CERF : Computation, ERosion and Flow V1.0

GOLAY Frédéric

Table of contents

Introduction **3**

Licence **4**

1 Modélisation **5**
1.1 La méthode des volumes finis . . . 5
1.1.1 Principe de la méthode des volumes finis . . . 5
1.1.2 Discrétisation temporelle . . . 7
1.1.3 Discrétisation spatiale . . . 8
1.2 Modèle de Saint Venant . . . 13
1.2.1 Équations constitutives . . . 13
1.2.2 Prise en compte de la bathymétrie . . . 15
1.2.3 Critère de raffinement . . . 15
1.2.4 Prise en compte de la friction . . . 15
1.2.5 Solveur de Riemann pour le modèle de Saint-Venant . . . 16
1.3 Modèle d'écoulement bifluide . . . 22
1.3.1 Équations constitutives . . . 22
1.3.2 Critère de raffinement . . . 24
1.3.3 Raidissement de la surface libre . . . 25
1.3.4 Notion de *"fluide rigide"* . . . 25
1.3.5 Solveur de Riemann pour le modèle bifluide . . . 26

2 Guide de l'utilisateur **33**
2.1 Installation du logiciel . . . 33
2.1.1 Prérequis . . . 33
2.1.2 Installation . . . 33
2.1.3 Déroulement d'un calcul . . . 35
2.2 Le fichier de configuration *.inp . . . 37
2.2.1 Généralités . . . 37
2.2.2 Définition des paramètres physiques: **PHYS** . . . 38
2.2.3 Définition du maillage maître : **MAME** . . . 39
2.2.4 Définition des conditions initiales: **INIT** . . . 40
2.2.5 Définition de la bathymétrie et de la friction : **BATH** . . . 41
2.2.6 Définition des conditions aux limites : **COND** . . . 42
2.2.7 Définition du maillage : **MESH** . . . 43

2.2.8 Définition d'"obstacle" : **OBST** 44
2.2.9 Définition des paramètres numériques: **NUME** 46
2.3 Les fichiers sources de **CERF** 47
2.3.1 Structure et modules 47
2.3.2 Le graphe de dépendance du programme principal 48
2.3.3 Le graphe de dépendance du programme de pré-traitement 48
2.3.4 Le graphe de dépendance du programme de modification du maillage . . 48

3 Exemples documentés **53**
3.1 Exemples avec le modèle de Saint-Venant 53
3.1.1 Problème de Riemann 53
3.1.2 Ressaut hydraulique 61
3.1.3 Friction, cas test de McDonald 74
3.2 Exemples avec le modèle bifluide eulérien 79
3.2.1 Problème de Riemann 79
3.2.2 Rupture de barrage 86
3.2.3 Chute d'un cylindre sur un plan d'eau 91

Références **102**

Introduction

Ce document est un guide d'utilisation de **CERF**. **CERF** est un code de calcul par volumes finis dévolu à la simulation d'écoulements à surface libre. **CERF** est un code de recherche développé au sein de l'institut de mathématique de Toulon sous la direction de Frédéric Golay. **CERF** résout les équations de Saint-Venant et d'Euler bi-fluide isotherme. Ces systèmes hyperboliques d'équations permettent d'utiliser une intégration temporelle explicite et parallèle. La structure de données de **CERF** est basée sur une approche BB-AMR (Block Based Adaptive Mesh Refinement) qui permet de gérer des maillages adaptatifs en temps et en espace. **CERF** est un code de calcul parallèle qui utilise la bibliothèque MPI pour la communication entre les différents processus. **CERF** est écrit en Fortran 90. Dans sa version V1.0 le maillage de **CERF** est exclusivement constitué d'hexaèdres.

Il est organisé de la manière suivante. Dans le premier chapitre, nous présentons les modèles physiques utilisés dans **CERF** pour simuler des écoulements à surface libre. Les modèles d'érosion sont en cours de développement. Nous décrivons également les schémas numériques utilisés. Le deuxième chapitre est consacré à l'installation et la mise en œuvre de **CERF** ainsi que la description des fichiers d'entrée et de sortie de **CERF**. Nous expliquons comment configurer une simulation et comment visualiser les résultats. Enfin, le troisième chapitre est ensemble d'exemples documentés.

Ce document a été rédigé avec Quarto.
Ce logiciel est régi par la licence CeCILL-B soumise au droit français et respectant les principes de diffusion des logiciels libres.

Licence

Copyright Institut de Mathématique de Toulon, (2025).

frederic.golay@univ-tln.fr

Ce logiciel est un code de calcul par volumes finis dévolu à la simulation d'écoulements à surface libre.

Ce logiciel est régi par la licence CeCILL-B soumise au droit français et respectant les principes de diffusion des logiciels libres. Vous pouvez utiliser, modifier et/ou redistribuer ce programme sous les conditions de la licence CeCILL-B telle que diffusée par le CEA, le CNRS et l'INRIA sur le site "http://www.cecill.info".

En contrepartie de l'accessibilité au code source et des droits de copie, de modification et de redistribution accordés par cette licence, il n'est offert aux utilisateurs qu'une garantie limitée. Pour les mêmes raisons, seule une responsabilité restreinte pèse sur l'auteur du programme, le titulaire des droits patrimoniaux et les concédants successifs.

A cet égard l'attention de l'utilisateur est attirée sur les risques associés au chargement, à l'utilisation, à la modification et/ou au développement et à la reproduction du logiciel par l'utilisateur étant donné sa spécificité de logiciel libre, qui peut le rendre complexe à manipuler et qui le réserve donc à des développeurs et des professionnels avertis possédant des connaissances informatiques approfondies. Les utilisateurs sont donc invités à charger et tester l'adéquation du logiciel à leurs besoins dans des conditions permettant d'assurer la sécurité de leurs systèmes et ou de leurs données et, plus généralement, à l'utiliser et l'exploiter dans les mêmes conditions de sécurité.

Le fait que vous puissiez accéder à cet en-tête signifie que vous avez pris connaissance de la licence [CeCILL-B], et que vous en avez accepté les termes.

1 Modélisation

Ce chapitre décrit les modèles physiques utilisés dans **CERF** pour simuler des écoulements (sans érosion dans la version V1.0), ainsi que la manière dont ces modèles sont implémentés dans le code. Il est divisé en trois sections. La première section présente dans le cas général la méthode des volumes finis. La deuxième section décrit le modèle d'écoulement en eaux peu profondes de Saint Venant. La troisième section présente le modèle d'écoulement bifluide eulérien utilisé dans **CERF**.

1.1 La méthode des volumes finis

La méthode des volumes finis est une méthode numérique utilisée pour résoudre des équations aux dérivées partielles. Cette méthode est utilisée dans **CERF** pour résoudre les équations de Saint Venant et les équations d'écoulement bifluide.

1.1.1 Principe de la méthode des volumes finis

Dans cette section, nous rappelons rapidement l'approximation par volumes finis, en dimension n_{dim}, d'un système d'équations de conservation de la forme :

$$\frac{\partial \overrightarrow{w}}{\partial t} + \nabla \cdot \overline{\overline{F}}(\overrightarrow{w}) = \vec{S}(\overrightarrow{w}) \tag{1.1}$$

où $\overrightarrow{w}(\vec{x}, t)$ est le vecteur des variables conservatives de taille n_{dof} dépendant de l'espace et du temps, $\overline{\overline{F}}$ la matrice des flux de taille $n_{dof} \times n_{dim}$, $\vec{S}$ le vecteur des termes sources de taille n_{dof} et $\nabla\cdot$ l'opérateur *divergence*.

Le système est résolu dans $\mathbb{R}^{n_{dim}}$. Le domaine de calcul $\Omega \subset \mathbb{R}^{n_{dim}}$ est divisé en un ensemble de volumes de contrôle, également appelés cellules, $\Omega = \cup_k C_k$. Nous commençons par intégrer l'Equation 1.1 sur une cellule C_k au moyen de la formule de Green :

$$\int_{C_k} \frac{\partial \overrightarrow{w}}{\partial t} d\Omega + \int_{C_k} \nabla \cdot \overline{\overline{F}} d\Omega = \int_{C_k} \vec{S}(\overrightarrow{w}) d\Omega$$

$$\int_{C_k} \frac{\partial \overrightarrow{w}}{\partial t} d\Omega + \sum_a \int_{\partial C_{k/a}} \overline{\overline{F}} \cdot \vec{n}_{k/a} d\partial\Omega = \int_{C_k} \vec{S}(\overrightarrow{w}) d\Omega \tag{1.2}$$

où $\vec{n}_{k/a}$ désigne le vecteur normal unitaire sur la frontière $\partial C_{k/a}$ entre la cellule C_k et la cellule voisine C_a. La sommation est effectuée sur toutes les cellules voisines de C_k.

La méthode des volumes finis consiste alors à approcher la solution par des fonctions constantes par cellule, si bien qu'en posant

$$\overline{w}_k(t) \approx \frac{1}{|C_k|} \int_{C_k} \overline{w}(\vec{x}, t) d\Omega$$

où $|C_k|$ représente le volume de la cellule C_k, nous approchons l'Equation 1.2 par

$$|C_k| \frac{\partial \overline{w}_k(t)}{\partial t} + \sum_a |\partial C_{k/a}| \, \hat{\vec{F}}\left(\overline{w}_k(t), \overline{w}_a(t), \vec{n}_{k/a}\right) = |C_k| S_k(t) \tag{1.3}$$

où $S_k(t) \approx \frac{1}{|C_k|} \int_{C_k} S(\vec{x}, t) d\Omega$ représente la valeur moyenne du vecteur sollicitation sur la cellule, $|\partial C_{k/a}|$ la surface de la face entre les cellules C_k et C_a et $\hat{\vec{F}}$ une approximation du flux.

Comme les inconnues $\overline{w}$ ne sont pas définies à l'interface entre les cellules, le flux ne peut être calculé aisément. On l'approche donc par un flux numérique dépendant de l'interface et des valeurs des variables dans les cellules adjacentes. On note :

$$\vec{F}_{k/a}(t) = \hat{\vec{F}}\left(\overline{w}_k(t), \overline{w}_a(t), \vec{n}_{k/a}\right) \approx \frac{1}{|\partial C_{k/a}|} \int_{\partial C_{k/a}} \overline{\overline{F}} \cdot \vec{n}_{k/a} ds$$

Il existe de très nombreuses approches : Rusanov, HLLC, … Dans **CERF** le flux numérique, par la méthode de Godunov, est calculé en utilisant la solution exacte du problème de Riemann à l'interface k/a (pour plus de détails, voir Section 1.2.5 et Section 1.3.5).

On peut atteindre l'ordre 2 en espace en utilisant la méthode MUSCL (Monotone Upstream Scheme for Conservation Laws) pour reconstruire les variables primitives aux interfaces. À partir des valeurs des variables sur les cellules voisines, on estime le gradient des variables primitives. Soit w_{p_i} la ième composante des variables primitives et ${w_{p_i}}^k$ sa valeur au centre de la cellule, on peut estimer le gradient sur la cellule C_k comme :

$$|C_k| \widetilde{\nabla} w_{p_i} = \int_{C_k} \nabla w_{p_i} d\Omega = \sum_a \int_{\partial C_{k/a}} w_{p_i} \vec{n}_{k/a} d\partial\Omega = \sum_a |\partial C_{k/a}| \frac{1}{2} \left({w_{p_i}}^k + {w_{p_i}}^a \right) \vec{n}_{k/a}$$

et on peut alors reconstruire les variables primitives de la cellule C_k de centre K sur les centres F des faces de la cellule telles que :

$$w_{p_i}^F = w_{p_i}^k + \widetilde{\nabla} w_{p_i} \cdot \overrightarrow{KF}$$

Pour atteindre l'ordre 2, on remplace alors $\hat{\vec{F}}\left(\overline{w}_k(t), \overline{w}_a(t), \vec{n}_{k/a}\right)$ par $\hat{\vec{F}}\left(\overline{w}_k^F(t), \overline{w}_a^F(t), \vec{n}_{k/a}\right)$

> **Note**
>
> En général il est nécessaire d'adjoindre un limiteur à la méthode MUSCL. Dans **CERF** le limiteur de Barth-Jespersen est utilisé : $Min(w_{p_F}, w_{p_F}) \leq w_{p_i}^F \leq Max(w_{p_F}, w_{p_F})$.

1.1.2 Discrétisation temporelle

Afin de faciliter la parallélisation du code, **CERF** ne dispose que de schémas explicites: le schéma de Runge-Kutta d'ordre 1 ou 2. L'intervalle de temps $[0, T]$ est discrétisé de telle façon que $t_{n+1} = t_n + \delta t_n$ où δt_n représente le pas de temps au temps t_n. La méthode de Runge-Kutta d'ordre 1, ou encore méthode d'Euler explicite d'ordre 1, consiste donc à réécrire l'Equation 1.3 à l'instant t_n en approchant la dérivée temporelle comme une différence finie, de telle sorte que:

$$|C_k| \frac{\vec{w}_k(t_{n+1}) - \vec{w}_k(t_n)}{\delta t_n} + \sum_a |\partial C_{k/a}| \, \vec{F}_{k/a}(t_n) = |C_k| \vec{S}_k(t_n)$$

En notant $\vec{w}_k(t_n) = \vec{w}_k^n$, on calcule donc de façon explicite la nouvelle valeur des variables conservatives à l'instant t_{n+1}:

$$\vec{w}_k^{n+1} = \vec{w}_k^n + \frac{\delta t_n}{|C_k|} \sum_a |\partial C_{k/a}| \, \vec{F}_{k/a}^n - \delta t_n \vec{S}_k^n$$

La méthode de Runge-Kutta d'ordre 2 consiste à calculer la solution à l'instant t_{n+1} en deux étapes. La première étape consiste à calculer la solution à l'instant $t_{n+\frac{1}{2}}$ en utilisant le schéma d'Euler explicite d'ordre 1. La deuxième étape consiste à calculer la solution à l'instant t_{n+1} en utilisant le schéma d'Euler explicite d'ordre 1 avec la solution obtenue à l'instant $t_{n+\frac{1}{2}}$.

$$\vec{w}_k^{n+\frac{1}{2}} = \vec{w}_k^n + \frac{\delta t_n}{2|C_k|} \sum_a |\partial C_{k/a}| \, \vec{F}_{k/a}^n - \frac{\delta t_n}{2} \vec{S}_k^n$$

$$\vec{w}_k^{n+1} = \vec{w}_k^n + \frac{\delta t_n}{|C_k|} \sum_a |\partial C_{k/a}| \, \vec{F}_{k/a}^{n+\frac{1}{2}} - \delta t_n \vec{S}_k^{n+\frac{1}{2}}$$

> **Important 1: Condition de stabilité de Courant-Friedrichs-Lewy**
>
> Les méthodes explicites sont simples à mettre en oeuvre mais elles sont conditionnées par la stabilité de Courant-Friedrichs-Lewy (CFL). La condition CFL est une condition nécessaire pour garantir la stabilité du schéma numérique. Cette condition est dépendante de la vitesse de propagation des ondes dans le milieu et de la taille des cellules. En pratique, elle est déterminée pour chaque modèle hyperbolique traité. Dans **CERF** la taille des cellules d_k est déterminée par le ratio entre le volume de la cellule et la surface

de sa frontière:

$$d_k = \frac{|C_k|}{\sum_a |\partial C_{k/a}|}$$

Dans le cas 1D ou 2D, les faces de symétrie ne sont pas considérées dans le calcul de la surface des frontières. La condition CFL est alors donnée par :

$$\delta t_n \leq \alpha_{CFL} \frac{\min_k d_k}{\max_k \lambda_k}$$

où α_{CFL} est un coefficient à définir par l'utilisateur et λ_k est la vitesse de propagation des ondes dans la cellule C_k dépendant du modèle utilisé (Note 1 pour Saint-Venant et Note 2 pour l'écoulement bifluide).

1.1.3 Discrétisation spatiale

! Important

Dans sa version V1.0 **CERF** dispose **exclusivement** de cellules hexaédriques et donc de faces quadrangulaires ! La généralisation est en cours de développement.

i Note

CERF permet l'utilisation de maillages dynamiques, c'est à dire que régulièrement il est possible de raffiner ou déraffiner le maillage (**AMR** : Adaptive Mesh Refinement).

CERF est un code nativement tridimensionnel. Dans sa version initiale, **CERF** ne dispose que de cellules hexaédriques et donc de faces quadrangulaires. Les cellules et les faces ne sont pas nécessairement régulières. Pour autant, comme illustré Figure 1.1, il est possible d'effectuer des simulations bidimensionnelles en réalisant un maillage avec **une** seule cellule dans la direction z et en imposant des conditions de type **"miroir"** sur les faces dans le plan x, y. De même, il est possible de réaliser des simulations unidimensionnelles en réalisant un maillage avec **une** seule cellule dans les directions y et z et en imposant des conditions de type *"miroir"* sur les faces dans le plan x, z et x, y.

1.1.3.1 Stratégie de maillage

CERF permet d'utiliser des outils de raffinement et de déraffinement de maillage (AMR: Adaptive Mesh Refinement). Afin de ne pas pénaliser le temps de calcul, la modification du maillage n'est pas réalisée à chaque pas de temps, mais à des instants déterminés par l'utilisateur.

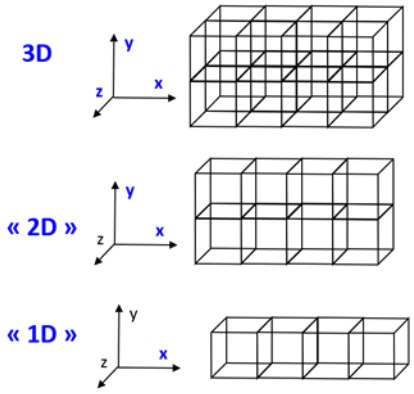


Figure 1.1: Maillages tri-bi-unidimensionnel avec des cellules hexaédriques

Cette approche permet de conserver une bonne qualité de maillage tout en limitant le coût de calcul. Cette méthode est dénommée BB-AMR (Block Based Adaptive Mesh Refinement). Nous allons décrire ci-après les différentes étapes de la stratégie de maillage utilisée dans **CERF**.

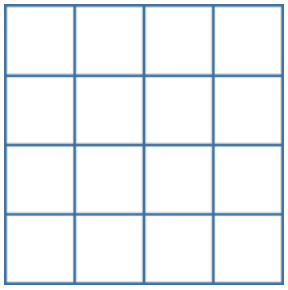


Figure 1.2: Maillage initial conforme : *"Master Mesh"*.

Étape 1 : Pour simplifier l'explication le maillage présenté Figure 1.2 est constitué de quadrangles réguliers, qui peuvent représenter un maillage hexaédrique vu suivant z avec une seule cellule suivant z. Comme illustré Figure 1.2, on part d'un maillage initial **conforme** constitué uniquement d'hexaèdres de forme quelconque. Dans un cas simple le maillage peut être créé par **CERF**, sinon il peut être importé au format GMSH V2.2. Ce maillage est dénommé le ***maillage maître*** ou ***Master Mesh***. Il est primordial que le maillage soit conforme afin de définir aisément les faces reliant les cellules. on appellera ***bloc*** les cellules de ce maillage initial.

Étape 2: On ordonne les blocs du maillage initial en les numérotant suivant le *Z-order* ou *Code de Morton* (Figure 1.3). Par cette approche, que l'on appelle *Space Filling Curve* (Section 1.1.3.2), la numérotation permet de parcourir les blocs, même en trois dimensions, en suivant une numérotation telle que la plupart du temps des blocs successifs dans la numérotation soient voisins dans l'espace.

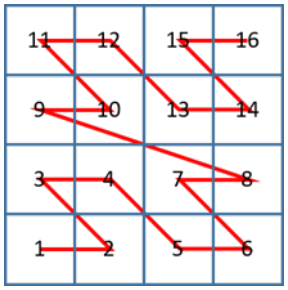


Figure 1.3: Numérotation du maillage initial suivant les ***codes de Morton***.

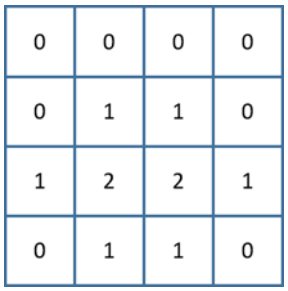


Figure 1.4: Définition du niveau de raffinement des blocs.

Étape 3 : On définit ensuite le niveau de raffinement des blocs, comme illustré Figure 1.4. Initialement, le niveau de raffinement est défini par l'utilisateur. En cours de calcul, le niveau de raffinement est défini par un critère Section 1.1.3.3. Comme vérifié dans Ersoy, Golay, and Yushchenko (2013) ou Thomas Altazin and Yushchenko (2016) la différence de niveau de raffinement entre deux blocs ne doit pas excéder 1. Lors du raffinement le bloc est divisé par 2 dans chaque direction et ce pour chaque niveau nécessaire. Ainsi, si on note Niv le niveau de raffinement requis (où $Niv = 0$ représente un bloc qui n'est pas raffiné) et $dim = 1, 2, 3$ la dimension du problème alors le nombre de cellules dans un bloc sera : $2^{dim \times Niv}$. Dans l'exemple Figure 1.4, le bloc 1 n'est pas raffiné, le bloc 2 est raffiné de 1 niveau et le bloc 4 de 2 niveaux. Ce qui conduit alors, comme illustré Figure 1.5 dans le cas où $n_{dim} = 2$, à un nombre de 1 cellule dans le bloc 1, 4 cellules dans le bloc 2 et 16 cellules dans le bloc 4.

Figure 1.5: Décomposition du maillage en domaines.

Étape 4 : Comme les schémas d'intégration en temps dans **CERF** sont explicites, il est aisé de paralléliser l'algorithme de résolution. Pour cela, il est nécessaire de *découper* le maillage en plusieurs domaines, si possible avec un nombre de cellules équilibré et des interfaces minimales afin de minimiser les échanges de données entre les domaines. Il existe de nombreux algorithmes, parfois complexes, qui effectuent cette opération de façon performante. Dans **CERF** nous avons choisi un compromis entre simplicité et efficacité. Nous avons opté pour un découpage piloté par les codes de Morton. Connaissant le nombre de cellules par bloc, on parcourt les blocs en suivant la numérotation de Morton et on les attribue à un domaine. On s'assure que le nombre de cellules par domaine est équilibré. Comme illustré Figure 1.5, le maillage initial est découpé en 3 domaines composés de 25, 21 et 18 cellules. Chaque domaine de calcul sera ensuite affecté à un processus de calcul

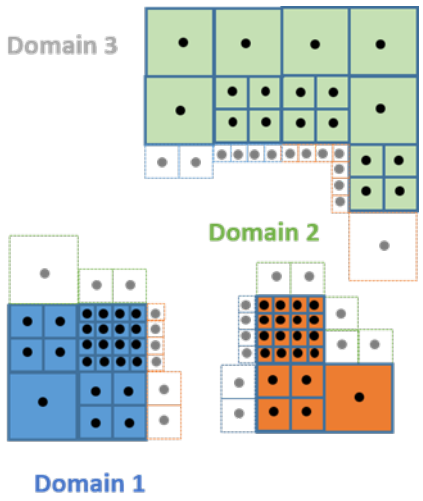


Figure 1.6: Réalisation du maillage.

Étape 5 : La dernière étape consiste alors à réaliser le maillage. Afin de communiquer les informations entre les domaines, **CERF** créé des cellules fantômes. Ces cellules sont des copies des cellules voisines du domaine. Elles sont utilisées pour calculer les flux numériques aux interfaces entre les domaines. Comme illustré Figure 1.6, les cellules fantômes sont représentées en gris. On notera que le maillage ainsi réalisé n'est **pas conforme**! Par exemple, la première cellule en bas à gauche du domaine 1 de Figure 1.6 est entourée de 6 faces (dans le plan x, y). Le raffinement des blocs définissant le maillage est bien sûr fonction de la dimension du problème à traiter (Figure 1.7). Enfin, si le maillage est modifié en cours de calcul, il est nécessaire d'effectuer une projection des variables conservatives sur le nouveau maillage. Dans une opération de remaillage on ne peut diminuer ou augmenter d'un niveau supérieur à 1, si bien que l'étape de projection est relativement simple.

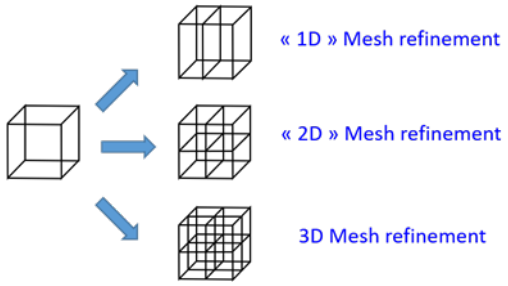


Figure 1.7: Raffinement en fonction de la dimension du problème.

1.1.3.2 Code de Morton

Les ***courbes de remplissages*** ou ***Space Filling Curves*** sont des courbes qui permettent de parcourir un espace de dimension supérieure. Courbes de Peano, puis de Hilbert ou Lebesgue elles sont utilisées par exemple pour indexer des données et en particulier dans la construction de ***quadtree*** ou ***octree***. Dans **CERF**, la numérotation par ***code de Morton*** ou ***Z-oder curve*** est utilisée. Cette numérotation permet d'indexer un ensemble de données (points, cellules, faces, ...) de telle sorte que des données voisines dans l'espace soient voisines dans l'indexation. Cette propriété est utilisée dans **CERF** pour découper le maillage en domaines de calcul en indexant les blocs du maillage maître. Considérons un point de coordonnées (x, y, z) dans un espace tridimensionnel. Pour indexer ce point, on commence par convertir les coordonnées en entiers positifs. Pour ce faire on définit les coordonnées les plus petites du domaine d'étude $(x_{min}, y_{min}, z_{min})$ et une distance d_{morton} définissant la grille du domaine, par exemple un dixième de la taille de la plus petite cellule du domaine. On définit alors les coordonnées entières du point (ix, iy, iz) par:

$$ix = \frac{x - x_{min}}{d_{morton}},\ iy = \frac{y - y_{min}}{d_{morton}},\ iz = \frac{z - z_{min}}{d_{morton}}$$

La numérotation de Morton consiste alors à convertir ces coordonnées entières en base 2 (ix_2, iy_2, iz_2), puis à entrelacer les bits de chaque coordonnées afin de former un seul nombre entier en base 2 M_2. Enfin, revenant en base 10, on obtient l'index de Morton M du point (x, y, z).

Example 1.1. $ix = 2$, $iy = 5$, $iz = 0$.
On a alors $ix_2 = 010$, $iy_2 = 101$ et $iz_2 = 000$.
En entrelaçant les bits on obtient $M_2 = 010100010$ et en base 10 $M = 162$.

> **Note**
>
> Tous les outils d'indexation de **CERF** se trouvent dans le module *~/sources/UTI/mod_zorder.f90*.

1.1.3.3 Critère de raffinement

En général, un critère de raffinement est défini pour chaque modèle physique traité. Ce critère est souvent issu de considérations physiques à partir, par exemple, du gradient d'une des variables primitives ou de façon plus pertinente et plus complexe d'un calcul d'erreur *a posteriori* ou de la construction d'un indicateur d'erreur. Dans le cas des problèmes hyperboliques, il existe un critère plus général issu des travaux de Croisille (1990). Ce critère est basé sur la production numérique d'entropie. Il a été utilisé par Frédéric Golay (2009), Ersoy, Golay, and Yushchenko (2013) et longuement étudié par Puppo (2004). L'inégalité d'entropie de Lax (1971), nous permet de dire que pour tout système hyperbolique de lois de conservation de la forme Equation 1.1, il existe une entropie mathématique $s(\overrightarrow{w})$ et un flux d'entropie $\vec{\psi}_s(\overrightarrow{w})$ tels que :

$$\frac{\partial s(\overrightarrow{w})}{\partial t} + \nabla \cdot \vec{\psi}_s(\overrightarrow{w}) \leq 0 \tag{1.4}$$

où le flux d'entropie $\vec{\psi}_s(\overrightarrow{w})$ doit satisfaire une équation de compatibilité $\nabla_w \psi_s = \nabla_w \overline{\overline{F}} \cdot \nabla_w s$. Pour des solutions régulières l'Equation 1.4 est une égalité. Dans le cas d'un choc, c'est une inégalité stricte. Or, lors d'une simulation numérique, on constate que la production numérique d'entropie n'est pas nulle! Il s'avère alors que la production numérique d'entropie est un très bon indicateur d'erreur, à l'exception des chocs où elle tend vers l'infini. On définit donc p_k^s la production numérique d'entropie sur la cellule C_k par :

$$p_k^s = \frac{1}{|C_k|} \int_{C_k} \left(\frac{\partial s(\overrightarrow{w})}{\partial t} + \nabla \cdot \vec{\psi}_s(\overrightarrow{w}) \right) d\Omega$$

.

Le critère de raffinement C_b est défini par bloc comme le ratio de la production numérique d'entropie dans le bloc par rapport à la production numérique d'entropie totale sur le domaine.

$$C_b = \frac{\int_{Bloc} \left(\frac{\partial s(\overrightarrow{w})}{\partial t} + \nabla \cdot \vec{\psi}_s(\overrightarrow{w}) \right) d\Omega}{\int_{\Omega} \left(\frac{\partial s(\overrightarrow{w})}{\partial t} + \nabla \cdot \vec{\psi}_s(\overrightarrow{w}) \right) d\Omega}$$

Dans **CERF**, le critère de raffinement est défini par bloc. Si le critère est supérieur à une valeur seuil α_{max}, le bloc est raffiné. Si le critère est inférieur à une valeur seuil α_{min}, le bloc est déraffiné. Les valeurs seuils sont définies par l'utilisateur.

1.2 Modèle de Saint Venant

1.2.1 Équations constitutives

Le modèle d'écoulement en eaux peu profondes ou modèle de Saint-Venant est un modèle d'écoulement simplifié. Il est dérivé des équations de Navier-Stokes en négligeant les termes de

viscosité, assumant une répartition hydrostatique de la pression et en supposant que la hauteur de l'eau est beaucoup plus petite que la longueur caractéristique de l'écoulement. Le modèle original de Barré de Saint-Venant est un modèle 1D Saint-Venant (1871), la dérivation du modèle est expliqué par exemple par Marche (2007) ou Poussel (2024). Les développements du modèle de **CERF** ont été initiés par Pons (2018). Dans un repère (O, x, y, z), on considère un écoulement en eaux peu profondes sur un domaine où la bathymétrie est définie par $z_b(x, y)$ avec une hauteur d'eau $h(x, y)$, si bien que le niveau de la surface libre est défini par $\zeta = h + z_b$, comme illustré Figure 1.8.

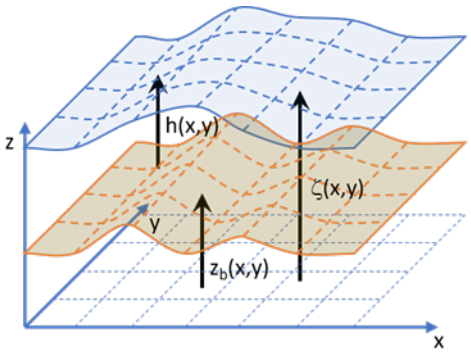


Figure 1.8: Notations du modèle d'écoulement en eaux peu profondes

Les équations de Saint-Venant sont les suivantes :

$$\frac{\partial h}{\partial t} + \nabla \cdot (h\vec{u}) = 0 \tag{1.5}$$

$$\frac{\partial h\vec{u}}{\partial t} + \nabla \cdot (h\vec{u} \otimes \vec{u} + g\frac{h^2}{2}\bar{\bar{I}}) = -gh\nabla z_b - gh\overrightarrow{S_f} \tag{1.6}$$

où $\vec{u} =< u, v >^T$ représente la vitesse de l'écoulement, g l'accélération de la pesanteur, S_f le terme de frottement et où l'on note $\bar{\bar{I}}$ la matrice identité, ∇ l'opérateur gradient, $\nabla\cdot$ l'opérateur divergence et $\otimes$ le produit tensoriel. On définit les variables conservatives $\overrightarrow{w} =< h, hu, hv >^T$ et les variables primitives $\overrightarrow{w_p} =< h, u, v >^T$. On retrouve la forme générale l'Equation 1.1 avec :

$$\bar{\bar{F}}(\overrightarrow{w}) = \begin{bmatrix} hu & hv \\ hu^2 + g\frac{h^2}{2} & huv \\ huv & hv^2 + g\frac{h^2}{2} \end{bmatrix}$$

et

$$\vec{S} = -gh\nabla z_b - gh\overrightarrow{S_f}$$

1.2.2 Prise en compte de la bathymétrie

La prise en compte de la bathymétrie dans le schéma de résolution par volumes finis nécessite une attention particulière. En effet, le saut de bathymétrie inhérent au fait que la bathymétrie soit considérée constante par cellule, peut générer des vagues parasites. La méthode de reconstruction, proposée par Audusse et al. (2004) est donc implémentée dans **CERF**. Soit $\Delta z_{k/a}$ le *saut de bathymétrie* à l'interface entre les cellules. À l'interface, on distingue alors le flux $\vec{F}_k$ contribuant à la cellule C_k et le flux $\vec{F}_a$ contribuant à la cellule C_a, tels que:

$$\vec{F}_k\left(\overrightarrow{w}_k(t), \overrightarrow{w}_a(t), \vec{n}_{k/a}, \Delta z_{k/a}\right) = \vec{F}\left(\overrightarrow{w}_k^*(t), \overrightarrow{w}_a^*(t), \vec{n}_{k/a}\right) + \left\{ \begin{array}{c} 0 \\ \frac{g}{2}\left(h_k^2 - {h_k^*}^2\right)\vec{n}_{k/a} \end{array} \right\}$$

$$\vec{F}_a\left(\overrightarrow{w}_k(t), \overrightarrow{w}_a(t), \vec{n}_{k/a}, \Delta z_{k/a}\right) = \vec{F}\left(\overrightarrow{w}_k^*(t), \overrightarrow{w}_a^*(t), \vec{n}_{k/a}\right) + \left\{ \begin{array}{c} 0 \\ \frac{g}{2}\left(h_a^2 - {h_a^*}^2\right)\vec{n}_{k/a} \end{array} \right\}$$

où $\overrightarrow{w}_k^* =< h_k^*, \vec{u}_k >^T$ et $\overrightarrow{w}_a^* =< h_a^*, \vec{u}_a >^T$, avec :

$$h_k^* = max\left(0, h_k - max\left(0, \Delta z_{k/a}\right)\right)$$

$$h_a^* = max\left(0, h_a - max\left(0, \Delta z_{k/a}\right)\right)$$

1.2.3 Critère de raffinement

Dans le cas du modèle de Saint-Venant, pour définir la production numérique d'entropie Equation 1.4, on utilise l'entropie définie par :

$$s(\overrightarrow{w}) = \frac{1}{2}\left(h||\vec{u}||^2 + gh^2 + 2ghz_b\right)$$

et le flux d'entropie

$$\vec{\psi}_s(\overrightarrow{w}) = \left(s + \frac{gh^2}{2}\right)\vec{u}.$$

1.2.4 Prise en compte de la friction

Dans l'équation de conservation de la quantité de mouvement Equation 1.6, le terme de frottement est pris en compte. Dans **CERF**, le terme de frottement est modélisé par le terme de Manning-Strickler ou le terme de Darcy-Weisbach. Le terme de Manning-Strickler est défini par :

$$\overrightarrow{S_f} = C_f \frac{||\vec{u}||\vec{u}}{h^{4/3}}$$

où $C_f = n^2$ est le coefficient de frottement de Manning-Strickler.

Le terme de Darcy-Weisbach est défini par :

$$\overrightarrow{S_f} = C_f \frac{||\vec{u}||\vec{u}}{h}$$

Le terme de friction est traité par une méthode de *"splitting"*. On résout d'abord le problème Equation 1.1 sans le terme de friction,

$$\frac{\partial \overrightarrow{w}}{\partial t} + \nabla \cdot \overline{\overline{F}}(\overrightarrow{w}) = -gh\nabla z_b \tag{1.7}$$

puis on résout

$$\frac{\partial \overrightarrow{w}}{\partial t} = -gh\overrightarrow{S_f} \tag{1.8}$$

On fait alors l'hypothèse que la friction ne modifie ni la hauteur d'eau, ni la direction de l'écoulement, mais simplement l'intensité de la vitesse de l'écoulement. On peut alors résoudre le problème de friction sur la vitesse par un schéma implicite d'ordre 1 sur un pas de temps δt_n et *"pénaliser"* la vitesse comme dans Varra et al. (2024).
Soit $\vec{u}^*$ la solution du problème Equation 1.7, on peut alors résoudre le problème Equation 1.8 et pénaliser la vitesse par :

$$\vec{u} = \frac{2\vec{u}^*}{1 + \sqrt{1 + 4\beta}}$$

où

$$\beta = \delta t_n g C_f h^{-q}$$

avec $q = 4/3$ pour le modèle de Manning-Strickler et $q = 1$ pour le modèle de Darcy-Weisbach.

1.2.5 Solveur de Riemann pour le modèle de Saint-Venant

1.2.5.1 Généralités

Les flux numériques dans **CERF** sont calculés à l'aide d'un solveur de Riemann. Il existe de très nombreuses références concernant ce solveur comme par exemple Toro (1997). Ce chapitre décrit le solveur de Riemann utilisé pour le modèle de Saint-Venant en s'appuyant sur le cours de Gloria Faccanoni.

On cherche donc à résoudre le problème de Saint-Venant en une dimension. On considère un écoulement en eaux peu profondes de hauteur $h(x,t)$, de vitesse horizontale $u(x,t)$ sur un fond plat décrit par le problème suivant :

$$\begin{cases} \dfrac{\partial h}{\partial t} + \dfrac{\partial (hu)}{\partial x} = 0 \\ \dfrac{\partial (hu)}{\partial t} + \dfrac{\partial (hu^2 + \frac{1}{2}gh^2)}{\partial x} = 0 \end{cases} \tag{1.9}$$

Avec les conditions initiales telles que :

$$\overrightarrow{w_p}(x,0) = \begin{cases} \overrightarrow{w_p}_L & \text{si } x < 0 \\ \overrightarrow{w_p}_R & \text{si } x > 0 \end{cases}$$

où $\overrightarrow{w_p}_L =< h_L, u_L >^T$ et $\overrightarrow{w_p}_R =< h_R, u_R >^T$ sont deux états constants.

En développant l'Equation 1.9, on peut réécrire le système sous la forme suivante :

$$\frac{\partial}{\partial t}\begin{Bmatrix} h \\ u \end{Bmatrix} + \begin{bmatrix} u & h \\ g & u \end{bmatrix} \frac{\partial}{\partial t}\begin{Bmatrix} h \\ u \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

On cherche alors les valeurs propres λ_1 et λ_2 ainsi que les vecteurs propres $\vec{r}_1$ et $\vec{r}_2$ de la matrice ainsi mise en évidence. Un calcul rapide nous donne :

$$\lambda_1 = u - \sqrt{gh}, \lambda_2 = u + \sqrt{gh}$$

avec $\lambda_1 \leq \lambda_2$ et

$$\vec{r}_1 = \begin{Bmatrix} -\sqrt{h} \\ \sqrt{g} \end{Bmatrix}, \vec{r}_2 = \begin{Bmatrix} \sqrt{h} \\ \sqrt{g} \end{Bmatrix}$$

.

> **Note 1**
>
> Les valeurs propres permettent de déterminer la vitesse de propagation des ondes dans la cellule Important 1 comme : $\lambda_k = Max\left(|u - \sqrt{gh}|, |u + \sqrt{gh}|\right)$

Pour déterminer la nature des deux champs caractéristiques, on calcule alors $\nabla_{w_p}\lambda_1 \cdot \vec{r}_1 = 3/2 \neq 0$ et $\nabla_{w_p}\lambda_2 \cdot \vec{r}_2 = 3/2 \neq 0$. Les champs caractéristiques sont donc vraiment non linéaires. Il n'y aura donc pas de discontinuité de contact et les deux ondes sont soit des chocs soit des ondes de raréfaction.

Comme il y a 2 ondes, on peut définir pour chaque onde 1 invariant de Riemann $I_{1;1}$, $I_{2;1}$ tels que $\nabla_{w_p} I_{1;1} \cdot \vec{r}_1 = 0$ et $\nabla_{w_p} I_{2;1} \cdot \vec{r}_2 = 0$. On peut aisément vérifier que $I_{1;1} = u + 2\sqrt{gh}$ et $I_{2;1} = u - 2\sqrt{gh}$ sont des invariants de Riemann.

La solution faible entropique du problème de Riemann est constituée d'au plus trois états constant $\overrightarrow{w_p}_L$, $\overrightarrow{w_p}_*$, $\overrightarrow{w_p}_R$ séparés par deux discontinuités. Ces discontinuités sont soit des chocs soit des ondes de raréfaction. Par la suite, nous allons déterminer Les états $\overrightarrow{w_p}_*$.

1.2.5.2 Étude des détentes

Considérons un état $\overrightarrow{w_p}_g$ et un état $\overrightarrow{w_p}_d$, reliés par une onde de raréfaction k. On doit alors satisfaire les conditions suivantes :

- $\lambda_k(\overrightarrow{w_p}_g) < \lambda_k(\overrightarrow{w_p}_d)$
- L'invariant de Riemann est conservé le long de l'onde de raréfaction $I_{k;1}(\overrightarrow{w_p}_g) = I_{k;1}(\overrightarrow{w_p}_d)$

Appliquons ces conditions au problème de Saint-Venant

Considérons en premier lieu une 1-détente reliant un état $\overrightarrow{w_p}_L$ à un état $\overrightarrow{w_p}_*$, séparés par une onde de raréfaction. Les invariants de Riemann donnent

$$I_{k;1}(\overrightarrow{w_p}_L) = I_{k;1}(\overrightarrow{w_p}_*) \rightarrow u_L + 2\sqrt{g\ h_L} = u_* + 2\sqrt{g\ h_*}$$

soit

$$u_* = u_L + 2\sqrt{g}\left(\sqrt{h_L} - \sqrt{h_*}\right) \tag{1.10}$$

De plus,

$$\lambda_k(\overrightarrow{w_p}_L) < \lambda_k(\overrightarrow{w_p}_*) \rightarrow u_L - \sqrt{g\ h_L} < u_* - \sqrt{g\ h_*} \rightarrow h_* < h_L$$

et

$$u_* > u_L$$

Considérons enfin une 2-détente reliant un état $\overrightarrow{w_p}_*$ à un état $\overrightarrow{w_p}_R$, séparés par une onde de raréfaction. Les invariants de Riemann donnent

$$I_{k;1}(\overrightarrow{w_p}_*) = I_{k;1}(\overrightarrow{w_p}_R) \rightarrow u_* - 2\sqrt{g\ h_*} = u_R - 2\sqrt{g\ h_R}$$

soit

$$u_* = u_R + 2\sqrt{g}\left(\sqrt{h_*} - \sqrt{h_R}\right) \tag{1.11}$$

De plus,

$$\lambda_k(\overrightarrow{w_p}_*) < \lambda_k(\overrightarrow{w_p}_R) \rightarrow u_* + \sqrt{g\ h_*} < u_R + \sqrt{g\ h_R} \rightarrow h_* < h_R$$

et

$$u_* < u_R$$

1.2.5.3 Étude des chocs

Considérons un état $\overrightarrow{w_p}_g$ et un état $\overrightarrow{w_p}_d$, reliés par une onde de choc k. On doit alors satisfaire les conditions suivantes:

- Les conditions de Rankine-Hugoniot où l'on note $\dot{\sigma}_k$ la vitesse du choc : $\dot{\sigma}_k = \dfrac{\left[\vec{F}(\overrightarrow{w})\right]}{[\overrightarrow{w}]}$
- Les conditions entropiques de Lax pour un champs k vraiment non linéaire :

$$\begin{cases} \lambda_{k-1}(\overrightarrow{w_p}_g) < \dot{\sigma}_k \\ \lambda_k(\overrightarrow{w_p}_d) < \dot{\sigma}_k < \lambda_k(\overrightarrow{w_p}_g) \\ \dot{\sigma}_k < \lambda_{k+1}(\overrightarrow{w_p}_d) \end{cases}$$

Appliquées au problème de Saint-Venant, les conditions de Rankine-Hugoniot donnent :

$$\dot{\sigma}_k = \frac{h_d u_d - h_g u_g}{h_d - h_g} = \frac{h_d u_d^2 + \frac{g}{2} h_d^2 - h_g u_g^2 + \frac{g}{2} h_g^2}{h_d u_d - h_g u_g}$$

on peut définir en particulier j_k tel que

$$j_k = h_g u_g - \dot{\sigma}_k h_g = h_d u_d - \dot{\sigma}_k h_d.$$

Par élimination de la vitesse du choc, on obtient :

$$\left(u_g - u_d\right)^2 = \frac{g}{2h_d h_g} \left(h_g - h_d\right)^2 \left(h_g + h_d\right)$$

et donc

$$j_k = \frac{g}{2} \frac{h_g - h_d}{u_d - u_g}$$

Considérons en premier lieu un 1-choc reliant un état $\overrightarrow{w_{p_L}}$ à un état $\overrightarrow{w_{p_*}}$, séparés par une onde de choc. Les conditions de Lax donnent :

$$\left\{ \begin{array}{l} \lambda_1(\overrightarrow{w_{p_*}}) < \dot{\sigma}_1 < \lambda_1(\overrightarrow{w_{p_L}}) \\ \dot{\sigma}_1 < \lambda_2(\overrightarrow{w_{p_*}}) \end{array} \right. \rightarrow \left\{ \begin{array}{l} u_* - \sqrt{g h_*} < \dot{\sigma}_1 < u_L - \sqrt{g h_L} \\ \dot{\sigma}_1 < u_* + \sqrt{g h_*} \end{array} \right.$$

Les conditions de Rankine-Hugoniot donnent :

$$\left\{ \begin{array}{l} j_1 = h_* \left(u_* - \dot{\sigma}_1\right) = h_L \left(u_L - \dot{\sigma}_1\right) \\ \dot{\sigma}_1 = \dfrac{h_* u_*^2 + \frac{g}{2} h_*^2 - h_L u_L^2 - \frac{g}{2} h_L^2}{h_* u_* - h_L u_L} \end{array} \right.$$

Donc à partir des conditions de Lax et en utilisant la première condition de Rankine Hugonio, on obtient :

$$\left\{ \begin{array}{l} u_* - \sqrt{g h_*} < \dot{\sigma}_1 \\ u_L - \sqrt{g h_L} > \dot{\sigma}_1 \\ u_* + \sqrt{g h_*} > \dot{\sigma}_1 \end{array} \right. \rightarrow \left\{ \begin{array}{l} h_* \sqrt{g h_*} > j_1 \\ h_L \sqrt{g h_L} < j_1 \\ -h_* \sqrt{g h_*} < j_1 \end{array} \right.$$

Soit

$$h_* > h_L$$

et comme $j_1 = \frac{g}{2} \frac{h_L - h_*}{u_* - u_L} > 0$, on a

$$u_* < u_L$$

et en utilisant l'expression de $\dot{\sigma}_1$, on trouve la vitesse

$$u_* = u_L + \left(h_L - h_*\right) \sqrt{\frac{g}{2} \frac{h_L + h_*}{h_L h_*}} \tag{1.12}$$

Considérons enfin un 2-choc reliant un état $\overrightarrow{w_{p_*}}$ à un état $\overrightarrow{w_{p_R}}$, séparés par une onde de choc. Les conditions de Lax donnent :

$$\begin{cases} \lambda_1(\overrightarrow{w_{p_*}}) < \dot{\sigma}_2 \\ \lambda_2(\overrightarrow{w_{p_R}}) < \dot{\sigma}_2 < \lambda_2(\overrightarrow{w_{p_*}}) \end{cases} \rightarrow \begin{cases} u_* - \sqrt{gh_*} < \dot{\sigma}_2 \\ < u_R + \sqrt{gh_R} < \dot{\sigma}_2 < u_* + \sqrt{gh_*} \end{cases}$$

Les conditions de Rankine-Hugoniot donnent :

$$j_2 = h_* (u_* - \dot{\sigma}_2) = h_R (u_R - \dot{\sigma}_2) = \frac{g}{2} \frac{h_* - h_R}{u_R - u_*}$$

Donc à partir des conditions de Lax et en utilisant la première condition de Rankine Hugonio, on obtient :

$$\begin{cases} u_* - \sqrt{gh_*} < \dot{\sigma}_2 \\ u_R + \sqrt{gh_R} < \dot{\sigma}_2 \\ u_* + \sqrt{gh_*} > \dot{\sigma}_2 \end{cases} \rightarrow \begin{cases} h_* \sqrt{gh_*} > j_2 \\ -h_R \sqrt{gh_R} > j_2 \\ -h_* \sqrt{gh_*} > j_2 \end{cases}$$

Soit

$$h_* > h_R$$

et comme $j_2 < 0$, on a

$$u_* > u_L$$

et en utilisant l'expression de $\dot{\sigma}_2$, on trouve la vitesse

$$u_* = u_R + (h_* - h_R) \sqrt{\frac{g}{2} \frac{h_R + h_*}{h_R h_*}} \tag{1.13}$$

1.2.5.4 Construction de la solution

Pour déterminer la solution du problème de Riemann, on doit alors déterminer l'état $\overrightarrow{w_{p_*}}$ en fonction des états $\overrightarrow{w_{p_L}}$ et $\overrightarrow{w_{p_R}}$. On relie l'état intermédiaire à l'état gauche par une 1-onde avec Equation 1.12 et Equation 1.10 :

$$u_* = v^L(h_*) = \begin{cases} u_L - \left(\sqrt{h_*} - \sqrt{h_L}\right) \dfrac{2\sqrt{g}}{\sqrt{h_L} + \sqrt{h_*}} \text{ si } h_* \leq h_L \text{ (1-détente)} \\ u_L - \left(\sqrt{h_*} - \sqrt{h_L}\right) \sqrt{\dfrac{g}{2} \dfrac{h_* + h_L}{h_* h_L}} \text{ si } h_* > h_L \text{ (1-choc)} \end{cases}$$

et on relie l'état intermédiaire à l'état droit par une 2-onde avec Equation 1.11 et Equation 1.13 :

$$u_* = v^R(h_*) = \begin{cases} u_R + \left(\sqrt{h_*} - \sqrt{h_R}\right) \dfrac{2\sqrt{g}}{\sqrt{h_R} + \sqrt{h_*}} \text{ si } h_* \leq h_R \text{ (2-détente)} \\ u_R + \left(\sqrt{h_*} - \sqrt{h_R}\right) \sqrt{\dfrac{g}{2} \dfrac{h_* + h_R}{h_* h_R}} \text{ si } h_* > h_R \text{ (2-choc)} \end{cases}$$

Donc pour trouver h_*, on doit résoudre l'équation $v^L(h_*) - v^R(h_*) = 0$ par un schéma de Newton-Raphson.

En introduisant la fonction $\zeta(h,\psi) = \begin{cases} \dfrac{2\sqrt{g}}{\sqrt{h}+\sqrt{\psi}} \text{ si } h \leq \psi \\ \sqrt{\dfrac{g}{2}\dfrac{h+\psi}{h\psi}} \text{ si } h > \psi \end{cases}$, on notera simplement

$$u_* = u_R + \left(\sqrt{h_*} - \sqrt{h_R}\right)\zeta(h_*, h_R)$$

Les calculs précédents nous permettent de définir les vitesses des ondes de choc $\dot{\sigma}_1 = u_L - h_*\sqrt{\dfrac{g}{2}\dfrac{h_L+h_*}{h_L h_*}}$ et $\dot{\sigma}_2 = u_R + h_*\sqrt{\dfrac{g}{2}\dfrac{h_R+h_*}{h_R h_*}}$.

et les vitesses des ondes de raréfaction $\lambda_1 = u_L - \sqrt{gh_L}$, $\lambda_1^* = u_* - \sqrt{gh_*}$, $\lambda_2^* = u_* + \sqrt{gh_*}$ et $\lambda_2 = u_R + \sqrt{gh_R}$.

La solution dans une zone de détente est donnée par:

$$\overrightarrow{w_p}_{1det}(x,t) = \left\{ \begin{array}{l} \dfrac{1}{9g}\left(u_L + 2\sqrt{gh_L} - \dfrac{x}{t}\right)^2 \\ \dfrac{1}{3}\left(u_L + 2\sqrt{gh_L} + 2\dfrac{x}{t}\right) \end{array} \right\}$$

et

$$\overrightarrow{w_p}_{2det}(x,t) = \left\{ \begin{array}{l} \dfrac{1}{9g}\left(-u_R + 2\sqrt{gh_R} + \dfrac{x}{t}\right)^2 \\ \dfrac{1}{3}\left(u_R - 2\sqrt{gh_R} + 2\dfrac{x}{t}\right) \end{array} \right\}.$$

Enfin nous pouvons écrire la solution complète du problème de Riemann.

Cas 1: $h_* > h_L$ et $h_* > h_R$ la solution est composée d'un 1-choc et d'un 2-choc :

$$\overrightarrow{w_p}(x,t) = \left\{ \begin{array}{ll} \overrightarrow{w_p}_L & \text{si } x < \dot{\sigma}_1 t \\ \overrightarrow{w_p}_* & \text{si } \dot{\sigma}_1 t < x < \dot{\sigma}_2 t \\ \overrightarrow{w_p}_R & \text{si } x > \dot{\sigma}_2 t \end{array} \right.$$

Cas 2: $h_L < h_* < h_R$ la solution est composée d'un 1-choc et d'une 2-détente :

$$\overrightarrow{w_p}(x,t) = \left\{ \begin{array}{ll} \overrightarrow{w_p}_L & \text{si } x < \dot{\sigma}_1 t \\ \overrightarrow{w_p}_* & \text{si } \dot{\sigma}_1 t < x < \lambda_2^* t \\ \overrightarrow{w_p}_{2det}(x,t) & \text{si } \lambda_2^* t < x < \lambda_2^R t \\ \overrightarrow{w_p}_R & \text{si } x > \lambda_2^R t \end{array} \right.$$

Cas 3: $h_R < h_* < h_L$ la solution est composée d'une 1-détente et d'un 2-choc :

$$\overrightarrow{w_p}(x,t) = \begin{cases} \overrightarrow{w_p}_L & \text{si } x < \lambda_1^L t \\ \overrightarrow{w_p}_{1det}(x,t) & \text{si } \lambda_1^L t < x < \lambda_1^* t \\ \overrightarrow{w_p}_* & \text{si } \lambda_1^* t < x < \dot{\sigma}_2 t \\ \overrightarrow{w_p}_R & \text{si } x > \dot{\sigma}_2 t \end{cases}$$

Cas 4: $h_* < h_L$ et $h_* < h_R$ la solution est composée d'une 1-détente et d'une 1-détente :

$$\overrightarrow{w_p}(x,t) = \begin{cases} \overrightarrow{w_p}_L & \text{si } x < \lambda_1^L t \\ \overrightarrow{w_p}_{1det}(x,t) & \text{si } \lambda_1^L t < x < \lambda_1^* t \\ \overrightarrow{w_p}_* & \text{si } \lambda_1^* t < x < \lambda_2^* t \\ \overrightarrow{w_p}_{2det}(x,t) & \text{si } \lambda_2^* t < x < \lambda_2^R t \\ \overrightarrow{w_p}_R & \text{si } x > \lambda_2^R t \end{cases}$$

> **Note**
>
> Dans le cas où il apparaît des zones sèches l'algorithme est adapté comme explicité par exemple dans ce cours de Gloria Faccanoni ou la thèse de Pons (2018).

> **Note**
>
> Ce solveur de Riemann est utilisé dans **CERF**. Il est programmé en Fortran90 dans *~/sources/PHY/svriemann.F90*.

> **Important 2**
>
> Dans la pratique, il est nécessaire de définir une valeur seuil définissant la zone "sèche". Dans **CERF**, on définit donc h_{crit} tel que si $h < h_{crit}$, alors on considère que la zone est sèche et on impose $h = 0$ et $\vec{u} = \vec{0}$. Cette valeur est modifiable par l'utilisateur.

1.3 Modèle d'écoulement bifluide

1.3.1 Équations constitutives

Ce modèle a été développé à l'origine pour simuler des écoulements en hydrodynamique maritime, propagation et déferlement de vagues. Pour le déferlement l'approche physiquement la plus pertinente est de considérer les deux fluides air et eau avec un comportement régi par un modèle Navier-Stokes incompressible avec ou sans turbulence. La prise en compte de

l'interface est par ailleurs un sujet en soi ! Quelle que soit l'approche de discrétisation utilisée, il s'agit de simulations très coûteuses en temps de calcul. De plus, du fait de dissipation numérique, il est difficile de simuler des écoulements à très grandes échelles. C'est pourquoi, pour simuler des phénomènes de propagation de vagues, on préférera souvent des modèles de type Saint-Venant, Serre-Green-Nagdhi ou écoulement irrotationnel non visqueux. Ces modèles, par contre, modélisent mal, ou pas, les phénomènes de déferlement. La littérature sur ces sujets est très vaste et nous ne prétendons pas faire un état de l'art ici. Nous nous contenterons de proposer une étude comparative partielle dans Helluy, Philippe et al. (2005).
Le modèle bifluide présenté ici est issu de la volonté de trouver un compromis entre la pertinence des modèles de type Navier-Stokes et la simplicité des modèles de type Saint-Venant. On considère donc un **mélange** de deux fluides, si bien qu'aucun "suivi" d'interface n'est nécessaire et aucune tension de surface ne peut être prise en compte. On fait de plus l'hypothèse, qui se vérifie dans la plupart des cas étudiés en hydrodynamique, que la viscosité peut être négligée, ce qui d'un point de vue numérique nous affranchit avantageusement de la modélisation de l'opérateur Laplacien. Enfin, comme dans Chorin (1967), on considère que les deux fluides sont très faiblement compressibles. Ainsi, on s'affranchit de la nécessité d'imposer l'incompressibilité, ce qui est coûteux en temps de calcul, par contre il devient nécessaire d'adjoindre une équation d'état pour le mélange. Après étude, d'après Frédéric Golay and Helluy (2007), il semble qu'une équation d'état isotherme soit suffisante pour la plupart des cas étudiés, si bien que l'on peut également s'affranchir de la résolution de l'équation sur l'énergie. On obtient ainsi un modèle Eulérien de la forme :

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \tag{1.14}$$

$$\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot \left(\rho \vec{u} \otimes \vec{u} + p \bar{\bar{I}} \right) = \rho \vec{g} \tag{1.15}$$

où ρ est la masse volumique du mélange, $\vec{u} =< u, v, w >^T$ est la vitesse du mélange, $\vec{g}$ est le vecteur de gravité et où l'on note $\bar{\bar{I}}$ la matrice identité, $\nabla\cdot$ l'opérateur divergence et $\otimes$ le produit tensoriel. Equation 1.14 représente la conservation de la masse et Equation 1.15 la conservation de la quantité de mouvement. p est la pression du mélange qui est déterminée par l'équation d'état :

$$p = p_0 + c_0^2 \left(\rho - (\phi \rho_a + (1 - \phi) \rho_w) \right) \tag{1.16}$$

où ρ_a représente la densité de l'air, ρ_w la densité de l'eau, p_0 la pression de référence et c_0 la vitesse du son dans le mélange. ϕ représente la fraction volumique de l'air dans le mélange. ϕ satisfait une équation de transport, que l'on prendra sous la forme non conservative (Frédéric Golay and Helluy (2007)) de la forme :

$$\frac{\partial \phi}{\partial t} + \vec{u} \cdot \nabla \phi = 0 \tag{1.17}$$

Usuellement, on prendra $p_0 = 10^5 Pa$, $\rho_a = 1.kg/m^3$, $\rho_w = 1000.kg/m^3$ et $c_0 = 20.m/s$.
En hydrodynamique, les écoulements sont de l'ordre de quelques m/s, si bien qu'une vitesse du son de l'ordre de $20m/s$ conduit à un nombre de Mach $M = \dfrac{||\vec{u}||}{c_0} < 0.3$. On peut alors considérer que le mélange est quasiment incompressible. Cette vitesse du son est complètement artificielle et ne correspond à aucune réalité physique. Elle est choisie pour des raisons numériques. En effet, le choix de c_0 va influencer la stabilité du schéma numérique explicite utilisé pour résoudre les équations par la condition de CFL (Important 1).

Cependant, de façon fortuite, d'après l'équation de Wood (Figure 1.9), alors que la vitesse du son dans l'eau est de l'ordre de 1481 m/s et dans l'air de l'ordre de 343 m/s, la vitesse du son dans un mélange air-eau est de l'ordre de 23 m/s. Il ne s'agit nullement d'une justification du modèle utilisé ici, mais il est intéressant de noter que la vitesse du son dans un mélange air-eau est de l'ordre de la vitesse du son artificielle utilisée dans le modèle bifluide.

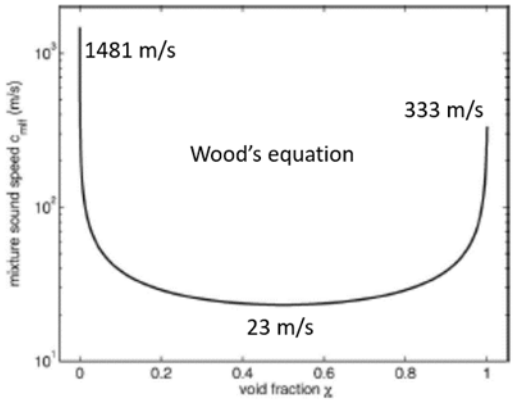


Figure 1.9: Vitesse du son dans un mélange air-eau d'après l'équation de Wood

Les équations Equation 1.14, Equation 1.15 et Equation 1.17, complétées par Equation 1.16 constituent le problème hyperbolique résolu par **CERF** pour modéliser les écoulements bifluides. On définit les variables conservatives $\overrightarrow{w} =< \rho, \rho u, \rho v, \rho w, \phi >^T$ et les variables primitives $\overrightarrow{w_p} =< \rho, u, v, w, p >^T$. On retrouve la forme générale Equation 1.1.

1.3.2 Critère de raffinement

Dans le cas du modèle bifluide, pour définir la production numérique d'entropie Equation 1.4, on utilise l'entropie définie par:

$$s(\overrightarrow{w}) = \frac{1}{2}\rho||\vec{u}||^2 + c_0^2 \rho ln(\rho) - c_0^2 \left(\rho_w - \rho_a\right)\phi$$

et le flux d'entropie

$$\vec{\psi}_s(\overrightarrow{w}) = \left(s + c_0^2\rho + c_0^2\left(\rho_w - \rho_a\right)\right)\vec{u} = \left(\frac{1}{2}\rho||\vec{u}||^2 + c_0^2\rho\left(ln(\rho)+1\right)\right)\vec{u}.$$

1.3.3 Raidissement de la surface libre

Le schéma de résolution par volumes finis du système hyperbolique Equation 1.14, Equation 1.15 et Equation 1.17, même à l'ordre 2, est connu pour être dissipatif. En conséquence l'interface air-eau diffuse et s'élargit. Pour contrer ce phénomène, on peut utiliser un modèle de raidissement de l'interface. L'idée est de rajouter un terme de source à l'équation de transport de ϕ Equation 1.17 qui va pénaliser les zones de mélange avec une méthode de splitting. Sur un pas de temps fictif on calcule donc :

$$\frac{\partial \phi}{\partial \tau} = \phi^2(1-\phi)^2(\phi - c)$$

où c est déterminé de façon à conserver la fraction volumique, c'est à dire:

$$c = \frac{\int_\Omega \phi^3(1-\phi)^2 d\Omega}{\int_\Omega \phi^2(1-\phi)^2 d\Omega}$$

On peut remarquer que pour $\phi = 0$ ou $\phi = 1$, ce terme source est sans effet. Enfin, comme ϕ est modifié, par soucis de conservation on modifie également la masse volumique et la quantité de mouvement du mélange :

$$\frac{\partial \rho}{\partial \tau} = \phi^2(1-\phi)^2(\phi - c)(\rho_a - \rho_w)$$
$$\frac{\partial (\rho \vec{u})}{\partial \tau} = \phi^2(1-\phi)^2(\phi - c)\vec{u}(\rho_a - \rho_w)$$

Le pas de temps fictif est calibré pour représenter une part du flux. Cette part est déterminée par un coefficient de raidissement α_{sharp}. Dans la pratique, ce coefficient est de l'ordre de $10^{-3} - 10^{-2}$.

1.3.4 Notion de *"fluide rigide"*

En suivant le principe des domaines fictifs, on peut pénaliser une partie des cellules pour simuler un obstacle en imposant une vitesse nulle. En suivant les travaux de Coquerelle and Cottet (2008) on peut également pénaliser une partie des cellules pour obtenir un comportement de type "solide rigide". L'extension des travaux de Coquerelle and Cottet (2008) à un modèle bifluide a été réalisé par Altazin (2017). Considérons un "solide" occupant le domaine Ω_s, de densité uniforme ρ_s, de centre de gravité G, de masse M_s, de matrice d'inertie $\overline{\overline{J_s}}$. La vitesse de tout point P du solide est donnée par :

$$\vec{v}_s = \vec{v}_G + \vec{\omega}_G \wedge \overrightarrow{GP}$$

où $\vec{v}_G$ est la vitesse du centre de gravité, $\vec{\omega}_G$ est la vitesse angulaire du solide et $\overrightarrow{GP}$ est le

vecteur reliant le centre de gravité au point P. En minimisant $\int_{\Omega_s} ||\rho\vec{u} - \rho_s\vec{v}_s||^2 d\Omega_s$, on obtient pour une densité uniforme du solide :

$$\vec{v}_G = \frac{\int_{\Omega_s} \rho\vec{u} d\Omega_s}{M_s}$$

et

$$\vec{\omega}_G = \overline{\overline{J_s}}^{-1} \int_{\Omega_s} \overrightarrow{GP} \wedge \vec{u}.$$

À l'issue de chaque pas de temps, on calcule la position et la vitesse du centre de gravité du solide et la vitesse de rotation du solide. On affecte ensuite ρ_s et $\vec{v}_s$ à toutes les cellules du "solide". Dans **CERF** le solide est déterminé par son enveloppe. Cette enveloppe est définie par un maillage triangulaire. Les cellules à l'intérieur de l'enveloppe sont déterminées par une technique de "ray casting". Dans **CERF** on peut considérer un solide "fixe", à "vitesse imposée" ou "libre".

1.3.5 Solveur de Riemann pour le modèle bifluide

1.3.5.1 Généralités

Les flux numériques dans **CERF** sont calculés à l'aide d'un solveur de Riemann. Il existe de très nombreuses références concernant ce solveur comme par exemple Toro (1997). Ce chapitre décrit le solveur de Riemann utilisé pour le modèle d'écoulement bifluide en s'appuyant sur Frédéric Golay and Helluy (2007).

On cherche donc à résoudre le problème bifluide isotherme en une dimension. On considère un écoulement de densité $\rho(x,t)$, de vitesse horizontale $u(x,t)$, de pression $p(x,t)$, de fraction volumique $\phi(x,t)$ décrit par le problème suivant :

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial(\rho u)}{\partial x} = 0 \\ \dfrac{\partial(\rho u)}{\partial t} + \dfrac{\partial(\rho u^2 + p)}{\partial x} = 0 \\ \dfrac{\partial \rho\phi}{\partial t} + \dfrac{\partial \rho\phi u}{\partial x} = 0 \end{cases} \tag{1.18}$$

Avec une équation d'état de la forme :

$$p = p_0 + c_0^2 \left(\rho - (\phi\rho_a + (1-\phi)\rho_w)\right)$$

et les conditions initiales telles que :

$$\overrightarrow{w_p}(x,0) = \begin{cases} \overrightarrow{w_p}_L & \text{si } x < 0 \\ \overrightarrow{w_p}_R & \text{si } x > 0 \end{cases}$$

où $\overrightarrow{w_p}_L =< \rho_L, u_L, \phi_L >^T$ et $\overrightarrow{w_p}_R =< \rho_R, u_R, \phi_R >^T$ sont deux états constants.

En développant l'Equation 1.18, on peut réécrire le système sous la forme suivante :

$$\frac{\partial}{\partial t}\begin{Bmatrix} \rho \\ u \\ \phi \end{Bmatrix} + \begin{bmatrix} u & \rho & 0 \\ \frac{c_0^2}{\rho} & u & \frac{c_0^2(\rho_w - \rho_a)}{\rho} \\ 0 & 0 & u \end{bmatrix} \frac{\partial}{\partial x}\begin{Bmatrix} \rho \\ u \\ \phi \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix}.$$

On cherche alors les valeurs propres λ_1, λ_2 et λ_3 ainsi que les vecteurs propres $\vec{r}_1$, $\vec{r}_2$ et $\vec{r}_3$ de la matrice ainsi mise en évidence. Un calcul rapide nous donne:

$$\lambda_1 = u - c_0, \lambda_2 = u, \lambda_3 = u + c_0$$

avec $\lambda_1 \leq \lambda_2 \leq \lambda_3$ et

$$\vec{r}_1 = \begin{Bmatrix} -\rho/c_0 \\ 1 \\ 0 \end{Bmatrix}, \vec{r}_2 = \begin{Bmatrix} (\rho_w - \rho_a) \\ 0 \\ -1 \end{Bmatrix}, \vec{r}_3 = \begin{Bmatrix} \rho/c_0 \\ 1 \\ 0 \end{Bmatrix}.$$

> **Note 2**
>
> Les valeurs propres permettent de déterminer la vitesse de propagation des ondes dans la cellule Important 1 comme : $\lambda_k = Max\left(|u - c_0|, |u|, |u + c_0|\right)$

Pour déterminer la nature des trois champs caractéristiques, on calcule alors $\nabla_{w_p}\lambda_1 \cdot \vec{r}_1 = 1 \neq 0$, $\nabla_{w_p}\lambda_2 \cdot \vec{r}_2 = 0$ et $\nabla_{w_p}\lambda_3 \cdot \vec{r}_3 = 1 \neq 0$. Les champs caractéristiques 1 et 3 sont donc **vraiment non linéaires** alors que le champs 2 est **linéairement dégénéré**. Il y aura donc une de discontinuité de contact et deux ondes qui sont soit des chocs soit des ondes de raréfaction.

Comme il y a 3 ondes, on peut définir pour chaque onde 2 invariants de Riemann. Les invariants de Riemann de l'onde k sont tels que $\nabla_{w_p} I_{k;1} \cdot \vec{r}_k = 0$ et $\nabla_{w_p} I_{k;2} \cdot \vec{r}_k = 0$. On peut aisément vérifier que :
$I_{1;1} = \phi$, $I_{1;2} = u + c_0 ln(\rho)$,
$I_{2;1} = u$, $I_{2;2} = p$,
$I_{3;1} = \phi$, $I_{3;2} = u - c_0 ln(\rho)$.

La solution faible entropique du problème de Riemann est constituée d'au plus quatre états constants $\overrightarrow{w_p}_L$, $\overrightarrow{w_p}_I$, $\overrightarrow{w_p}_{II}$, $\overrightarrow{w_p}_R$ séparés par un choc ou une détente, une discontinuité de contact, puis un choc ou une détente (Figure 1.10). Par la suite, nous allons déterminer Les états $\overrightarrow{w_p}_I$ et $\overrightarrow{w_p}_{II}$.

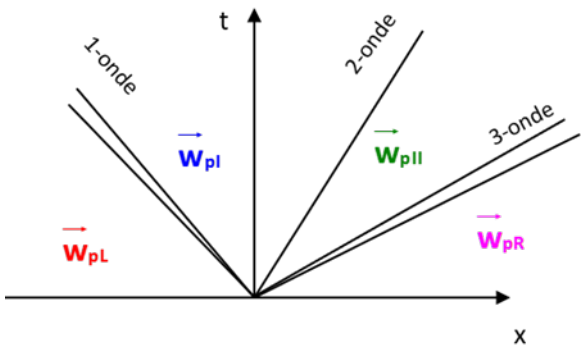


Figure 1.10: Schéma des solutions du problème de Riemann avec le modèle bi-fluide isotherme

1.3.5.2 Étude des détentes

Considérons un état $\overrightarrow{w_{p_g}}$ et un état $\overrightarrow{w_{p_d}}$, reliés par une onde de raréfaction k. On doit alors satisfaire les conditions suivantes:

- $\lambda_k(\overrightarrow{w_{p_g}}) < \lambda_k(\overrightarrow{w_{p_d}})$
- Les invariants de Riemann sont conservés le long de l'onde de raréfaction $I_{k;i}(\overrightarrow{w_{p_g}}) = I_{k;i}(\overrightarrow{w_{p_d}})$ pour $i = 1, 2$

Appliquons ces conditions au problème bifluide isotherme.

Considérons en premier lieu une 1-détente reliant un état $\overrightarrow{w_{p_L}}$ à un état $\overrightarrow{w_{p_I}}$, séparés par une onde de raréfaction. Les invariants de Riemann donnent :

$$\begin{cases} I_{1;1}(\overrightarrow{w_{p_L}}) = I_{1;1}(\overrightarrow{w_{p_I}}) \\ I_{1;2}(\overrightarrow{w_{p_L}}) = I_{1;2}(\overrightarrow{w_{p_I}}) \end{cases} \rightarrow \begin{cases} \phi_L = \phi_I \\ u_L + c_0 ln(\rho_L) = u_I + c_0 ln(\rho_I) \end{cases}$$

soit

$$u_I = u_L + c_0 ln\left(\frac{\rho_L}{\rho_I}\right) \tag{1.19}$$

De plus,

$$\lambda_1(\overrightarrow{w_{p_L}}) < \lambda_1(\overrightarrow{w_{p_I}}) \rightarrow u_L < u_I$$

et en conséquence d'après Equation 1.19, $\rho_L > \rho_I$.

Considérons enfin une 3-détente reliant un état $\overrightarrow{w_{p_{II}}}$ à un état $\overrightarrow{w_{p_R}}$, séparés par une onde de raréfaction. La condition sur les vitesses s'écrit :

$$\lambda_3(\overrightarrow{w_{p_{II}}}) < \lambda_3(\overrightarrow{w_{p_R}}) \rightarrow u_{II} < u_R.$$

Les invariants de Riemann donnent

$$\left\{ \begin{array}{l} I_{3;1}(\overrightarrow{w_p}_{II}) = I_{3;1}(\overrightarrow{w_p}_R) \\ I_{3;2}(\overrightarrow{w_p}_{II}) = I_{3;2}(\overrightarrow{w_p}_R) \end{array} \right. \rightarrow \left\{ \begin{array}{l} \phi_{II} = \phi_R \\ u_{II} - c_0 ln(\rho_{II}) = u_R - c_0 ln(\rho_R) \end{array} \right.$$

soit

$$u_{II} = u_R - c_0 ln\left(\frac{\rho_R}{\rho_{II}}\right) \tag{1.20}$$

donc comme $u_{II} < u_R$, on a $\rho_{II} < \rho_R$.

1.3.5.3 Étude des chocs

Considérons un état $\overrightarrow{w_p}_g$ et un état $\overrightarrow{w_p}_d$, reliés par une onde de choc k. On doit alors satisfaire les conditions suivantes :

- Les conditions de Rankine-Hugoniot où l'on note $\dot{\sigma}_k$ la vitesse du choc : $\dot{\sigma}_k = \dfrac{\left[\vec{F}(\vec{w})\right]}{[\vec{w}]}$
- Les condition entropiques de Lax pour un champs k vraiment non linéaire :

$$\left\{ \begin{array}{l} \lambda_{k-1}(\overrightarrow{w_p}_g) < \dot{\sigma}_k < \lambda_k(\overrightarrow{w_p}_g) \\ \lambda_k(\overrightarrow{w_p}_d) < \dot{\sigma}_k < \lambda_{k+1}(\overrightarrow{w_p}_d) \end{array} \right.$$

Appliquées au problème bifluide, les conditions de Rankine-Hugoniot donnent :

$$\dot{\sigma}_k = \frac{\rho_d u_d - \rho_g u_g}{\rho_d - \rho_g} = \frac{\rho_d u_d^2 + p_d - \rho_g u_g^2 - p_g}{\rho_d u_d - \rho_g u_g} = \frac{\rho_d u_d \phi_d - \rho_g u_g \phi_g}{\rho_d \phi_d - \rho_g \phi_g}$$

$$\left\{ \begin{array}{l} \rho_d \dot{\sigma}_k - \rho_g \dot{\sigma}_k = \rho_d u_d - \rho_g u_g \\ \rho_d u_d \dot{\sigma}_k - \rho_g u_g \dot{\sigma}_k = \rho_d u_d^2 + p_d - \rho_g u_g^2 - p_g \\ \rho_d \phi_d \dot{\sigma}_k - \rho_g \phi_g \dot{\sigma}_k = \rho_d u_d \phi_d - \rho_g u_g \phi_g \end{array} \right.$$

$$\left\{ \begin{array}{l} \rho_g(u_g - \dot{\sigma}_k) = \rho_d(u_d - \dot{\sigma}_k) \\ u_d \rho_d(u_d - \dot{\sigma}_k) - u_g \rho_g(u_g - \dot{\sigma}_k) + p_d - p_g = 0 \\ \phi_g \rho_g(u_g - \dot{\sigma}_k) = \phi_d \rho_d(u_d - \dot{\sigma}_k) \end{array} \right.$$

Soit, si on note $j_k = \rho_g(u_g - \dot{\sigma}_k) = \rho_d(u_d - \dot{\sigma}_k)$, on obtient :

$$\phi_g = \phi_d$$

et

$$j_k = \frac{p_d - p_g}{u_g - u_d}$$

Par élimination de la vitesse du choc, on a également :

$$\left(\rho_d u_d - \rho_g u_g\right)^2 = \left(\rho_d - \rho_g\right)\left(\rho_d u_d^2 - \rho_g u_g^2 + p_d - p_g\right)$$

ou encore

$$\rho_d \rho_g \left(u_d - u_g\right)^2 = \left(\rho_d - \rho_g\right)\left(p_d - p_g\right)$$

Considérons en premier lieu un 1-choc reliant un état $\overrightarrow{w_{p}}_L$ à un état $\overrightarrow{w_{p}}_I$, séparés par une onde de choc. Les conditions d'admissibilité de Lax s'écrivent :

$$\left\{\begin{array}{l} \dot{\sigma}_1 < \lambda_1(\overrightarrow{w_{p}}_L) \\ \lambda_1(\overrightarrow{w_{p}}_I) < \dot{\sigma}_1 < \lambda_2(\overrightarrow{w_{p}}_I) \end{array}\right. \rightarrow \left\{\begin{array}{l} \dot{\sigma}_1 < u_L - c_0 \\ u_I - c_0 < \dot{\sigma}_1 \\ \dot{\sigma}_1 < u_I \end{array}\right. \rightarrow \left\{\begin{array}{l} \rho_L c_0 < j_1 \\ j_1 < \rho_I c_0 \\ \dot{\sigma}_1 < u_I \end{array}\right.$$

Donc en particulier $u_I < u_L$ et $\rho_L < \rho_I$. Les conditions de Rankine-Hugoniot donnent :

$$\left\{\begin{array}{l} \phi_L = \phi_I \\ \left(u_I - u_L\right)^2 = \dfrac{\rho_I - \rho_L}{\rho_I \rho_L}\left(p_I - p_L\right) \end{array}\right.$$

d'où, comme $u_I < u_L$ et $\rho_L < \rho_I$:

$$u_I = u_L - \sqrt{\frac{\rho_I - \rho_L}{\rho_I \rho_L}\left(p_I - p_L\right)} = u_L + c_0 \frac{\rho_L - \rho_I}{\sqrt{\rho_I \rho_L}} \tag{1.21}$$

Considérons enfin un 3-choc reliant un état $\overrightarrow{w_{p}}_{II}$ à un état $\overrightarrow{w_{p}}_R$, séparés par une onde de choc. Les conditions de Lax donnent :

$$\left\{\begin{array}{l} \lambda_2(\overrightarrow{w_{p}}_{II}) < \dot{\sigma}_3 < \lambda_3(\overrightarrow{w_{p}}_{II}) \\ \lambda_3(\overrightarrow{w_{p}}_R) < \dot{\sigma}_3 \end{array}\right. \rightarrow \left\{\begin{array}{l} u_{II} < \dot{\sigma}_3 \\ \dot{\sigma}_3 < u_{II} + c_0 \\ u_R + c_0 < \dot{\sigma}_3 \end{array}\right. \rightarrow \left\{\begin{array}{l} u_{II} < \dot{\sigma}_3 \\ -c_0 \rho_{II} < j_3 \\ j_3 < -c_0 \rho_R \end{array}\right.$$

Donc en particulier $u_R < u_{II}$ et $\rho_R < \rho_{II}$.Les conditions de Rankine-Hugoniot s'écrivent :

$$\left\{\begin{array}{l} \phi_{II} = \phi_R \\ \left(u_R - u_{II}\right)^2 = \dfrac{\rho_R - \rho_{II}}{\rho_R \rho_{II}}\left(p_R - p_{II}\right) \end{array}\right.$$

d'où , comme $u_R < u_{II}$ et $\rho_R < \rho_{II}$:

$$u_{II} = u_R + \sqrt{\frac{\rho_R - \rho_{II}}{\rho_R \rho_{II}}\left(p_R - p_{II}\right)} = u_R - c_0 \frac{\rho_R - \rho_{II}}{\sqrt{\rho_R \rho_{II}}} \tag{1.22}$$

1.3.5.4 Étude de la discontinuité de contact

Considérons un état $\overrightarrow{w_{p_g}}$ et un état $\overrightarrow{w_{p_d}}$, reliés par une discontinuité de contact. Au travers d'une k-discontinuité de contact l'entropie et les invariants de Riemann sont conservés. La discontinuité se déplaçant à la vitesse $\dot{\sigma}_k$ on a :

$$\dot{\sigma}_k = \lambda_k(\overrightarrow{w_{p_g}}) = \lambda_k(\overrightarrow{w_{p_d}}) = \frac{\left[\vec{F}(\vec{w})\right]}{[\vec{w}]}$$

Considérons donc une 2-discontinuité de contact reliant l'état $\overrightarrow{w_{p_I}}$ à l'état $\overrightarrow{w_{p_{II}}}$. On a alors

$$\dot{\sigma}_2 = u_I = u_{II} = \frac{\rho_{II}u_{II} - \rho_I u_I}{\rho_{II} - \rho_I} = \frac{\rho_{II}u_{II}^2 + p_{II} - \rho_I u_I^2 - p_I}{\rho_{II}u_{II} - \rho_I u_I} = \frac{\rho_{II}u_{II}\phi_{II} - \rho_I u_I \phi_I}{\rho_{II}\phi_{II} - \rho_I \phi_I}$$

D'où l'on déduit :

$$u_I = u_{II} \text{ et } p_I = p_{II} \tag{1.23}$$

1.3.5.5 Construction de la solution

Pour simplifier les notations, définissons une fonction $H(a,b)$ telle que $H(a,b) = \begin{cases} c_0 \dfrac{\rho_a - \rho_b}{\sqrt{\rho_a \rho_b}} & \text{si } \rho_a < \rho_b \\ c_0 ln\left(\dfrac{\rho_a}{\rho_b}\right) & \text{si } \rho_a > \rho_b \end{cases}$.

On peut alors écrire pour la première onde, en utilisant Equation 1.19, Equation 1.21 $u_I = u_L + H(\rho_L, \rho_I)$ et pour la troisième onde, en utilisant Equation 1.20, Equation 1.22 $u_{II} = u_R - H(\rho_R, \rho_{II})$.

On sait par l'étude de la discontinuité de contact que :

$$u_I - u_{II} = 0$$

donc

$$u_L + H(\rho_L, \rho_I) - u_R + H(\rho_R, \rho_{II}) = 0$$

Or, $\rho_I = \rho_I(p_I, \phi_I)$ et $\rho_{II} = \rho_{II}(p_{II}, \phi_{II})$. On sait par ailleurs que $p_I = p_{II}$, nous noterons alors la pression intermédiaire p^* telle que $p^* = p_I = p_{II}$. De plus on sait $\phi_I = \phi_L$ et $\phi_{II} = \phi_R$. On peut alors écrire :

$$u_L - u_R + H(\rho_L, \rho(p^*, \phi_L)) + H(\rho_R, \rho(p^*, \phi_R)) = 0$$

C'est donc une équation non-linéaire en p^* que l'on peut résoudre par une méthode de Newton-Raphson. Connaissant p^*, on peut alors déterminer les états $\overrightarrow{w_{p_I}}$ et $\overrightarrow{w_{p_{II}}}$.

> **Note**
>
> Ce solveur de Riemann est utilisé dans **CERF**. Il est programmé en Fortran90 dans *~/sources/PHY/riemisot.F90*.

2 Guide de l'utilisateur

Ce chapitre décrit l'installation du logiciel **CERF**, ainsi que son utilisation.

2.1 Installation du logiciel

2.1.1 Prérequis

CERF est un logiciel écrit en **Fortran 90** utilisant la bibliothèque **MPI** pour la communication entre les différents processus.

> ❗ Important
>
> Il est donc nécessaire d'avoir :
> - un système d'exploitation ou émulateur **Linux** (Ubuntu, CentOS, WSL sous Windows 10 ou 11, Mac OS, ….)
> - un compilateur **Fortran 90** (gfortran, ifort, …)
> - la bibliothèque **MPI** (OpenMPI)

Les fichiers de sortie de **CERF** sont au format **Tecplot** (**VTK** à venir). Il est donc recommandé d'avoir un logiciel de visualisation tel que **ParaView** ou **Visit**.

2.1.2 Installation

Pour installer **CERF**, après avoir téléchargé l'archive 'cerf_v1.0.tar', il suffit de la décompresser dans un répertoire de votre choix.

```
tar xvf cerf_v1.0.tar .
```

L'archive déployée présentera alors une arborescence comme illustré sur Figure 2.1.

Le répertoire principal de **CERF** comporte 4 sous-répertoires : *doc*, *sources*, *lib* et *exec*.

- le répertoire *doc* contient la documentation du logiciel

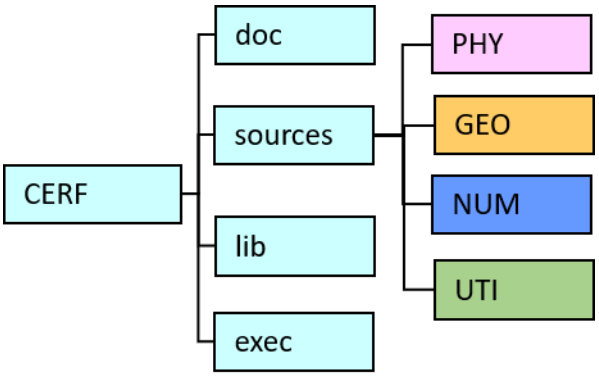


Figure 2.1: Arborescence de **CERF**

- le répertoire *sources* contient les sources du logiciel
- le répertoire *lib* contient les bibliothèques créées à la compilation et nécessaires à la création des exécutables
- le répertoire *exec* contient tous les exécutables du logiciel. C'est le répertoire de travail

Pour installer **CERF**, il suffit de se rendre dans le répertoire *exec* et de lancer le *Makefile*.

```
cd CERF/exec
make
```

> **Note**
>
> Par défaut, le *Makefile* utilise le compilateur **gfortran** et la bibliothèque **OpenMPI**. Cependant il est possible de modifier ces paramètres dans le *Makefile*.
>
> - Paramètre **FC** = "gfortran" ou "ifort" ou ..... pour un calcul séquentiel
> - Paramètre **FC** = "mpif90" ou ..... pour un calcul parallèle
> - Paramètre **MPI** = "_MPI" ou "NONE_MPI" suivant que la librairie **MPI** est installée ou non
> - Paramètre **FFLAGS** pour redéfinir les paramètres de compilation

> **Note**
>
> Le *Makefile* est configuré pour ne recompiler que les fichiers sources nouvellement créés ou modifiés. En cas de recompilation complète, par exemple suite à un changement de machine, il est conseillé de nettoyer les fichiers objets et exécutables avant de relancer le *Makefile*, par la commande:
>
> ```
> make clean
> ```

À l'issue de l'exécution du *Makefile*, les exécutables de **CERF** sont créés dans le répertoire *exec* : ***cerf***, ***cerf_amr***, ***cerf_input*** et ***cerf2tec***.

> **Note**
>
> En fonction du système d'exploitation utilisé, les exécutables peuvent comporter l'extension **".exe"** ou non.

2.1.3 Déroulement d'un calcul

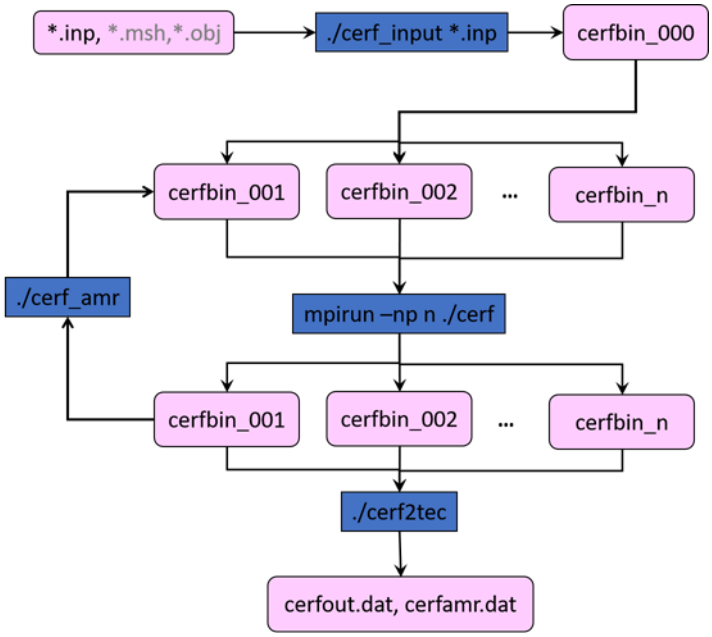


Figure 2.2: Déroulement d'un calcul de **CERF**

Comme illustré sur Figure 2.2, le déroulement d'un calcul de **CERF** se fait en plusieurs étapes. Une phase de *pré-traitement* permet de générer le maillage, les conditions initiales du calcul

et les paramètre du calcul. Cette phase est réalisée par l'exécutable ***cerf__input***. Une fois le maillage généré, le calcul est effectué par l'exécutable ***cerf*** et le maillage peut être modifié par l'exécutable ***cerf__amr***. Enfin, l'exécutable ***cerf2tec*** permet de convertir les fichiers de sortie en format **Tecplot** pour une phase de *post-traitement*.

- ***cerf__input***
  En entrée, ***cerf__input*** lit un fichier de configuration ***.inp*** dans un format spécifique à **CERF**, dans lequel sont définis les paramètres physiques et numériques du modèle à traiter, le maillage, les conditions initiales, les conditions aux limites, … etc … Dans le cas où le maillage est défini par un fichier au format **GMSH V2.2** il lira également le fichier ***.msh***. Dans le cas où des pénalisations de *fluides rigides* sont définies, il lira également les *objets* dans les fichiers ***.obj***.
  En sortie, ***cerf__input*** génère un fichier binaire ***cerfbin__000*** contenant la définition du *maillage maître* et autant de fichiers binaires ***cerfbin__**** que de *domaines* requis, contenant toutes les informations des domaines : paramètres, maillage, degrés de liberté, conditions aux limites, …. etc …

- ***cerf***
  En entrée, ***cerf*** lit les fichiers binaires ***cerfbin__**** générés par ***cerf__input*** ou modifiés par ***cerf__amr***. Puis il effectue un cacul en parallèle (via les processus **MPI**) sur les domaines définis durant la période de temps requise. Dans cette étape le maillage est fixe. En sortie, à l'issue du calcul, ***cerf*** met à jour les fichiers binaires ***cerfbin__**** avec les nouvelles valeurs des champs calculés.

- ***cerf__amr***
  En entrée, ***cerf__amr*** lit les fichiers binaires ***cerfbin__**** générés par ***cerf***. Puis il effectue un raffinement du maillage en fonction de critères définis dans le fichier de configuration ***.inp***. En sortie, ***cerf__amr*** met à jour les fichiers binaires ***cerfbin__**** avec le nouveau maillage. Par ailleurs, ***cerf__amr*** génère un fichier ***cerfamr.dat*** contenant les informations liées au *maillage maître* : critère de raffinement, niveau de raffinement, …. Dans le cas où des pénalisations de *fluides rigides* sont définies, ***cerf__amr*** génère également les fichiers ***solide__**** contenant la nouvelle position des *objets*.

- ***cerf2tec***
  En entrée, ***cerf2tec*** lit les fichiers binaires ***cerfbin__**** . Puis il convertit les fichiers binaires en un seul fichier ASCII, nommé ***cerfout.dat*** au format **Tecplot** pour une visualisation des résultats.

Lancer un calcul en utilisant que des lignes de commande peut devenir fastidieux. C'est pourquoi on utilise souvent des scripts. Voici, ci-après, un exemple de script qui permet de lancer un calcul complet de **CERF** en une seule commande. Il suffit de le modifier en fonction de vos besoins.

Exemple de script shell

Voici un exemple de script shell permettant de lancer un calcul à partir du fichier ***dam.inp***. Ce script lance le calcul sur 4 processeurs puis la modification du maillage 20 fois. Les fichiers de sortie sont renommés en fonction de l'itération afin de pouvoir visualiser l'évolution du calcul.

```
#!/bin/bash
./cerf_input dam.inp
./cerf2tec
mv cerfout.dat d_0.dat
mv cerfamr.dat d_mame_0.dat
for i in $(seq 1 20 )
do
mpirun -np 4 ./cerf
./cerf2tec
mv cerfout.dat d_$i.dat
./cerf_amr
mv cerfamr.dat d_mame_$i.dat
done
```

2.2 Le fichier de configuration *.inp

2.2.1 Généralités

Le fichier de configuration est un fichier texte permettant de définir tous les paramètres et entités nécessaires à la simulation. Il est divisé en plusieurs sections, chacune débutant par une ***entête*** de **4 caractères**. Les entêtes, à **renseigner dans cet ordre**, sont les suivants : **PHYS**, **MAME**, **BATH**, **INIT**, **COND**, **MESH**, **OBST**, **NUME**. Chaque entête est suivi sur la même ligne de 1 ou 2 entiers qui définissent le **code** et éventuellement l'**argument**. Toutes les lignes commençant par un "**!**" sont des commentaires et sont ignorées par le programme. On peut mettre des commentaires entre les sections, mais pas à l'intérieur d'une section. Chaque section est ensuite complétée par des lignes de données réelles ou entières, de mots clés, qui définissent les paramètres de la simulation.

Exemple de ligne avec entête, code et argument

```
! voici une ligne avec entête, code et argument
MAME    1    1
```

La première ligne est une ligne de commentaire. La seconde ligne est une ligne avec un entête. Elle commence par l'entête **MAME**, suivi de 2 entiers **1** et **1**qui définissent le **code** et l'**argument**.

2.2.2 Définition des paramètres physiques: PHYS

L'entête **PHYS** permet de définir les paramètres physiques du modèle. Le modèle est appliqué à tout le domaine de calcul. les modèles disponibles sont le modèle de Saint-Venant Section 1.2 et le modèle bifluide Section 1.3.

On sélectionne le modèle utilisé en renseignant obligatoirement le ***mot clé*** **MODE**, qui prend la valeur 2 pour le modèle bifluide et 1 pour le modèle de Saint-Venant.

Pour le modèle bifluide isotherme on peut au besoin définir les paramètres à valeurs réelles suivants :

- **GPES** : Orientation du champ de gravité suivant $1/x$, $2/y$, $3/z$, ou aucune 0 (valeur par défaut 2.)
- **PREF** : Pression de référence p_0 de la loi d'état Equation 1.16 (valeur par défaut 10^5)
- **CSSM** : Vitesse du son artificielle c_0 de la loi d'état Equation 1.16 (valeur par défaut 20.)
- **REFA** : Densité de l'air ρ_a de la loi d'état Equation 1.16 (valeur par défaut 1.)
- **REFW** : Densité de l'eau ρ_w de la loi d'état Equation 1.16 (valeur par défaut 1000.)
- **SHAR** : Coefficient de raidissement d'interface Section 1.3.3 (valeur par défaut 0.)

Pour le modèle de Saint-Venant on peut au besoin définir les paramètres suivants :

- **WLTR** : Valeur seuil de définition de la zone sèche (valeur par défaut 10^{-4})
- **FRMO** : Prise en compte de la friction. 0 pas de friction, 1 Manning-Strickler, 2/Darcy-Weisbach (valeur par défaut 0). La valeur du coefficient de friction qui peut varier spatialement est définie ultérieurement dans la section **BATH**.
- **BAGR** : Le gradient de la bathymétrie est soit donné (par une fonction utilisateur ou par interpolation bilinéaire) **BAGR**=0, soit calculé par la méthode de Barth **BAGR**=1 (valeur par défaut 0).

Exemple de définition des paramètres pour le modèle bifluide

```
PHYS 0
MODE 2.
SHAR 0.01
```

on définit ainsi un modèle bifluide pour un écoulement air-eau avec la gravité suivant y avec un coefficient de raidissement d'interface de 0.01.

Exemple de définition des paramètres pour le modèle de Saint-Venant

```
PHYS 0
MODE 1.
FRMO 1.
```

on définit ainsi un modèle d'écoulement en eaux peu profondes avec une friction de type Manning-Strickler.

2.2.3 Définition du maillage maître : MAME

L'entête **MAME** permet de définir le *maillage maître*. Pour rappel, ce maillage doit être conforme! **CERF** dispose d'un mailleur rudimentaire (**code** 1) et lit les fichiers ascii de maillages au format GMSH V2.2 (**code** 2).

Le maillage réalisé par **CERF** est un maillage cartésien, il faut définir sur une ligne les limites du domaine et sur une autre ligne la discrétisation dans chaque direction.

- Pour un maillage "1D" : **argument** 0
  - 1 ligne constituée de 2 réels : x_{min}, x_{max}
  - 1 ligne constituée de 1 entier : n_x
- Pour un maillage "2D" : **argument** 1
  - 1 ligne constituée de 4 réels : x_{min}, x_{max}, y_{min}, y_{max}
  - 1 ligne constituée de 2 entiers : n_x, n_y
- Pour un maillage "3D" : **argument** 2
  - 1 ligne constituée de 6 réels : x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}
  - 1 ligne constituée de 3 entiers : n_x, n_y, n_z

En 1D, les dimensions suivant y et z sont déterminées automatiquement par **CERF**, telles que $y_{min} = z_{min} = -\frac{x_{max}-x_{min}}{2n_x}$ et $y_{max} = z_{max} = \frac{x_{max}-x_{min}}{2n_x}$.

En 2D, la dimension suivant z est déterminée automatiquement par **CERF**, telle que $z_{min} = -Min\left(\frac{x_{max}-x_{min}}{2n_x}, \frac{y_{max}-y_{min}}{2n_y}\right)$ et $z_{max} = Min\left(\frac{x_{max}-x_{min}}{2n_x}, \frac{y_{max}-y_{min}}{2n_y}\right)$.

Exemple de définition d'un maillage maître via **CERF**

```
MAME    1    1
0. 1.  0.  1.
2 2
```

On définit ainsi un maillage "2D", constitué de 4 blocs de forme hexagonale, 2 suivant x, 2 suivant y et 1 suivant z. Les blocs sont de taille 0.5×0.5 suivant x et y. La dimension suivant z est déterminée automatiquement par **CERF**

Pour des maillages plus complexes, on peut utiliser un maillage généré par un mailleur externe, tel que **GMSH**. Dans ce cas, il faut définir le maillage dans un fichier au format **GMSH V2.2**. L'entête **MAME** doit donc être affectée du **code** 2 et la ligne suivante contient le nom du fichier qui doit impérativement se trouver dans le répertoire ***exec***.

Exemple de définition d'un maillage maître via **GMSH**

```
MAME    2    0
toto.msh
```

On définit ainsi un maillage "2D" à partir du fichier ***toto.msh*** au format **GMSH** V2.2.

2.2.4 Définition des conditions initiales: INIT

L'entête **INIT** permet de définir les conditions initiales du problème à traiter. On peut définir les conditions initiales pour le cas du *tube à choc* (**code** 0), par une fonction définie par l'utilisateur (**code** 1) ou par zones (**code** 2).

Pour rappel, les variables primitives du modèle de Saint-Venant sont au nombre de $n_{dof} = 3$: h, u, v.
Pour rappel, les variables primitives du modèle bifluide sont au nombre de $n_{dof} = 5$: ρ, u, v, w, p.

- Pour une initialisation "tube à choc" : **code** 0
  L'**argument** est 0. Puis:
  - 1 ligne constituée de 2 réels pour le modèle de Saint-Venant (h, u) ou 3 réels pour le modèle bifluide (ρ, u, p) à l'instant initial à *"gauche"* ($x < 0$)
  - 1 ligne constituée de 2 réels pour le modèle de Saint-Venant (h, u) ou 3 réels pour le modèle bifluide (ρ, u, p) à l'instant initial à *"droite"* ($x > 0$)

- Pour une initialisation par fonction utilisateur : **code** 1
  L'**argument** définit le numéro de la fonction utilisateur, écrit dans la routine *~/CERF/sources/PHY/USER/fct_iniw.f90*.
- Pour une initialisation par zones : **code** 2
  L'**argument** définit le nombre de zone à créer. L'ordre de définition des zones est important. Cela permet d' "imbriquer" les zones. Ainsi, il est recommandé de définir la première zone couvrant largement le domaine d'étude. Puis pour chaque zone on utilise 2 lignes :
  - 1 ligne constituée de 6 réels pour définir la zone : x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}
  - 1 ligne constituée de n_{dof} réels définissant les variables primitives à l'instant initial

Tip 2: Exemple d'initialisation par zones

```
INIT    2    2
-1. 4.1 -1. 3.1 -1. 1.
1. 0. 0. 0.  1.e5
-1. 1. -1. 2. -1. 1.
1000. 0. 0. 0.  1.e5
```

Dans cet exemple, on cherche à initialiser un problème de "rupture de barrage".Dans un domaine "2D" de 4×3 rempli d'air, on initialise une colonne d'eau à gauche de hauter 2 et largeur 1. On définit alors 2 zones. La première zone couvre largement le domaine entier et la seconde le domaine occupé par l'eau. Dans tout le domaine on initialise $\rho = 1$, $\vec{u} = \vec{0}$ et $p_0 = 10.^5$. Par la deuxième zone, dans la partie occupée par l'eau, on initialise $\rho = 1000$, $\vec{u} = \vec{0}$ et $p_0 = 10.^5$.

Important

Dans le cas où une fonction utilisateur est utilisée pour l'initialisation, il est nécessaire de relancer la compilation et l'édition de lien par la commande *make*!

2.2.5 Définition de la bathymétrie et de la friction : BATH

L'entête **BATH** permet de définir les paramètre de la bathymétrie et de la friction dans le cas, bien évidemment où le modèle de Saint-Venant est choisi.

- Dans le cas où l'on considère un fond plat sans friction : **code** 0
- Dans le cas où l'on considère une bathymétrie et une friction définies par une fonction utilisateur : **code** 1

L'**argument** définit le numéro de la fonction utilisateur, écrit dans la routine *~/CERF/sources/PHY/USER/fct_bathy.f90*.

- Dans le cas où l'on considère une bathymétrie et une friction définie bi-linéairement par bloc : **code** 2
  L'**argument** 0 permet de lire le fichier et l'argument -1 permet de le créer. Sur la ligne suivante on indique le nom du fichier ASCII contenant les informations de la bathymétrie et de la friction. Ce fichier doit se trouver dans le répertoire ***exec***. Lors de la création (argument -1) le fichier est constitué de 2 colonnes contenant les coordonnées x et y des quatre sommets de chaque bloc du maillage maître. L'utilisateur doit ensuite compléter le fichier, en rajoutant deux colonnes de réels indiquant pour chaque coordonnées les valeurs de la bathymétrie et de la friction.

2.2.6 Définition des conditions aux limites : COND

L'entête **COND** permet de définir les conditions aux limites du problème. Les conditions aux limites sont définies par un numéro (écrit sur une ligne) et éventuellement des valeurs réelles (écrites sur une autre ligne). Les conditions aux limites disponibles sont :

- **0** : condition de sortie libre (on recopie)
- **1** : condition miroir
- **2** : condition de Dirichlet sur les variables primitives. On doit alors définir les valeurs des n_{dof} variables primitives sur la ligne suivante. Si la valeur 10^{20} est utilisée, alors la variable sera considérée comme libre.
- **3** : condition définie par l'utilisateur. L'utilisateur doit alors définir la fonction de condition aux limites dans la routine *~/CERF/sources/PHY/USER/fct_cond.f90*. Sur la ligne suivante, on définit le numéro de la fonction utilisateur.
- **4** : condition de Dirichlet sur les variables conservatives. On doit alors définir les valeurs des n_{dof} variables conservatives sur la ligne suivante. Si la valeur 10^{20} est utilisée, alors la variable sera considérée comme libre.
- **999** : Aucune condition aux limites. Les flux ne sont pas calculés.

La définition des conditions aux limites sur les bords du domaine dépend alors du type de maillage maître utilisé :

- Dans le cas où l'on considère un "tube à choc" : **code** 0
  On est dans le cas où la section **MAME** a pour **code** 1, **argument** 0, et la section **INIT** a pour **code** 0. Dans ce cas, aucune définition n'est nécessaire.
- Dans le cas où l'on considère un maillage "2D" : **code** 1
  On est dans le cas où la section **MAME** a pour **code** 1, **argument** 1. On définit alors les conditions aux limites sur les bords du domaine dans l'ordre x_{min}, x_{max}, y_{min}, y_{max}. Pour chaque bord, on utilise 1 ou 2 lignes pour définir la condition aux limites.

- Dans le cas où l'on considère un maillage "3D" : **code** 2
  On est dans le cas où la section **MAME** a pour **code** 1, **argument** 2. On définit alors les conditions aux limites sur les bords du domaine dans l'ordre x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}. Pour chaque bord, on utilise 1 ou 2 lignes pour définir la condition aux limites.
- Dans le cas où l'on considère un maillage "**GMSH**" : **code** 3
  On est dans le cas où la section **MAME** a pour **code** 2. On utilise alors les numéros de "*zones*" définis par **GMSH**. L'**argument** définit le nombre de *zones* à traiter. Ces zones correspondent aux bords du domaine. Pour chaque zone, on utilise 1 ou 2 lignes pour définir la condition aux limites.

Exemple de conditions aux limites

```
COND    1    0
1
1
1
1
```

Dans cet exemple, on applique des conditions "*miroir*" sur les 4 bords d'un domaine "2D" rectangulaire.

2.2.7 Définition du maillage : MESH

L'entête **MESH** permet de définir le maillage du domaine, c'est à dire la discrétisation des blocs définis dans la section **MAME**. De plus, on définit les paramètres de (dé)raffinement de maillage. Le **code** par défaut est 0.

On peut alors définir/modifier les **mots clés** suivants :

- **NBDS** : Nombre de domaines à créer, c'est-à-dire nombre de processus **MPI** (valeur par défaut 1.)
- **MARL** : Niveau maximum de raffinement ("MAximal Refinement Level") (valeur par défaut 0.)
- **COPA** : Valeur seuil du critère de déraffinement ("mesh COarsening PArameter") (valeur par défaut 0.002)
- **REPA** : Valeur seuil du critère de raffinement ("mesh REfinement PArameter") (valeur par défaut 0.02)
- **FDRA** : Numéro de la fonction utilisateur définissant le raffinement à appliquer. Cette fonction est définie dans la routine *~/CERF/sources/UTI/USER/fct_mesh.f90*

- **NZRA** :Nombre de zones définissant le raffinement à appliquer (default 1.). Suivi, pour chaque zone, de 1 entier et 6 réels définissant le niveau de raffinement à appliquer à chacun des blocs de la zone et les limites de la zone x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}.

Tip 2: Exemple de raffinement par zones

```
MESH 0
NBDS 4
MARL 3
COPA 0.8
REPA 1.
NZRA 3
0 -1. 4.1 -1. 3.1 -1. 1.
3 -1. 1.25 -1. 2.25 -1. 1.
0 -1. .75 -1. 1.75 -1. 1.
```

On reprend l'exemple de "rupture de barrage" Tip 2. Dans un domaine "2D" de 4×3 rempli d'air, on considère une colonne d'eau à gauche de hauteur 2 et largeur 1. L'objectif est de raffiner autour de l'interface air-eau. On définit alors 3 zones. La première zone couvre largement le domaine entier pour affecter un niveau de raffinement 0. La seconde couvre le domaine occupé par l'eau avec une marge de 0.25, auquel on affecte le niveau maximal de raffinement 3. Et enfin, la troisième couvre le domaine occupé par l'eau avec une marge de -0.25, auquel on affecte le niveau minimal de raffinement 0, afin de minimiser le nombre de cellules. **CERF** ajustera automatiquement (c.a.d sans intervention de l'utilisateur) le niveau de maillage entre les niveaux 0 et 3 afin de respecter les critères imposés par le mailleur. Par ailleurs, on décompose le domaine en 4 domaines et on modifie les paramètres de gestion du critère de raffinement de maillage Section 1.1.3.3.

2.2.8 Définition d'"obstacle" : OBST

L'entête **OBST** permet de définir un ensemble de cellules à qui on impose une pénalisation de type "*fluide rigide*" dans le cas, bien évidemment où le modèle bi-fluide est choisi. Le **code** par défaut est 0 et l'**argument** définit le nombre d'obstacles à traiter.
Pour chaque obstacle on écrit sur une ligne : un *nom de fichier*, la *densité de l'obstacle* et le *type de pénalisation*.

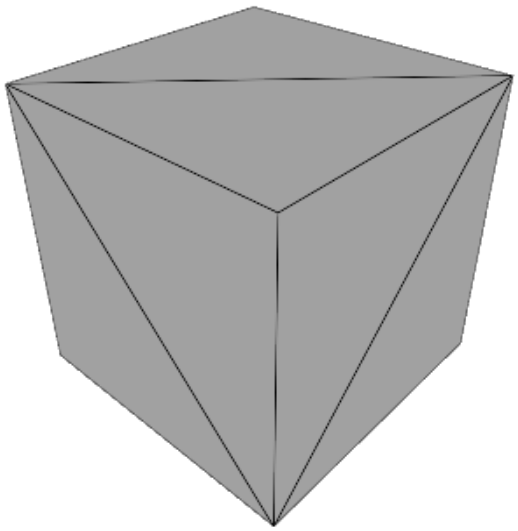
- *nom de fichier* : il s'agit d'une chaîne de caractère (20 caractères max) définissant le nom du fichier ASCII contenant le maillage de l'enveloppe de l'obstacle avec des éléments triangulaires au format **.OBJ**. Ce fichier doit se trouver dans le répertoire ***exec***.
- *densité de l'obstacle* : il s'agit d'un réel définissant la densité de l'obstacle. Cette densité est nécessairement $\rho_a \leq \rho_{obs} \leq \rho_w$.

- *type de pénalisation* : il s'agit d'un entier définissant le type de pénalisation. Les types de pénalisation disponibles sont :
  - **-1** : pénalisation de type "fixe". On impose une vitesse nulle.
  - **0** : pénalisation de type "libre". L'obstacle a un comportement de solide rigide libre.
  - **>0** : pénalisation de type "imposée". L'obstacle a un comportement de solide rigide auquel on impose la vitesse du centre de gravité et la vitesse de rotation. La valeur de l'entier définit le numéro de la fonction utilisateur définissant le mouvement dans la routine *~/CERF/sources/PHY/USER/fct_mvsol.f90*.

Exemple de fichier obstacle au format .obj

```
v 0. 0. 0.
v 1. 0. 0.
v 1. 1. 0.
v 0. 1. 0.
v 0. 0. 1.
v 1. 0. 1.
v 1. 1. 1.
v 0. 1. 1.
f 1 2 3
f 1 3 4
f 2 6 7
f 2 7 3
f 6 5 7
f 5 8 7
f 5 1 4
f 5 4 8
f 3 4 7
f 4 7 8
f 2 5 6
f 2 1 5
```

Dans cet exemple, on définit un obstacle de forme cubique de côté 1 centré en $(0.5, 0.5, 0.5)$. On définit les sommets de l'obstacle puis les faces de l'obstacle. Les sommets sont définis par la lettre **v** suivie des coordonnées du sommet. Les faces sont définies par la lettre **f** suivie des indices des sommets de la face. La figure ci-après illustre la forme de l'obstacle ainsi défini.



2.2.9 Définition des paramètres numériques: NUME

L'entête **NUME** permet de définir les paramètres numériques du problème à traiter. Le **code** par défaut est 0.

On peut alors définir/modifier les **mots clés** suivants:

- **TINT** : Intervalle de temps du calcul (Time INTerval) (valeur par défaut 0.)
- **CCFL** : Coefficient CFL Important 1 (valeur par défaut 0.9)
- **TDOR** : Ordre de la discrétisation temporelle (Time Discretization ORder) 1 ou 2 (valeur par défaut 1.)
- **SDOR** : Ordre de la discrétisation spatiale (Space Discretization ORder) 1 ou 2 (valeur par défaut 1.)
- **LIMI** : Type de limiteur 0/Barth, 1/Cartesian (valeur par défaut 0.)
- **NPRO** : Nombre de sondes < 10 (valeur par défaut 0.). Puis pour chaque sonde une ligne contenant 3 réels (coordonnées x, y, z de la sonde)

- **SAVE** : Type de sauvegarde 0/Non, 1/Oui. Le chiffre des unités se rapporte à la sauvegarde dans les fichiers *cerfbin-xxx*. Le chiffre des dizaines se rapporte à une sauvegarde au format gnuplot en 1D. (valeur par défaut 01.)

2.3 Les fichiers sources de CERF

2.3.1 Structure et modules

Comme illustré sur Figure 2.1, le code source de **CERF** est organisé en plusieurs répertoires:

- **./sources/PHY** contient toutes les routines de plus bas niveau concernant la manipulation des variables et entités physiques. Les variables physiques globales et les *types* de données (au sens Fortran 90), comme les degrés de liberté, leur gradients, les flux ou les caractéristiques physiques, sont définis dans le module **phy_typ** (*./sources/PHY/mod_phy_typ.f90*). Le module **phy** (*./sources/PHY/mod_phy.f90*), quant à lui, contient le module **phy_typ** ainsi que toutes les interfaces des routines contenues dans le répertoire **PHY**. Lors de la compilation, le résultat de compilation des routines de ce répertoire est archivé dans la librairie *./lib/lib_PHY.a*.
- **./sources/GEO** contient toutes les routines concernant la manipulation des variables et entités géométriques: calcul des flux aux interface, calcul des termes sources, ….. Les variables géométriques globales et les *types* de données (au sens Fortran 90), comme la liste des faces, la liste des cellules, le maillage, … etc …, sont définis dans le module **geo_typ** (*./sources/GEO/mod_geo_typ.f90*). Le module **geo_typ** contient le module **phy**. Ils sont imbriqués. Le module **geo** (*./sources/GEO/mod_geo.f90*), quant à lui, contient le module **geo_typ** ainsi que toutes les interfaces des routines contenues dans le répertoire **GEO**. Lors de la compilation, le résultat de compilation des routines de ce répertoire est archivé dans la librairie *./lib/lib_GEO.a*.
- **./sources/NUM** contient toutes les routines de plus haut niveau concernant la manipulation des variables et entités numériques: algorithme rk2, sauvegarde,communication MPI, …. Les variables numériques globales et les *types* de données (au sens Fortran 90), comme le "calcul", sont définis dans le module **num_typ** (*./sources/NUM/mod_num_typ.f90*). Le module **num_typ** contient le module **geo**. Ils sont imbriqués. Le module **num** (*./sources/NUM/mod_num.f90*), quant à lui, contient le module **num_typ** ainsi que toutes les interfaces des routines contenues dans le répertoire **NUM**. Lors de la compilation, le résultat de compilation des routines de ce répertoire est archivé dans la librairie *./lib/lib_NUM.a*.
- **./sources/UTI** contient les programmes et les toutes les routines nécessaires au pré et post-traitement, donc en particulier à la gestion du maillage AMR. Le module **uti** (./sources/UTI/mod_uti.f90), contient le module **num** ainsi que toutes les interfaces des routines contenues dans le répertoire **UTI**. Lors de la compilation, le résultat de compilation des routines de ce répertoire est archivé dans la librairie *./lib/lib_UTI.a*.

Les modules **phy**, **geo**, **num** et **uti** sont imbriqués et interdépendants. Les types de données qui y sont définies sont les outils de base de la méthode des volumes finis. Dans le cadre de la gestion de maillages **AMR**, il a été necessaire de développer de nouveaux outils qui ont été regroupés dans deux modules.

- **zorder** (*./sources/UTI/mod_zorder.f90*) : ce module contient les outils de gestion de la structure de données "z-order" qui est utilisée pour gérer le maillage AMR. Il contient l'ensemble des routines nécessaire à la gestion des Code de Morton.
- **msh** (*./sources/UTI/mod_msh.f90*) : ce module contient les outils de gestion de la structure de données "maillage" qui est utilisée pour gérer le maillage AMR. Il contient l'ensemble des routines nécessaire à la gestion du maillage AMR. Les strucutures de données du code volumes finis sont stockées dans des tableaux "fixes" par soucis de performance. La gestion des graphes de maillage, par contre est plus complexe et nécessite des structures de données dynamiques avec des listes chainées. Le module **msh** contient l'ensemble des outils nécessaires à la gestion de ces structures de données.

2.3.2 Le graphe de dépendance du programme principal

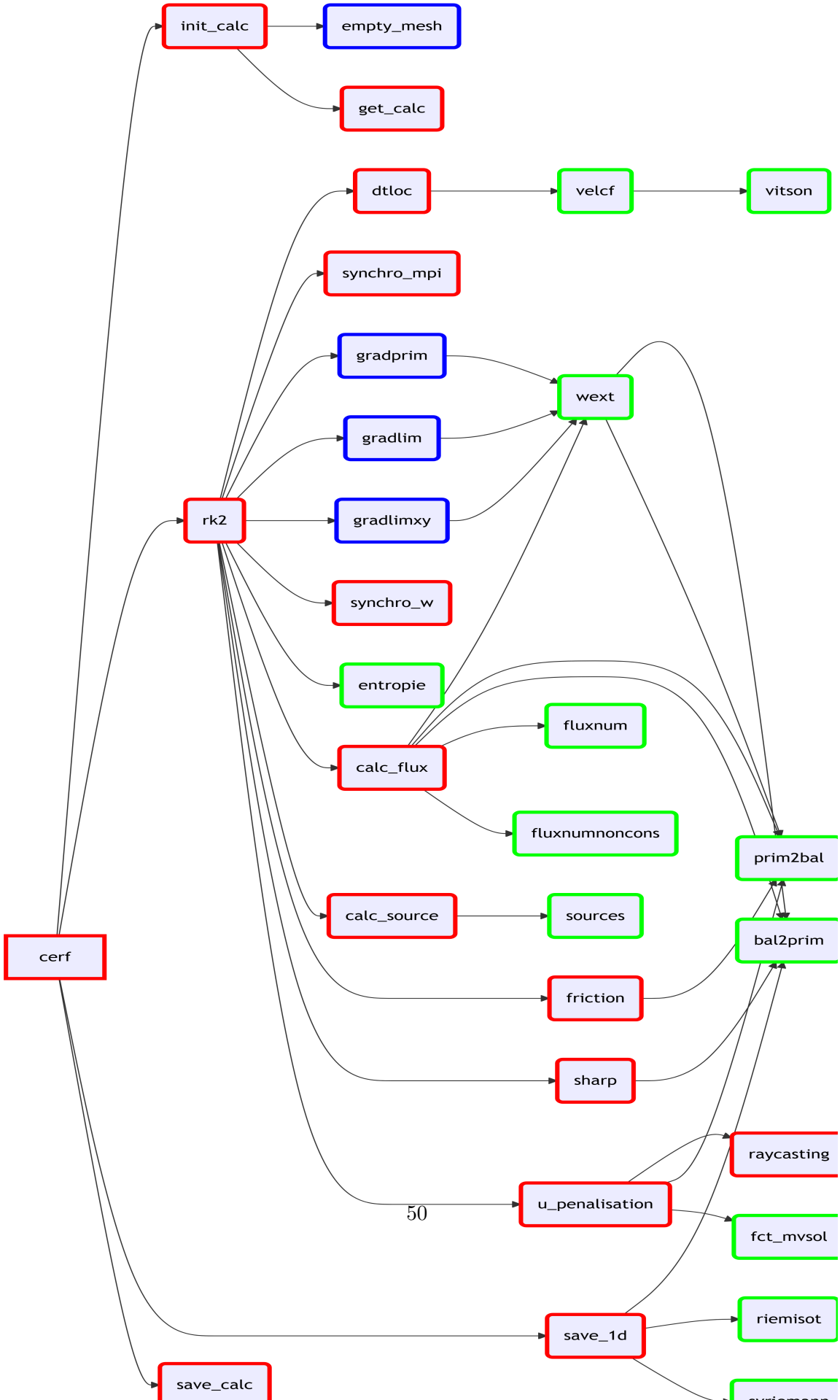
Nous présentons ci-apprès le graphe de dépendance du programme principal ***cerf*** (hormis les routines MPI et les routines d'affichage). Les routines dépendant du module **phy** sont entourées de **vert**, les routines dépendant du module **geo** sont entourées de **bleu** et les routines dépendant du module **num** sont entourées de **rouge**. La documentation au format html permet d'accéder directement au code source de la routine en cliquant sur le nom de la routine dans le graphe Figure 2.3.

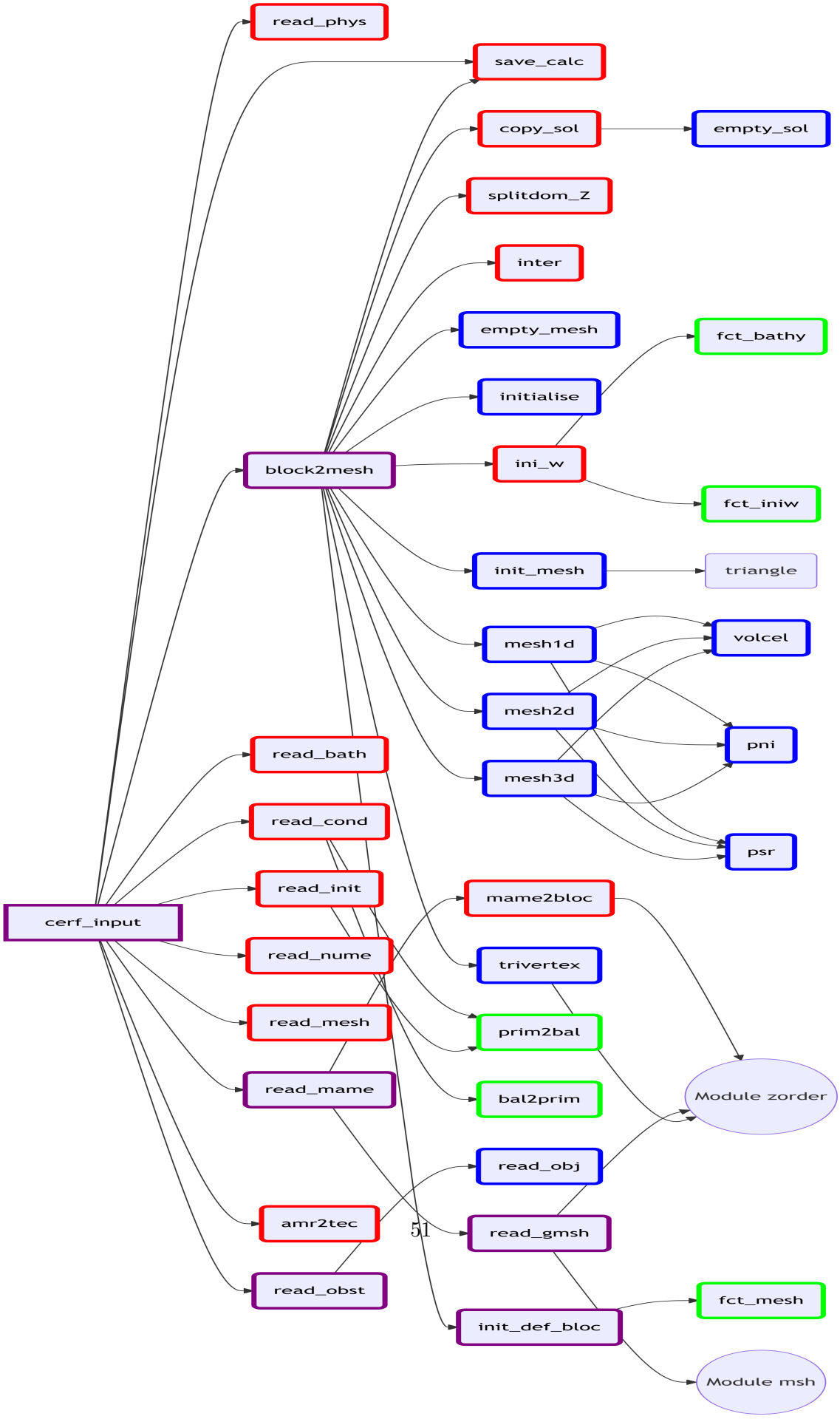
2.3.3 Le graphe de dépendance du programme de pré-traitement

Nous présentons ci-apprès le graphe de dépendance du programme de pré-traitement ***cerf_input*** (hormis les routines MPI et les routines d'affichage). Les routines dépendant du module **phy** sont entourées de **vert**, les routines dépendant du module **geo** sont entourées de **bleu**, les routines dépendant du module **num** sont entourées de **rouge** et les routines dépendant du module **uti** sont entourées de **violet**. La documentation au format html permet d'accéder directement au code source de la routine en cliquant sur le nom de la routine dans le graphe Figure 2.4.

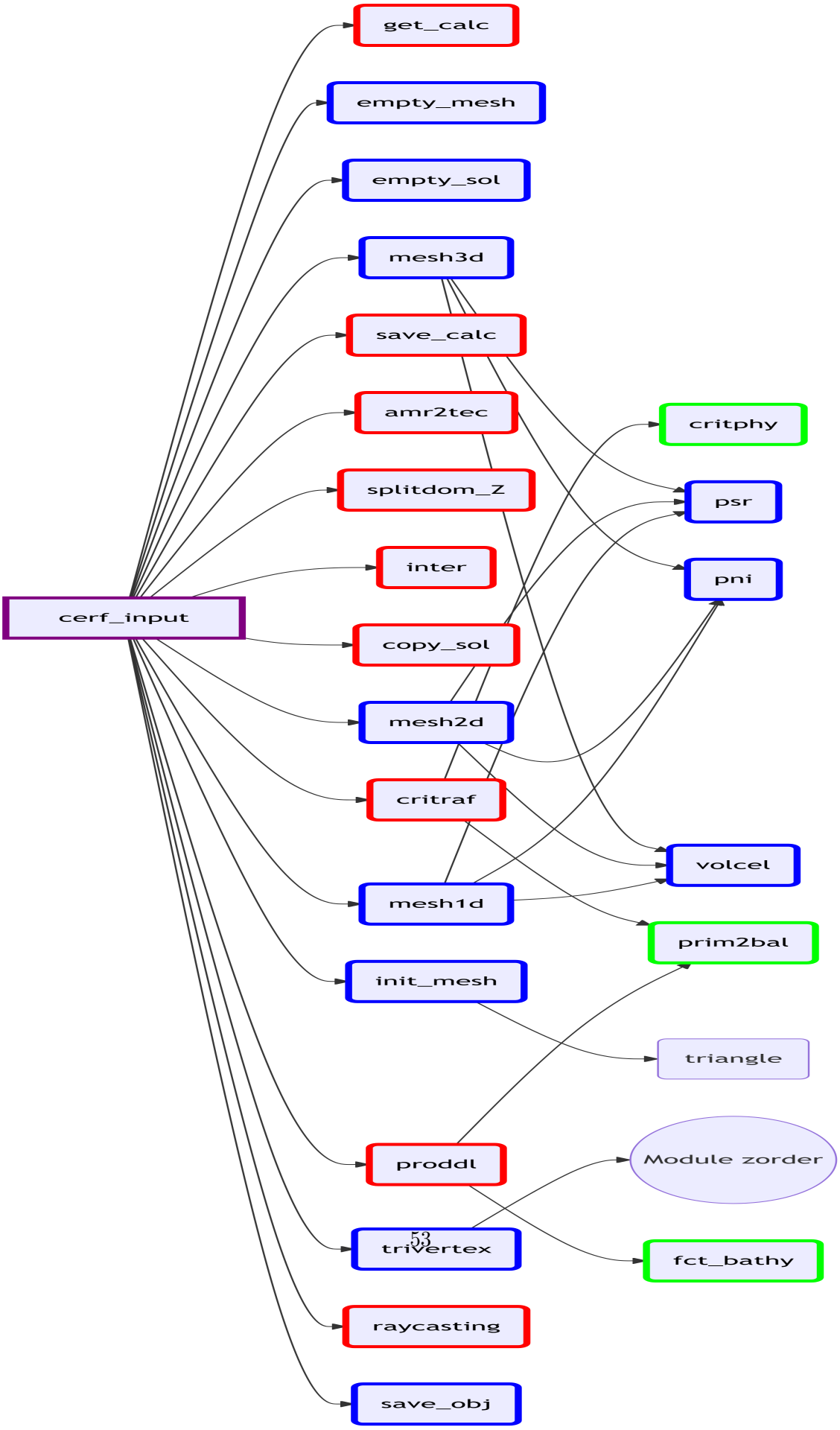
2.3.4 Le graphe de dépendance du programme de modification du maillage

Nous présentons ci-apprès le graphe de dépendance du programme de de modification du maillage ***cerf_amr*** (hormis les routines MPI et les routines d'affichage). Les routines dépendant du module **phy** sont entourées de **vert**, les routines dépendant du module **geo** sont entourées de **bleu**, les routines dépendant du module **num** sont entourées de **rouge** et les routines


init_calc
empty_mesh
get_calc
dtloc
velcf
vitson
synchro_mpi
gradprim
wext
gradlim
rk2
gradlimxy
synchro_w
entropie
fluxnum
calc_flux
fluxnumnoncons
prim2bal
calc_source
sources
bal2prim
cerf
friction
sharp
raycasting
u_penalisation
fct_mvsol
riemisot
save_1d
save_calc


read_phys
save_calc
copy_sol
empty_sol
splitdom_Z
inter
empty_mesh
fct_bathy
initialise
block2mesh
ini_w
fct_iniw
init_mesh
triangle
mesh1d
volcel
mesh2d
pni
mesh3d
psr
read_bath
read_cond
read_init
cerf_input
mame2bloc
read_nume
trivertex
read_mesh
prim2bal
read_mame
bal2prim
Module zorder
read_obj
amr2tec
read_gmsh
read_obst
fct_mesh
init_def_bloc
Module msh

dépendant du module **uti** sont entourées de **violet**. La documentation au format html permet d'accéder directement au code source de la routine en cliquant sur le nom de la routine dans le graphe Figure 2.5.


get_calc
empty_mesh
empty_sol
mesh3d
save_calc
amr2tec
splitdom_Z
inter
cerf_input
copy_sol
mesh2d
critraf
mesh1d
init_mesh
proddl
trivertex
raycasting
save_obj
critphy
psr
pni
volcel
prim2bal
triangle
Module zorder
fct_bathy

3 Exemples documentés

3.1 Exemples avec le modèle de Saint-Venant

Dans ce chapitre, nous présentons la mise en oeuvre de quelques cas tests classiques d'écoulements pour le modèle de Saint-Venant. Les cas traités sont les suivants :

- Résolution d'un problème de Riemann
- Ressaut hydraulique
- Friction, cas test de McDonald
- ...

3.1.1 Problème de Riemann

On considère le cas classique de la rupture de barrage 1D pour le modèle de Saint-Venant sur un fond plat. On reprend le cas test de Faccanoni and Mangeney (2013) (cas test 5). Le domaine est défini par $x \in [-2., 2.]$ et $t \in [0, 0.5]$. Les conditions initiales sont données par :

$$\overrightarrow{w_p}(x,0) = \begin{cases} \overrightarrow{w_p}_L =< h_L, u_L >^T =< 0.0046, 0. >^T & \text{si } x < 0 \\ \overrightarrow{w_p}_R =< h_R, u_R >^T =< 0.1446, 0. >^T & \text{si } x > 0 \end{cases}$$

On ainsi cas de rupture de barrage sans zone sèche, avec un choc et une détente, dont la solution exacte peut être calculée par le solveur de Riemann.

Dans un premier temps, on considère un maillage 1D uniforme et fixe de 400 cellules sur un seul domaine. Le fichier de données correspondant au cas traité est fourni ci-après:

Tip 3: Fichier de données *ex1_sv_Riemann.inp*

```
!---------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
```

```
!---------------------------------------------------------------------

!---------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!---------------------------------------------------------------------
! MODE : physical model
!        1/ Shallow water
!        2/ Euler isothermal two-phases               (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z   (default 2.)
! PREF : isothermal pressure of reference             (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)          (default 20.)
! REFA : Isothermal reference density for Air         (default 1.)
! REFW : Isothermal reference density for Water       (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                       (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level            (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach       (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!---------------------------------------------------------------------
PHYS 0
MODE 1.

!---------------------------------------------------------------------
! Reading the MAster MEsh
! Header  MAME
!---------------------------------------------------------------------
! code 1 -> creation of a mesh
!
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
! next line: nx
!
!           argument: 1 type "2d box"
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
```

```
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!---------------------------------------------------------------------
MAME 1 0
-2. 2.
400

!---------------------------------------------------------------------
! Reading Initial conditions
! Header INIT
!---------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!---------------------------------------------------------------------
INIT 0 0
0.0046 0. 0.
0.1446 0. 0.

!---------------------------------------------------------------------
! Reading bathymetry and friction
! Header BATH
!---------------------------------------------------------------------
! code 0 -> Default : flat bottom and no friction

! code 1 -> bathymetry and friction defined by user function (defined in sources/PHY/fct_bat
!           argument: Function's Number
! code 2 -> bathymetry and friction defined linearly per block
```

```
!           argument: 0  reading the file
!           argument:-1  creation of a file containing the coordinates of the vertices of the
! next line: file name
!-------------------------------------------------------------------------
BATH    0   0

!-------------------------------------------------------------------------
! Reading boundary condition
! (maximum 10 for the moment)
! Header COND
!-------------------------------------------------------------------------
!
! code 0 -> Description in the case  "shock tube" : Nothing
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!-------------------------------------------------------------------------
COND    0    0

!-------------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!-------------------------------------------------------------------------
!
! code 0 -> default value
```

```
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
! FDRA : Function defining the mesh refinement to be applied
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-------------------------------------------------------------------------
MESH    0
NBDS 1
MARL 0
NZRA 1
0 -2.  2. -100. 100. -100. 100.

!-------------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-------------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
! TINT : Time interval to compute                   (default 0.)
! CCFL : cfl condition                              (default 0.9)
! TDOR : Time discretization order 1 ou 2         (defaut 1.)
! SDOR : Space discretization order 1 ou 2        (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian      (default 0.)
! SAVE : kind of backup, three-digit number         (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                     (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!-------------------------------------------------------------------------
NUME 0
TINT .5
CCFL .9
SDOR 2.
TDOR 2.
SAVE 11.
```

Pour lancer le calcul, comme il n'y a qu'un domaine à traiter sur un seul intervalle de temps :

```
./cerf_input ex1_sv_Riemann.inp
./cerf
```

Comme dans le fichier de données *ex1_sv_Riemann.inp* on a demandé une sauvegarde au format 1D (paramètre **SAVE**=11 section **NUME**), des fichiers de sorties 1D ont été générés. Des fichiers pour la solution numérique en hauteur d'eau et vitesse *'n_h_001'* et *'n_u_001'*, et des fichiers pour la solution exacte en hauteur d'eau et vitesse : *'e_h_001'* et *'e_u_001'*. Ces fichiers peuvent être visualisés par exemple avec gnuplot :

```
gnuplot
gnuplot> set term x11
gnuplot> plot 'e_h_001' w l, 'n_h_001' w l
gnuplot> plot 'e_u_001' w l, 'n_u_001' w l
gnuplot> quit
```

On peut ainsi comparer la solution numérique à la solution exacte.

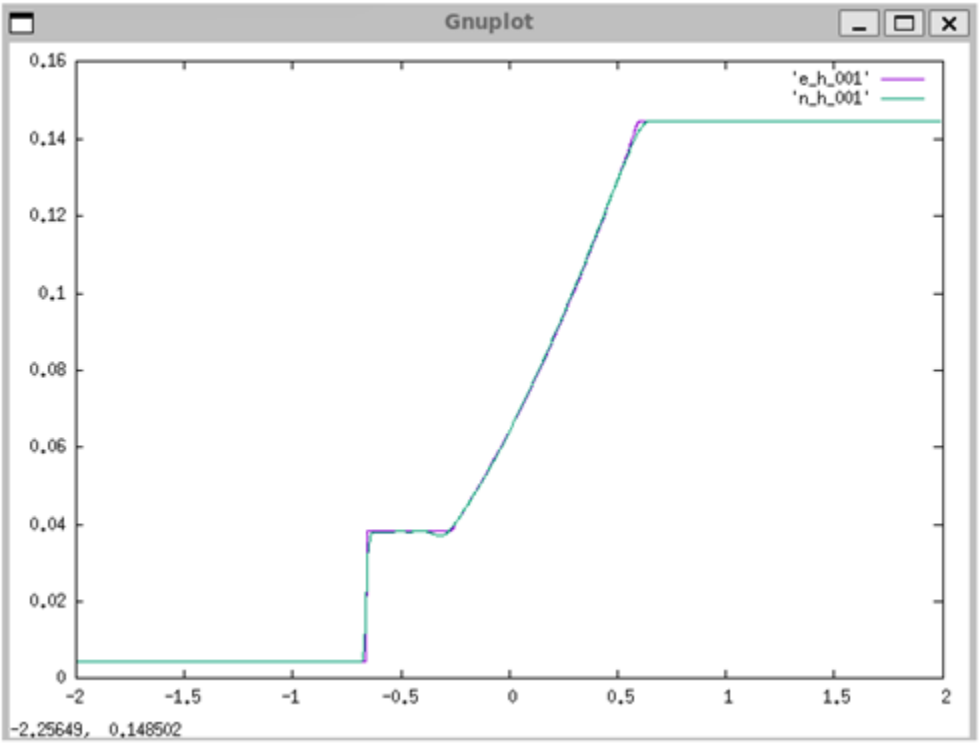


Figure 3.1: Comparaison numérique/exacte hauteur d'eau à $t = 0.5s$, maillage uniforme 400 cellules

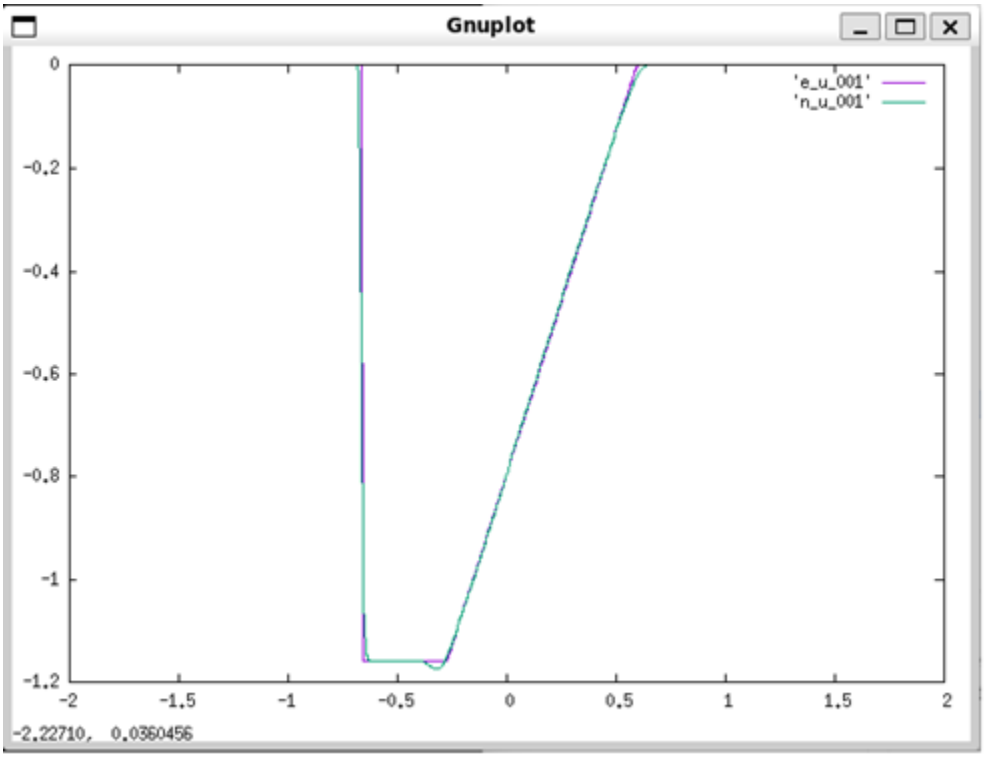


Figure 3.2: Comparaison numérique/exacte vitesse à $t = 0.5s$, maillage uniforme 400 cellules

Pour tester le code avec une zone "*sèche*" (Faccanoni and Mangeney (2013) cas test 1), on peut par exemple reprendre le même problème avec une condition initiale différente en modifiant la section **INIT** du fichier de données *ex1_sv_Riemann.inp*:

```
INIT    0   0
0. 0. 0.
0.1446 0. 0.
```

On obtiendra alors :

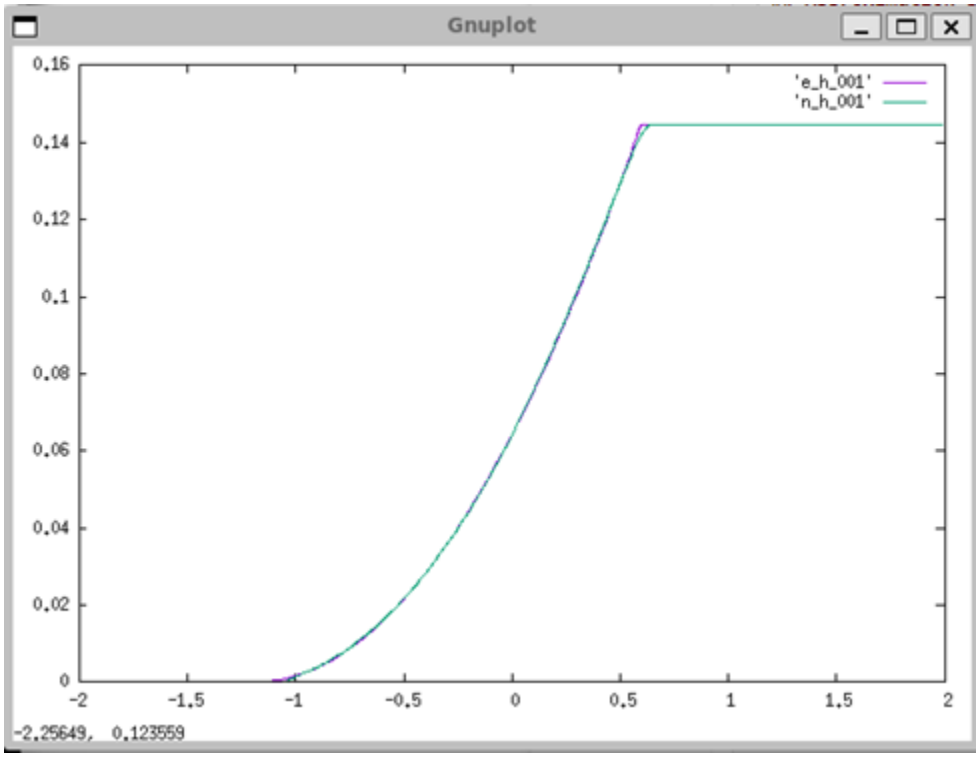


Figure 3.3: Comparaison numérique/exacte hauteur d'eau à $t = 0.5s$, maillage uniforme 400 cellules avec zone sèche

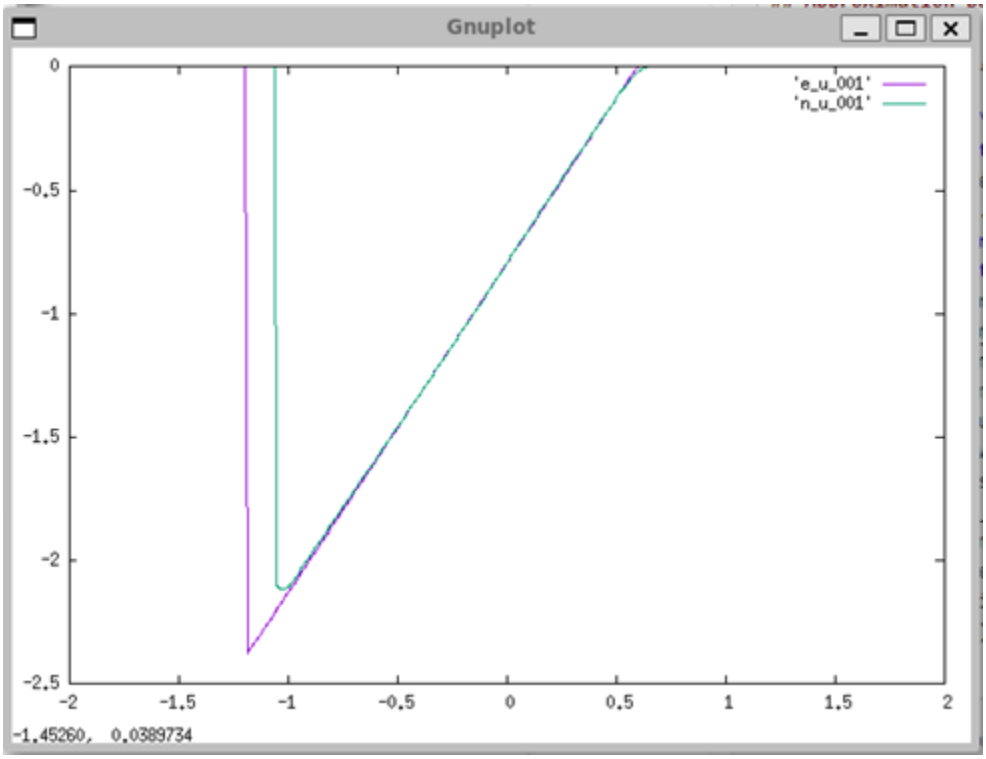


Figure 3.4: Comparaison numérique/exacte vitesse à $t = 0.5s$, maillage uniforme 400 cellules avec zone sèche

3.1.2 Ressaut hydraulique

On considère ici un cas tests stationnaire, sans friction, avec un fond non plat. La solution analytique est issue de Delestre et al. (2013). Sur un domaine $x \in [0., 25.]$, on considère une topographie définie par :

$$z_b(x) = \begin{cases} 0.2 - 0.05\,(x-10)^2 & \text{si } 8 < x < 12 \\ 0. & \text{sinon} \end{cases}$$

On considère le cas de l'écoulement transcritique avec choc, caractérisé par les conditions aux limites suivantes :

- À gauche, on impose $h = 0.33m$ et on laisse le débit libre.
- À droite, on impose $hu = 0.18m^2/s$ et on laisse la hauteur d'eau.

Comme préconisé par Delestre et al. (2013), les conditions initiales sont données par : $h + z_b = 0$ et $u = 0$.

On effectue le calcul sur un maillage fixe et régulier de 2500 cellules réparties sur 8 domaines. À la recherche d'un état stationnaire, on effectue le calcul jusqu'à $T = 100s$. Le fichier de données correspondant au cas traité est fourni ci-après Tip 4 :

Tip 4: Fichier de données *ex2_sv_Bump.inp*

```
!-----------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
!-----------------------------------------------------------------------

!-----------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!-----------------------------------------------------------------------
! MODE : physical model
!        1/ Shallow water
!        2/ Euler isothermal two-phases               (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z   (default 2.)
! PREF : isothermal pressure of reference             (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)          (default 20.)
! REFA : Isothermal reference density for Air         (default 1.)
! REFW : Isothermal reference density for Water       (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                       (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level            (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach       (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!-----------------------------------------------------------------------
PHYS 0
MODE 1.
FRMO 0.
BAGR 0.
!-----------------------------------------------------------------------
```

```
! Reading the MAster MEsh
! Header  MAME
!---------------------------------------------------------------------
! code 1 -> creation of a mesh
!
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
! next line: nx
!
!           argument: 1 type "2d box"
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!---------------------------------------------------------------------
MAME    1    0
0. 25.
2500
!---------------------------------------------------------------------
! Reading Initial conditions
! Header INIT
!---------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!---------------------------------------------------------------------
INIT    1   1
```

```
!-----------------------------------------------------------------------
! Reading bathymetry and friction
! Header BATH
!-----------------------------------------------------------------------
! code 0 -> Default : flat bottom and no friction

! code 1 -> bathymetry and friction defined by user function (defined in sources/PHY/fct_batl
!           argument: Function's Number
! code 2 -> bathymetry and friction defined linearly per block
!           argument: 0  reading the file
!           argument:-1  creation of a file containing the coordinates of the vertices of the
! next line: file name
!-----------------------------------------------------------------------
BATH    1   1

!-----------------------------------------------------------------------
! Reading boundary condition
! (maximum 10 for the moment)
! Header COND
!-----------------------------------------------------------------------
!
! code 0 -> Description in the case  "shock tube" : Nothing
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!-----------------------------------------------------------------------
```

```
COND    2    0
4
1.e20 0.18 0.
2
0.33 1.e20 0.
999
999
999
999

!-----------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!-----------------------------------------------------------------------
!
! code 0 -> default value
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
! FDRA : Function defining the mesh refinement to be applied
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-----------------------------------------------------------------------
MESH 0
NBDS 8
NZRA 1
0 0.  25. -100. 100. -100. 100.

!-----------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-----------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
! TINT : Time interval to compute                   (default 0.)
! CCFL : cfl condition                              (default 0.9)
```

```
! TDOR : Time discretization order 1 ou 2         (defaut 1.)
! SDOR : Space discretization order 1 ou 2        (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian       (default 0.)
! SAVE : kind of backup, three-digit number          (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                      (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!---------------------------------------------------------------------
NUME 0
TINT 100.
CCFL .9
SDOR 2.
TDOR 2.
SAVE 01.
```

Pour lancer le calcul, comme il n'y a qu'un domaine à traiter sur un seul intervalle de temps, mais sur 8 domaines, on utilise les commandes suivantes :

```
./cerf_input ex2_sv_Bump.inp
mpirun -np 8 ./cerf
./cerf2tec
```

On peut récupérer la solution analytique de ce problème sur le site de Swashes. Dans le répertoire *doc* de **CERF** on trouvera à titre d'information la solution analytique de $\eta = h + z_b$ sur 2500 points au format Tecplot. On peut alors comparer la solution numérique à la solution exacte sur la figure ci-après (Figure 3.5).

On peut également visualiser la solution numérique avec *Visit* ou *Paraview*:

Sur la figure Figure 3.6, on visualise les isovaleurs du niveau d'eau à $t = 100s$. On remarquera que le maillage a été modifié pour le post-traitement. Les coordonnées des nœuds de la base des hexaèdres correspondent à z_b, alors que les nœuds de la partie supérieure des hexaèdres correspondent à $z_b + h$. On peut ainsi visualiser la topographie et le niveau d'eau en même temps.

On peut également réaliser ce calcul en utilisant le raffinement de maillage automatique. Pour cela, on modifie le fichier de données en conséquence *ex2_sv_Bump.inp*. On crée alors un maillage constitué de seulement 250 blocs et on autorise un raffinement de maillage seulement jusqu'au niveau 2. On garde tous les autres paramètres du calcul comme pour le maillage fixe.

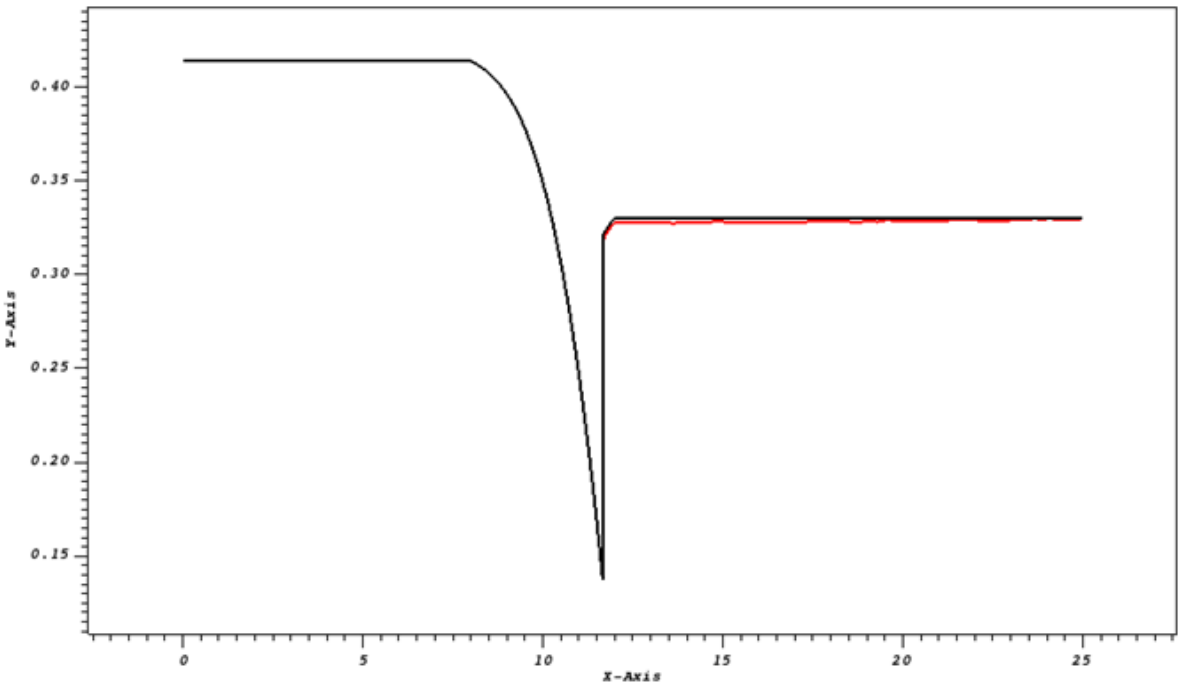


Figure 3.5: Comparaison numérique (Rouge) / exacte (Noire) niveau d'eau à $t = 100.s$, maillage uniforme 2500 cellules

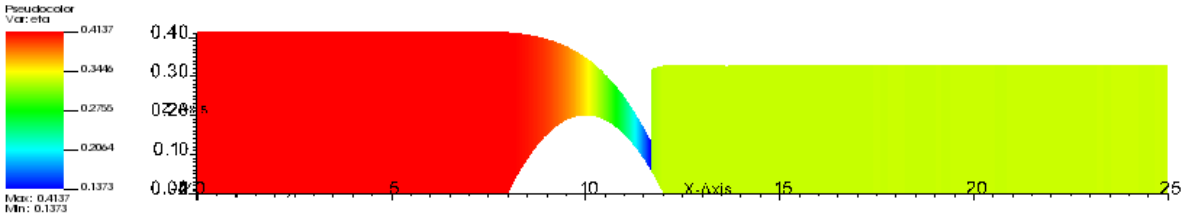


Figure 3.6: Isovaleur niveau d'eau à $t = 100.s$, maillage uniforme 2500 cellules

Tip 5: Fichier de données *ex2_sv_Bump_amr.inp*

```
!----------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
!----------------------------------------------------------------------

!----------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!----------------------------------------------------------------------
! MODE : physical model
!        1/ Shallow water
!        2/ Euler isothermal two-phases                (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z    (default 2.)
! PREF : isothermal pressure of reference              (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)           (default 20.)
! REFA : Isothermal reference density for Air          (default 1.)
! REFW : Isothermal reference density for Water        (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                        (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level            (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach        (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!----------------------------------------------------------------------
PHYS 0
MODE 1.

!----------------------------------------------------------------------
! Reading the MAster MEsh
! Header  MAME
!----------------------------------------------------------------------
! code 1 -> creation of a mesh
!
```

```
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
! next line: nx
!
!           argument: 1 type "2d box"
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!---------------------------------------------------------------------
MAME    1    0
0. 25.
2500
!---------------------------------------------------------------------
! Reading Initial conditions
! Header INIT
!---------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!---------------------------------------------------------------------
INIT    1   1

!---------------------------------------------------------------------
! Reading bathymetry and friction
! Header BATH
!---------------------------------------------------------------------
```

```
! code 0 -> Default : flat bottom and no friction

! code 1 -> bathymetry and friction defined by user function (defined in sources/PHY/fct_bath
!           argument: Function's Number
! code 2 -> bathymetry and friction defined linearly per block
!           argument: 0
! next line: file name
!
! code 3 -> bathymetry and friction defined  per block at the highest mesh refinement level
!           argument: 0
! next line: file name
!-----------------------------------------------------------------------
BATH    1   1

!-----------------------------------------------------------------------
! Reading boundary condition
! (maximum 10 for the moment)
! Header COND
!-----------------------------------------------------------------------
!
! code 0 -> Description in the case  "shock tube" : Nothing
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!-----------------------------------------------------------------------
COND    2    0
```

```
4
1.e20 0.18 0. 0. 0. 0.
2
0.33 1.e20 0. 0. 0. 0.
999
999
999
999

!-----------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!-----------------------------------------------------------------------
!
! code 0 -> default value
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
! FDRA : Function defining the mesh refinement to be applied
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-----------------------------------------------------------------------
MESH 0
NBDS 8
MARL 2
NZRA 2
0 0.  25. -100. 100. -100. 100.
2 7.8  12.1 -100. 100. -100. 100.


!-----------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-----------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
```

```
! TINT : Time interval to compute                    (default 0.)
! CCFL : cfl condition                                (default 0.9)
! TDOR : Time discretization order 1 ou 2          (defaut 1.)
! SDOR : Space discretization order 1 ou 2         (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian       (default 0.)
! SAVE : kind of backup, three-digit number          (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                      (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!-----------------------------------------------------------------------
NUME 0
TINT 10.
CCFL .9
SDOR 2.
TDOR 2.
SAVE 01.
```

Afin de lancer le calcul avec raffinement de maillage, on utilise un script :

```
#!/bin/bash
./cerf_input ex2_sv_Bump2.inp
./cerf2tec
mv cerfout.dat b_0.dat
mv cerfamr.dat b_mame_0.dat
for i in $(seq 1 20 )
do
mpirun -np 8 ./cerf
./cerf2tec
mv cerfout.dat b_$i.dat
./cerf_amr
mv cerfamr.dat b_mame_$i.dat
done
```

Comme précédemment, on peut comparer avec la solution exacte.

On peut également visualiser la solution numérique avec *Visit* ou *Paraview* :

Que soit sur Figure 3.7 ou Figure 3.8, on remarquera que le raffinement de maillage a permis de réduire le nombre de cellules tout en conservant une bonne précision sur la solution numérique.

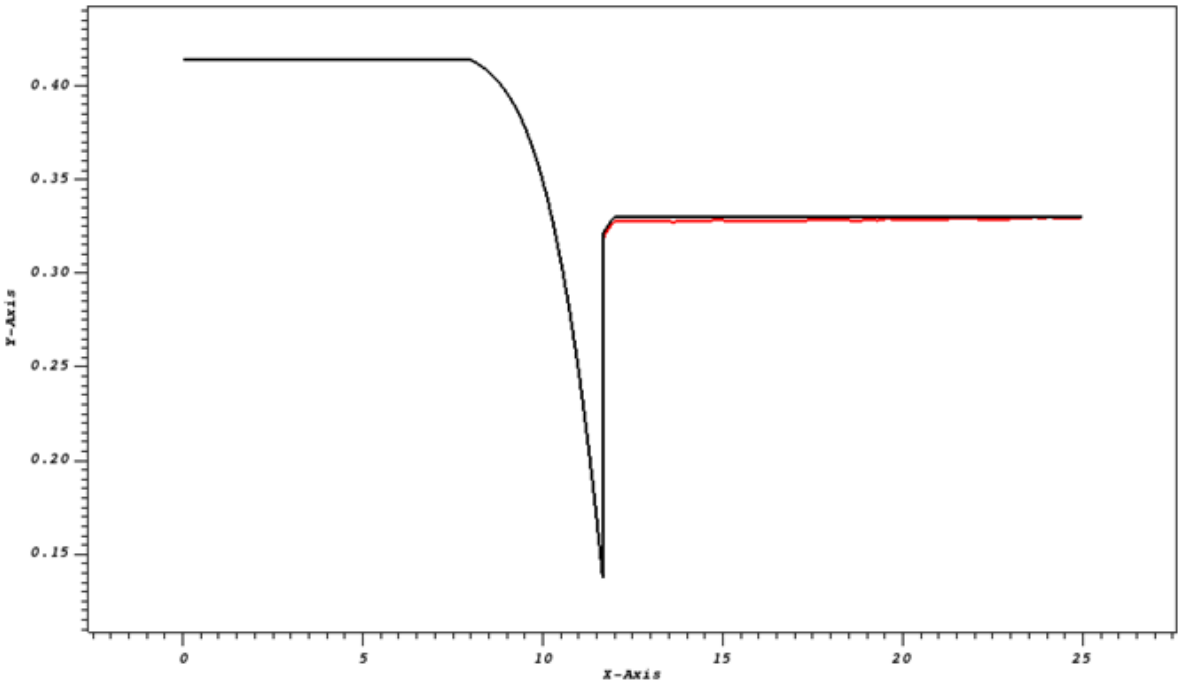


Figure 3.7: Comparaison numérique sur maillage fixe (Rouge) / numérique sur maillage dynamique (Vert) / exacte (Noire) niveau d'eau à $t = 100.s$

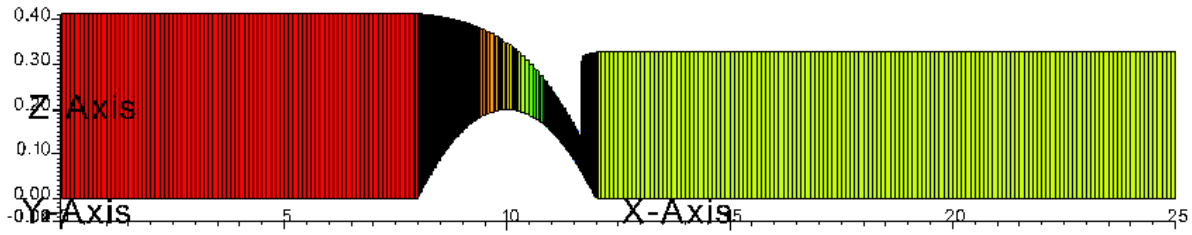


Figure 3.8: Isovaleur niveau d'eau, maillage dynamique 344 cellules à $t = 100.s$

3.1.3 Friction, cas test de McDonald

On considère ici un cas tests stationnaire, avec friction sur un fond non plat. La solution semi-analytique est issue de Delestre et al. (2013) (cas 3.2.1).

Sur un domaine $x \in [0., 1000.]$, on considère un écoulement sub-critique avec un débit constant à l'entrée de $q = hu = 2m^2/s$ et une hauteur d'eau prescrite à la sortie $h_e(1000)$. Le domaine est initialement sec, avec $h = 0$ et $u = 0$. On impose une friction de Manning avec $n = 0.033$. La hauteur d'eau est donnée par :

$$h_{ex}(x) = \left(\frac{4}{g}\right)^{1/3} \left(1 + \frac{1}{2} exp\left(-16\left(\frac{x}{1000} - \frac{1}{2}\right)\right)\right)$$

En utilisant Equation 1.6, dans le cas stationnaire, on peut alors retrouver la topographie par intégration de :

$$\frac{\partial z_b}{\partial x} = \left(\frac{q^2}{gh^3} - 1\right) \frac{\partial h}{\partial x} - S_f$$

On effectue le calcul sur un maillage fixe et régulier de 1000 cellules. À la recherche d'un état stationnaire, on effectue le calcul jusqu'à $T = 1500s$. Le fichier de données correspondant au cas traité est fourni ci-après Tip 6 :

> Tip 6: Fichier de données *ex3_sv_McDo.inp*

```
!----------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
!----------------------------------------------------------------------

!----------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!----------------------------------------------------------------------
! MODE : physical model
!        1/ Shallow water
!        2/ Euler isothermal two-phases               (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z   (default 2.)
```

```
! PREF : isothermal pressure of reference              (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)           (default 20.)
! REFA : Isothermal reference density for Air          (default 1.)
! REFW : Isothermal reference density for Water        (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                        (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level             (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach        (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!-----------------------------------------------------------------------
PHYS 0
MODE 1.
FRMO 1.

!-----------------------------------------------------------------------
! Reading the MAster MEsh
! Header  MAME
!-----------------------------------------------------------------------
! code 1 -> creation of a mesh
!
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
! next line: nx
!
!           argument: 1 type "2d box"
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!-----------------------------------------------------------------------
MAME    1    0
0. 1000.
1000
!-----------------------------------------------------------------------
! Reading Initial conditions
```

```
! Header INIT
!---------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!---------------------------------------------------------------------
INIT    2   1
-1. 1001. -100. 100. -100. 100.
0. 0. 0.

!---------------------------------------------------------------------
! Reading bathymetry and friction
! Header BATH
!---------------------------------------------------------------------
! code 0 -> Default : flat bottom and no friction

! code 1 -> bathymetry and friction defined by user function (defined in sources/PHY/fct_bath
!           argument: Function's Number
! code 2 -> bathymetry and friction defined linearly per block
!           argument: 0  reading the file
!           argument:-1  creation of a file containing the coordinates of the vertices of the
! next line: file name
!---------------------------------------------------------------------

BATH    1   2
!---------------------------------------------------------------------
! Reading boundary condition
! (maximum 10 for the moment)
! Header COND
!---------------------------------------------------------------------
!
! code 0 -> Description in the case  "shock tube" : Nothing
```

```
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!-----------------------------------------------------------------------
COND    2    0
4
1.e20 2. 0.
2
0.74832355831838937 1.e20 0. 0.
999
999
999
999

!-----------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!-----------------------------------------------------------------------
!
! code 0 -> default value
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
```

```
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
! FDRA : Function defining the mesh refinement to be applied
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-------------------------------------------------------------------------
MESH 0
NBDS 1
MARL 0
NZRA 1
0 0.  1000. -100. 100. -100. 100.

!-------------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-------------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
! TINT : Time interval to compute                    (default 0.)
! CCFL : cfl condition                               (default 0.9)
! TDOR : Time discretization order 1 ou 2         (defaut 1.)
! SDOR : Space discretization order 1 ou 2        (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian       (default 0.)
! SAVE : kind of backup, three-digit number          (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                       (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!-------------------------------------------------------------------------
NUME 0
TINT 50.
CCFL .9
SDOR 1.
TDOR 1.
SAVE 01.
```

Pour lancer le calcul, afin de visualiser l'évolution de la recherche de l'état stationnaire, on décompose le temps de calcul en 30 étapes en utilisant le script ci-après :

Script *ex3_sv_McDo.sh*

```
#!/bin/bash
./cerf_input ex3_sv_McDo.inp
./cerf2tec
mv cerfout1d.dat md_0.dat
for i in $(seq 1 30 )
do
./cerf
./cerf2tec
mv cerfout1d.dat md_$i.dat
done
```

On peut récupérer la solution analytique de ce problème sur le site de Swashes. Dans le répertoire *doc* de **CERF** on trouvera à titre d'information la solution analytique sur 10000 points au format Tecplot. On peut alors comparer la solution numérique à la solution exacte sur la figure ci-après (Figure 3.9).

3.2 Exemples avec le modèle bifluide eulérien

Dans ce chapitre, nous présentons la mise en oeuvre de quelques cas tests classiques d'écoulements bifluides. Les cas traités sont les suivants :

- Résolution d'un problème de Riemann
- Rupture de barrage
- Chute d'un cylindre sur un plan d'eau
- ...

3.2.1 Problème de Riemann

Afin de valider la mise en oeuvre du modèle, on considère le cas classique d'un "tube à choc" avec le modèle isotherme bi-fluide. On reprend le cas test de Frédéric Golay and Helluy (2007). Le domaine est défini par $x \in [-2., 2.]$ et $t \in [0, 0.25]$. Les conditions initiales sont données par :

$$\overrightarrow{w_p}(x,0) = \begin{cases} \overrightarrow{w_p}_L =< \rho_L, u_L, p_L >^T =< 0.0046,\ 0,\ 1.5\ 10^5 >^T & \text{si } x < 0 \\ \overrightarrow{w_p}_R =< \rho_R, u_R, p_R >^T =< 0.1446,\ 0.,\ 1.\ 10^5 >^T & \text{si } x > 0 \end{cases}$$

On considère un maillage 1D uniforme et fixe de 400 cellules sur un seul domaine. Le fichier de données correspondant au cas traité est fourni ci-après:

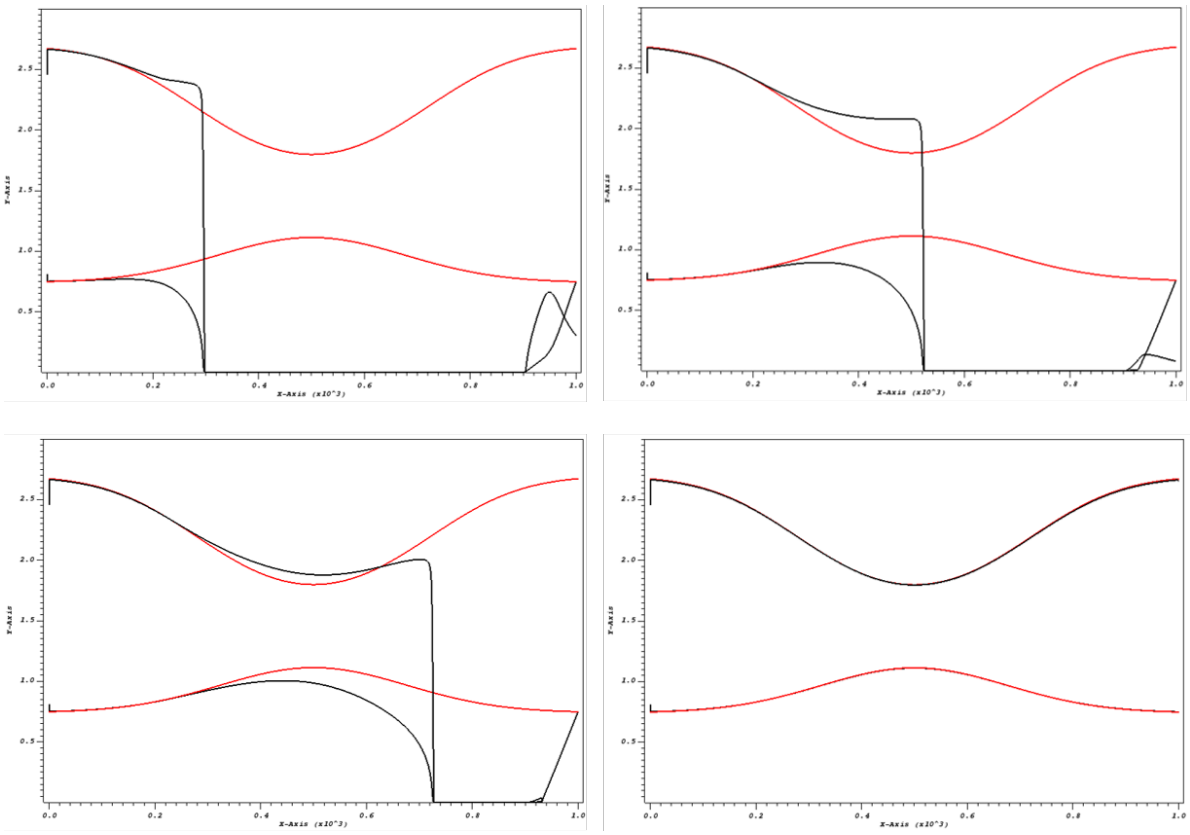


Figure 3.9: Comparaison numérique (Noire) / exacte (Rouge) sur le niveau d'eau h (en haut) et la vitesse u (en bas), avec un maillage uniforme 1000 cellules à l'ordre 1 en espace et en temps aux instants $t = 100s$ (en haut à gauche), $t = 200s$ (en haut à droite),$t = 300s$ (en bas à gauche),$t = 1500s$ (en bas à droite),

Tip 7: Fichier de données *ex1_bf_Riemann.inp*

```
!---------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
!---------------------------------------------------------------------

!---------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!---------------------------------------------------------------------
! MODE : physical model
!        1/  Shallow water
!        2/  Euler isothermal two-phases              (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z   (default 2.)
! PREF : isothermal pressure of reference             (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)          (default 20.)
! REFA : Isothermal reference density for Air         (default 1.)
! REFW : Isothermal reference density for Water       (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                       (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level            (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach       (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!---------------------------------------------------------------------
PHYS 0
MODE 2.


!---------------------------------------------------------------------
! Reading the MAster MEsh
! Header  MAME
!---------------------------------------------------------------------
! code 1 -> creation of a mesh
```

```
!
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
! next line: nx
!
!           argument: 1 type "2d box"
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!-------------------------------------------------------------------------
MAME    1    0
-10. 10.
400
!-------------------------------------------------------------------------
! Reading Initial conditions
! Header INIT
!-------------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!-------------------------------------------------------------------------
INIT    0   0
1125. 0. 1.5e5
1. 0. 1.e5

!-------------------------------------------------------------------------
```

```
! Reading boundary condition
! (maximum 10 for the moment)
! Header COND
!--------------------------------------------------------------------
!
! code 0 -> Description in the case  "shock tube" : Nothing
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!--------------------------------------------------------------------
COND    0    0


!--------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!--------------------------------------------------------------------
!
! code 0 -> default value
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
```

```
! FDRA : Function defining the mesh refinement to be applied
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-----------------------------------------------------------------------
MESH 0
NBDS 1
MARL 0
NZRA 1
0 -10.  10. -100. 100. -100. 100.

!-----------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-----------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
! TINT : Time interval to compute                    (default 0.)
! CCFL : cfl condition                               (default 0.9)
! TDOR : Time discretization order 1 ou 2         (defaut 1.)
! SDOR : Space discretization order 1 ou 2        (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian       (default 0.)
! SAVE : kind of backup, three-digit number          (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                      (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!-----------------------------------------------------------------------
NUME 0
TINT 0.25
CCFL .9
SDOR 2.
TDOR 2.
SAVE 11.
```

Pour lancer le calcul, comme il n'y a qu'un domaine à traiter sur un seul intervalle de temps :

```
./cerf_input ex1_bf_Riemann.inp
./cerf
```

Comme dans le fichier de données *ex1_bf_Riemann.inp* on a demandé une sauvegarde au format 1D (paramètre **SAVE**=11 section **NUME**), des fichiers de sorties 1D ont été générés. Des fichiers pour la solution numérique en densité, vitesse et pression *'n_r_001'*, *'n_u_001'* et *'n_p_001'*, et des fichiers pour la solution exacte en densité, vitesse et pression : *'e_r_001'*, *'e_u_001'* et *'e_p_001'*. Ces fichiers peuvent être visualisés par exemple avec gnuplot :

```
gnuplot
gnuplot> set term x11
gnuplot> plot 'e_r_001' w l, 'n_r_001' w l
gnuplot> plot 'e_u_001' w l, 'n_u_001' w l
gnuplot> plot 'e_p_001' w l, 'n_p_001' w l
gnuplot> quit
```

On peut ainsi comparer la solution numérique à la solution exacte.

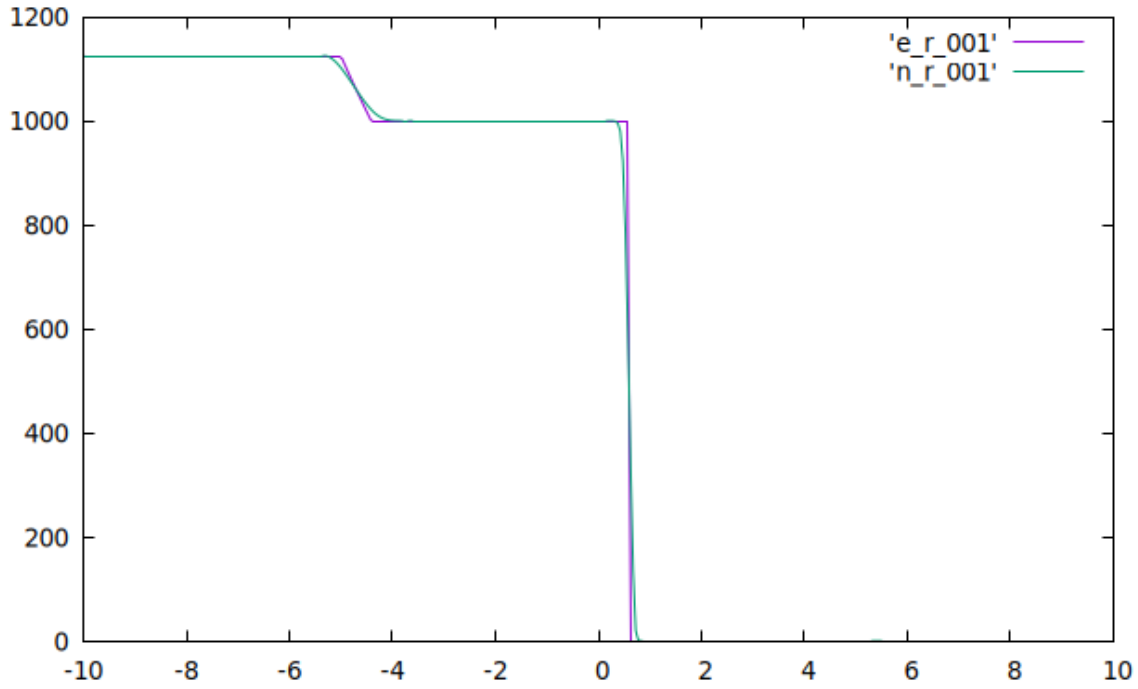


Figure 3.10: Comparaison numérique/exacte de la densité à $t = 0.25s$, maillage uniforme 400 cellules

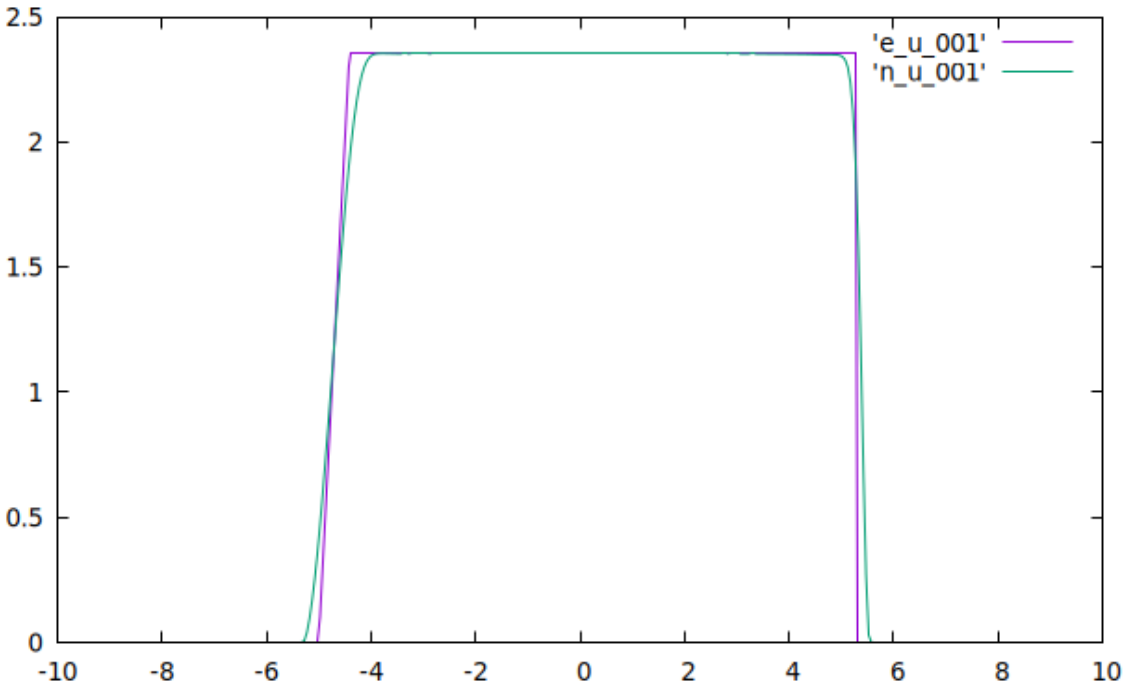


Figure 3.11: Comparaison numérique/exacte de la vitesse à $t = 0.25s$, maillage uniforme 400 cellules

3.2.2 Rupture de barrage

On considère ici un cas test 2D de rupture de barrage. Il s'agit d'un cas très largement utilisé et documenté. Dans point de vue expérimental, on peut par exemple s'appuyer sur les expériences réalisées par Martin and Moyce (1952). On considère un domaine de calcul $x \in [0., 4.]$ et $y \in [0., 3.]$. Les bords du domaine sont caractérisés par des conditions aux limites de type "miroir" ou Neumann nul. À l'instant initial, le domaine est constitué d'air, hormis une zone définie par $x \in [0., 2.]$ et $y \in [0., 3.]$ où l'eau est présente. Le maillage maître est constitué de 40×30 blocs qui sont raffinés à un niveau 4 autour de l'interface air / eau. Pour améliorer la qualité de l'interface, on pénalise le calcul par un coefficient de raidissement d'interface de 0.02. Le fichier de données correspondant au cas traité est fourni ci-après :

Tip 8: Fichier de données *ex2_bf_Dam.inp*

```
!-----------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
```

```
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
!---------------------------------------------------------------------

!---------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!---------------------------------------------------------------------
! MODE : physical model
!        1/ Shallow water
!        2/ Euler isothermal two-phases               (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z   (default 2.)
! PREF : isothermal pressure of reference             (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)          (default 20.)
! REFA : Isothermal reference density for Air         (default 1.)
! REFW : Isothermal reference density for Water       (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                       (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level            (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach       (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!---------------------------------------------------------------------
PHYS 0
MODE 3.
GPES 2
SHAR 0.02
!---------------------------------------------------------------------
! Reading the MAster MEsh
! Header  MAME
!---------------------------------------------------------------------
! code 1 -> creation of a mesh
!
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
! next line: nx
!
!           argument: 1 type "2d box"
```

```
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!---------------------------------------------------------------------
MAME    1    1
0. 4.  0. 3.
40 30
!---------------------------------------------------------------------
! Reading Initial conditions
! Header INIT
!---------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!---------------------------------------------------------------------
INIT    2    2
-1. 4.1 -1. 3.1 -1. 1.
1. 0. 0. 0.  1.e5 0.
-1. 1. -1. 2. -1. 1.
1000. 0. 0. 0.  1.e5 0.

!---------------------------------------------------------------------
! Reading boundary condition
! (maximum 10 for the moment)
! Header COND
!---------------------------------------------------------------------
```

```
!
! code 0 -> Description in the case  "shock tube" : Nothing
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!-----------------------------------------------------------------------
COND    1    0
1
1
1
1

!-----------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!-----------------------------------------------------------------------
!
! code 0 -> default value
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
! FDRA : Function defining the mesh refinement to be applied
```

```
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-------------------------------------------------------------------------
MESH 0
NBDS 8
MARL 4
COPA 0.9
REPA 3
NZRA 3
0 -1. 4.1 -1. 3.1 -1. 1.
4 -1. 1.5 -1. 2.25 -1. 1.
0 -1. .75 -1. 1.75 -1. 1.

!-------------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-------------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
! TINT : Time interval to compute                    (default 0.)
! CCFL : cfl condition                               (default 0.9)
! TDOR : Time discretization order 1 ou 2         (defaut 1.)
! SDOR : Space discretization order 1 ou 2        (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian       (default 0.)
! SAVE : kind of backup, three-digit number          (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                       (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!-------------------------------------------------------------------------
NUME 0
TINT 0.01
CCFL .9
SDOR 2.
TDOR 2.
SAVE 1.
```

Les vitesses attendues sont de l'ordre de quelques mètres par seconde. Donc, comme les blocs

ont une taille de $0.1m$, on peut estimer que l'intervalle de temps entre chaque remaillage doit être de $0.01s$. Pour accélérer le calcul on décompose le maillage en 8 domaines (ou encore 8 process Mpi). Pour effectuer un calcul jusqu'à $T = 0.8s$ on peut utiliser le script suivant :

```
#!/bin/bash
./cerf_input ex2_bf_Dam.inp
./cerf2tec
mv cerfout.dat d_0.dat
mv cerfamr.dat d_mame_0.dat
for i in $(seq 1 80 )
do
mpirun -np 8 ./cerf
./cerf2tec
mv cerfout.dat d_$i.dat
./cerf_amr
mv cerfamr.dat d_mame_$i.dat
done
```

Sur les images ci-après Figure 3.12, la densité du mélange est représentée (Bleu pour l'eau et rouge pour l'air). On vérifie que l'interface est "*fine*", c'est-à-dire sur 3-4 cellules.

3.2.3 Chute d'un cylindre sur un plan d'eau

On considère ici un cas test 2D de chute d'un objet solide sur un plan d'eau. Ce problème d'interaction fluide-structure est traité ici avec l'outil de "fluide-rigide" de CERF (Frederic Golay (2018)). Il s'agit d'un cas très largement utilisé et documenté (voir par exemple H. Sun and Faltinsen (2006), P. Sun, Ming, and Zhang (2015)). D'un point de vue expérimental, on peut par exemple s'appuyer sur les expériences réalisées par Greenhow and Lin (2015). On considère un domaine de calcul $x \in [0., 2.]$ et $y \in [0., 2.]$. Les bords du domaine sont caractérisés par des conditions aux limites de type "miroir" ou Neumann nul. À l'instant initial, le domaine est constitué d'air, hormis une zone définie par $x \in [0., 2.]$ et $y \in [0., 0.65]$ où l'eau est présente. Le solide rigide est défini par un maillage au format .OBJ. Les cellules à l'intérieur de cette enveloppe auront un comportement de solide rigide libre, soumis à la pesanteur et de densité $500kg/m^3$. Le maillage maître est constitué de 40×40 blocs qui sont raffinés à un niveau 4 autour de l'interface air / eau et autour du cylindre dont le centre est positionné à $0.5m$ de la surface du plan d'eau comme illustré sur la figure Figure 3.13. Pour améliorer la qualité de l'interface, on pénalise le calcul par un coefficient de raidissement d'interface de 0.01.

Le fichier de données correspondant au cas traité est fourni ci-après :

> Tip 9: Fichier de données *ex3_bf_cyl.inp*

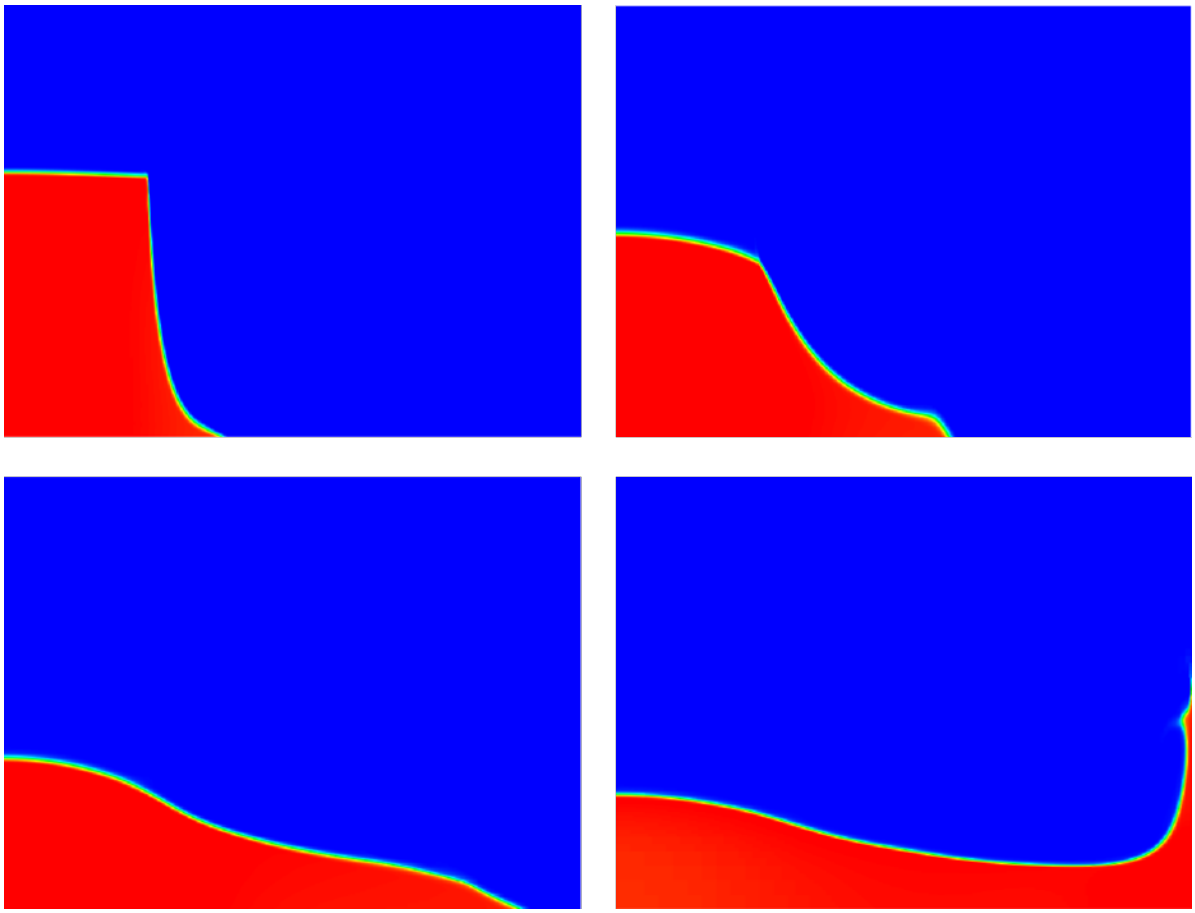


Figure 3.12: Rupture de barrage, densité à $t = 0.2s, 0.4s, 0.6s, 0.8s$

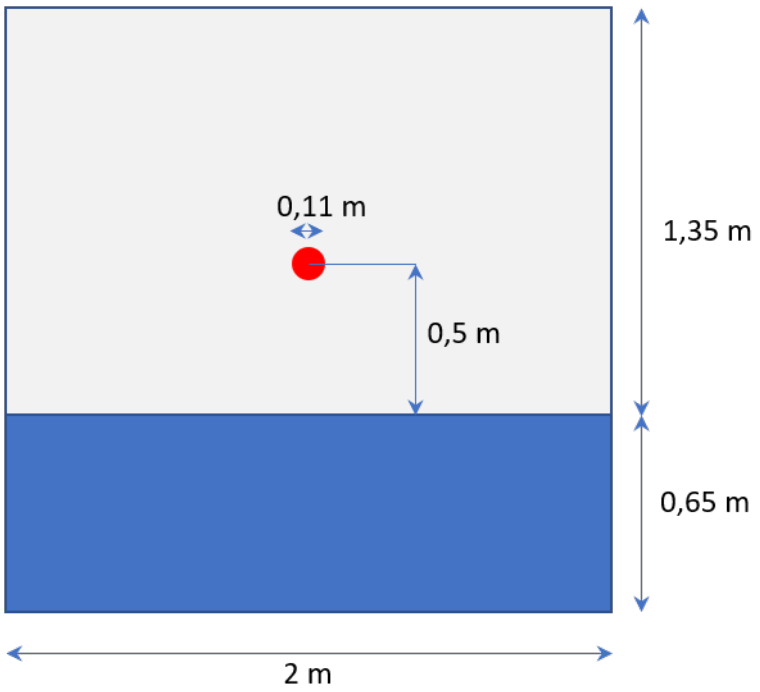


Figure 3.13: Descriptif du cas test : chute d'un cylindre sur un plan d'eau

```
!-----------------------------------------------------------------------
! Data file in CERF format
! Each block of data is characterized in the first line by a header (4 characters)
! then a code (1 integer) and an argument (1 integer)
! PHYS MAME INIT BATH COND MESH OBST NUME
!-----------------------------------------------------------------------

!-----------------------------------------------------------------------
! Reading physical parameters
! Header PHYS
! 1 keyword (4 characters) and a code (1 integer)
! for now we have the same values for all domains
!-----------------------------------------------------------------------
! MODE : physical model
!        1/ Shallow water
!        2/ Euler isothermal two-phases               (default 0.)
!---- for isothermal Euler model
! GPES : Gravity Orientation  0/none, 1/x, 2/y, 3/z   (default 2.)
! PREF : isothermal pressure of reference             (default 1.d5)
! CSSM : Mixture sound velocity (isothermal)          (default 20.)
! REFA : Isothermal reference density for Air         (default 1.)
! REFW : Isothermal reference density for Water       (default 1000.)
! SHAR : isothermal interface sharpening parameter
!        recommended value 0.01                       (defaut 0.)
!---- for Shallow water model
! WLTR : Value to truncate the Water level            (default 0.0001)
! FRMO : Friction Model
! 0/none, 1/Manning-Strickler, 2/Darcy-Weisbach       (default 0.)
! BAGR : Bathymetry gradient 0/ Given , 1/ numerical approximation
!-----------------------------------------------------------------------
PHYS 0
MODE 2.
GPES 2
SHAR 0.01
!-----------------------------------------------------------------------
! Reading the MAster MEsh
! Header  MAME
!-----------------------------------------------------------------------
! code 1 -> creation of a mesh
!
!           argument: 0 type "1D shock tube"
! next line: xmin,xmax
```

```
! next line: nx
!
!           argument: 1 type "2d box"
! next line: xmin,xmax,ymin,ymax
! next line: nx,ny
!
!           argument: 2 type "3d box"
! next line: xmin,xmax,ymin,ymax,zmin,zmax
! next line: nx,ny,nz
!
! code 2 -> Reading an ascii file format GMSH V2.2
! next line: file name
!---------------------------------------------------------------------
MAME    1    1
0. 2.  0. 2.
40 40
!---------------------------------------------------------------------
! Reading Initial conditions
! Header INIT
!---------------------------------------------------------------------
! code 0 -> Definition for a shock tube
!           primitive variables on left
!           primitive variables on right
!           argument: 0
! code 1 -> user function
!           argument: Function's Number
! code 2 -> Definition using areas
!           argument: number of areas
!           and for each area 2 lignes
!           - xmin,xmax,ymin,ymax,zmin,zmax
!           - nvar values ( primitive variables)
! Shallow water model: water level, x velocity, y velocity
! Air-water flow model: density, x velocity, y velocity,z velocity, pressure
!---------------------------------------------------------------------
INIT    2    2
-1. 2.1 -1. 2.1 -1. 1.
1. 0. 0. 0.  1.e5 0.
-1. 2. -1. 0.65 -1. 1.
1000. 0. 0. 0.  1.e5 0.

!---------------------------------------------------------------------
! Reading boundary condition
```

```
! (maximum 10 for the moment)
! Header COND
!---------------------------------------------------------------------
!
! code 0 -> Description in the case  "shock tube" : Nothing
!
! code 1 -> Description in the case  "2d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax
!
! code 2 -> Description in the case  "3d box" (at least 1 line per type) :
!           kind of boundary condition in xmin, xmax, ymin, ymax, zmin,zmax
!
! code 3 -> Definition per zone for  GMSH mesh
!           argument: number of zones defining a boundary condition
!           number of zone, kind of boundary condition
!
! kind of boundary condition:
! 0 outlet (we copy)
! 1 miror
! 2 Dirichlet condition (if 1.e20 then copy), then next line nvar values
! 3 user's function , then next line function number
! 4 Dirichlet condition on conservative variables  (if 1.e20 then copy),
!   then next line nvar values
! 999 Non used
!---------------------------------------------------------------------
COND    1    0
1
1
1
1

!---------------------------------------------------------------------
! Reading mesh parameters in order to define the mesh according to
! the Master mesh is the one represented by the domain "0".
! Header MESH
!---------------------------------------------------------------------
!
! code 0 -> default value
!
! 1 keyword (4 characters) and a real
! NBDS : NumBer of DomainS                          (default 001.)
! MARL : MAximal Refinement Level                   (default 000.)
```

```
! COPA : Mesh COarsening PArameter          0<..<1 (default 0.002)
! REPA : Mesh REfinement PArameter          0<..<1 (default 0.02)
! FDRA : Function defining the mesh refinement to be applied
! NZRA : Number of zones where initial blocks are refined (default 1.)
!        then for each zone
!        nrb,xmin,xmax,ymin,ymax,zmin,zmax
!-----------------------------------------------------------------------
MESH 0
NBDS 10
MARL 4
COPA 1
REPA 2
NZRA 4
0 -1. 2.1 0. 2.1 -1. 1
4 -1. 2.1 0.6 0.7 -1. 1
0 -1. 2.1 0. 0.6 -1. 1
4 0.9 1.1  1.0 1.3  -1. 1

!-----------------------------------------------------------------------
! Reading rigid obstacle parameters
! Header OBST
!-----------------------------------------------------------------------
!
! code 0 -> default value
!           argument: number of obstacles
! 1 line per obstacle: file name, density, kind of motion
! (  -1: fixed, 0: free, >0 : function number imposing the movement
!
!-----------------------------------------------------------------------
OBST    0    1
ex3_bf_cyl.obj 500. 0

!-----------------------------------------------------------------------
! Reading numerical parameter
! Header  NUME
!-----------------------------------------------------------------------
! 1 keyword (4 characters) then a real
! code 0 -> default value
!
! TINT : Time interval to compute                   (default 0.)
! CCFL : cfl condition                              (default 0.9)
! TDOR : Time discretization order 1 ou 2         (defaut 1.)
```

```
! SDOR : Space discretization order 1 ou 2        (defaut 1.)
! LIMI : kind of limiter  0/Barth, 1/Cartesian       (default 0.)
! SAVE : kind of backup, three-digit number          (default 001.)
!        decade  : shock tube backup 0/no 1/yes
!        unit    : binary backup     0/no 1/yes
! NPRO : number of probe  < 10                      (defaut 000.)
!        then coordinates x,y,z of the probe (1 probe per line)
!-----------------------------------------------------------------------
NUME 0
TINT 0.01
CCFL .9
SDOR 2.
TDOR 2.
TIME 1
SAVE 1.
```

Les vitesses attendues sont de l'ordre de quelques mètres par seconde. Donc, comme les blocs ont une taille de $0.05m$, on peut estimer que l'intervalle de temps entre chaque remaillage doit être de $0.01s$. Pour accélérer le calcul on décompose le maillage en 10 domaines (ou encore 10 process Mpi). Pour effectuer un calcul jusqu'à $T = 2s$ on peut utiliser le script suivant :

```
#!/bin/bash
./cerf_input ex3_bf_cyl.inp
./cerf2tec
mv cerfout.dat c_000.dat
mv cerfamr.dat c_mame_000.dat
cp ex3_bf_cyl.obj cs_000.obj

for i in $(seq 1 9 )
do
mpirun -np 10 ./cerf
./cerf2tec
mv cerfout.dat c_00$i.dat
./cerf_amr
mv cerfamr.dat c_mame_00$i.dat
mv solide_001.obj cs_00$i.obj
done
for i in $(seq 10 99 )
do
mpirun -np 10 ./cerf
```

```
./cerf2tec
mv cerfout.dat c_0$i.dat
./cerf_amr
mv cerfamr.dat c_mame_0$i.dat
mv solide_001.obj cs_0$i.obj
done
for i in $(seq 100 200 )
do
mpirun -np 10 ./cerf
./cerf2tec
mv cerfout.dat c_$i.dat
./cerf_amr
mv cerfamr.dat c_mame_$i.dat
mv solide_001.obj cs_$i.obj
done
```

Sur les images ci-après, la densité du mélange est représentée (Bleu pour l'eau et rouge pour l'air). On vérifie que l'interface est "*fine*", c'est-à-dire sur 3-4 cellules et que le maillage dynamique "suit" bien les zones d'intérêt. La première figure Figure 3.14, à $t = 0.3s$, correspond à l'instant où le cylindre touche la surface de l'eau à une vitesse de l'ordre de $3m/s$, ce qui est en accord avec la théorie ($\sqrt{2gh}$) et l'expérience. La seconde figure Figure 3.15, à $t = 0.45s$, illustre la génération de la vague qui se crée lors de l'entrée du cylindre dans l'eau.

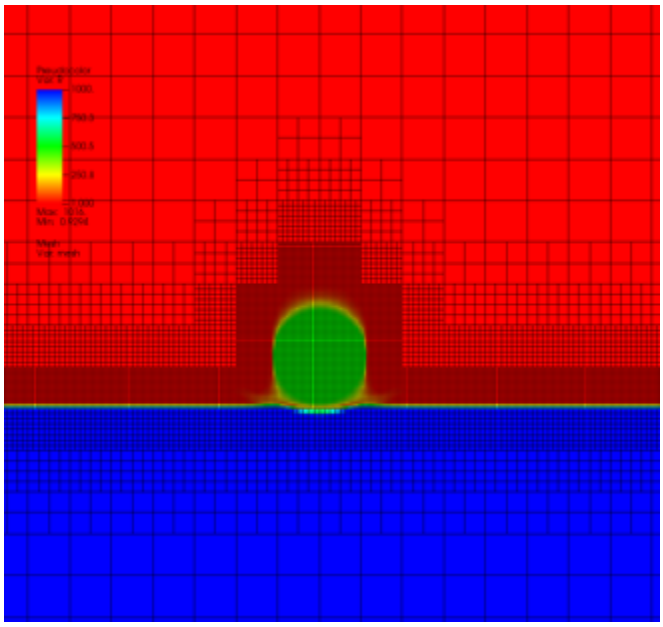


Figure 3.14: Chute d'un cylindre, densité et maillage à $t = 0.3s$

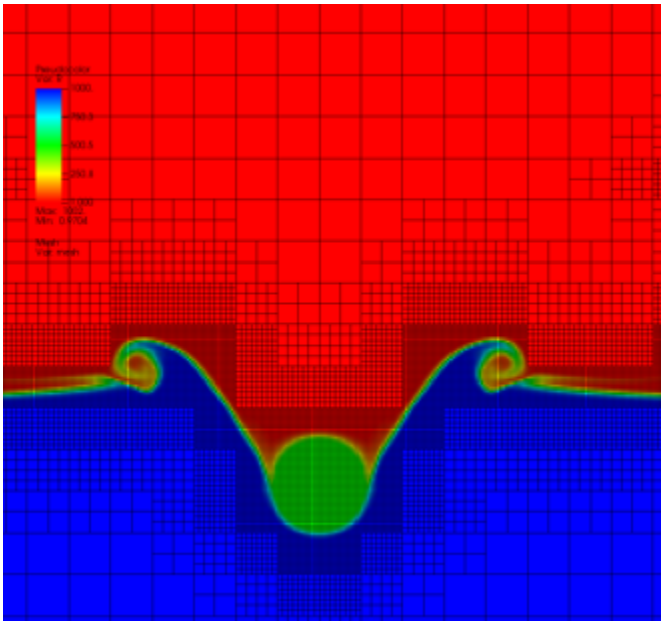


Figure 3.15: Chute d'un cylindre, densité et maillage à $t = 0.45s$

Sur la figure ci-après Figure 3.16, on compare la trajectoire verticale du cylindre au cours du temps avec les résultats expérimentaux de Greenhow and Lin (2015). On constate un bon accord entre les deux approches.

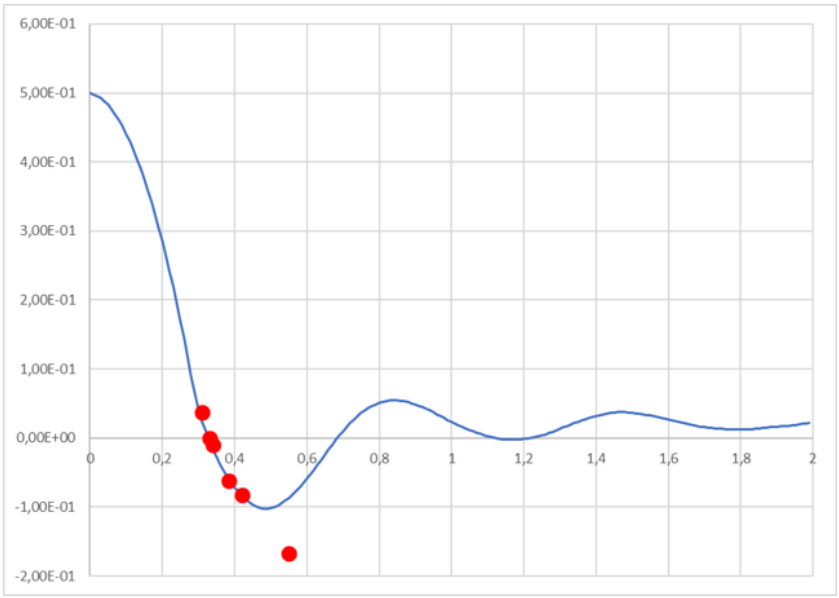


Figure 3.16: Trajectoire verticale du cylindre au cours du temps: comparaison de la simulation numérique (trait plein) avec l'expérience (point)

Références

Altazin, Thomas. 2017. “Un Modèle d’interaction Fluide-Structure En Régime Compressible Faible Mach.” PhD thesis. http://www.theses.fr/2017TOUL0021/document.

Audusse, Emmanuel, François Bouchut, Marie-Odile Bristeau, Rupert Klein, and Benoı t Perthame. 2004. “A Fast and Stable Well-Balanced Scheme with Hydrostatic Reconstruction for Shallow Water Flows.” *SIAM Journal on Scientific Computing* 25 (6): 2050–65. https://doi.org/10.1137/S1064827503431090.

Chorin, Alexandre Joel. 1967. “A Numerical Method for Solving Incompressible Viscous Flow Problems.” *Journal of Computational Physics* 2 (1): 12–26. https://doi.org/https://doi.org/10.1016/0021-9991(67)90037-X.

Coquerelle, M., and G.-H. Cottet. 2008. “A Vortex Level Set Method for the Two-Way Coupling of an Incompressible Fluid with Colliding Rigid Bodies.” *Journal of Computational Physics* 227 (21): 9121–37. https://doi.org/https://doi.org/10.1016/j.jcp.2008.03.041.

Croisille, Jean-Pierre. 1990. “Contribution à l’étude Théorique Et à l’approximation Par Éléments Finis Du Système Hyperbolique de La Dynamique Des Gaz Multidimensionnelle Et Multiespèces.” PhD thesis.http://www.theses.fr/1990PA066095.

Delestre, Olivier, Carine Lucas, Pierre-Antoine Ksinant, Frédéric Darboux, Christian Laguerre, T.-N.-Tuoi Vo, François James, and Stéphane Cordier. 2013. “SWASHES: A Compilation of Shallow Water Analytic Solutions for Hydraulic and Environmental Studies.” *International Journal for Numerical Methods in Fluids* 72 (3): 269–300. https://doi.org/https://doi.org/10.1002/fld.3741.

Ersoy, Mehmet, Frédéric Golay, and Lyudmyla Yushchenko. 2013. *Open Mathematics* 11 (8): 1392–1415. https://doi.org/doi:10.2478/s11533-013-0252-6.

Faccanoni, Gloria, and Anne Mangeney. 2013. “Exact Solution for Granular Flows.” *International Journal for Numerical and Analytical Methods in Geomechanics* 37 (10): 1408–33. https://doi.org/https://doi.org/10.1002/nag.2124.

Golay, Frederic. 2018. “Monolithic Fluid Structure Interaction Model, Application to Water Entry Problem.” In *Topical Problems of Fluid Mechanics 2018*. Prague, Czech Republic: Institute of Thermomechanics AS CR v.v.i. https://doi.org/10.14311/TPFM.2018.016.

Golay, Frédéric. 2009. “Numerical Entropy Production and Error Indicator for Compressible Flows.” *Comptes Rendus. Mécanique* 337 (4): 233–37. https://doi.org/10.1016/j.crme.2009.04.004.

Golay, Frédéric, and Philippe Helluy. 2007. “Numerical Schemes for Low Mach Wave Breaking.” *International Journal of Computational Fluid Dynamics* 21 (2): 69–86. https://doi.org/10.1080/10618560701343382.

Greenhow, Martin, and and Lin. 2015. "Greenhow m. And Lin w-m."nonlinear Free Surface Effects: Experiments and Theory" MIT Internal Report 83-19 (1983)," June. https://doi.org/10.13140/RG.2.1.4129.2964.

Helluy, Philippe, Golay, Frédéric, Caltagirone, Jean-Paul, Lubin, Pierre, Vincent, Stéphane, Drevard, Deborah, Marcer, Richard, et al. 2005. "Numerical Simulations of Wave Breaking." *ESAIM: M2AN* 39 (3): 591–607. https://doi.org/10.1051/m2an:2005024.

Lax, Peter. 1971. "Shock Waves and Entropy." In *Contributions to Nonlinear Functional Analysis*, edited by Eduardo H. Zarantonello, 603–34. Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-12-775850-3.50018-2.

Marche, Fabien. 2007. "Derivation of a New Two-Dimensional Viscous Shallow Water Model with Varying Topography, Bottom Friction and Capillary Effects." *European Journal of Mechanics - B/Fluids* 26 (1): 49–63. https://doi.org/https://doi.org/10.1016/j.euromechflu.2006.04.007.

Martin, J. C., and W. J. Moyce. 1952. "Part IV. An Experimental Study of the Collapse of Liquid Columns on a Rigid Horizontal Plane." *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 244 (882): 312–24. http://www.jstor.org/stable/91519.

Pons, Kevin. 2018. "Dynamique Des Écoulements à Surface Libre Dans Les Plages Sableuses." Thèse de doctorat, IMATH, Université de Toulon: Université de Toulon.

Poussel, Camille. 2024. "Dynamique Des Écoulements à Surface Libre Dans Les Plages Sableuses." Thèse de doctorat, IMATH, Université de Toulon: Université de Toulon.

Puppo, Gabriella. 2004. "Numerical Entropy Production for Central Schemes." *SIAM Journal on Scientific Computing* 25 (4): 1382–1415. https://doi.org/10.1137/S1064827502386712.

Saint-Venant, Barré de. 1871. "AJC (1871b)." Théorie Et Equations générales Du Mouvement Non Permanent Des Eaux, Avec Application Aux Crues Des Rivières Et à l'introduction Des Marées Dans Leur Lit (2ème Note)."." *Comptes Rendus Des séances de l'Académie Des Sciences*, 237–40.

Sun, Hui, and Odd M. Faltinsen. 2006. "Water Impact of Horizontal Circular Cylinders and Cylindrical Shells." *Applied Ocean Research* 28 (5): 299–311. https://doi.org/https://doi.org/10.1016/j.apor.2007.02.002.

Sun, Pengnan, Furen Ming, and Aman Zhang. 2015. "Numerical Simulation of Interactions Between Free Surface and Rigid Body Using a Robust SPH Method." *Ocean Engineering* 98: 32–49. https://doi.org/https://doi.org/10.1016/j.oceaneng.2015.01.019.

Thomas Altazin, Frédéric Golay, Mehmet Ersoy, and Lyudmyla Yushchenko. 2016. "Numerical Investigation of BB-AMR Scheme Using Entropy Production as Refinement Criterion." *International Journal of Computational Fluid Dynamics* 30 (3): 256–71. https://doi.org/10.1080/10618562.2016.1194977.

Toro, E. F. 1997. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer. https://books.google.fr/books?id=6QFAAQAAIAAJ.

Varra, Giada, Veronica Pepe, Renata Della Morte, and Luca Cozzolino. 2024. "A Novel Efficient and Robust Treatment of the Friction Source Term in 2D Shallow Water Inundation Models." *Journal of Hydrology* 634: 131045. https://doi.org/10.1016/j.jhydrol.2024.131045.