Projet

ENE6002

Thermohydraulique des systèmes diphasiques

–

Polytechnique Montréal

Thomas Charpentier - 2036024
thomas.charpentier@polymtl.ca

Gaétan Raynaud - 2022063
gaetan.raynaud@polymtl.ca

24 avril 2020

Abstract

Pour de nombreuses applications industrielles, une bonne estimation de la perte de pression d'un écoulement dans une conduite est nécessaire afin de préparer, suivre et contrôler au mieux le fonctionnement d'un grand nombre de machines. Cette perte de pression a tendance à s'accentuer lorsque l'écoulement est diphasique, là où la DNS devient encore plus complexe. Des modèles de corrélation remplacent et simplifient certaines équations du problème et permettent de mener des simulations numériques unidimensionnelles. Les corrélations de Inoue et Chexal sont utilisées dans le modèle à vitesse séparée ainsi que la modélisation de frottement interfacial de Friedel. Ces modèles s'intègrent dans un set d'équations couplées et non linéaires, dont on propose deux algorithmes de résolution. Le premier est un algorithme dit « classique » avec une discrétisation spatiale uniforme et une convergence par itération directe. Le second type d'algorithme est basé sur un modèle très récent de *Physics Informed Neural Networks* et présente un certain nombre d'avantages pratiques : absence de maillage, dérivation exacte possible des quantités calculées, des propriétés thermodynamiques ou toute combinaison de fonction. Il montre également certaines limites en termes de robustesse qui seront discutées. Les calculs sont effectués dans les configurations de deux campagnes expérimentales ce qui permet de comparer les résultats avec des écarts variant entre 1 et 50% sur la perte de pression.

Table des matières

1 Introduction et présentation de l'étude

1.1 Introduction

Dans les applications industrielles, dès lors que l'on travaille avec un écoulement dans un conduite chauffée, on peut observer en parallèle de la phase liquide l'apparition de vapeur. Entre ces deux phases il s'échange de la masse, de l'énergie et de la quantité de mouvement et le désordre généré s'accompagne généralement d'une chute de pression. Si la conduite est suffisamment longue, la prise en compte des pertes de charges dans un modèle monophasique risque d'être peu représentatif de la réalité, les pertes liées à l'ébullition n'étant pas négligeables.

Cette sous-estimation de la perte de pression lors du dimensionnement peut avoir des répercussions sur le rendement de l'installation ou bien sa sécurité. Dans le cas d'un réacteur à eau pressurisé, il est important de connaître les conditions de l'écoulement à tous les endroits de la conduite. On ne souhaite pas se retrouver avec un écoulement diphasique dans une zone où cela n'était pas prévu, avec des conséquences majeures pour la sécurité. De plus les différents scénarios d'accidents (brèche, fusion du coeur...) peuvent placer le système dans des configurations favorables à l'apparition de vapeur qu'il s'agit d'anticiper et de designer convenablement pour s'assurer des marges.

Dans le domaine des nouvelles technologies, la micro électronique peut tirer parti d'une bonne utilisation de micro canaux pour dissiper l'énergie lorsque ceci est compliqué (satellites, micro puces...). Ceux-ci possèdent en effet de bien meilleurs coefficients de transfert thermique que les méthodes conventionnelles ce qui permet d'en réduire la taille et potentiellement le coût de fabrication [Revellin and Thome, 2007]. Néanmoins la chute de pression doit être évaluée précisément pour assurer un fonctionnement optimal du système de dissipation et cela se traduit par des modèles uni-dimensionnel pour des écoulements diphasiques.

1.2 Présentation de l'étude

Pour ce projet, nous nous intéressons à la perte de pression ainsi qu'au taux de vide moyen pour un écoulement bouillant dans une conduite chauffée. Une représentation d'un banc d'essai typique pour ce type de mesures est disponible à la fig. 1

Le but de ce projet est de réaliser un modèle numérique et de déterminer les différents paramètres le long de la conduite. L'analyse des résultats se fait en comparant à des tables de données expérimentales. On travaille dans le cas d'un écoulement vertical ascendant.

Pour réaliser cela, plusieurs hypothèses ont été réalisées :
- **H1** : L'écoulement est incompressible.
- **H2** : La perte de pression par accélération dans la région non bouillante est négligeable.
- **H3** : La perte de pression totale est négligeable par rapport à la pression de l'écoulement.
- **H4** : L'enthalpie de l'eau sous-refroidie à une température donnée est égale à l'enthalpie de saturation de l'eau correspondant à cette température

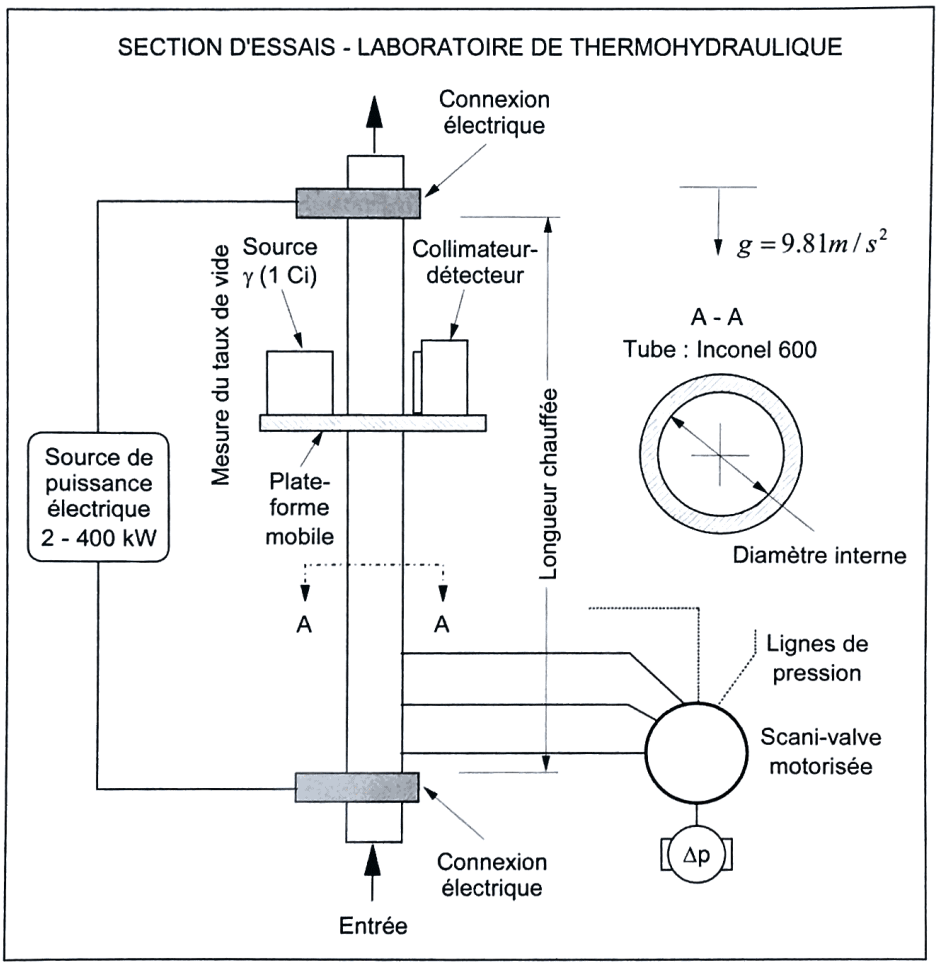

FIGURE 1 – Installation pour mesure de perte de pression et du taux de vide - D'après l'énoncé

La détermination des propriétés thermodynamiques a été faite avec `pyXSteam`, qui est une librairie Python sur la base de `X-Steam` disponible sur Excel et Matlab.
Le titre et le taux de vide moyen sont déterminés à l'aide des corrélations de CHEXAL-LELLOUCHE et de INOUE. Le multiplicateur diphasique ainsi que les facteurs de frottements qui servent au calcul de la perte de pression proviennent eux de la corrélation de FRIEDEL.

2 Méthode de résolution

Afin de trouver la perte de pression dans la conduite, il convient de définir les équations qui régissent le problème.

2.1 Équations du problème

Les différentes équations dont on dispose sont :
des équations d'état lors que le mélange est à saturation :

$$T_{sat} = T_{sat}(P) \tag{1}$$
$$h_{l,sat} = h_{l,sat}(T) = h_{l,sat}(P) \tag{2}$$
$$h_{v,sat} = h_{v,sat}(T) = h_{v,sat}(P) \tag{3}$$

Des lois de comportement, notamment pour calculer la perte de pression

$$-\frac{\partial p}{\partial z} = \left(\frac{\partial p}{\partial z}\right)_a + \left(\frac{\partial p}{\partial z}\right)_f + \left(\frac{\partial p}{\partial z}\right)_g \tag{4}$$

Que l'on peut écrire comme :

$$-\frac{\partial p}{\partial z} = \underbrace{\left[\overbrace{\frac{\partial}{\cancel{\partial \tau}}\cancel{G}}^{\text{Ecoul. permanent}} + \frac{\partial}{\partial z}G^2 v'\right]}_{\text{Acceleration}} + \underbrace{\frac{4\tau_w}{D}}_{\text{Frottements}} + \underbrace{\rho_m g}_{\text{Gravitation}} \tag{5}$$

Avec :

$$v' = \frac{x^2}{\varepsilon\rho_v} + \frac{(1-x)^2}{(1-\varepsilon)\rho_l} \tag{6}$$

Etant donné la complexité de gérer les cas $\varepsilon \to 0$ et $\varepsilon \to 1$ dans l'équation précédente, on a utilisé une équation dérivée du modèle homogène pour v' :

$$v' = \frac{1}{\rho_m} = \frac{1}{\varepsilon\rho_v + (1-\varepsilon)\rho_l} \tag{7}$$

Le terme de frottement pouvant être écrit comme la perte de pression (par frottement) si l'écoulement était seulement liquide et une correction, le « multiplicateur diphasique ». Ce dernier sera obtenu par la corrélation de Friedel rappelée dans [Revellin and Thome, 2007].

$$\left(\frac{\partial p}{\partial z}\right)_f = f\frac{G^2}{2\rho_l}\frac{1}{D}\phi_{lo}^2 \tag{8}$$

Comme rappelé par [Revellin and Thome, 2007], le facteur de frottement f est défini par :

$$f = 0.316\left[\mathrm{Re}_L\right]^{-0.25} = 0.316\left[\frac{GD}{\mu_l}\right]^{-0.25} \tag{9}$$

On peut donc écrire la perte de pression que nous allons évaluer comme :

$$-\frac{\partial p}{\partial z} = G^2 \frac{\partial}{\partial z}\left[\frac{1}{\varepsilon\rho_v + (1-\varepsilon)\rho_l}\right] + f\frac{G^2}{2\rho_l}\frac{1}{D}\phi_{lo}^2 + \left[\varepsilon\rho_v + (1-\varepsilon)\rho_l\right] g \tag{10}$$

On dispose ensuite d'une relation entre le taux de vide moyen ε et le titre de l'écoulement x provenant du modèle à vitesses séparées. Il est accompagné de lois de comportements pour fixer les constantes de corrélation :

$$\frac{1}{\varepsilon} = \frac{\rho_g}{\hat{x}G} < V_{gj} >_{2g} + C_0\left(\frac{\rho_g}{\rho_l}\frac{1-\hat{x}}{\hat{x}} + 1\right) \tag{11}$$

$$C_0 \text{ fonction des paramètres de l'écoulement}$$
$$< V_{gj} >_{2g} \text{ fonction des paramètres de l'écoulement}$$

En pratique on l'utilisera de deux manières. Dans l'algorithme classique, on s'en sert pour obtenir ε à partir de x et p en inversant l'équation 11. Dans le PINN, on utilise plutôt la forme suivante où on retourne x à partir d'un set initial de ε, x et p :

$$x = \varepsilon\left[\frac{\rho_g}{G} < V_{gj} >_{2g} + C_0\left(\frac{1-x}{\rho_l}\rho_g + x\right)\right] \tag{12}$$

Il faut bien noter que C_0 et $< V_{gj} >$ dépendent des solutions ε, x et p ainsi que des propriétés thermodynamiques qui varient elles-même avec les solutions, d'où la complexité du problème. En particulier on a vérifié les versions des corrélations utilisées avec l'Appendix de l'article suivant [Coddington and Macian, 2002]. En effet nous avons relevé quelques erreurs dans la version proposée dans l'énoncé.

Enfin on utilise des lois de conservation, notamment de la masse qui permet de fixer G pour tout le problème :

$$\frac{\partial G}{\partial z} = 0 \tag{13}$$

et de l'énergie du mélange :

$$\frac{\partial}{\partial z}\left[h_{l,sat}(T_{(z)}) + x_{(z)}h_{vl,sat}(T_{(z)})\right] = \frac{4}{DG}q'' \tag{14}$$

où q'' est le flux de chaleur ajouté en paroi obtenu en supposant la puissance thermique P_{th} apportée uniformément répartie sur la longueur chauffée L_x du cylindre : $q'' = \frac{P_{th}}{\pi D L_c}$. On suppose que l'on travaille à saturation, où la pression p et la température T sont directement liées. Ainsi nous n'avons pas besoin de résoudre explicitement $T = T_{sat}(p_{(z)})$

3 Zone de saturation

Comme nous négligeons la perte de pression par accélération dans la zone où l'ébullition est sous-refroidie (**H2**), il nous faut calculer la position où le fluide est à température de saturation. Cela correspond d'ailleurs à l'endroit où le titre thermodynamique devient nul. Pour cela plusieurs méthodes sont possibles.

3.1 Puissance chauffage

On peut écrire que la puissance nécessaire pour arriver à la température de saturation en partant de la température d'entrée. Cette puissance dépend du flux massique et de la capacité thermique massique de l'eau. Il s'agit donc de faire un bilan énergétique sur le fluide qui passe dans la conduite. On a :

$$\dot{Q}_{sat} = \dot{m} Cp \Delta T \tag{15}$$

Le terme Cp pouvant être pris pour de l'eau à saturation à la pression de sortie (la variation est faible pour un ΔT de cet ordre de grandeur), et le terme ΔT correspond à la différence entre la température de saturation du fluide à la pression donnée et la température d'entrée.
Comme on considère que la conduite est chauffée uniformément, le rapport entre puissance nécessaire pour arriver à saturation et puissance totale thermique ajoutée P_{th} est égale au rapport entre L_{sat} et longueur totale chauffée L_c. On a alors :

$$L_{sat} = \frac{\dot{Q}_{sat} \times L_c}{P_{th}} \tag{16}$$

3.2 Différence enthalpie

Dans les notes de cours, on peut trouver à la page 181, une expression qui nous donne L_{SC}, la longueur *subcooled*. Elle s'écrit comme :

$$L_{sc} = (h_{l,sat} - h_i) \frac{\dot{m}}{\pi D q''_w} \tag{17}$$

Cette équation prend mieux en compte les variations des propriétés du fluide avec la température. Le terme q''_w correspondant à la puissance de chauffage par unité de surface sur le tube :

$$q''_w = \frac{P}{S_{\text{paroi}}} = \frac{P}{\pi D L} \tag{18}$$

On peut maintenant s'intéresser aux valeurs prises par les deux termes d'enthalpies.

L'enthalpie à saturation $h_{l,sat}$ est simple à trouver, il s'agit de prendre l'enthalpie à saturation pour la pression de sortie donnée. La perte de pression que l'on cherche à quantifier est trop faible pour pour faire apparaître des variations notables dans la valeur de l'enthalpie à saturation.

Pour ce qui est de l'enthalpie d'entrée h_i, l'hypothèse **H4** nous indique d'utiliser l'enthalpie à saturation pour cette température.
Notons qu'il est possible de travailler avec la « vraie » enthalpie en utilisant la température d'entrée et pression corrigée avec la perte de pression expérimentale.

3.3 Comparaison des méthodes

Comme nous avons les données pour 2 expériences, nous allons pouvoir comparer les deux méthodes. Rappelons les différentes valeurs caractéristiques pour ces dernières.

TABLE 1 – Paramètres pour les deux expériences

	Expérience 65BV	**Expérience 19**
Diamètre (m)	1.34×10^{-2}	2.29×10^{-2}
Longueur chauffée (m)	1.8	1.8
Puissance thermique (kW)	250	151,8
Débit massique (kg/s)	6.40×10^{-1}	4.70×10^{-1}
Température entrée (°C)	184	215.3
Pression sortie (bar)	20.3	42.1

On peut déterminer des valeurs pour Cp de l'eau à saturation basés sur la pression de sortie, ainsi que la température de saturation. On a alors :

TABLE 2 – Cp et T_{sat} pour les deux expériences

	Expérience 65BV	**Expérience 19**
Cp (kJ/kgK)	4.58	4.90
Température de saturation (°C)	212.1	253.3

On trouve alors $\dot{Q}_{sat}$ et on en déduit la longueur nécessaire pour atteindre un titre nul.

TABLE 3 – Puissance et longueur sous-refroidie pour les deux expériences

	Expérience 65BV	**Expérience 19**
Puissance nécessaire $\dot{Q}_{sat}$ (kW)	81.7	77.6
Longueur sous-refroidie L_{sc} (m)	0.59	0.92

On peut maintenant passer à la seconde méthode avec les enthalpies. On a :

TABLE 4 – Enthalpies pour les deux expériences

	Expérience 65BV	**Expérience 19**
Enthalpie saturation $h_{l,sat}$ (kJ/kg)	912	1102
Enthalpie entrée h_i (kJ/kg)	781.2	922.6
Enthalpie entrée corrigée $h_{i,cor}$ (kJ/kg)	781.6	922.5

Après application numérique on a alors :

TABLE 5 – Longueurs sous-refroidie pour les deux expériences

	Expérience 65BV	**Expérience 19**
Longueur sous-refroidie L_{sc} (m)	0.602	0.999
Longueur sous-refroidie corrigée $L_{sc,cor}$ (m)	0.600	1.000

On remarque que les résultats avec les deux méthodes donnent des valeurs du même ordre de grandeur, ce qui confirme que les deux comparent bien les mêmes choses. De plus les résultats pour les enthalpies, qu'elles soient corrigées ou non sont très proche, l'hypothèse **H4** est donc cohérente.

La différence des résultats entre les deux méthodes est que dans la première, on ne prend pas en compte le changement des propriétés thermiques (Cp) du fluide dans la conduite et on travaille seulement avec la valeur à saturation. Il pourrait être plus réaliste de moyenner avec la valeur à l'entrée ou bien de faire l'intégrale sur toute la zone en considérant que le gradient de température est constant dans la conduite.

4 Algorithme classique

4.1 Principe de l'algorithme classique

La première partie de ce rapport détaille les équations qui régissent notre problème. Nous pouvons donc nous concentrer sur la mise en place de celles-ci maintenant.

La présentation des méthodes que nous avons utilisés ne se fait pas dans l'ordre chronologique, ce que nous avons appelé `Algorithme classique` et qui est présenté dans cette section nous a servi de comparaison pour le second algorithme qui est lui plus complexe et qui rend la détection de fautes plus difficile (il peut parfois s'agir de la non convergence de l'algorithme et pas de fautes de à proprement parler).

Dans cette méthode nous avons effectué une discrétisation de la zone d'étude en N_z points équidistants et résolus les équations à chacun de ces points. Nous avons travaillé avec le modèle à vitesses séparées et les corrélations de Chexal [Chexal and Lellouche, 1986], Fridel [Freidel, 1979] ainsi que de Inoue [Inoue et al., 1993] ont été utilisées. Une version légèrement différente de ces modèles [Revellin and Thome, 2007] a pu être utilisée pour corriger un certain nombre d'erreurs.

Une hypothèse forte qui a été faite, est celle de négliger la variation des enthalpies par rapport à la position dans la conduite. Elles ne sont pas constantes pour autant, elles restent dépendantes de la de la pression et donc de la température. Plus formellement :

$$\left| x_{(z)} \frac{\partial h}{\partial P} \frac{\partial P}{\partial z} \right| \ll \left| \frac{\partial x_{(z)}}{\partial z} h(P_{(z)}) \right| \tag{19}$$

Ce qui revient à isoler le terme de dérivée du titre $\frac{\partial x}{\partial z}$ dans l'équation 14 :

$$\frac{\partial x_{(z)}}{\partial z} h_{l,sat}(P_{(z)}) \approx \frac{\partial}{\partial z} \left[h_{l,sat}(P_{(z)}) + x_{(z)} h_{vl,sat}(P_{(z)}) \right] = \frac{4}{DG} q'' \tag{20}$$

L'algorithme part de ce que l'on connaît déjà (ie. la pression ainsi que le titre en sortie) et intègre en « remontant » la conduite. Nous avons des équations (qui ont été définies précédemment dans la partie 2) pour les termes x, ϵ et p que nous pouvons actualiser à chaque itération.

Cette technique nécessite donc un critère d'arrêt, car sinon le programme tourne indéfiniment. Nous l'avons pris tel que le RMS de la variation entre deux itérations soit inférieur à un seuil de tolérance fixé à une valeur proche de l'epsilon machine en simple précision :

$$\text{RMS var} = \sqrt{\frac{1}{N_z} \sum_{k=1}^{N_z} \left[\left(x_k^i - x_k^{i+1} \right)^2 + \left(\frac{p_k^i - p_k^{i+1}}{p_s} \right)^2 + \left(\epsilon_k^i - \epsilon_k^{i+1} \right)^2 \right]} < \text{tol} = 1 \times 10^{-7} \tag{21}$$

Dans l'équation ci-dessus, x_k^i fait référence au titre évalué au k$^{\text{ème}}$ point de discrétisation ($z_k = z_e + \frac{k}{N_z} L_c$) lors de la i$^{\text{ème}}$ itération . Cela permet de faire converger rapidement l'algorithme et ne pas trop contraindre la variation des différents paramètres.

4.2 Formalisation de l'algorithme classique

On se propose de synthétiser la structure de la simulation classique avec l'algorithme 1. La version complète peut-être trouvée dans le fichier python `Python/Algorithme_classique.py`.

Algorithm 1: Algorithme de résolution classique

Result: Return converged vectors x, ϵ and p
initialisation;
x,ϵ,$p \leftarrow 0 \in \mathbb{R}^{N_z}$;
while *RMS var* $>$ *tol* **do**
  Compute thermodynamic properties at each point;
  $\epsilon^{new} \leftarrow$ Correlation Model$(\epsilon, x, p, ...)$;
  $x^{new} \leftarrow x_s + \int_{z_s}^{z}$ Energy Equation$(\epsilon, x, p, ...)$;
  $p^{new} \leftarrow P_s + \int_{z_s}^{z}$ Momentum Equation$(\epsilon, x, p, ...)$;
  Compute RMS var $(x^{new}, \epsilon^{new}, p^{new}) - (x, \epsilon, p)$;
  $x, \epsilon, p \leftarrow x^{new}, \epsilon^{new}, p^{new}$
end
Compute total pressure drop;
Output x, ϵ, p and plot;

Les mises à jour des paramètres se font avec les équations qui régissent notre modèle. Plus précisément, le taux de vide ϵ est modifié en suivant l'équation (11), le titre et la pression quant à eux suivent respectivement les équations (20) et (5).

4.3 Convergence en discrétisation spatiale

Afin de vérifier que la discrétisation spatiale engendre des erreurs qui convergent vers 0 avec N_z, on effectue une série de simulations et on considèrera le paramètre scalaire $\Delta p = |P(z_s) - P(z_e)|$. On se limitera au cadre de l'expérience 19 avec le modèle de corrélation d'Inoue [Inoue et al., 1993]. Le seuil d'arrêt est fixé dans tous les cas à $tol = 1 \times 10^{-7}$ pour le RMS de la variation entre deux itérations consécutives. Enfin on s'intéressera à l'écart relatif entre la chute de pression pour une discrétisation N_z avec la discrétisation la plus fine obtenue, défini par :

$$err_{\Delta p} = \left| \frac{\Delta p^{N_z} - \Delta p^{\max N_z}}{\Delta p^{\max N_z}} \right| \tag{22}$$

Les résultats sont obtenus à l'aide du code contenu dans le fichier situé dans le dossier `Python/Convergence_Discretisation_Algo_Classique.py` et sont présentés à la figure 2. On observe que le nombre d'itérations pour arriver au même seuil de tolérance ne varie pas avec la finesse de discrétisation spatiale dans le cadre de l'étude présentée.

D'autre part la convergence de la chute de pression avec le nombre d'éléments N_z semble suivre un ordre 1 en espace. Ce n'est pas un résultat évident car on ne peut pas faire d'analyse classiques d'une discrétisation d'une équation aux dérivées partielles linéaire (car le critère d'arrêt dépend de la variation entre itérations successives et n'a rien d'absolu). Néanmoins, cela peut se comprendre par

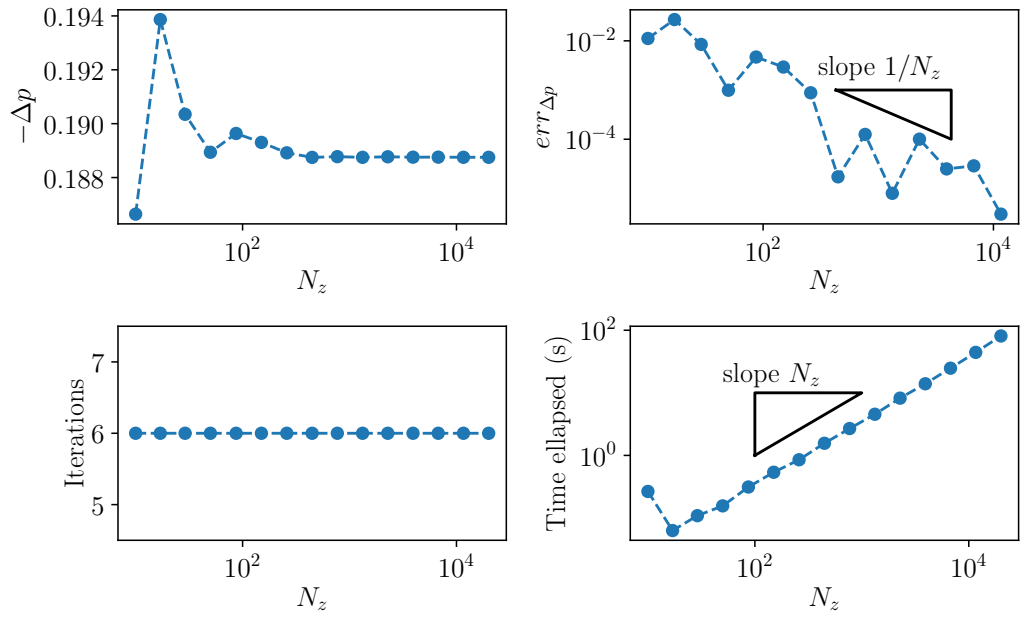

FIGURE 2 – Étude de convergence en discrétisation spatiale sur l'algorithme classique. Pour différents N_z, on donne la chute de pression obtenue, l'écart relatif $err_{\Delta p}$, le nombre d'itérations ainsi que le temps écoulé

le fait d'avoir choisi un schéma de discrétisation pour la dérivée (matrice Dz) ainsi que l'intégrale (par une technique de rectangles) qui est d'ordre 1 en espace.

A ce titre, étant donné que le nombre d'itérations reste constant, chaque opération est globalement en $\mathcal{O}(N_z)$ par itération ce qui permet d'expliquer l'évolution linéaire du temps écoulé pour la résolution du problème avec N_z (pas d'inversion de matrice, ou autre procédé plus gourmand en ressources). On remarque que l'analyse aurait été à priori assez différente si la discrétisation influait davantage sur la vitesse de convergence, i.e. le nombre d'itérations nécessaires.

5 Présentation de la méthode PINN

5.1 Introduction et état de l'art

L'utilisation des techniques d'apprentissage machine aux domaines dits « classiques » comme la mécanique n'est pas nouveau. De nombreux travaux de recherche ont été menés jusqu'aux applications industrielles pour des problèmes d'aide à l'optimisation, de contrôle, de pré-développement de pièces ou de modèles réduits dont on pourra se faire une idée avec la synthèse de [Brunton et al., 2019]. Néanmoins l'approche privilégiée a été celle des données, *data-driven* comme on le trouve dans la littérature. Cela signifie que les réseaux de neurones encodent les mécanismes physiques (dans le meilleur des cas) uniquement à partir d'un certain nombre de données préparées au préalable. Par exemple, [Guo et al., 2016] montre qu'il est possible d'estimer très rapidement l'écoulement moyen autour de formes complexes à l'aide de réseaux de neurones convolutionnels (CNNs) (Pour les concepts de base on pourra se référer à un ouvrage de référence [Goodfellow et al., 2016]).

Néanmoins une autre approche a été récemment explorée, celle de l'approximation des solutions d'un problème de type PDE (équations aux dérivées partielles) par des familles de fonctions sous l'hypothèse du théorème général d'approximation. S. Brunton a notamment travaillé dans ce domaine en déterminant les fonctions les plus adaptées pour décrire quantitativement (avec des données brutes) les solutions d'un problème à partir d'une famille suffisamment large de fonctions candidates [Brunton et al., 2016]. Une suite logique est présentée chez [Corbetta and Field,] où la représentation d'un phénomène par une famille de fonctions continues permet de remonter aux équations qui le gouvernent. C'est aussi une manière d'établir des modèles réduits de problèmes complexes.

Le formalisme plus classique des réseaux de neurones comme détaillé dans la section suivante apparaît un peu plus tard. [Raissi et al., 2019a] présente un formalisme assez général de résolution de PDE avec différentes échelles de complexité et non-linéarités que l'on appelera PINN par la suite (pour *Physics-Informed Neural Networks*). Il s'agit de représenter les solutions d'une PDE par une combinaison de fonctions de bases, connues, dont les inputs sont les coordonnées physiques : le temps, l'espace. Il n'y a donc pas de discrétisation spatiale ou temporelle. De plus, il introduit des fonctions d'erreurs basées sur les équations locales du problème accompagnées de conditions aux limites (spatiales et temporelles) ainsi que d'écarts aux mesures. On pourra également trouver une justification mathématique de la méthode avec une analogie avec la méthode de Galerkin chez [Al-Aradi et al.,]. Le manuscrit de thèse [Rudy, 2019] est également une piste d'approfondissement.

Des applications dans des domaines plus spécifiques sont rapidement publiées : [Raissi et al., 2018] pour la reconstitution d'écoulement et de forces à partir de mesures de concentration d'un scalaire passif. Dans le domaine de l'interaction fluide-structure, [Raissi et al., 2019b] propose la reconstitution d'écoulement 2D autour d'un cylindre monté sur ressort à partir de divers types de données initiales. Plus récemment encore, on pourra se pencher du côté de [Mao et al., 2020] pour des écoulements hautes-vitesses, [Haghighat et al., 2020, Lu et al., 2020] pour de la mécanique des solides ou encore [Chen et al., 2020] dans le domaine de la nano-optique et des méta matériaux.

5.2 Principe de fonctionnement

L'idée fondamentale de la méthode PINN réside dans le théorème général d'approximation [Hornik et al., 1989] que l'on pourra énoncer dans une version simplifiée par :

Theorem 1 *Soit $u : x \in C \to u(x) \in \mathbb{R}^m$ une fonction continue d'un sous-ensemble compact C de $\mathbb{R}^n$ dans $\mathbb{R}^m$. Alors u peut-être approchée avec une fonction d'activation f avec une précision (étant donné une norme $\|.\|$ sur l'espace de fonctions $\mathcal{C}(C, \mathbb{R}^m)$) arbitraire (i.e. à $\epsilon \in \mathbb{R}^+$ près) par un réseau à une couche cachée suffisamment large (taille $p \in \mathbb{N}$).*

$$\forall \epsilon > 0, \exists p \in \mathbb{N}, B_1 \in \mathbb{R}^p, B_2 \in \mathbb{R}^m, W_1 \in \mathbb{R}^{p \times n}, W_2 \in \mathbb{R}^{m \times p}$$

définissant

$$\tilde{u}_{NN}(x) = W_2 f\left(W_1.x + B_1\right) + B_2$$

Avec

$$\|\tilde{u}_{DNN} - u\| \leq \epsilon$$

L'approximation de u par un réseau de neurones mono-couche $\tilde{u}_{DNN}$ est construite en plusieurs étapes successives :

1. D'abord, on effectue une première opération matricielle de l'ensemble de départ ($x \in \mathbb{R}^n$) jusqu'à $\mathbb{R}^p$ qui est l'espace de la couche cachée (*hidden layer* dans la littérature). On multiplie x par la matrice W_1 puis on ajoute un terme constant B_1.

$$y_1 = W_1.x + B_1 \in \mathbb{R}^p$$

2. On effectue une opération non-linéaire sur les éléments du hidden-layer avec la fonction d'activation f. Par exemple on peut prendre la tangente hyperbolique de chacun des éléments du vecteur $y_1 \in \mathbb{R}^p$. Généralement, $f(y_1) \in \mathbb{R}^p$
3. On envoie $f(y_1)$ dans l'espace d'arrivée $\mathbb{R}^m$ avec une seconde opération matricielle (comme au 1.) utilisant W_2 et B_2 :

$$\tilde{u}_{DNN} = W_2 f(y_1) + B_2$$

En pratique le choix de la taille de la couche cachée p ainsi que les fonctions d'activations f ne sont pas donnés et peuvent dépendre du problème en question. On trouve beaucoup de littérature à ce sujet, par exemple sur les fonctions d'activation [Leshno et al., 1993].

D'autre part, les matrices $(W_k)_k$ et les vecteurs $(B_k)_k$ sont appelés poids (W pour *Weights* en anglais) et biais (et *Bias*) et forment l'ensemble des paramètres ajustables du modèle. En pratique, pour éviter que p ne soit trop grand quand u se complexifie, on rajoute des couches intérieures « cachées ». Cela signifie qu'après l'étape 2, on reprend l'étape 1 avec de nouveaux poids W et biais B. On peut recommencer autant de fois que l'on veut de couches cachées (on parle de profondeur du réseau). On effectue l'étape 3 pour revenir dans l'espace d'arrivée de $u(x)$.

Intuitivement ce résultat peut se comprendre avec l'exemple plus classique de l'approximation des fonctions continues par un polynôme (Théorème de Stone-Weierstrass) et décomposition en série de polynômes de Taylor. La méthode classique pour approcher une fonction continue consiste à

travailler dans un espace de fonctions de bases en escalier (constantes ou $\mathcal{C}^k$ par morceaux) sur une subdivision d'un intervalle $I \subset R$ (une sorte de maillage, comme dans la construction de l'intégrale de Riemann). Au lieu de ça, on travaille dans un espace de paramètres définissant des fonctions continues. Par exemple dans le cas de Stone-Weierstrass, ces paramètres sont les coefficients associés à une base de polynômes. Pour un réseau de neurones, l'espace des paramètres est celui des poids et des biais. On désigne généralement ces paramètres par θ. Cette façon de faire se rapproche d'une méthode de Galerkin mais ici les fonctions de bases sont elles-même modifiables.

Finalement, connaissant la structure du réseau, c'est à dire ses fonctions d'activation f, le nombre de couches cachées ainsi que la taille de chacune de ses couches cachées, $\tilde{u}_{DNN}$ est entièrement définie avec l'ensemble des paramètres θ (poids et biais) pour tout $x \in \mathbb{R}^n$. L'objectif du problème est donc d'**optimiser** ces paramètres en **minimisant une erreur** $\mathcal{L}$. La difficulté étant de bien construire cette erreur.

L'optimisation d'un très grand nombre de paramètres (θ peut contenir plusieurs milliers de scalaires !) est rendue possible par des outils de calcul symbolique. En effet, pour chacun des θ_i, on peut calculer **symboliquement** $\frac{\partial \mathcal{L}}{\partial \theta_i}$ par dérivation en chaîne : $\frac{\partial f(g)}{\partial \theta_i} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial \theta_i}$ puisque on connaît explicitement l'ensemble des opérations qui permettent d'obtenir $\tilde{u}_{DNN}$ puis $\mathcal{L}$ à partir de x et θ. Et plus important, connaissant f', on sait les dériver exactement ! Les librairies actuelles le font automatiquement (Tensorflow par exemple [TensorFlow,]) : on appelle cela l'**auto-différenciation**.

L'auto-différenciation permet donc de dériver symboliquement l'erreur $\mathcal{L}$ par rapport aux paramètres du modèle et ainsi faire des descentes de gradients ou de Hessiennes. Plusieurs optimiseurs existent et font cela en boite noire. En particulier, on utilise dans le projet l'optimiseur Adam [Kingma and Ba, 2017]. Mais l'auto-différentiation permet en particulier de dériver des quantités en sortie ($\tilde{u}_{DNN}$ par exemple) par rapport aux variables d'entrée x qui sont, dans un PINN, les coordonnées physiques.

Cette dérivation symbolique par rapport aux coordonnées physiques est un des points importants de la méthode PINN puisque l'on travaille avec des problèmes régis par des PDE souvent non linéaires. Ainsi pour un problème décrit par une PDE de la forme

$$\mathcal{N}_x(u) = 0 \tag{23}$$

où $\mathcal{N}_x$ est un opérateur (potentiellement non-linéaire), en pratique la PDE du problème. Une méthode classique est de linéariser $\mathcal{N}_x(u) \approx \mathcal{N}_x(u_0) + \frac{\partial \mathcal{N}_x}{\partial u}(u - u_0)$ et de faire converger la solution u en inversant l'opérateur linéarisé $\frac{\partial \mathcal{N}_x}{\partial u}$. La méthode PINN permet de s'affranchir de cette linéarisation en calculant symboliquement $\mathcal{N}_x(u)$ et en le dérivant par rapport aux paramètres θ du modèle.

Dans le cas d'un PINN, on décompose $\mathcal{L} = \mathcal{L}_{PDE} + \mathcal{L}_{BC} + \mathcal{L}_{data}$:

— le résidu des équations aux dérivées partielles sur un ensemble de points **choisis** dans le domaine de la PDE :

$$\mathcal{L}_{PDE} = \|\mathcal{N}_x(\tilde{u}_{DNN})\|^2 \tag{24}$$

Par exemple, on peut les choisir aléatoirement dans le domaine. Il est important d'évaluer cette erreur sur un nombre de points représentatif de l'ensemble du domaine. En pratique,

plus le problème est compliqué, plus il y a de paramètres θ et plus il faut de points de pénalisation dans les zones « pathogènes ». D'autre part, d'un point de vue mathématique, on se trouve souvent dans un problème sur-contraint, c'est-à-dire que l'on veut imposer $\mathcal{N}_x(\tilde{u}_{DNN}) = 0$ en davantage de points qu'il n'y a de degrés de liberté.

— La pénalisation des conditions aux frontières, spatiales comme temporelles (conditions initiales ou finales). Cela permet de mettre en place « simplement » l'équivalent d'une méthode de tir traditionnelle

$$\mathcal{L}_{BC} = \|\tilde{u}_{DNN}(x_{BC}) - u_{BC}\|^2 \tag{25}$$

— L'écart à une série de données. Par exemple des données de mesure si l'on en dispose

$$\mathcal{L}_{data} = \|\tilde{u}_{DNN}(x_{data}) - u_{data}\|^2 \tag{26}$$

5.3 Adaptation au problème du projet

Pour résoudre ce problème, on cherche à approximer avec la méthode PINN le titre x, la pression p ainsi que le taux de vide ϵ comme des fonctions de $z \in [z_e, z_s]$. On construit alors trois réseaux de neurones avec $n = m = 1$ et dont les poids et biais sont les inconnues du modèle à optimiser. On aurait d'ailleurs pu faire le choix d'utiliser un seul réseau de neurones dont la couche de sortie est dans $\mathbb{R}^3$, ce qui est discuté chez [Haghighat et al., 2020].

$$\begin{aligned} \mathrm{DNN}_\epsilon : z \in \mathbb{R} &\to \tilde{\epsilon}_{DNN}(z) \in \mathbb{R} \\ \mathrm{DNN}_x : z \in \mathbb{R} &\to \tilde{x}_{DNN}(z) \in \mathbb{R} \\ \mathrm{DNN}_p : z \in \mathbb{R} &\to \tilde{p}_{DNN}(z) \in \mathbb{R} \end{aligned} \tag{27}$$

Par la suite on désignera les fonctions $\tilde{\epsilon}_{DNN}, \tilde{x}_{DNN}$ et , $\tilde{p}_{DNN}$ respectivement par ϵ, x et p par souci d'alléger les notations. On construit l'erreur en sommant les écarts quadratiques aux trois équations, à l'écart à la pression de sortie. Il faut également rajouter une pénalisation pour $\epsilon \in [0, 1]$. En effet cette dernière contrainte est codée en dur dans l'algorithme classique. Pour le PINN on choisit de lui donner cette information par une pénalisation.

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{Énergie}} + \mathcal{L}_{\text{Momentum}} + \mathcal{L}_{\text{Drift Flux Model}}}_{\mathcal{L}_{\text{PDE}}} + \mathcal{L}_{\text{BC}} + \mathcal{L}_{\text{Pénalisation}} \tag{28}$$

Dans le cas du projet on n'utilise pas de données de mesure, ainsi $\mathcal{L}_{data} = 0$. D'autre part, les conditions aux frontières sont incomplètes. On travaille avec un ensemble de coordonnées z qui vont permettre d'évaluer l'erreur sur des points générés dans $[z_e, z_s]$. On appelle cet ensemble V^{int} de cardinal N^{int}.

$$\mathcal{L}_{\text{Pénalisation}} = \frac{1}{N^{\text{int}}} \sum_{z \in V^{\text{int}}} |\max(0, -\epsilon(z)) + \max(0, \epsilon - 1)|^2 \tag{29}$$

Pour l'erreur associé aux conditions aux bords, on a fait le choix de ne pénaliser seulement la condition en pression en sortie P_s. L'idée est de tester s'il est nécessaire de donner l'information sur

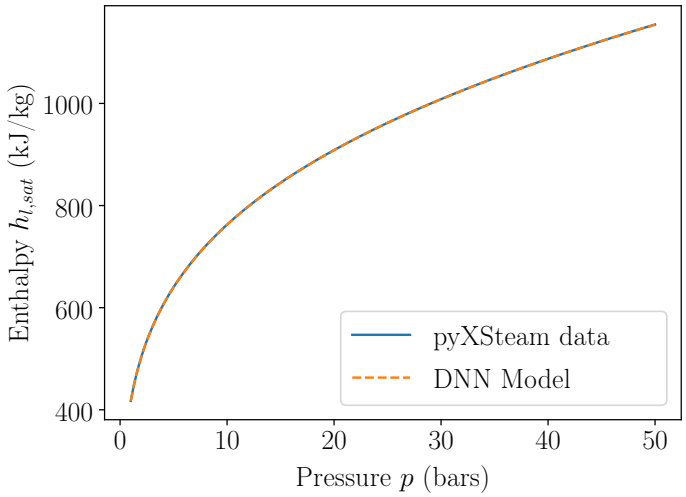

FIGURE 3 – Entraînement d'un réseau de neurones annexe pour l'enthalpie massique du liquide à saturation à partir de données issues de pyXSteam. Le *root mean square* (RMS) de l'écart relatif vaut $\text{RMS}\left(\frac{h_{DNN}-h_{XSteam}}{h_{XSteam}}\right) = 6.29 \times 10^{-5}$.

le titre en sortie x_s sachant que celui-ci est calculé à partir de l'intégration de l'équation d'énergie. Néanmoins on ne donne pas la température en entrée qui est contenu dans le x_s.

$$\mathcal{L}_{\text{BC}} = |p(z_s) - P_s|^2 \tag{30}$$

Pour les trois équations constituant $\mathcal{L}_{\text{PDE}}$, il faut reconstruire toutes les fonctions et les coefficients de corrélation avec des fonctions de TensorFlow de telle manière qu'on puisse dériver ces quantités par rapport à z ainsi qu'aux paramètres du modèle θ. Notamment pour l'équation d'énergie, on a voulu prendre en compte les variations d'enthalpie avec z, c'est-à-dire ne pas faire l'approximation formalisée à l'équation 20. Pour ce faire, il faut pouvoir dériver $\frac{\partial h_{l,sat}(p)}{\partial p}$, ce qui n'est pas possible avec la bibliothèque `XSteam` puisque celle-ci ne propose qu'une interpolation de points. On a donc créé des réseaux de neurones auxiliaires qui approximent avec une précision largement suffisante les fonctions thermodynamiques comme $\text{DDN}_{h_{l,sat}} : p \to h_{l,sat}(p)$ sur une plage de pression $p \in [1, 50]$ bars. On a fait de même pour $h_{v,sat}(p), \rho_l(p), \rho_m(p), \mu_l(p), \mu_v(p)$ et $\sigma(p)$.

Ces modèles ont été entraînés au préalable avec des données générées par `pyXSteam` puis les poids et biais sont stockés sous forme d'archive `pickle` (dans le dossier `Python/Models`). Il suffit alors de les importer dans le code principal et de définir ces paramètres comme des réseaux **constants**, c'est à dire qu'on ne touchera pas à ces réseaux lors de l'optimisation de $\mathcal{L}$. Le cas de $h_{l,sat}$ est présenté à la figure 3.

Ainsi, pour l'équation d'énergie, on est capable de construire un graphe d'opérations dérivables

$$H : z \to h_{l,sat}(p(z)) + x(z)h_{vl,sat}(p(z)) \tag{31}$$

Avec $\frac{\partial H}{\partial z}$ obtenu en un bloc. Concrètement, on utilise `tf.gradients(H,z)` avec Tensorflow qui sera

évalué pour tous les points de V^{int}. Ainsi :

$$\mathcal{L}_{\text{Energie}} = \frac{1}{N^{\text{int}}} \sum_{z \in V^{\text{int}}} \left(\frac{DG}{4q''} \frac{\partial}{\partial z} \left[h_{l,sat}(p(z)) + x(z) h_{vl,sat}(p(z)) \right] - 1 \right)^2 \tag{32}$$

La même structure d'erreur est constituée pour les deux autres équations. A ce titre, il faut aborder le point de la **normalisation**. En effet, on travaille avec des grandeurs dimensionnées qui peuvent être d'un ordre de grandeur très différent : x varie usuellement entre 0 et 1 et donc on s'attend à ce que $\mathcal{L}_{\text{Drift Flux Model}} \sim 1$. En revanche, $4q''/DG \sim 10^2$ donc $\mathcal{L}_{\text{Energie}}$ peut être de l'ordre de 10^4 si on l'implémente comme dans l'équation 14. Cela explique pourquoi l'équation 32 a un -1 comme terme de droite. En effet, il est difficile d'optimiser simultanément une somme de plusieurs termes qui sont d'ordre de grandeurs très différents. Une solution plus simple aurait été de travailler dès le début avec des grandeurs adimensionnés. Cela ne s'est pas fait car il y a beaucoup de relations thermodynamiques et de coefficients dimensionnés définis implicitement.

Enfin, la question de l'initialisation des réseaux de neurones pour la résolution de ϵ, x et p est un sujet compliqué dont on ne traitera pas en profondeur dans ce rapport. En pratique, après une initialisation aléatoire, on effectue une pré-optimisation vers un set de fonctions affines afin de gommer « l'indéterminisme ».

A travers cette section nous avons tenté d'introduire le formalisme PINN et quelques détails qui nous semblaient importants. Comme vous le constaterez par les dates des articles (la plupart n'étant pas paru avant 2019), le sujet est tout récent et manque donc encore de maturité. Nous n'avons pas non plus cherché à être exhaustif car ce n'est pas l'objet de ce rapport. De même la bibliographie a été réduite à quelques exemples pertinents et est loin d'être complète. L'idée a été motivée par un projet de maîtrise portant sur les PINNs dans un domaine connexe. Si vous avez des questions ou souhaitez en savoir davantage, n'hésitez pas à nous contacter.

5.4 Limites rencontrées

Dans le cadre du projet, nous avons rencontré plusieurs difficultés à adapter la méthode PINN pour des raisons parfois spécifiques à la nature du problème. En particulier les modèles de corrélation complexes avec des seuils, des conditions posent des soucis de différentiation. De même il faut bien veiller à n'appliquer les équations du modèle diphasique lorsque $0 < x < 1$ et veiller à ce que l'on retombe sur des équations monophasiques aux bords ce qui créé d'autres sources d'instabilités avec des gradients infinis. Enfin nous avons tenté de supprimer toute les possibilités d'obtenir un numérateur nul en écrivant les équations de la manière la plus adéquat. Il demeure quelques sources de problèmes tout de même. En particulier nous n'avons pas pu résoudre correctement le problème du sueil $x = 0$ dans le modèle de corrélation (où on pénalise l'écart à $\varepsilon = 0$ quand $x < 0$).

De manière plus générale, le choix des fonctions d'activation et de la taille des réseaux est encore assez empirique et sujet à discussion. Ceci dit, il reste un grand nombre de possibilités à explorer tant dans les détails que dans la méthode. L'une d'entre elle serait d'établir des modèles de corrélation directement avec des réseaux de neurones en utilisant des données expérimentales dans davantage de cas.

6 Résultats obtenus

6.1 Résumé synthétique

On trouvera à la table 6 un tableau synthétisant les pertes de pression obtenues pour les différentes expériences, modèles et méthodes retenues. Les écarts relatifs avec les données expérimentales sur la chute de pression totale sont, quant à eux, présentés à la table 7 pour les même paramètres.

TABLE 6 – Chute de pression (bars)

	Expérience 65BV	**Expérience 19**
PINN Chexal	4.85	1.78×10^{-1}
Classique Chexal	3.08	1.70×10^{-1}
PINN Inoue	4.68	2.14×10^{-1}
Classique Inoue	3.09	1.89×10^{-1}

TABLE 7 – Ecart relatif de la chute de pression avec les expériences $err = \frac{\|\Delta p - \Delta p_{data}\|}{\Delta p_{data}}$ (%)

	Expérience 65BV	**Expérience 19**
PINN Chexal	51.6	4.8
Classique Chexal	3.7	9
PINN Inoue	46.5	14.4
Classique Inoue	3.5	0.98

6.2 Résultats détaillés

Enfin les figures 4, 5, 6 et 7 présentent graphiquement l'évolution du titre x, du taux de vide ε ainsi que de la pression p avec l'abscisse z pour la méthode classique puis la PINN respectivement.

Les lignes en pointillés rouge sur les figures suivantes correspondent à la position où le fluide est à saturation ($x = 0$). La partie précédent cette position n'est pas prise en compte dans l'algorithme classique. Dans le cas de l'algorithme PINN, il n'est pas possible de fixer cette condition, c'est pour cette raison que le titre n'est pas nul au niveau de la zone de saturation. Cela résulte du choix d'implanter ε comme une inconnue supplémentaire et pas une conséquence directe de l'équation du modèle de corrélation.

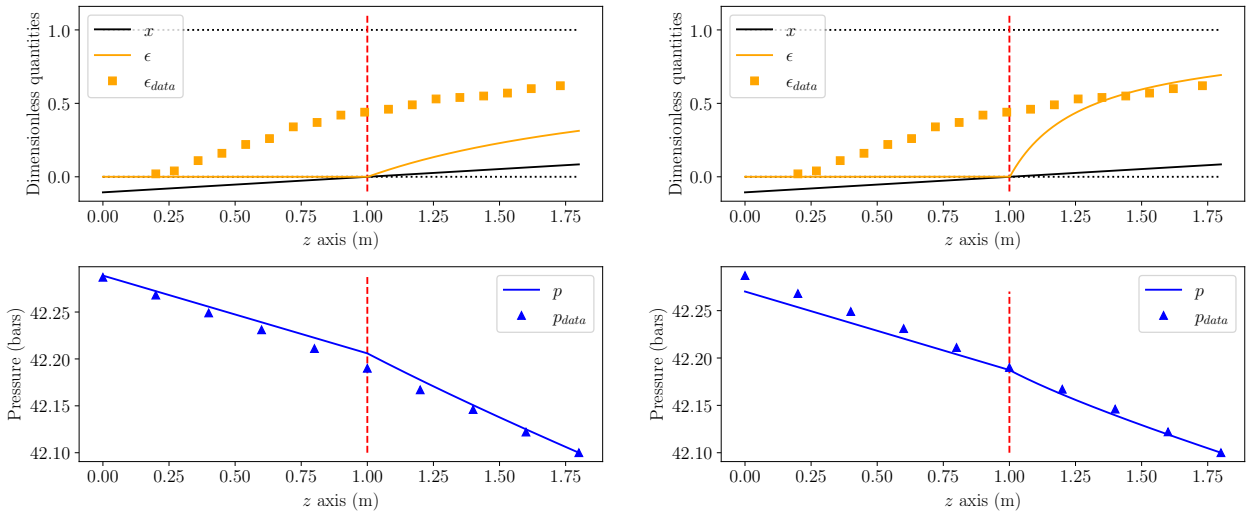

FIGURE 4 – Résultats obtenus avec la méthode classique dans le cas de l'expérience 19 avec les corrélations d'Inoue [Inoue et al., 1993] (à gauche) et celle de Chexal [Chexal and Lellouche, 1986] (à droite)

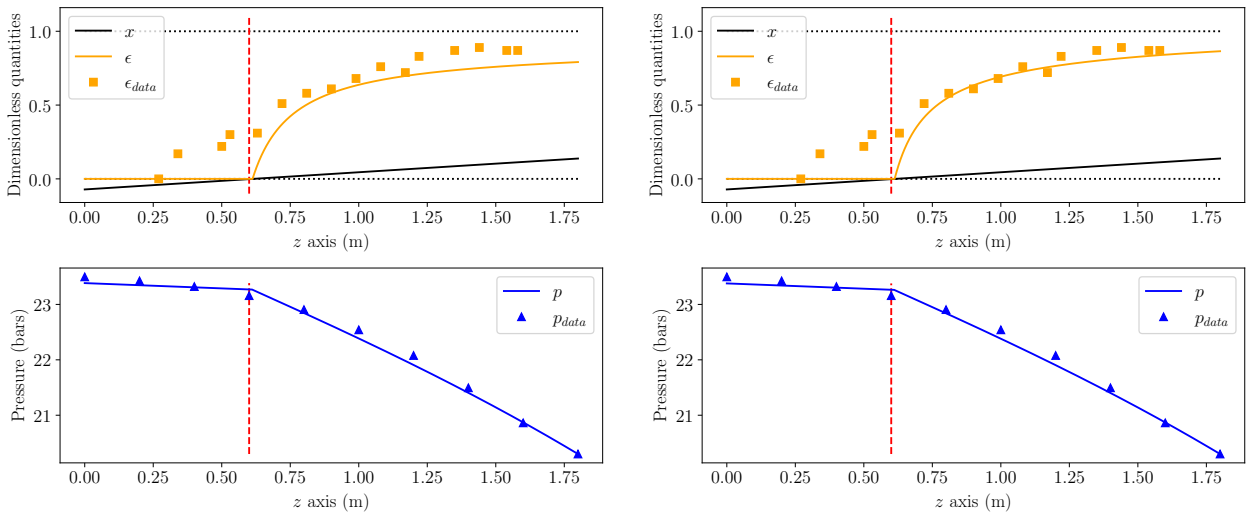

FIGURE 5 – Résultats obtenus avec la méthode classique dans le cas de l'expérience 65BV avec les corrélations d'Inoue [Inoue et al., 1993] (à gauche) et celle de Chexal [Chexal and Lellouche, 1986] (à droite)

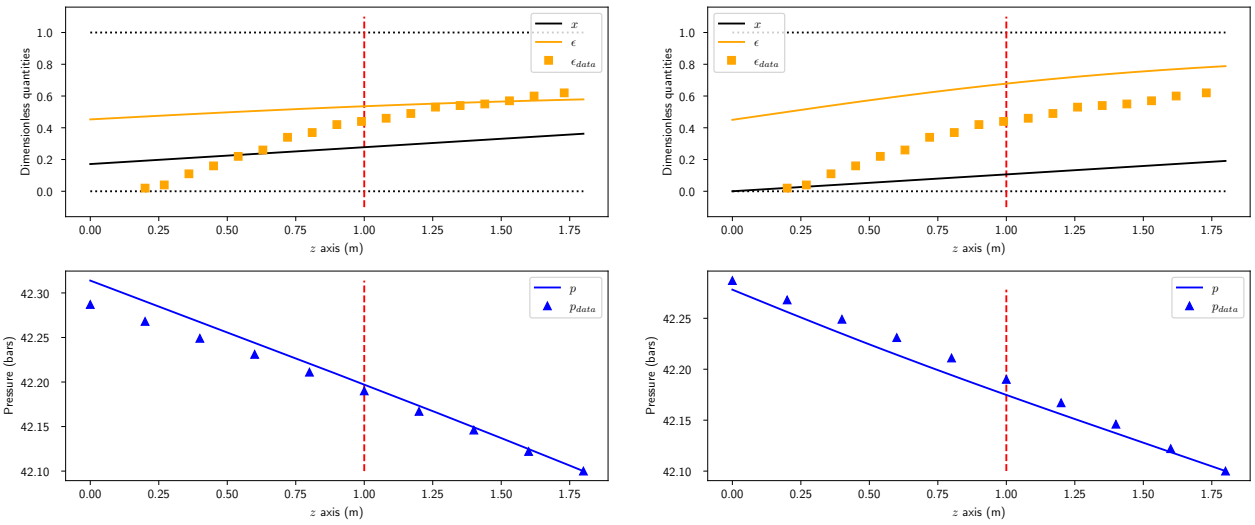

FIGURE 6 – Résultats obtenus avec la méthode PINN dans le cas de l'expérience 19 avec les corrélations d'Inoue [Inoue et al., 1993] (à gauche) et celle de Chexal [Chexal and Lellouche, 1986] (à droite)

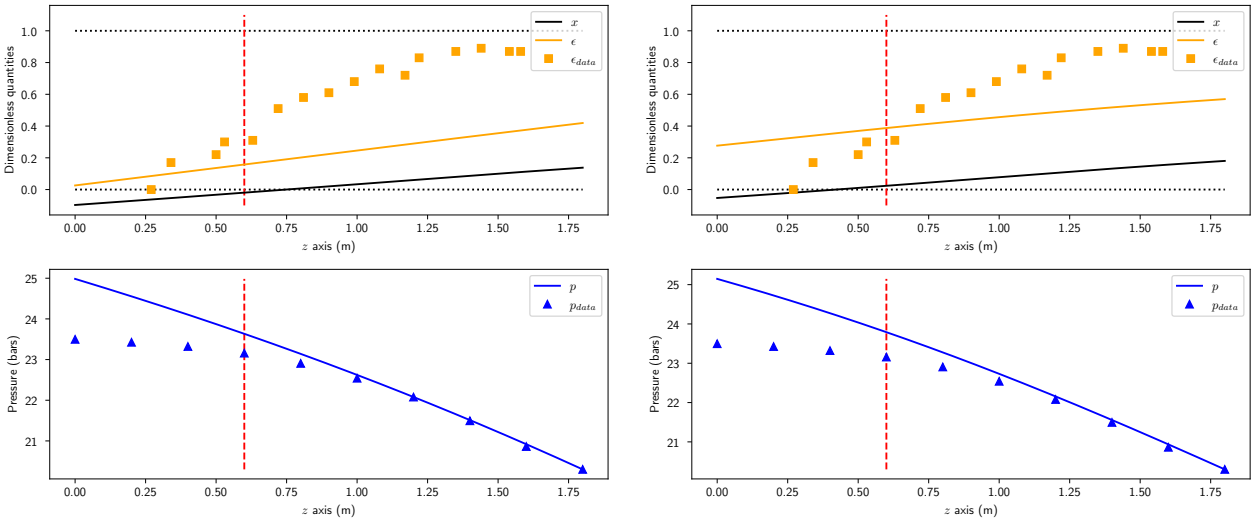

FIGURE 7 – Résultats obtenus avec la méthode PINN dans le cas de l'expérience 65BV avec les corrélations d'Inoue [Inoue et al., 1993] (à gauche) et celle de Chexal [Chexal and Lellouche, 1986] (à droite)

7 Discussion et conclusion

D'après les différentes sections précédentes présentant les modèles et les méthodes de résolutions, ainsi que des graphes des résultats nous pouvons nous interroger sur les résultats obtenus.

Nous avons réalisés pour les deux jeux de donnés des expériences, une résolution avec l'algorithme classique et l'algorithme PINN. Pour chacune de ces méthodes, les modèles de Chexal et Inoue on été testés. Nous avons ainsi 8 cas que nous pouvons comparer pour en tirer des conclusions.

7.1 Commentaires sur les résultats obtenus

Les premières comparaisons que nous pouvons faire sont entre les deux algorithmes. La principale différence que l'on remarque entre les résultats, est la discontinuité pour le taux de vide que l'on obtient en utilisant l'algorithme classique. La manière dont nous avons travaillé est de le contraindre égal à zéro dans toute la zone précédent la saturation (lorsque $x < 0$), d'où le fait qu'il n'augmente qu'après. Cela a pour conséquence de créer une fonction dont la dérivée est non continue pour ce paramètre mais qui respecte mieux les critères du modèle.

De son coté, l'algorithme PINN a plus de mal à converger vers les bons résultats, la tendance semble correcte cependant (croissante et du même ordre de grandeur) ce qui nous rassure par rapport à l'implantation de notre algorithme. Il faut noter qu'en théorie la zone avant saturation ne pénalise pas les « bonnes » équations.

Dans tous les cas que nous avons essayés, on peut voir que la pression correspond bien au jeux de donnés disponibles. Les tables 6 et 7 nous montre que la chute de pression est du même ordre de grandeur que ce qui était attendu quelque soit la méthode utilisée. Les écarts relatifs sont très faibles, et la solution trouvée à la même enveloppe que les données. Le changement de pente à l'abcisse qui correspond à $x = 0$ correspond à l'hypothèse **H2**, le fait de négliger la perte de pression par accélération dans la zone sous-refroidie.

Comme précédemment, la méthode PINN cherche une fonction qui fait tendre les résidus vers 0, il est donc normal de ne pas obtenir exactement les mêmes résultats que pour la méthode classique. En revanche, on remarque que les solutions ne sont pas éloignées, et que la courbe possède une pente presque constante basée sur les résultats dans la zone de saturation.

Enfin, pour ce qui est du titre de l'écoulement, les résultats semblent cohérents quand nous travaillons avec les mesures d'une même expérience.

L'algorithme PINN a plusieurs résultats difficiles à accepter, comme l'expérience 19 avec la corrélation d'Inoue (fig. 6). On a un titre qui n'est jamais nul, ce qui ne peut pas se produire étant donné que la première zone de la conduite est censée être sous-refroidie. Cela s'explique par le fait qu'on ne fixe nulle part une condition à la limite pour x. Sa dérivée intervient dans plusieurs équations mais sa valeur même n'intervient qu'implicitement (dans des propriétés thermodynamiques du mélange par exemple). Les résultats présentés fig. 7 montrent que dans certains cas, la longueur à saturation a été déterminée sans connaissance de conditions aux limites autres que la pression de

sortie.

Pour tous les résultats obtenus, il semble cohérent de dire que ceux qui proviennent de l'algorithme classique sont bons. Ils correspondent aux données expérimentales et le titre de l'écoulement (dont nous n'avons pas de données de comparaison) est le même pour les deux corrélations utilisées.

L'algorithme PINN nécessite un choix de paramètres précis pour qu'il fonctionne et qu'il converge vers les bonnes valeurs. Une fois fonctionnel il est plus flexible que l'algorithme classique car il permet d'avoir toutes les propriétés de l'écoulement à toutes les positions de la conduite. Mais son implémentation plus complexe nécessite d'avoir plus de jeux de données ou bien comme nous l'avons fait un algorithme plus classique, en discrétisation pour pouvoir faire des comparaisons.

7.2 Conclusion

La résolution de ce problème par un algorithme classique a révélé des résultats convaincants. La partie la plus complexe étant l'implantation des modèles de corrélation, la partie principale de l'algorithme a montré sa robustesse et sa rapidité d'exécution. Les écarts de chute de pression avec les données expérimentales sont contenues sous les 10% pour l'ensemble des cas étudiés et sont en accord avec les relevés ponctuels dont on dispose. En revanche il est difficile de dire si l'une des deux corrélations du modèle à écart de vitesse est meilleure que l'autre car les résultats différent selon les configurations expérimentales.

Implanter la méthode PINN fut un défi intéressant : avec de nombreux aspects pertinents d'un point de vue pratique (différentiation plus poussée et simple, modèles thermodynamiques, représentation continue...) il y a sûrement des pistes à approfondir. Néanmoins du fait du manque de régularité des modèles et du manque de recul sur ces méthodes, nous n'avons pas pu résoudre un certain nombre de points pour atteindre la précision et la robustesse escomptées.

Une perspective qu'il pourrait être intéressant de suivre pour canaliser la multitude de modèles de corrélations et leur complexité d'implémentation peut être cherchée du côté de modèles réduits conçus sous forme de réseaux de neurones "simples", bien plus mature dans ce domaine là, comme cela a été fait chez [Alvarez del Castillo et al., 2012]. La perte en interprétabilité pourrait alors être compensée par un regain en efficacité et éventuellement un meilleur couplage avec un PINN.

Références

[Al-Aradi et al.,] Al-Aradi, A., Correia, A., Naiff, D., Jardim, G., Vargas, F. G., and Saporito, Y. Solving Nonlinear and High-Dimensional Partial Differential Equations via Deep Learning. page 76.

[Alvarez del Castillo et al., 2012] Alvarez del Castillo, A., Santoyo, E., and García-Valladares, O. (2012). A new void fraction correlation inferred from artificial neural networks for modeling two-phase flows in geothermal wells. *Computers & Geosciences*, 41 :25–39.

[Brunton et al., 2019] Brunton, S., Noack, B., and Koumoutsakos, P. (2019). Machine Learning for Fluid Mechanics. *arXiv :1905.11075 [physics, stat]*. arXiv : 1905.11075.

[Brunton et al., 2016] Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS*, 113(15) :3932–3937.

[Chen et al., 2020] Chen, Y., Lu, L., Karniadakis, G. E., and Negro, L. D. (2020). Physics-informed neural networks for inverse problems in nano-optics and metamaterials. *arXiv :1912.01085 [physics]*. arXiv : 1912.01085.

[Chexal and Lellouche, 1986] Chexal, B. and Lellouche, G. (1986). Full-range drift-flux correlation for vertical flows. Revision 1. Technical Report EPRI-NP–3989-SR-REV.1, Electric Power Research Inst.

[Coddington and Macian, 2002] Coddington, P. and Macian, R. (2002). A study of the performance of void fraction correlations used in the context of drift-flux two-phase flow models. *Nuclear Engineering and Design*, 215(3) :199–216.

[Corbetta and Field,] Corbetta, M. and Field, M. Application of sparse identification of nonlinear dynamics for physics-informed learning. page 9.

[Freidel, 1979] Freidel, L. (1979). Improved friction pressure drop correlations for horizontal and vertical two-phase flow. In *European Two-Phase Flow Group Meeting*.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Guo et al., 2016] Guo, X., Li, W., and Iorio, F. (2016). Convolutional Neural Networks for Steady Flow Approximation. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 481–490, New York, NY, USA. ACM. event-place : San Francisco, California, USA.

[Haghighat et al., 2020] Haghighat, E., Raissi, M., Moure, A., Gomez, H., and Juanes, R. (2020). A deep learning framework for solution and discovery in solid mechanics : linear elasticity. *arXiv :2003.02751 [cs, stat]*. arXiv : 2003.02751.

[Hornik et al., 1989] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5) :359–366.

[Inoue et al., 1993] Inoue, A., Kurosu, T., Yagi, M., Morooka, S., Hoshide, A., Ishizuka, T., and Yoshimura, K. (1993). In-bundle void measurement of a BWR fuel assembly by an x-ray CT scanner : Assessment of BWR design void correlation and development of new void correlation. *2nd ASME-JSME international conference on nuclear engineering – 1993*.

[Kingma and Ba, 2017] Kingma, D. P. and Ba, J. (2017). Adam : A Method for Stochastic Optimization. *arXiv :1412.6980 [cs]*. arXiv : 1412.6980.

[Leshno et al., 1993] Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6) :861–867.

[Lu et al., 2020] Lu, L., Dao, M., Kumar, P., Ramamurty, U., Karniadakis, G. E., and Suresh, S. (2020). Extraction of mechanical properties of materials through deep learning from instrumented indentation. *PNAS*.

[Mao et al., 2020] Mao, Z., Jagtap, A. D., and Karniadakis, G. E. (2020). Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360 :112789.

[Raissi et al., 2019a] Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019a). Physics-informed neural networks : A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378 :686–707.

[Raissi et al., 2019b] Raissi, M., Wang, Z., Triantafyllou, M. S., and Karniadakis, G. E. (2019b). Deep learning of vortex-induced vibrations. *Journal of Fluid Mechanics*, 861 :119–137.

[Raissi et al., 2018] Raissi, M., Yazdani, A., and Karniadakis, G. (2018). *Hidden Fluid Mechanics : A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data*.

[Revellin and Thome, 2007] Revellin, R. and Thome, J. R. (2007). Adiabatic two-phase frictional pressure drops in microchannels. *Experimental Thermal and Fluid Science*, 31(7) :673–685.

[Rudy, 2019] Rudy, S. (2019). *Computational Methods for System Identification and Data-driven Forecasting*. Ph.D., Ann Arbor, United States.

[TensorFlow,] TensorFlow. Partial Differential Equations | TensorFlow Core.