr(t) + "s"✓
    )
        k6y = h * g(t + 1 / 2 * h, x - 8 / 27 * k1x + 2 * k2x - 3544 / 2565 * k3x + 1859 / 4104 * k4x - 11 / 40 * k5x, y - 8 / 27 * k1y + 2 * k2y - 3544 / 2565 * k3y + 1859 / 4104 * k4y - 11 / 40 * k5y)
        Console.WriteLine(ControlChars.Tab + "k6y = " + CStr(k6y) + " @ " + CStr(t) + "s"✓
    )
    Loop

```

```

4104 * k4y - 11 / 40 * k5y)
    Console.WriteLine(ControlChars.Tab + "k6y = " + CStr(k6y) + " @ " + CStr(t) + "s"
)
    'Solution
    t = t + h
    xc = x + (25 / 216 * k1x + 1408 / 2565 * k3x + 2197 / 4104 * k4x - 1 / 5 * k5x)
    x = x + (16 / 135 * k1x + 6656 / 12825 * k3x + 28561 / 56430 * k4x - 9 / 50 * k5x
+ 2 / 55 * k6x)
    yc = y + (25 / 216 * k1y + 1408 / 2565 * k3y + 2197 / 4104 * k4y - 1 / 5 * k5y)
    y = y + (16 / 135 * k1y + 6656 / 12825 * k3y + 28561 / 56430 * k4y - 9 / 50 * k5y
+ 2 / 55 * k6y)
    'Calcul de l'erreur absolue et de l'erreur relative
    Ex = System.Math.Abs(xc - x)
    Ey = System.Math.Abs(yc - y)
    ERx = System.Math.Abs(Ex / x)
    Ery = System.Math.Abs(Ey / y)
    'Déterminer l'erreur relative la plus grande
    If ERx >= Ery Then ER = ERx Else ER = Ery
    'Adaptive step size control
    h = h_new(ER, D, h)
    'Adjonction des résultats
    RKF452step(t, x, y, ER, h, StopHere)
Loop Until StopHere = True
End Sub
Function h_new(ByVal ER As Double, ByVal D As Double, ByVal h As Double) As Double
    'Fonction de contrôle adaptatif du pas d'intégration en fonction de la dernière
    erreur obtenue
    Dim ER_D As Double
    Dim q As Double = 4
    Dim S As Double = 0.9
    ER_D = System.Math.Abs(ER) / D
    If ER_D > 1.1 Then
        'Le pas doit être diminué
        If S * ER_D ^ (-1 / q) >= 5 Then
            'Augmentation du pas limitée à 5 fois sa valeur précédente
            h_new = h * 5
        ElseIf S * ER_D ^ (-1 / q) <= 1 / 5 Then
            'Diminution du pas limitée à 1/5 de sa valeur précédente
            h_new = h * 1 / 5
        Else
            'Règle de diminution du pas
            h_new = h * S * ER_D ^ (-1 / q)
        End If
    ElseIf ER_D < 0.5 Then
        'Le pas peut être augmenté
        If S * ER_D ^ (-1 / (q + 1)) >= 5 Then
            h_new = h * 5
        ElseIf S * ER_D ^ (-1 / (q + 1)) <= 1 / 5 Then
            h_new = h * 1 / 5
        Else
            'Règle d'augmentation du pas
            h_new = h * S * ER_D ^ (-1 / (q + 1))
        End If
    Else
        h_new = h
    End If
End Function
End Module

```

Module RKF453

```

'Module d'intégration numérique par la méthode de Runge-Kutta-Fehlberg à 3 équations
Delegate Function ftxyz(ByVal t As Double, ByVal x As Double, ByVal y As Double, ByVal z As Double) As Double
Sub RKF453(ByVal f1 As ftxyz, ByVal f2 As ftxyz, ByVal f3 As ftxyz, _
    ByRef t As Double, ByRef x As Double, ByRef y As Double, _
    ByRef z As Double, ByRef h As Double, ByRef ER As Double)
    Dim k1x, k2x, k3x, k4x, k5x, k6x As Double
    Dim k1y, k2y, k3y, k4y, k5y, k6y As Double
    Dim k1z, k2z, k3z, k4z, k5z, k6z As Double
    Dim xc, yc, zc As Double
    Dim Ex, Ey, Ez As Double
    Dim ERx, ERy, ERz As Double
    Dim D As Double = ER
    'Initialisation des variables et du pas
    k1x = h * f1(t, x, y, z)
    k1y = h * f2(t, x, y, z)
    k1z = h * f3(t, x, y, z)
    k2x = h * f1(t + 1 / 4 * h, x + 1 / 4 * k1x, y + 1 / 4 * k1y, z + 1 / 4 * k1z)
    k2y = h * f2(t + 1 / 4 * h, x + 1 / 4 * k1x, y + 1 / 4 * k1y, z + 1 / 4 * k1z)
    k2z = h * f3(t + 1 / 4 * h, x + 1 / 4 * k1x, y + 1 / 4 * k1y, z + 1 / 4 * k1z)
    k3x = h * f1(t + 3 / 8 * h, x + 3 / 32 * k1x + 9 / 32 * k2x, y + 3 / 32 * k1y + 9 / 32 * k2y, z + 3 / 32 * k1z + 9 / 32 * k2z)
    k3y = h * f2(t + 3 / 8 * h, x + 3 / 32 * k1x + 9 / 32 * k2x, y + 3 / 32 * k1y + 9 / 32 * k2y, z + 3 / 32 * k1z + 9 / 32 * k2z)
    k3z = h * f3(t + 3 / 8 * h, x + 3 / 32 * k1x + 9 / 32 * k2x, y + 3 / 32 * k1y + 9 / 32 * k2y, z + 3 / 32 * k1z + 9 / 32 * k2z)
    k4x = h * f1(t + 12 / 13 * h, x + 1932 / 2197 * k1x - 7200 / 2197 * k2x + 7296 / 2197 * k3x, y + 1932 / 2197 * k1y - 7200 / 2197 * k2y + 7296 / 2197 * k3y, z + 1932 / 2197 * k1z - 7200 / 2197 * k2z + 7296 / 2197 * k3z)
    k4y = h * f2(t + 12 / 13 * h, x + 1932 / 2197 * k1x - 7200 / 2197 * k2x + 7296 / 2197 * k3x, y + 1932 / 2197 * k1y - 7200 / 2197 * k2y + 7296 / 2197 * k3y, z + 1932 / 2197 * k1z - 7200 / 2197 * k2z + 7296 / 2197 * k3z)
    k4z = h * f3(t + 12 / 13 * h, x + 1932 / 2197 * k1x - 7200 / 2197 * k2x + 7296 / 2197 * k3x, y + 1932 / 2197 * k1y - 7200 / 2197 * k2y + 7296 / 2197 * k3y, z + 1932 / 2197 * k1z - 7200 / 2197 * k2z + 7296 / 2197 * k3z)
    k5x = h * f1(t + h, x + 439 / 216 * k1x - 8 * k2x + 3680 / 513 * k3x - 845 / 4104 * k4x, y + 439 / 216 * k1y - 8 * k2y + 3680 / 513 * k3y - 845 / 4104 * k4y, z + 439 / 216 * k1z - 8 * k2z + 3680 / 513 * k3z - 845 / 4104 * k4z)
    k5y = h * f2(t + h, x + 439 / 216 * k1x - 8 * k2x + 3680 / 513 * k3x - 845 / 4104 * k4x, y + 439 / 216 * k1y - 8 * k2y + 3680 / 513 * k3y - 845 / 4104 * k4y, z + 439 / 216 * k1z - 8 * k2z + 3680 / 513 * k3z - 845 / 4104 * k4z)
    k5z = h * f3(t + h, x + 439 / 216 * k1x - 8 * k2x + 3680 / 513 * k3x - 845 / 4104 * k4x, y + 439 / 216 * k1y - 8 * k2y + 3680 / 513 * k3y - 845 / 4104 * k4y, z + 439 / 216 * k1z - 8 * k2z + 3680 / 513 * k3z - 845 / 4104 * k4z)
    k6x = h * f1(t + 1 / 2 * h, x - 8 / 27 * k1x + 2 * k2x - 3544 / 2565 * k3x + 1859 / 4104 * k4x - 11 / 40 * k5x, y - 8 / 27 * k1y + 2 * k2y - 3544 / 2565 * k3y + 1859 / 4104 * k4y - 11 / 40 * k5y, z - 8 / 27 * k1z + 2 * k2z - 3544 / 2565 * k3z + 1859 / 4104 * k4z - 11 / 40 * k5z)
    k6y = h * f2(t + 1 / 2 * h, x - 8 / 27 * k1x + 2 * k2x - 3544 / 2565 * k3x + 1859 / 4104 * k4x - 11 / 40 * k5x, y - 8 / 27 * k1y + 2 * k2y - 3544 / 2565 * k3y + 1859 / 4104 * k4y - 11 / 40 * k5y, z - 8 / 27 * k1z + 2 * k2z - 3544 / 2565 * k3z + 1859 / 4104 * k4z - 11 / 40 * k5z)
    k6z = h * f3(t + 1 / 2 * h, x - 8 / 27 * k1x + 2 * k2x - 3544 / 2565 * k3x + 1859 / 4104 * k4x - 11 / 40 * k5x, y - 8 / 27 * k1y + 2 * k2y - 3544 / 2565 * k3y + 1859 / 4104 * k4y - 11 / 40 * k5y, z - 8 / 27 * k1z + 2 * k2z - 3544 / 2565 * k3z + 1859 / 4104 * k4z - 11 / 40 * k5z)
    'Solution
    t = t + h
    xc = x + (25 / 216 * k1x + 1408 / 2565 * k3x + 2197 / 4104 * k4x - 1 / 5 * k5x)
    x = x + (16 / 135 * k1x + 6656 / 12825 * k3x + 28561 / 56430 * k4x - 9 / 50 * k5x + 2 / 55 * k6x)
    yc = y + (25 / 216 * k1y + 1408 / 2565 * k3y + 2197 / 4104 * k4y - 1 / 5 * k5y)
    y = y + (16 / 135 * k1y + 6656 / 12825 * k3y + 28561 / 56430 * k4y - 9 / 50 * k5y + 2 / 55 * k6y)
    zc = z + (25 / 216 * k1z + 1408 / 2565 * k3z + 2197 / 4104 * k4z - 1 / 5 * k5z)
    z = z + (16 / 135 * k1z + 6656 / 12825 * k3z + 28561 / 56430 * k4z - 9 / 50 * k5z + 2 / 55 * k6z)
    'Calcul de l'erreur absolue et de l'erreur relative

```

```

    Ex = System.Math.Abs(xc - x)
    Ey = System.Math.Abs(yc - y)
    Ez = System.Math.Abs(zc - z)
    ERx = System.Math.Abs(Ex / x)
    Ery = System.Math.Abs(Ey / y)
    ERz = System.Math.Abs(Ez / z)
    'Déterminer l'erreur relative la plus grande
    If ERx >= Ery And ERx >= ERz Then ER = ERx
    If Ery >= ERx And Ery >= ERz Then ER = Ery
    If ERz >= ERx And ERz >= Ery Then ER = ERz
    'Adaptive step size control
    h = h_new(ER, D, h)
End Sub
Function h_new(ByVal ER As Double, ByVal D As Double, ByVal h As Double) As Double
    'Voir RKF452 pour commentaires
    Dim ER_D As Double
    Dim q As Double = 4
    Dim S As Double = 0.9
    ER_D = System.Math.Abs(ER) / D
    If ER_D > 1.1 Then
        If S * ER_D ^ (-1 / q) >= 5 Then
            h_new = h * 5
        ElseIf S * ER_D ^ (-1 / q) <= 1 / 5 Then
            h_new = h * 1 / 5
        Else
            h_new = h * S * ER_D ^ (-1 / q)
        End If
    ElseIf ER_D < 0.5 Then
        If S * ER_D ^ (-1 / (q + 1)) >= 5 Then
            h_new = h * 5
        ElseIf S * ER_D ^ (-1 / (q + 1)) <= 1 / 5 Then
            h_new = h * 1 / 5
        Else
            h_new = h * S * ER_D ^ (-1 / (q + 1))
        End If
    Else
        h_new = h
    End If
End Function
End Module

```

Module SODIFIG

```

'SODIFIG est un acronyme pour : «Steady One-Dimensional Isentropic Flow of an Imperfect Gas» ✓
'Module de calcul des propriétés du fluide au col en fonction de sa vitesse à cet endroit ✓
et
'des propriétés d'arrêt
Dim M, s, ht As Double
Sub PTV(ByVal Pt As Double, ByVal Tt As Double, ByVal Mc As Double, _
Optional ByRef Pc As Double = 0, Optional ByRef Tc As Double = 0, _
Optional ByRef Vc As Double = 0)
    'Calcul de la pression, de la température et de la vitesse au col en fonction
    'des propriétés d'arrêt et du nombre de Mach au col par méthode numérique
    If Mc > 1 Then System.Diagnostics.Debugger.Break()
    Dim cntr As Integer = 0
    Dim rhot, ut, cvt, cpt, ct As Double
    Dim rhoc, uc, hc, cvc, cpc, cc As Double
    Dim P0, DeltaM0 As Double
    Dim P1, DeltaM1 As Double
    Dim P2, DeltaM2 As Double
    Dim err As Boolean
    'Écriture de données dans un fichier pour déboguage
    Output.Header(Param.SODIFIG, "Pt", "Tt", "Mc")
    Output.Save(Param.SODIFIG, Pt, Tt, Mc)
    M = Mc
    'Calcul de l'entropie
    NIST.TPFLSHSI(Tt, Pt, rhot, ut, ht, s, cvt, cpt, ct, err)
    'Première approximation pour la pression : Gaz parfait
    P0 = Isentropic.p_p0(Mc, Param.k) * Pt
    DeltaM0 = DeltaM(P0)
    'Deuxième approximation : interpolation linéaire à partir de zéro
    P1 = P0 * (Mc + DeltaM0) / Mc
    DeltaM1 = DeltaM(P1)
    If DeltaM0 < DeltaM1 Then
        Dim buf As Double
        buf = P0
        P0 = P1
        P1 = buf
    End If
    Output.Save(Param.SODIFIG, "P0", "DeltaM0", "P1", "DeltaM1", "P2", "DeltaM2")
    Do
        'Méthode de la sécante
        DeltaM1 = DeltaM(P1)
        DeltaM0 = DeltaM(P0)
        P2 = P1 - DeltaM1 * (P0 - P1) / (DeltaM0 - DeltaM1)
        DeltaM2 = DeltaM(P2)
        Output.Save(Param.SODIFIG, P0, DeltaM0, P1, DeltaM1, P2, DeltaM2)
        P0 = P1
        P1 = P2
        cntr = cntr + 1
        If cntr > 500 Then System.Diagnostics.Debugger.Break()
    Loop Until System.Math.Abs(DeltaM2) < 0.00001
    Pc = P2
    NIST.PSFLSHSI(Pc, s, Tc, rhoc, uc, hc, cvc, cpc, cc, err)
    Vc = Mc * cc
    'Écriture de données dans un fichier pour déboguage
    Output.Save(Param.SODIFIG, "Pc", "Tc", "Vc")
    Output.Save(Param.SODIFIG, Pc, Tc, Vc)
End Sub
Function DeltaM(ByVal P As Double) As Double
    'Fonction qui renvoie la différence entre le nombre de Mach spécifié et le nombre de ✓
Mach
    'calculé en fonction de la pression au col.
    Dim t, rho, u, h, cv, cp, c As Double
    Dim err As Boolean
    NIST.PSFLSHSI(P, s, t, rho, u, h, cv, cp, c, err)
    DeltaM = M ^ 2 - (2 * (ht - h)) / (c ^ 2)
End Function
End Module

```


Bibliographie

- [1] B. Angers, A. Hourri, P. Bénard, P. Tessier, and J. Perrin. Simulations of hydrogen releases from a storage tanks : Dispersion and consequences of ignition. In *Proceedings of 1st International Conference on Hydrogen Safety*, 2005.
- [2] E. Gallego et al. An intercomparison exercise on the capabilities of cfd models to predict distribution and mixing of h_2 in a closed vessel. In *Proceedings of 1st International Conference on Hydrogen Safety*, 2005.
- [3] K. Mohamed and M. Paraschivoiu. Real gas simulation of hydrogen release from a high pressure chamber. *International Journal of Hydrogen Energy*, 30 :903–912, 2005.
- [4] Young-Do Jo and Bum Jong Ahn. A simple model for the release rate of hazardous gas from a hole on high-pressure pipelines. *Journal of Hazardous Materials*, A97 :31–46, 2003.
- [5] Gordon J. Van Wylen, Richard E. Sonntag, and Pierre Desrochers. *Thermodynamique appliquée*. Éditions du renouveau pédagogique, 1992.
- [6] Maurice J. Zucrow and Joe D. Hoffman. *Gas Dynamics*. J. Wiley, 1976.
- [7] Frank Kreith and Mark S. Bohn. *Principles of Heat Transfer*. Brooks/Cole, 2001.
- [8] James E. A. John. *Gas Dynamics*. Allyn and Bacon, 1969.