# Rapport de stage:

## Etude de méthodes spectrales discontinues pour le calcul haute performance

**Organisme d'accueil:** Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique (CERFACS) - Equipe AAM

**Encadrants:** Guillaume Daviller et Nadir-Alexandre Messaï
**Référent universitaire:** Mehdi Badsi

**Marie Compain**
M2 MACS - Nantes Université

du 2 Avril au 30 Septembre 2024

**Résumé**

Ce rapport de stage présente une adaptation de la méthode Face-Upwinded Spectral Element (FUSE) [8] à la méthode des Différences Spectrales (SD) pour la résolution de lois de conservation hyperboliques. Cette nouvelle méthode est décrite pour des cas scalaires linéaires et non linéaires en 1D, et des cas linéaires et de systèmes en 2D. De plus, une preuve de stabilité des deux méthodes est présentée pour un maillage hexahédrique 2D. La méthode FUSE est ensuite validée sur plusieurs cas tests et comparée à la méthode SD. Il est observé que les erreurs $L^2$ commises pour les deux méthodes sont du même ordre de grandeur. Enfin, une analyse de la complexité algorithmique des deux méthodes montrent que la méthode FUSE requiert moins d'opérations que la méthode SD standard. Ceci est validé par des tests 1D et 2D sur coeurs CPU et cartes GPU, qui montrent que le temps de calcul de la méthode FUSE est plus bas que celui de la méthode.

# Table des matières

## Remerciements

Je tiens à remercier tout d'abord Guillaume et Nadir, pour m'avoir si bien accueillie, aidée et encadrée tout au long de ce stage. Un grand merci aussi à Alexandre, qui était toujours là pour répondre à mes questions sur HOPPS et m'aider à débuguer le code, ainsi qu'aux autres collègues qui ont été très accueillants et très sympathiques durant ces six mois : merci Etienne, Thomas, Fabien, Félicia et Jules.

Je remercie aussi l'équipe pédagogique du master MACS de Nantes Université, pour leur bienveillance, leur pédagogie et leur écoute pendant ces deux années.

Enfin je voulais remercier les amis que j'ai rencontré pendant mes études, pour leur soutien et pour avoir toujours été là, dans les bons comme dans les mauvais moments. Un grand merci, entre autres, à Quentin, Hugo, Solenne, Imène et Matthieu.

# 1 Introduction

Les simulations numériques jouent un rôle essentiel en mécanique des fluides, autant dans la recherche que dans la conception industrielle. En effet, les essais peuvent être coûteux en terme de matériel, de temps et de main d'oeuvre. L'alternative de la simulation numérique est plus fiable, plus rapide et plus économique. De plus, elle permet d'accéder à des grandeurs inaccessibles à l'expérience. A titre d'illustration, citons la caractérisation des mouvements de très petite échelle caractéristiques de la turbulence.

La mécanique des fluides numérique (CFD pour Computational Fluid Dynamics) est un des axes de recherche principaux du CERFACS. Les phénomènes physiques à simuler en CFD étant complexes, la taille des problèmes à résoudre nécessite souvent d'utiliser des supercalculateurs. Ces derniers sont de plus en plus hybrides, c'est-à-dire composés de noeuds avec coeurs CPU mais aussi de cartes GPU. Et parmi ces cartes, plusieurs types existent. C'est pourquoi il est essentiel d'adapter les codes CFD à ces nouvelles architectures. C'est dans ce contexte que le CERFACS a décidé de développer un code de calcul CFD haute performance, nommé HOPPS, dont la parallélisation dépend de Kokkos [10]. Cette librairie C++ permet en effet d'automatiser la portabilité de HOPPS sur différentes architectures matérielles.

Depuis une vingtaine d'années, les méthodes numériques d'ordre élevé ont émergé en CFD comme une alternative aux méthodes utilisées jusque-là. En effet, les méthodes d'ordre élevé permettent de gagner en précision à un coût de calcul moins important que les méthodes volumes finis classiques. Elles ont aussi des niveaux de dissipation numérique beaucoup plus faibles. Parmi les méthodes d'ordre élevé les plus connues, nous pouvons citer la méthode Galerkin Discontinu (DG) [2]. Cependant, bien qu'elle soit précise, son coût de calcul est en général assez grand. C'est pourquoi des alternatives à cette méthode ont été construites. Nous avons par exemple la méthode Reconstruction par Flux (FR) [3], qui est plus rapide que la méthode DG mais renvoie des solutions moins précises, ou bien la méthode des Différences Spectrales (SD) [5], qui se trouve à mi-chemin entre les méthodes FR et DG en terme de coût de calcul et de précision [11]. Le choix du CERFACS s'est donc porté sur cette méthode numérique pour ses solveurs CFD.

Soit la loi de conservation suivante :

$$\partial_t \mathbf{U} + \nabla.\mathbf{f}(\mathbf{U}) = 0 \text{ sur } \Omega \times [0, T], \tag{1.1}$$

avec $\Omega \in \mathbb{R}^d$, $d = 1, 2, 3$, le domaine physique, $T \in \mathbb{R}_+^*$, $\mathbf{U} = (U_i)_{1 \leq i \leq N_{eq}}$ le vecteur des inconnues, où $N_{eq}$ est le nombre d'inconnues du système, et $\mathbf{f}(\mathbf{U})$ le flux. Il peut s'agir, par exemple, des

équations d'Euler ou des équations de Navier-Stokes.

La méthode SD approche chaque composante du vecteur solution $\mathbf{U}$ par un polynôme d'ordre $p$ en utilisant l'interpolation de Lagrange. Ainsi, pour définir ce polynôme, il nous faut la valeur de $U_i$ à $p+1$ points. Ces points seront appelés les points solution. Nous aurons aussi besoin d'approcher le flux $\mathbf{f}(\mathbf{U})$ par un polynôme, mais cette fois d'ordre $p+1$ pour être consistant avec l'ordre de $\mathbf{U}$. Il nous faudra alors $p+2$ points d'interpolation, appelés points flux. Les points solution et les points flux n'étant usuellement pas collocalisés, la méthode SD utilise une étape d'extrapolation afin d'exprimer $\mathbf{U}$ aux points flux.

Or il a été démontré par Jameson [4] que la stabilité de la méthode SD ne dépendait pas de la localisation des points solution. Il a donc été proposé par Pan et al. [8] de collocaliser les points solution avec certains points flux.

L'objectif de ce stage est de caractériser les possibilités que nous offre cette nouvelle méthode, nommée Face-Upwinded Spectral Element (FUSE), en terme de stabilité et de coût de calcul, en faisant une étude théorique de stabilité et en implémentant par la suite cette méthode numérique dans HOPPS.

Ce rapport de stage se décompose comme suit : dans la section 2, nous rappellerons le principe de la méthode SD standard en 1D puis en 2D et 3D pour des maillages hexahédriques. La partie 3 décrira la méthode FUSE et son équivalence avec la méthode SD, ainsi que l'extension de cette méthode à des problèmes non linéaires. Dans la section 4, une preuve de stabilité en 2D sera présentée pour un maillage hexahédrique de la méthode SD et de la méthode FUSE étendue aux SD. Ensuite, dans la section 5, nous nous intéresserons aux résultats numériques obtenus. Enfin, la partie 6 concernera l'analyse et la comparaison de la performance des deux méthodes.

# 2 Différences spectrales

Nous détaillerons dans cette partie la méthode SD en 1D puis en 2D et 3D expliquée par Marchal [6].

## 2.1 Méthode en 1D

### 2.1.1 Notations et discrétisation spatiale

Considérons un système d'équations hyperboliques en 1D :

$$\frac{\partial \mathbf{U}}{\partial t}(x,t) + \frac{\partial \mathbf{E}}{\partial x}(x,t) = 0, \ \forall (x,t) \in \Omega \times [0,T], \tag{2.1}$$

où $\Omega = [a,b]$, $a < b \in \mathbb{R}$, $T \in \mathbb{R}_+^*$, $\mathbf{U}$ est le vecteur solution et $\mathbf{E} = \mathbf{E}(\mathbf{U})$ est le vecteur flux de $\mathbf{U}$.

Nous discrétisons l'espace $\Omega$ en $N_e$ éléments. La méthode des Différences Spectrales repose sur le fait que nous résolvons l'équation (2.1) sur le segment $[0,1]$, appelé domaine isoparamétrique et dont la variable sera notée $\xi \in [0,1]$. Nous appliquerons donc une transformation $\mathcal{F}_e$ de l'espace isoparamétrique vers la cellule $e \in [\![1, N_e]\!]$ du maillage. La jacobienne de cette transformation sera notée $J$. En appliquant cette transformée, nous obtenons, pour chaque cellule, le système dans l'espace isoparamétrique :

$$\frac{\partial \hat{\mathbf{U}}}{\partial t} + \frac{\partial \hat{\mathbf{E}}}{\partial \xi} = 0, \tag{2.2}$$

avec $\hat{\mathbf{U}} = |J|\mathbf{U}$ et $\hat{\mathbf{E}} = |J|\xi_x \mathbf{E}$, où $\xi_x = \dfrac{\partial \xi}{\partial x}$.

Dans cet espace isoparamétrique, nous supposons que $\hat{\mathbf{U}}$ est un polynôme (par rapport à $\xi$) de degré $p$. Nous avons donc besoin de connaître $\hat{\mathbf{U}}$ sur $p+1$ points de $[0,1]$. Ces points seront appelés points solution et leur nombre sera noté :

$$N_{SP} = p + 1. \tag{2.3}$$

La stabilité de la méthode ne dépendant pas de la localisation des points solution, comme prouvé dans l'article [4], nous utilisons usuellement les $p+1$ points de Gauss-Tchebychev :

$$\boldsymbol{\xi}_{SP} = [\xi_{SP}(i)]_{1 \leq i \leq N_{SP}} = \left[ \frac{1}{2} \left( 1 - \cos\left( \frac{2i-1}{2N_{SP}} \pi \right) \right) \right]_{1 \leq i \leq N_{SP}}. \tag{2.4}$$

Pour la discrétisation du flux, il faut que $\frac{\partial \hat{\mathbf{E}}}{\partial \xi}$ soit consistant avec $\frac{\partial \hat{\mathbf{U}}}{\partial t}$, ce dernier étant de degré

$p$ en $\xi$. Donc $\hat{\mathbf{E}}$ sera un polynôme de degré $p+1$, et il nous faut par conséquent $p+2$ points d'interpolation. Nous les appellerons points flux et leur nombre sera noté :

$$N_{FP} = p + 2. \tag{2.5}$$

Les $p$ points flux intérieurs seront les points de Gauss-Legendre, et les deux points restants seront les extrémités du segment isoparamétrique, soit $\xi_{FP}(1) = 0$ et $\xi_{FP}(N_{FP}) = 1$. Cette distribution de points est utilisée car elle ne rend pas le schéma instable, comme démontré dans [4].

### 2.1.2 Interpolation et extrapolation

Pour interpoler et extrapoler $\hat{\mathbf{U}}$ et $\hat{\mathbf{E}}$, nous utiliserons l'interpolation de Lagrange. Soit la base de polynômes de Lagrange de degré $p$ associés aux points solution et évalués au point $\xi$ :

$$\mathbf{L}_{SP}(\xi) = [L_{i,SP}(\xi)]_{1\leq i\leq N_{SP}} = \left[\prod_{k=1,k\neq i}^{N_{SP}} \frac{\xi - \xi_{SP}(k)}{\xi_{SP}(i) - \xi_{SP}(k)}\right]_{1\leq i\leq N_{SP}}. \tag{2.6}$$

Nous pouvons, de façon analogue, définir une base de polynômes de Lagrange de degré $p+1$ associés aux points flux :

$$\mathbf{L}_{FP}(\xi) = [L_{i,FP}(\xi)]_{1\leq i\leq N_{FP}} = \left[\prod_{k=1,k\neq i}^{N_{FP}} \frac{\xi - \xi_{FP}(k)}{\xi_{FP}(i) - \xi_{FP}(k)}\right]_{1\leq i\leq N_{FP}}. \tag{2.7}$$

Alors nous pouvons exprimer $\hat{\mathbf{U}}$ (respectivement $\hat{\mathbf{E}}$) dans la base (2.6) (resp. dans la base (2.7)).

Nous aurons aussi besoin de dériver $\hat{\mathbf{E}}$ aux points solution, donc de la dérivée des polynômes $L_{i,FP}$ par rapport à $\xi$, dont l'expression est donnée par :

$$\frac{\partial L_{i,FP}}{\partial \xi}(\xi) = \frac{\displaystyle\sum_{k=1,k\neq i}^{N_{FP}} \prod_{m=1,m\neq k}^{N_{FP}} (\xi - \xi_{FP}(m))}{\displaystyle\prod_{k=1,k\neq i}^{N_{FP}} (\xi_{FP}(i) - \xi_{FP}(k))}, \ \forall i \in [\![1, N_{FP}]\!]. \tag{2.8}$$

### 2.1.3 Algorithme de résolution numérique

Nous commençons par initialiser le maillage, $\xi_x$, la jacobienne $|J|$, les vecteurs de points $\boldsymbol{\xi}_{SP}$ et $\boldsymbol{\xi}_{FP}$. Nous initialisons aussi $\hat{\mathbf{U}}_{SP} = |J|\mathbf{U}_{SP}$, le tenseur des solutions dans l'espace isoparamétrique évaluées aux points solution de tous les éléments $e \in [\![1, N_e]\!]$ du maillage. $\mathbf{U}_{SP}$ correspond au tenseur des solutions dans le domaine physique évaluées aux points solution de tous les éléments

$e \in [\![1, N_e]\!]$ du maillage. $\mathbf{U}_{FP}$ et $\hat{\mathbf{U}}_{FP}$ sont définis de manière analogue pour les points flux.

Soit un élément $e$ du maillage. A un instant donné $0 \leq t_n \leq T$, le vecteur solution aux points solution $\hat{\mathbf{U}}^e_{SP}$ est connu. Nous pouvons donc l'extrapoler aux points flux tel que :

$$\hat{\mathbf{U}}^e_{FP} = \left[\overline{\mathbf{U}}^e(\boldsymbol{\xi}_{FP}(i))\right]_{1\leq i\leq N_{FP}}, \tag{2.9}$$

où :

$$\overline{\mathbf{U}}^e(\xi) = \sum_{i=1}^{N_{SP}} \hat{\mathbf{U}}^e_{SP}(i) L_{i,SP}(\xi). \tag{2.10}$$

Nous pouvons donc évaluer le vecteur flux $\hat{\mathbf{E}}^e_{FP}$ aux points flux. Nous commençons par calculer les éléments intérieurs de ce vecteur :

$$\left[\hat{\mathbf{E}}^e_{FP}(i)\right]_{2\leq i\leq N_{FP}-1} = \left[\hat{\mathbf{E}}\left(\overline{\mathbf{U}}^e(\xi_{FP}(i))\right)\right]_{2\leq i\leq N_{FP}-1}. \tag{2.11}$$

Puis, pour assurer la continuité du flux sur tout le domaine physique, un solveur de Riemann est utilisé à chaque interface entre deux éléments. Après cela, le flux est défini de façon unique sur tous les points flux. Nous pouvons donc dériver $\hat{\mathbf{E}}^e_{FP}$ aux points solution en calculant :

$$\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} = \left[\sum_{k=1}^{N_{FP}} \hat{\mathbf{E}}^e_{FP}(k) \frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i))\right]_{1\leq i\leq N_{SP}}. \tag{2.12}$$

Enfin, nous récupérons $\hat{\mathbf{U}}^e_{SP}$ au temps suivant $t_{n+1}$ en résolvant numériquement (avec n'importe quel schéma d'intégration temporel explicite) l'équation :

$$\frac{d\hat{\mathbf{U}}^e_{SP}}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP}. \tag{2.13}$$

Si $\mathbf{E}$ est une fonction de $\mathbf{U}$ mais aussi de $\nabla\mathbf{U}$ (par exemple dans les équations de Navier-Stokes), nous avons besoin de calculer le gradient de $\mathbf{U}$ aux points flux. Comme pour le flux, $\hat{\mathbf{U}}^e_{FP}$ n'est pas défini de façon unique aux interfaces. Il existe plusieurs méthodes pour obtenir la continuité de $\mathbf{U}$, Sun et al. [9] proposent la méthode suivante : aux interfaces, nous utilisons la moyenne arithmétique des deux valeurs pour chaque solution $U^i = \frac{U^i_L + U^i_R}{2}$, $1 \leq i \leq N_{eq}$, où $N_{eq}$ est le nombre de variables conservatives dans le système, $U^i$ la $i$ème variable, et $U^i_L$ (respectivement $U^i_R$) l'état de $U^i$ à gauche (respectivement à droite) de l'interface. Nous évaluons ensuite la dérivée de $\hat{\mathbf{U}}$ aux points solution :

$$\left(\frac{\partial \hat{\mathbf{U}}}{\partial \xi}\right)^e_{SP} = \left[\sum_{k=1}^{N_{FP}} \hat{\mathbf{U}}^e_{FP}(k) \frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i))\right]_{1\leq i\leq N_{SP}}. \tag{2.14}$$

Cependant pour calculer le flux il nous faut l'évaluation de ce gradient aux points flux :

$$\left(\frac{\partial \hat{\mathbf{U}}}{\partial \xi}\right)_{FP}^{e} = \left[\sum_{k=1}^{N_{SP}} \left(\frac{\partial \hat{\mathbf{U}}}{\partial \xi}\right)_{k,SP}^{e} L_{k,SP}(\xi_{FP}(i))\right]_{1\leq i\leq N_{FP}}. \tag{2.15}$$

Cette quantité n'est pas continue aux interfaces. Nous utilisons ici encore la moyenne arithmétique à chaque interface : $\frac{\partial U^i}{\partial \xi} = \frac{\frac{\partial U_L^i}{\partial \xi} + \frac{\partial U_R^i}{\partial \xi}}{2}$. Ainsi, nous pouvons maintenant évaluer le flux aux points flux.

## 2.2 Méthode en 2D/3D

Nous pouvons étendre la méthode présentée dans la partie 2.1 à des maillages hexahédriques en 2D et 3D.

### 2.2.1 Notations et discrétisation spatiale

Nous considérons ici le système d'équations :

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla.\mathbf{f}(\mathbf{U}) = 0 \text{ sur } \Omega \times [0,T], \tag{2.16}$$

avec $\Omega \subset \mathbb{R}^d$, $d = 2,3$ la dimension de l'espace physique, $T \in \mathbb{R}_+^*$, $\mathbf{U}$ le vecteur des variables conservatives et $\mathbf{f}$ le flux. Nous pouvons réécrire l'équation (2.16) dans le cas 3D (le cas 2D est analogue) sous la forme :

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{F}}{\partial y} + \frac{\partial \mathbf{G}}{\partial z} = 0, \tag{2.17}$$

avec $\mathbf{E}$ (respectivement $\mathbf{F}$ et $\mathbf{G}$) le flux dans la direction $x$ (respectivement directions $y$ et $z$).

Comme en 1D nous souhaitons passer du domaine physique au domaine isoparamétrique. Pour un maillage hexahédrique, il s'agit du domaine $[0,1]^d$. La transformation du domaine isoparamétrique au domaine physique s'écrit :

$$x(\xi,\eta,\zeta) = \sum_{i=1}^{k} M_i(\xi,\eta,\zeta)x_i^e, \tag{2.18}$$

$$y(\xi,\eta,\zeta) = \sum_{i=1}^{k} M_i(\xi,\eta,\zeta)y_i^e, \tag{2.19}$$

$$z(\xi,\eta,\zeta) = \sum_{i=1}^{k} M_i(\xi,\eta,\zeta)z_i^e, \tag{2.20}$$

avec $k$ le nombre de points utilisés pour définir l'élément $e$ (par exemple 8 pour définir un hexahèdre en 3D), $(x_i^e, y_i^e, z_i^e)$ les coordonnées physiques des points utilisés pour définir $e$, et $M_i$ définie pour les SD telle que :

$$M_i(\xi,\eta,\zeta) = L_{\xi_i,e}(\xi)L_{\eta_i,e}(\eta)L_{\zeta_i,e}(\zeta), \tag{2.21}$$

où $L_{\xi_i,e}$ (respectivement $L_{\eta_i,e}$ et $L_{\zeta_i,e}$) est le polynôme de Lagrange 1D (défini dans la partie 2.1.2) au point $\xi_i$ (resp. $\eta_i$ et $\zeta_i$) utilisant les autres coordonnées $\xi_k$ (resp. $\eta_k$ et $\zeta_k$), $k \neq i$, qui définissent l'élément $e$, avec $(\xi_i, \eta_i, \zeta_i)$ étant les coordonnées isoparamétriques du point $(x_i^e, y_i^e, z_i^e)$.

La preuve du passage en coordonnées isoparamétriques est détaillée dans le rapport [6]. Par conséquent nous obtenons le système dans l'espace isoparamétrique sous forme conservative suivant :

$$\frac{\partial \hat{\mathbf{U}}}{\partial t} + \frac{\partial \hat{\mathbf{E}}}{\partial \xi} + \frac{\partial \hat{\mathbf{F}}}{\partial \eta} + \frac{\partial \hat{\mathbf{G}}}{\partial \zeta} = 0, \tag{2.22}$$

où $\hat{\mathbf{U}} = |J|\mathbf{U}$, $\hat{\mathbf{E}} = |J|(\xi_x\mathbf{E} + \xi_y\mathbf{F} + \xi_z\mathbf{G})$, $\hat{\mathbf{F}} = |J|(\eta_x\mathbf{E} + \eta_y\mathbf{F} + \eta_z\mathbf{G})$ et $\hat{\mathbf{G}} = |J|(\zeta_x\mathbf{E} + \zeta_y\mathbf{F} + \zeta_z\mathbf{G})$.

Pour généraliser la méthode en 2D ou 3D, nous répétons le même processus qu'en 1D pour chaque dimension. Donc nous aurons dans un espace de dimension $d$, le nombre de points solution suivant pour un élément de degré $p$ :

$$N_{SP} = (p+1)^d. \tag{2.23}$$

Nous prenons cette fois encore pour chaque direction les points de Gauss-Tchebychev. L'expression d'un point solution sera donc :

$$(\xi_{SP}^i, \eta_{SP}^j, \zeta_{SP}^k) = (\xi_{SP}(i), \xi_{SP}(j), \xi_{SP}(k)). \tag{2.24}$$

De plus, il y aura un total de $N_{FP}$ points flux, avec :

$$N_{FP} = d(p+2)(p+1)^{d-1}, \tag{2.25}$$

et dans chaque direction $(p+2)(p+1)^{d-1}$ points.

Soit les points flux dans la direction $\xi$. Nous prendrons pour la coordonnée pour cette direction la distribution de points flux utilisée en 1D définie dans la partie 2.1.1. Pour les deux autres directions $\eta$ et $\zeta$, nous utiliserons les points solution 1D. Nous obtenons donc pour un point flux dans la direction $\xi$ l'expression suivante :

$$(\xi^i_{FP}, \eta^j_{FP}, \zeta^k_{FP})_\xi = (\xi_{FP}(i), \xi_{SP}(j), \xi_{SP}(k)), (i,j,k) \in [\![1, p+2]\!] \times [\![1, p+1]\!]^{d-1}. \tag{2.26}$$

De manière analogue, les points flux dans les directions $\eta$ et $\zeta$ seront :

$$(\xi^i_{FP}, \eta^j_{FP}, \zeta^k_{FP})_\eta = (\xi_{SP}(i), \xi_{FP}(j), \xi_{SP}(k)), (i,j,k) \in [\![1, p+1]\!] \times [\![1, p+2]\!] \times [\![1, p+1]\!], \tag{2.27}$$

$$(\xi^i_{FP}, \eta^j_{FP}, \zeta^k_{FP})_\zeta = (\xi_{SP}(i), \xi_{SP}(j), \xi_{FP}(k)), (i,j,k) \in [\![1, p+1]\!]^{d-1} \times [\![1, p+2]\!]. \tag{2.28}$$

### 2.2.2 Algorithme de résolution numérique

Soit un temps donné $0 \leq t_n \leq T$ et un élément du maillage $e$. Nous connaissons le vecteur solution aux points solution $\hat{\mathbf{U}}^e_{SP} = \left[\hat{\mathbf{U}}^e(i,j,k)\right]_{1 \leq i,j,k \leq p+1}$. Nous pouvons donc l'extrapoler aux points flux dans chaque direction :

$$\hat{\mathbf{U}}^e_{FP,\xi} = \left[\bar{\mathbf{U}}^e(\xi_{FP}(i), \xi_{SP}(j), \xi_{SP}(k))\right]_{\substack{1 \leq i \leq p+2 \\ 1 \leq j,k \leq p+1}}, \tag{2.29}$$

$$\hat{\mathbf{U}}^e_{FP,\eta} = \left[\bar{\mathbf{U}}^e(\xi_{SP}(i), \xi_{FP}(j), \xi_{SP}(k))\right]_{\substack{1 \leq j \leq p+2 \\ 1 \leq i,k \leq p+1}}, \tag{2.30}$$

$$\hat{\mathbf{U}}^e_{FP,\zeta} = \left[\bar{\mathbf{U}}^e(\xi_{SP}(i), \xi_{SP}(j), \xi_{FP}(k))\right]_{\substack{1 \leq k \leq p+2 \\ 1 \leq i,j \leq p+1}}, \tag{2.31}$$

où :

$$\bar{\mathbf{U}}^e(\xi,\eta,\zeta) = \sum_{i=1}^{p+1}\sum_{j=1}^{p+1}\sum_{k=1}^{p+1} \hat{\mathbf{U}}^e_{SP}(i,j,k) L_{i,SP}(\xi) L_{j,SP}(\eta) L_{k,SP}(\zeta). \tag{2.32}$$

Nous pouvons donc maintenant évaluer les vecteurs flux dans toutes les directions aux points

flux intérieurs :

$$\left[\hat{\mathbf{E}}^e_{FP}(i,j,k)\right]_{\substack{2\leq i\leq N_{FP}-1\\1\leq j,k\leq p+1}} = \left[\hat{\mathbf{E}}(\hat{\mathbf{U}}^e_{FP}(i,j,k))\right]_{\substack{2\leq i\leq N_{FP}-1\\1\leq j,k\leq p+1}}, \tag{2.33}$$

$$\left[\hat{\mathbf{F}}^e_{FP}(i,j,k)\right]_{\substack{2\leq j\leq N_{FP}-1\\1\leq i,k\leq p+1}} = \left[\hat{\mathbf{F}}(\hat{\mathbf{U}}^e_{FP}(i,j,k))\right]_{\substack{2\leq j\leq N_{FP}-1\\1\leq i,k\leq p+1}}, \tag{2.34}$$

$$\left[\hat{\mathbf{G}}^e_{FP}(i,j,k)\right]_{\substack{2\leq k\leq N_{FP}-1\\1\leq i,j\leq p+1}} = \left[\hat{\mathbf{G}}(\hat{\mathbf{U}}^e_{FP}(i,j,k))\right]_{\substack{2\leq k\leq N_{FP}-1\\1\leq i,j\leq p+1}}. \tag{2.35}$$

Comme en 1D, nous résolvons un problème de Riemann sur les bords de l'élément pour avoir une valeur unique du flux aux interfaces. Ainsi nous pouvons maintenant dériver $\hat{\mathbf{E}}$ (respectivement $\hat{\mathbf{F}}$ et $\hat{\mathbf{G}}$) par rapport à $\xi$ (resp. $\eta$ et $\zeta$) aux points solution :

$$\left(\frac{\partial \hat{E}}{\partial \xi}\right)^e_{SP} = \left[\sum_{l=1}^{p+2}\sum_{m=1}^{p+1}\sum_{n=1}^{p+1} \hat{E}^e_{FP}(l,m,n)\frac{\partial L_{l,FP}}{\partial \xi}(\xi^i_{SP})L_{m,SP}(\eta^j_{SP})L_{n,SP}(\zeta^k_{SP})\right]_{1\leq i,j,k\leq p+1}, \tag{2.36}$$

$$\left(\frac{\partial \hat{F}}{\partial \eta}\right)^e_{SP} = \left[\sum_{l=1}^{p+1}\sum_{m=1}^{p+2}\sum_{n=1}^{p+1} \hat{F}^e_{FP}(l,m,n)L_{l,SP}(\xi^i_{SP})\frac{\partial L_{m,FP}}{\partial \eta}(\eta^j_{SP})L_{n,SP}(\zeta^k_{SP})\right]_{1\leq i,j,k\leq p+1}, \tag{2.37}$$

$$\left(\frac{\partial \hat{G}}{\partial \zeta}\right)^e_{SP} = \left[\sum_{l=1}^{p+1}\sum_{m=1}^{p+1}\sum_{n=1}^{p+2} \hat{G}^e_{FP}(l,m,n)L_{l,SP}(\xi^i_{SP})L_{m,SP}(\eta^j_{SP})\frac{\partial L_{n,FP}}{\partial \zeta}(\zeta^k_{SP})\right]_{1\leq i,j,k\leq p+1}. \tag{2.38}$$

Enfin nous intégrons temporellement pour obtenir $\hat{\mathbf{U}}^e_{SP}$ au temps suivant :

$$\frac{d\hat{\mathbf{U}}^e_{SP}}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi} + \frac{\partial \hat{\mathbf{F}}}{\partial \eta} + \frac{\partial \hat{\mathbf{G}}}{\partial \zeta}\right)^e_{SP}. \tag{2.39}$$

# 3 Nouvelle méthode : la méthode FUSE

## 3.1 Introduction de la méthode

Pan et al. [8] présentent une nouvelle méthode d'ordre élevé pour la résolution numérique des équations hyperboliques en 1D et en 2D et 3D, nommée "Face-Upwinded Spectral Element" (abrégée en FUSE). Nous introduirons brièvement dans ce paragraphe le principe de cette nouvelle méthode en 1D telle qu'expliquée dans le papier.

Considérons l'équation hyperbolique 1D qui est de la forme :

$$\partial_t u + \partial_x f(u) = 0 \text{ sur } \Omega \times [0,T], \tag{3.1}$$

avec $\Omega = [0,1]$, $T \in \mathbb{R}_+^*$. Cette équation peut être réécrite sous la forme :

$$\partial_t u + a(u)\partial_x u = 0 \text{ sur } \Omega \times [0,T]. \tag{3.2}$$

Le domaine physique $\Omega$ est partitionné en éléments $K \in T_h$ tel que $\Omega = \bigcup_{K \in T_h} K$. Deux ensembles de points sont définis pour chaque élément $K$ :

- L'ensemble des points solution $\{s_0, ..., s_p\}$, $p \geq 0$, qui sont utilisés pour discrétiser la solution $u$.
- L'ensemble des points flux $\{f_0, ..., f_q\}$, avec $q \geq p$, qui sont utilisés pour discrétiser le flux $f$.

Soit les espaces de fonctions suivants :

$$V_s(T_h) = \left\{ v \in H^1(\Omega) : v_{|K} \in \mathbb{P}_p, \forall K \in T_h \right\}, \tag{3.3}$$

$$V_f(T_h) = \left\{ v \in H^1(\Omega) : v_{|K} \in \mathbb{P}_q, \forall K \in T_h \right\}, \tag{3.4}$$

où $\mathbb{P}_p$ est l'espace des polynômes de degré $p$. Soit la base $\{\phi_i^s\}$ (respectivement $\left\{\phi_i^f\right\}$) de l'espace $V_s(T_h)$ (resp. $V_f(T_h)$), où les $\phi_i^s$ sont des polynômes interpolateurs associés aux points $s_i$ (resp. associés aux points $f_i$) définis tels que :

$$\phi_i^s(s_j) = \delta_{ij} \text{ et } \phi_i^f(f_j) = \delta_{ij}, \tag{3.5}$$

avec $\delta_{ij}$ définie telle que :

$$\delta_{ij} = \begin{cases} 1 \text{ si } i = j, \\ 0 \text{ sinon.} \end{cases} \tag{3.6}$$

Le papier [8] propose de prendre les polynômes utilisés en méthode des éléments finis usuellement. Ainsi soit $u_h$ l'approximation de $u$ dans l'espace $V_s$ et $f_h$ l'approximation de $f$ dans l'espace $V_f$. L'algorithme de résolution numérique sera donc à chaque pas de temps, sur une cellule :

1. Le polynôme solution est formé : $u_h(x) = \sum_{i=0}^{p} \phi_i^s(x) u_h(s_i)$.
2. Le flux $f_h$ est évalué à chaque point flux $f_h(f_i) = f(u_h(f_i))$, pour $i = 0, ..., q$.
3. Le polynôme $f_h$ est formé : $f_h(x) = \sum_{i=0}^{q} \phi_i^f(x) f_h(f_i)$.
4. La dérivée de $f_h$ est évaluée à chaque point solution et est utilisée pour l'intégration temporelle.

Cependant sur les interfaces de chaque cellule, $\partial_x f_h$ a en général plusieurs valeurs. Il faut donc choisir une valeur unique. L'article [8] propose un exemple où $p = q = 2$, donc les points solution et les points flux coïncident. Nous ferons aussi le choix de prendre des points équidistants. Notons donc :

$$h = s_i - s_{i-1} = f_i - f_{i-1}. \tag{3.7}$$

Notons aussi $s_{i,k}$ le $i$-ème point solution du $k$-ème élément $K_k \in T_h$. Alors l'évaluation de la dérivée de $f_h$ au point solution intérieur sera :

$$f_h'(s_{1,k}) = \frac{1}{2h} f_h(f_{i+1,k}) - \frac{1}{2h} f_h(f_{i-1,k}). \tag{3.8}$$

Pour les points sur le bord de l'élément $k$, le choix de $f_h'$ dépend du signe de $a(u) = f'(u)$. L'article [8] propose de prendre par exemple pour le point $s_{2,k}$ :

$$f_h'(s_{2,k}) = \begin{cases} \frac{3}{2h} f_h(f_{2,k}) - \frac{2}{h} f_h(f_{1,k}) + \frac{1}{2h} f_h(f_{0,k}) \text{ si } a(s_{2,k}) > 0, \\ -\frac{3}{2h} f_h(f_{0,k+1}) + \frac{2}{h} f_h(f_{1,k+1}) - \frac{1}{2h} f_h(f_{2,k+1}) \text{ si } a(s_{2,k}) < 0. \end{cases} \tag{3.9}$$

## 3.2 Application de la méthode FUSE aux différences spectrales

Dans la partie 2.3.1 de l'article [8] est présenté un parallèle entre la méthode FUSE et les différences spectrales pour un domaine 1D. Nous présenterons donc cette équivalence et nous l'adapterons à des problèmes non linéaires et des maillages hexahédriques 2D. Par abus de langage, nous appellerons aussi cette extension FUSE.

### 3.2.1 Cas linéaire 1D

Dans cette partie, nous cherchons à résoudre numériquement l'équation de transport linéaire suivante :

$$\partial_t u + \partial_x E(u) = 0 \text{ sur } \Omega \times [0, T], \tag{3.10}$$

où $E(u) = cu$, avec $c \in \mathbb{R}$ la vitesse de transport.

De façon analogue à la méthode SD, l'espace est discrétisé en $N_e$ éléments et nous utilisons une transformation de chaque cellule $K$ dans l'espace isoparamétrique pour résoudre :

$$\frac{\partial \hat{u}}{\partial t} + \frac{\partial \hat{E}}{\partial \xi} = 0 \text{ sur } [0,1] \times [0,T], \tag{3.11}$$

avec $\xi \in [0,1]$, $\hat{u} = |J|u$ et $\hat{E} = |J|\xi_x E$, où $\xi_x = \frac{\partial \xi}{\partial x}$.

A l'intérieur de chaque cellule $K$, nous utilisons toujours $N_{SP} = p+1$ points solution et $N_{FP} = p+2$ points flux pour l'interpolation de la solution et du flux. La distribution des points flux reste la même, c'est-à-dire que nous utilisons les points de Gauss-Legendre pour les $p$ points intérieurs, et $\xi_{FP}(1) = 0$ et $\xi_{FP}(N_{FP}) = 1$ pour les deux points restants aux bords, car comme expliqué précédemment, cette localisation des points flux ne rend pas le schéma instable. Cependant la stabilité ne dépend pas de la distribution des points solution, c'est pourquoi nous ferons le choix dans cette nouvelle méthode de collocaliser les points solution avec les points flux. Or comme il y a un point solution de moins que de points flux, Pan et al. [8] proposent de choisir la localisation des points solution en fonction du sens du flux, qui est déterminé par le signe de $c$. Ainsi le choix de la distribution des points solution se fera de la manière suivante :

- Si $c > 0$, les points solution seront les $p+1$ points flux les plus à droite, soit : $\boldsymbol{\xi}_{SP} = [\xi_{FP}(i)]_{2 \leq i \leq N_{FP}}$ ;
- Si $c < 0$, les points solution seront les $p+1$ points flux les plus à droite, soit : $\boldsymbol{\xi}_{SP} = [\xi_{FP}(i)]_{1 \leq i \leq N_{FP}-1}$.

La figure 3.1 illustre la différence de disposition des points solution entre la méthode SD standard et la méthode FUSE.


FIGURE 3.1 – Position des points solution en 1D pour les différentes méthodes avec $p = 2$ : ● pour la méthode SD standard ; ■ pour la méthode FUSE quand $c < 0$ ; ★ pour la méthode FUSE quand $c > 0$.

Cette nouvelle localisation des points solution permet de ne pas avoir à extrapoler $\hat{\mathbf{U}}$ aux points flux. Nous obtenons donc l'algorithme 1 de résolution numérique.

L'étape du solveur de Riemann peut être évitée car pendant l'étape "d'extrapolation", nous

**Algorithm 1** Algorithme de résolution d'une équation de transport linéaire 1D

Lire le maillage, en déduire le déterminant de la jacobienne $|J|$ et la métrique $\xi_x$.
Calculer le vecteur de points flux $\xi_{FP}$, en déduire le vecteur de points solution $\xi_{SP}$.
Initialiser $\hat{\mathbf{U}}_{SP} = |J|\mathbf{U}_{SP}$ aux points solution.
**for** *iter* de 1 à $N_{iter}$ **do**
    **for** $e$ de 1 à $N_e$ **do**
        Remplir le vecteur solution aux points flux :
        **if** $c > 0$ **then**
            $\left[\hat{\mathbf{U}}^e_{FP}(i)\right]_{2\leq i\leq N_{FP}} = \hat{\mathbf{U}}^e_{SP}$.
            $\hat{\mathbf{U}}^e_{FP}(1) = \hat{\mathbf{U}}^{e-1}_{FP}(N_{FP})$.
        **else**
            $\left[\hat{\mathbf{U}}^e_{FP}(i)\right]_{1\leq i\leq N_{FP}-1} = \hat{\mathbf{U}}^e_{SP}$.
            $\hat{\mathbf{U}}^e_{FP}(N_{FP}) = \hat{\mathbf{U}}^{e+1}_{FP}(1)$.
        **end if**
    **end for**
    Appliquer les conditions aux limites à $\hat{\mathbf{U}}_{FP}$.
    **for** $e$ de 1 à $N_e$ **do**
        Calculer le flux aux points flux : $\left[\hat{\mathbf{E}}^e(i)\right]_{2\leq i\leq N_{FP}-1} = \left[\hat{\mathbf{E}}\left(\hat{U}^e_{FP}(i)\right)\right]_{2\leq i\leq N_{FP}-1}$
        Dériver $\hat{\mathbf{E}}^e$ aux points solution : $\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} = \left[\sum_{k=1}^{N_{FP}} \hat{\mathbf{E}}^e(k)\frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i))\right]_{1\leq i\leq N_{SP}}$
        Mettre à jour $\hat{\mathbf{U}}^e_{SP}$ en résolvant $\frac{d\hat{\mathbf{U}}^e_{SP}}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP}$ avec un schéma d'intégration temporel.
    **end for**
**end for**

récupérons la valeur de **U** au point flux manquant de la cellule d'à côté. Comme nous tenons compte du sens du flux pour le maillage, utiliser un schéma décentré (upwind) pour le solveur de Riemann nous donnerait exactement la même valeur de **U** à ce point-là.

### 3.2.2 Cas scalaire non linéaire 1D

Nous nous plaçons maintenant dans le cas scalaire où $E$ n'est pas forcément une fonction linéaire de $u$ et nous supposons que nous pouvons réécrire l'équation (2.1) telle que :

$$\frac{\partial u}{\partial t} + a(u)\frac{\partial u}{\partial x} = 0, \tag{3.12}$$

où $a(u) = E'(u)$.

Avec un $a$ plus général, le "sens" du flux n'est pas forcément le même sur tout le domaine et à tous les temps $t_n$. Dans ce cas-là, Pan et al. [8] ne proposent pas d'équivalence entre la méthode présentée et la méthode SD. Cependant, comme vu pour le cas linéaire, l'idée de collocaliser les points solution avec les points flux permet de faire moins d'opérations qu'une extrapolation classique (cf. partie 6 pour une analyse des complexités algorithmiques de chaque méthode numérique). Il a donc été décidé de garder cette idée de collocalisation en non linéaire. Deux manières de faire ont ainsi été identifiées :

1. La position des points solution change en fonction du sens du flux d'un élément à l'autre et d'une itération à l'autre. En effet, notons $E_{Godunov}$ la valeur unique sur l'interface déterminée par le solveur de Riemann, $u_L$ (respectivement $u_R$) la valeur de la solution sur la cellule à gauche de l'interface (resp. à droite). Alors le solveur de Godunov pour une fonction $E$ monotone sera :
$$E_{Godunov} = \begin{cases} E(u_L) \text{ si } E'(u_L) < 0, \\ E(u_R) \text{ si } E'(u_R) > 0. \end{cases} \tag{3.13}$$
Ainsi si le flux est monotone sur les deux cellules partageant une interface, nous pouvons collocaliser les points solution avec les points flux suivant le sens du flux de la même manière qu'en linéaire ce qui nous permet ainsi d'éviter d'utiliser un solveur de Riemann. Et pour les cellules où le sens est amené à changer, il suffit de faire la méthode SD classique et d'utiliser un solveur de Riemann.

   De façon plus formelle, soit $e$ un élément du maillage de l'espace $\Omega$. Notons $u_L^e$ la valeur de $u$ sur l'interface gauche de la cellule $e$, et $u_R^e$ sa valeur sur l'interface droite à un temps $t_n$ donné. Les points solution seront définis de la manière suivante :
   — Si $a(u_L^e) > 0$ et $a(u_R^e) > 0$ : nous supposons alors qu'il n'y a pas de changement de sens

du flux au sein de la cellule, donc nous choisirons les points solution comme les $p+1$ points flux les plus à droite : $\boldsymbol{\xi}_{SP} = [\xi_{FP}(i)]_{2\leq i\leq N_{FP}}$ ;

— Si $a(u_L^e) < 0$ et $a(u_R^e) < 0$ : nous supposons ici encore qu'il n'y a pas de changement de sens du flux au sein de la cellule, donc nous choisirons les points solution comme les $p+1$ points flux les plus à gauche : $\boldsymbol{\xi}_{SP} = [\xi_{FP}(i)]_{1\leq i\leq N_{FP}-1}$ ;

— Sinon on prend les points de Gauss-Tchebychev comme pour la méthode SD standard : $\boldsymbol{\xi}_{SP} = \left[\frac{1}{2}\left(1-\cos\left(\frac{2i-1}{2N_{SP}}\pi\right)\right)\right]_{1\leq i\leq N_{SP}}$.

Cette manière de faire nous donne l'algorithme 2 de résolution numérique.

2. La position des points solution est déterminée à l'initialisation et ne change pas d'une itération à l'autre. Pour choisir quelle position de points flux à prendre, la moyenne de la condition initiale $\bar{u_0} = \frac{1}{|\Omega|}\int_\Omega u_0(x)dx$ est calculée. Ainsi :

— Si $\bar{u_0} \geq 0$, nous choisirons les points solution comme les $p+1$ points flux les plus à droite : $\boldsymbol{\xi}_{SP} = [\xi_{FP}(i)]_{2\leq i\leq N_{FP}}$ ;

— Si $\bar{u_0} < 0$, nous choisirons les points solution comme les $p+1$ points flux les plus à gauche : $\boldsymbol{\xi}_{SP} = [\xi_{FP}(i)]_{1\leq i\leq N_{FP}-1}$.

Cette deuxième manière de faire nous amène à l'algorithme 3 de résolution numérique. C'est cet algorithme qui a été implémenté dans HOPPS.

**Algorithm 2** Algorithme de résolution d'une équation de transport non linéaire 1D

Lire le maillage, en déduire le déterminant de la jacobienne $|J|$ et la métrique $\xi_x$.
Calculer le vecteur de points flux $\boldsymbol{\xi}_{FP}$ et les différents vecteurs de points solution $\boldsymbol{\xi}_{SP}$.
Evaluer aux interfaces $a(U)$, en déduire la disposition de points solution $\boldsymbol{\xi}_{SP}$ pour chaque cellule.
Initialiser $\hat{\mathbf{U}}_{SP} = |J|\mathbf{U}_{SP}$ aux points solution.
**for** *iter* de 1 à $N_{iter}$ **do**
    **for** $e$ de 1 à $N_e$ **do**
        Remplir le vecteur solution aux points flux :
        **if** $a(U_L^e) > 0$ et $a(U_R^e) > 0$ **then**
            $\left[\hat{U}_{FP}^e(i)\right]_{2\leq i\leq N_{FP}} = \hat{\mathbf{U}}_{SP}^e$.
            $\hat{U}_{FP}^e(1) = \hat{U}_{FP}^{e-1}(N_{FP})$.
        **else if** $a(U_L^e) < 0$ et $a(U_R^e) < 0$ **then**
            $\left[\hat{U}_{FP}^e(i)\right]_{1\leq i\leq N_{FP}-1} = \hat{\mathbf{U}}_{SP}^e$.
            $\hat{U}_{FP}^e(N_{FP}) = \hat{U}_{FP}^{e-1}(1)$.
        **else**
            Extrapoler $\hat{\mathbf{U}}$ aux points flux : $\hat{\mathbf{U}}_{FP}^e = \left[\overline{U}^e(\xi_{FP}(i))\right]_{1\leq i\leq N_{FP}}$, avec $\overline{U}^e(\xi) = \sum_{i=1}^{N_{SP}} \hat{U}_{SP}^e(i) L_{i,SP}(\xi)$.
        **end if**
    **end for**
    Appliquer les conditions aux limites à $\hat{\mathbf{U}}_{FP}$.
    Evaluer à chaque interface $a(U)$.
    **for** $e$ de 1 à $N_e$ **do**
        Calculer le flux aux points flux intérieurs : $\left[\hat{E}^e(i)\right]_{2\leq i\leq N_{FP}-1} = \left[\hat{\mathbf{E}}\left(\hat{\mathbf{U}}_{FP}^e(i)\right)\right]_{2\leq i\leq N_{FP}-1}$
    **end for**
    Utiliser un solveur de Riemann aux interfaces du maillage qui voisinent une cellule avec une disposition de points solution de Gauss-Tchebychev pour obtenir une valeur unique du flux sur tout le domaine physique.
    **for** $e$ de 1 à $N_e$ **do**
        Dériver $\hat{\mathbf{E}}^e$ aux nouveaux points solution : $\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)_{SP}^e = \left[\sum_{k=1}^{N_{FP}} \hat{E}^e(k) \frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i))\right]_{1\leq i\leq N_{SP}}$.
        Mettre à jour $\hat{\mathbf{U}}_{SP}^e$ en résolvant $\frac{d\hat{\mathbf{U}}_{SP}^e}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)_{SP}^e$ avec un schéma d'intégration temporel.
    **end for**
**end for**

**Algorithm 3** Algorithme de résolution d'une équation de transport non linéaire 1D à points fixes

Lire le maillage, en déduire le déterminant de la jacobienne $|J|$ et la métrique $\xi_x$.
Calculer le vecteur de points flux $\boldsymbol{\xi}_{FP}$.
Evaluer $\bar{u_0}$, en déduire la disposition de points solution $\boldsymbol{\xi}_{SP}$ sur tout le domaine.
Initialiser $\hat{\mathbf{U}}_{SP} = |J|\mathbf{U}_{SP}$ aux points solution.
**for** *iter* de 1 à $N_{iter}$ **do**
    **for** $e$ de 1 à $N_e$ **do**
        Remplir le vecteur solution aux points flux :
        **if** $\bar{u_0} \geq 0$ : **then**
            $\left[\hat{U}^e_{FP}(i)\right]_{2\leq i\leq N_{FP}} = \hat{\mathbf{U}}^e_{SP}$.
            $\hat{U}^e_{FP}(1) = \sum_{i=1}^{N_{SP}} \hat{U}^e_{SP}(i) L_{i,SP}(\xi_{FP}(1))$.
        **else**
            $\left[\hat{U}^e_{FP}(i)\right]_{1\leq i\leq N_{FP}-1} = \hat{\mathbf{U}}^e_{SP}$.
            $\hat{U}^e_{FP}(N_{FP}) = \sum_{i=1}^{N_{SP}} \hat{U}^e_{SP}(i) L_{i,SP}(\xi_{FP}(N_{FP}))$.
        **end if**
    **end for**
    Appliquer les conditions aux limites à $\hat{\mathbf{U}}_{FP}$.
    **for** $e$ de 1 à $N_e$ **do**
        Calculer le flux aux points flux intérieurs : $\left[\hat{\mathbf{E}}^e(i)\right]_{2\leq i\leq N_{FP}-1} = \left[\hat{\mathbf{E}}\left(\hat{\mathbf{U}}^e_{FP}(i)\right)\right]_{2\leq i\leq N_{FP}-1}$
    **end for**
    Utiliser un solveur de Riemann aux interfaces du maillage pour obtenir une valeur unique du flux sur tout le domaine physique.
    **for** $e$ de 1 à $N_e$ **do**
        Dériver $\hat{\mathbf{E}}^e$ aux points solution : $\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} = \left[\sum_{k=1}^{N_{FP}} \hat{E}^e(k) \frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i))\right]_{1\leq i\leq N_{SP}}$.
        Mettre à jour $\hat{\mathbf{U}}^e_{SP}$ en résolvant $\frac{d\hat{\mathbf{U}}^e_{SP}}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP}$ avec un schéma d'intégration temporel.
    **end for**
**end for**

### 3.2.3 Cas linéaire 2D

Soit l'équation de transport linéaire suivante :

$$\partial_t u + \nabla.\mathbf{f}(u) = 0 \text{ sur } \Omega \times [0,T], \tag{3.14}$$

que l'on peut réécrire telle que :

$$\partial_t u + \partial_x E(u) + \partial_y F(u) = 0, \tag{3.15}$$

où $\Omega \subset \mathbb{R}^2$, $\mathbf{f}(u) = \begin{pmatrix} E(u) \\ F(u) \end{pmatrix} = \begin{pmatrix} c_x u \\ c_y u \end{pmatrix}$, $c_x, c_y \in \mathbb{R}$ des constantes.

Nous souhaitons généraliser la méthode FUSE proposée pour un modèle 1D à ce modèle 2D avec maillage hexahédrique. Nous voulons donc à nouveau collocaliser les points solution par rapport aux points flux afin d'éviter l'étape d'extrapolation. Comme pour la méthode SD standard, nous avons encore $(p+1)^2$ points solution et $(p+2)(p+1)$ points flux dans chaque direction, soit un total de $2(p+2)(p+1)$ points flux. Notons les points flux dans le domaine isoparamétrique tels que :

— Dans la $\xi$-direction : $[(\xi_{FP}, \eta_{FP})]_\xi = [(\xi_{FP}(i), \eta_{SP}(j))]_{\substack{1\leq i\leq p+2 \\ 1\leq j\leq p+1}}$
— Dans la $\eta$-direction : $[\xi_{FP}, \eta_{FP})]_\eta = [(\xi_{SP}(i), \eta_{FP}(j))]_{\substack{1\leq i\leq p+1 \\ 1\leq j\leq p+2}}$

Soit les points solution $(\xi_{SP}(i), \eta_{SP}(j))_{1\leq i,j\leq p+1}$, avec :

$$\boldsymbol{\xi}_{SP} = \begin{cases} [(\xi_{FP}(i))_\xi]_{2\leq i\leq p+2} & \text{si } c_x > 0, \\ [(\xi_{FP}(i))_\xi]_{1\leq i\leq p+1} & \text{sinon,} \end{cases} \quad \text{et } \boldsymbol{\eta}_{SP} = \begin{cases} [(\eta_{FP}(i))_\xi]_{2\leq i\leq p+2} & \text{si } c_y > 0, \\ [(\eta_{FP}(i))_\xi]_{1\leq i\leq p+1} & \text{sinon.} \end{cases}$$

La figure 3.2 illustre la différence de position des points solution et flux entre la méthode SD et la méthode FUSE. Nous obtenons ainsi l'algorithme de résolution numérique 4.

### 3.2.4 Cas d'un système non linéaire 2D

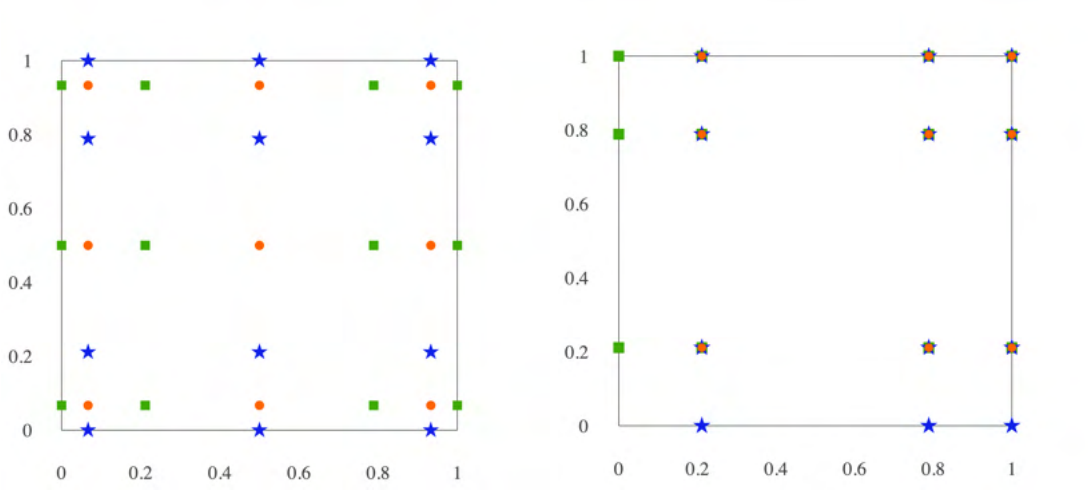
Nous nous intéressons maintenant au cas non linéaire en 2D, plus particulièrement à un système 2D, dont nous rappelons la forme ci-dessous :

$$\partial_t \mathbf{U} + \partial_x \mathbf{E}(\mathbf{U}) + \partial_y \mathbf{F}(\mathbf{U}) = 0 \text{ sur } \Omega \times [0,T]. \tag{3.16}$$

Comme pour le cas non linéaire en 1D, Pan et al. [8] ne proposent pas d'équivalence entre leur méthode et les différences spectrales. Dans le cas d'un système, les solutions peuvent ne pas aller toutes dans le même "sens". Une première idée pour adapter l'algorithme FUSE 2 proposé en sca-

**Algorithm 4** Algorithme de résolution d'une équation de transport linéaire 2D

Lire le maillage, en déduire le déterminant de la jacobienne $|J|$ et la métrique $\xi_x, \eta_y$.
Calculer les vecteurs de points flux $[(\xi_{FP}, \eta_{FP})]_\xi$ et $[(\xi_{FP}, \eta_{FP})]_\eta$, en déduire le vecteur de points solution $[(\boldsymbol{\xi}_{SP}, \boldsymbol{\eta}_{SP})]$.
Initialiser $\hat{\mathbf{U}}_{SP} = |J|\mathbf{U}_{SP}$ aux points solution.
**for** *iter* de 1 à $N_{iter}$ **do**
    **for** $e$ de 1 à $N_e$ **do**
        Remplir les vecteurs solution aux points flux dans chaque direction :
        **if** $c_x > 0$ **then**
            $\left[\hat{U}^e_{FP,\xi}(i,j)\right]_{\substack{2\leq i\leq p+2\\ 1\leq j\leq p+1}} = \hat{\mathbf{U}}^e_{SP}$.
            $\left[\hat{U}^e_{FP,\xi}(1,j)\right]_{1\leq j\leq p+1} = \left[\hat{U}^{e-1}_{SP}(p+2,j)\right]_{1\leq j\leq p+1}$.
        **else**
            $\left[\hat{U}^e_{FP,\xi}(i,j)\right]_{\substack{1\leq i\leq p+1\\ 1\leq j\leq p+1}} = \hat{\mathbf{U}}^e_{SP}$.
            $\left[\hat{U}^e_{FP,\xi}(p+2,j)\right]_{1\leq j\leq p+1} = \left[\hat{U}^{e+1}_{SP}(1,j)\right]_{1\leq j\leq p+1}$.
        **end if**
        **if** $c_y > 0$ **then**
            $\left[\hat{U}^e_{FP,\eta}(i,j)\right]_{\substack{1\leq i\leq p+1\\ 2\leq j\leq p+2}} = \hat{\mathbf{U}}^e_{SP}$.
            $\left[\hat{U}^e_{FP,\eta}(i,1)\right]_{1\leq i\leq p+1} = \left[\hat{U}^{e-1}_{SP}(i,p+2)\right]_{1\leq i\leq p+1}$.
        **else**
            $\left[\hat{U}^e_{FP,\eta}(i,j)\right]_{\substack{1\leq i\leq p+1\\ 1\leq j\leq p+1}} = \hat{\mathbf{U}}^e_{SP}$.
            $\left[\hat{U}^e_{FP,\eta}(i,p+2)\right]_{1\leq i\leq p+1} = \left[\hat{U}^{e+1}_{SP}(i,1)\right]_{1\leq i\leq p+1}$.
        **end if**
    **end for**
    Appliquer les conditions aux limites à $\hat{\mathbf{U}}_{FP,\xi}$ et $\hat{\mathbf{U}}_{FP,\eta}$.
    **for** $e$ de 1 à $N_e$ **do**
        Calculer les flux dans chaque direction aux points flux : $\hat{\mathbf{E}}^e = \hat{\mathbf{E}}(\hat{\mathbf{U}}^e_{FP,\xi})$ et $\hat{\mathbf{F}}^e = \hat{\mathbf{F}}(\hat{\mathbf{U}}^e_{FP,\eta})$.
        Dériver les flux aux points solution :
$\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} = \left[\sum_{k=1}^{p+2}\sum_{l=1}^{p+1} \hat{E}^e(k,l)\frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i))L_{l,SP}(\eta_{SP}(j))\right]_{1\leq i,j\leq p+1}$ et
$\left(\frac{\partial \hat{\mathbf{F}}}{\partial \eta}\right)^e_{SP} = \left[\sum_{k=1}^{p+1}\sum_{l=1}^{p+2} \hat{F}^e(k,l)L_{l,SP}(\xi_{SP}(i))\frac{\partial L_{k,FP}}{\partial \eta}(\eta_{SP}(j))\right]_{1\leq i,j\leq p+1}$
        Mettre à jour $\hat{\mathbf{U}}^e_{SP}$ en résolvant $\frac{d\hat{\mathbf{U}}^e_{SP}}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} - \left(\frac{\partial \hat{\mathbf{F}}}{\partial \eta}\right)^e_{SP}$ avec un schéma d'intégration temporel.
    **end for**
**end for**




(a) Méthode SD standard.

(b) Méthode FUSE avec $c_x > 0$ et $c_y > 0$.

FIGURE 3.2 – Positions des différents points en 2D pour la méthode SD et la méthode FUSE avec $p = 2$ : ● pour les points solution ; ■ pour les points flux dans la direction $x$ ; ★ pour les points flux dans la direction $y$.

laire 1D serait de prendre une disposition de points différente pour chaque composante du vecteur solution. Cependant cela serait coûteux en terme de temps de calcul et nous ferait perdre le temps de calcul gagné en évitant des solveurs de Riemann. C'est pourquoi il a été décidé de ne garder que l'idée de fixer les points solution à l'initialisation pour un système 2D.

Afin de choisir l'emplacement des points solution, nous pouvons par exemple nous appuyer comme en 1D sur la condition initiale d'une des composantes du vecteur solution. Pour les équations d'Euler par exemple, en s'appuyant sur le cas test du COVO (cf. partie 5.5), nous choisissons l'emplacement des points solution en fonction de la condition initiale de la vitesse $\mathbf{V}_0 = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix}$, soit $\bar{u_0} = \int_\Omega u_0(x)dx$ dans la direction $x$, et de $\bar{v_0} = \int_\Omega v_0(y)dy$ dans la direction $y$. Nous obtenons donc l'algorithme de résolution numérique 5.

**Algorithm 5** Algorithme de résolution d'un système 2D

Lire le maillage, en déduire le déterminant de la jacobienne $|J|$ et la métrique $\xi_x, \eta_y$.
Calculer les vecteurs de points flux $[(\xi_{FP}, \eta_{FP})]_\xi$ et $[(\xi_{FP}, \eta_{FP})]_\eta$.
Calculer $\bar{u_0}$ et $\bar{v_0}$, en déduire le vecteur de points solution $[(\xi_{SP}, \eta_{SP})]$.
Initialiser $\hat{\mathbf{U}}_{SP} = |J|\mathbf{U}_{SP}$ aux points solution.
**for** *iter* de 1 à $N_{iter}$ **do**
    **for** $e$ de 1 à $N_e$ **do**
        Remplir les vecteurs solution aux points flux dans chaque direction :
        **if** $\bar{u_0} > 0$ **then**
            $\left[\hat{U}^e_{FP,\xi}(i,j)\right]_{\substack{2\le i\le p+2\\ 1\le j\le p+1}} = \hat{\mathbf{U}}^e_{SP}$.
            **for** $k$ de 1 à $p+1$ **do**
                $\hat{U}^e_{FP,\xi}(1,k) = \sum_{i=1}^{N_{SP}} \sum_{j=1}^{N_{SP}} \hat{U}^e_{i,SP} L_{i,SP}(\xi_{FP}(1)) L_{j,SP}(\eta_{SP}(k))$.
            **end for**
        **else**
            $\left[\hat{U}^e_{FP,\xi}(i,j)\right]_{\substack{1\le i\le p+1\\ 1\le j\le p+1}} = \hat{\mathbf{U}}^e_{SP}$.
            **for** $k$ de 1 à $p+1$ **do**
                $\hat{U}^e_{FP,\xi}(p+2,k) = \sum_{i=1}^{N_{SP}} \sum_{j=1}^{N_{SP}} \hat{U}^e_{i,SP} L_{i,SP}(\xi_{FP}(p+2)) L_{j,SP}(\eta_{SP}(k))$.
            **end for**
        **end if**
        **if** $\bar{v_0} > 0$ **then**
            $\left[\hat{U}^e_{FP,\eta}(i,j)\right]_{\substack{1\le i\le p+1\\ 2\le j\le p+2}} = \hat{\mathbf{U}}^e_{SP}$.
            **for** $k$ de 1 à $p+1$ **do**
                $\hat{U}^e_{FP,\xi}(k,1) = \sum_{i=1}^{N_{SP}} \sum_{j=1}^{N_{SP}} \hat{U}^e_{i,SP} L_{i,SP}(\xi_{SP}(k)) L_{j,SP}(\eta_{FP}(1))$.
            **end for**
        **else**
            $\left[\hat{U}^e_{FP,\eta}(i,j)\right]_{\substack{1\le i\le p+1\\ 1\le j\le p+1}} = \hat{\mathbf{U}}^e_{SP}$.
            **for** $k$ de 1 à $p+1$ **do**
                $\hat{U}^e_{FP,\xi}(k,p+2) = \sum_{i=1}^{N_{SP}} \sum_{j=1}^{N_{SP}} \hat{U}^e_{i,SP} L_{i,SP}(\xi_{SP}(k)) L_{j,SP}(\eta_{FP}(p+2))$.
            **end for**
        **end if**
    **end for**
    Appliquer les conditions aux limites à $\hat{\mathbf{U}}_{FP,\xi}$ et $\hat{\mathbf{U}}_{FP,\eta}$.
    Utiliser un solveur de Riemann aux interfaces du maillage pour obtenir une valeur unique du flux sur tout le domaine physique.
    **for** $e$ de 1 à $N_e$ **do**
        Calculer les flux dans chaque direction aux points flux : $\hat{\mathbf{E}}^e = \hat{\mathbf{E}}(\hat{\mathbf{U}}^e_{FP,\xi})$ et $\hat{\mathbf{F}}^e = \hat{\mathbf{F}}(\hat{\mathbf{U}}^e_{FP,\eta})$.
        Dériver les flux aux points solution :
        $\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} = \left[\sum_{k=1}^{p+2}\sum_{l=1}^{p+1} \hat{E}^e(k,l) \frac{\partial L_{k,FP}}{\partial \xi}(\xi_{SP}(i)) L_{l,SP}(\eta_{SP}(j))\right]_{1\le i,j\le p+1}$ et
        $\left(\frac{\partial \hat{\mathbf{F}}}{\partial \eta}\right)^e_{SP} = \left[\sum_{k=1}^{p+1}\sum_{l=1}^{p+2} \hat{F}^e(k,l) L_{l,SP}(\xi_{SP}(i)) \frac{\partial L_{k,FP}}{\partial \eta}(\eta_{SP}(j))\right]_{1\le i,j\le p+1}$
        Mettre à jour $\hat{\mathbf{U}}^e_{SP}$ en résolvant $\frac{d\hat{\mathbf{U}}^e_{SP}}{dt} = -\left(\frac{\partial \hat{\mathbf{E}}}{\partial \xi}\right)^e_{SP} - \left(\frac{\partial \hat{\mathbf{F}}}{\partial \eta}\right)^e_{SP}$ avec un schéma d'intégration temporel.
    **end for**
**end for**

# 4 Stabilité en 2D de la méthode SD et de la méthode FUSE

Nous souhaitons dans cette partie démontrer la stabilité théorique de la méthode SD et de la méthode FUSE. L'article [4] démontre la stabilité de la méthode SD en 1D, en n'utilisant pas d'expression particulière des points solution. C'est pourquoi la méthode FUSE est aussi stable en 1D. Nous voulons maintenant prouver que les deux méthodes sont stables pour un maillage hexahédrique 2D.

**Proposition 1** *Soit la loi de conservation suivante :*

$$\begin{cases} \partial_t u + \nabla.\mathbf{f}(u) = 0 \text{ dans } \Omega \times [0,T], \\ u = g \ \text{ sur } \partial\Omega \times [0,T], \end{cases} \tag{4.1}$$

*avec $\Omega \subset \mathbb{R}^2$ le domaine physique, $\partial\Omega$ le bord de $\Omega$, $T \in \mathbb{R}_+^*$, $g \in L^2(\partial\Omega)$ la condition aux limites et $\mathbf{f}(u) = \mathbf{c}u = \begin{pmatrix} c_x \\ c_y \end{pmatrix} u = \begin{pmatrix} f_x \\ f_y \end{pmatrix}$ le flux, où $\mathbf{c}$ est la vitesse d'advection.*

*Soit le maillage conforme de $\Omega$ noté $T_h$ composé de quadrangles $K$ tel que $\Omega = \bigcup_{K \in T_h} K$. Soit la distribution de points flux dans la direction $x$ :*

$$\{(\xi_{FP}(i), \eta_{FP}(j)), (i,j) \in [\![1, p+2]\!] \times [\![1, p+1]\!]\}_\xi = \{(\xi_{FP}(i), \xi_{SP}(j)), (i,j) \in [\![1, p+2]\!] \times [\![1, p+1]\!]\},$$

*et dans la direction $y$ :*

$$\{(\xi_{FP}(i), \eta_{FP}(j)), (i,j) \in [\![1, p+1]\!] \times [\![1, p+2]\!]\}_\eta = \{(\xi_{SP}(i), \xi_{FP}(j)), (i,j) \in [\![1, p+1]\!] \times [\![1, p+2]\!]\},$$

*où $p$ est l'ordre du schéma SD utilisé, $\boldsymbol{\xi}_{SP} = [\xi_{SP}(i)]_{1 \le i \le p+1}$ sont les points solution 1D, et $\boldsymbol{\xi}_{FP} = [\xi_{FP}(i)]_{1 \le i \le p+2}$ sont les points flux définis tels que les $p$ points intérieurs sont les $p$ points de Gauss-Legendre, et $\xi_{FP}(1) = 0$ et $\xi_{FP}(p+2) = 1$.*

*Alors le schéma SD est stable avec cette disposition de points flux sous la norme :*

$$\|u\| = \int_K \left( u^2 + \sum_{\substack{\alpha=(\alpha_x,\alpha_y) \\ \alpha_x+\alpha_y=2p}} a_\alpha (\partial^\alpha u)^2 \right) dx, \tag{4.2}$$

*où les $a_\alpha$ sont des constantes positives à déterminer, et l'opérateur $\partial^\alpha$ est défini tel que : $\partial^\alpha u = \frac{\partial^{\alpha_x}}{\partial x^{\alpha_x}} \frac{\partial^{\alpha_y}}{\partial y^{\alpha_y}} u$.*

*De plus, le schéma FUSE est aussi stable sous la norme (4.2) avec cette disposition de points flux.*

*Preuve* Nous souhaitons montrer la stabilité de la méthode numérique SD pour une équation de transport linéaire en 2D avec un maillage hexahédrique, c'est-à-dire que nous souhaitons montrer que :

$$\partial_t \|u\|^2 \leq 0. \tag{4.3}$$

Soit l'espace isoparamérique $\hat{K} = [-1, 1]^2$. Nous pouvons définir une transformation $\mathcal{F}_K$ de $\hat{K}$ dans chaque cellule $K$ telle que :

$$\begin{aligned} \mathcal{F}_K : \hat{K} &\to K \\ \hat{\mathbf{x}} &\mapsto \mathbf{x}. \end{aligned}$$

Notons $\mathbf{x}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$, $i = 1, ..., 4$, les sommets qui définissent le quadrangle $K$. Afin que $\mathcal{F}_K({}^t(0,0)) = \mathbf{x}_1$, $\mathcal{F}_K({}^t(1,0)) = \mathbf{x}_2$, $\mathcal{F}_K({}^t(1,1)) = \mathbf{x}_3$ et $\mathcal{F}_K({}^t(0,1)) = \mathbf{x}_4$, nous avons :

$$\mathcal{F}_K(\hat{\mathbf{x}}) = \begin{pmatrix} x_3 + x_1 - x_2 - x_4 \\ y_3 + y_1 - y_2 - y_4 \end{pmatrix} \xi\eta + \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \end{pmatrix} \xi + \begin{pmatrix} x_4 - x_1 \\ y_4 - y_1 \end{pmatrix} \eta + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \tag{4.4}$$

où $\hat{\mathbf{x}} = \begin{pmatrix} \xi \\ \eta \end{pmatrix}$.

Notons $J_K$ la jacobienne de $\mathcal{F}_K$. Alors l'équation (4.1) devient sur $\hat{K}$ :

$$\partial_t \hat{u} + \nabla.\hat{\mathbf{f}}(\hat{u}) = 0, \tag{4.5}$$

avec $\hat{u} = |J_K|u$ et $\hat{\mathbf{f}} = \begin{pmatrix} |J_K|(\xi_x f_x + \xi_y f_y) \\ |J_K|(\eta_x f_x + \eta_y f_y) \end{pmatrix}$, où $\xi_x = \frac{\partial \xi}{\partial x}$ ($\xi_y$, $\eta_x$ et $\eta_y$ sont définis de manière analogue).

Soit les espaces de fonctions suivants :

$$V_h = \left\{ v \in L^2(\Omega) |\ \forall K \in T_h, |J_K| v_{|K} \in \mathbb{Q}_p(\hat{K}) \right\}, \tag{4.6}$$

$$F_h = \left\{ \mathbf{g} \in (L^2(\Omega))^2 |\ \forall K \in T_h, J_K \mathbf{g}_{|K} \in \tilde{\mathbb{Q}}_{p+1}(\hat{K}) \right\}, \tag{4.7}$$

où $\mathbb{Q}_p(\hat{K})$ désigne les polynômes de degré $p$ et $x$ et en $y$ (soit de degré global $2p$) et $\tilde{\mathbb{Q}}_{p+1}$ est défini tel que :

$$\tilde{\mathbb{Q}}_{p+1}(\hat{K}) = \left\{ \begin{pmatrix} L_1(x)L_2(y) \\ L_3(x)L_4(y) \end{pmatrix}, L_1, L_4 \in \mathbb{P}_{p+1}([-1,1]), L_2, L_3 \in \mathbb{P}_p([-1,1]) \right\}, \tag{4.8}$$

avec $\mathbb{P}_p([-1,1])$ désignant l'ensemble des polynômes de degré $p$ sur $[-1,1]$.

Notons $u_h$ l'approximation de $u$ dans l'espace $V_h$ et $\mathbf{f}_h = \mathbf{c}u_h$ l'approximation de $\mathbf{f}$ dans $F_h$. Alors $u_h$ et $f_h$ vérifient :

$$\partial_t u_h + \nabla.\mathbf{f}_h = 0 \text{ sur } \Omega \times [0,T]. \tag{4.9}$$

Pour toute cellule $K$ de $T_h$, $\mathbf{f}_{h|K}$ est construit à partir de $u_{h|K}$. Or $u_h$ n'est pas forcément continu sur $\Omega$. Mais pour obtenir la conservation du flux, $\mathbf{f}_{h|K}$ doit, quant à lui, être continu. Un solveur de Riemann est donc utilisé à chaque interface entre deux cellules pour rétablir la continuité du flux, que nous pouvons alors décomposer ainsi :

$$\mathbf{f_h} = \mathbf{c}u_h + \mathbf{f^c}, \tag{4.10}$$

où $\mathbf{f^c}$ est la correction apportée sur le bord de $K$ par le solveur de Riemann. Alors, en remplaçant $\mathbf{f}_h$ par son expression dans l'équation (4.9), nous obtenons :

$$\partial_t u_h = -\nabla.(\mathbf{c}u_h) - \nabla.\mathbf{f^c}. \tag{4.11}$$

L'équation (4.11) est multipliée par $u_h$ puis intégrée sur un élément du maillage $K$. Une intégration par parties nous donne ainsi :

$$\frac{1}{2}\int_K \partial_t u_h^2 dx = -\int_K u_h \nabla.(\mathbf{c}u_h)dx + \int_K \mathbf{f^c}.\nabla u_h dx - \int_{\partial K} u_h \mathbf{f^c}.\mathbf{n}d\sigma. \tag{4.12}$$

Or :

$$\begin{aligned} \int_K u_h \nabla.(\mathbf{c}u_h)dx &= \int_{\partial K} u_h^2 \mathbf{c}.\mathbf{n}d\sigma - \int_K u_h \nabla.(\mathbf{c}u_h)dx \\ \Longrightarrow \int_K u_h \nabla.(\mathbf{c}u_h)dx &= \frac{1}{2}\int_{\partial K} u_h^2 \mathbf{c}.\mathbf{n}d\sigma. \end{aligned}$$

Donc :

$$\frac{1}{2}\partial_t \|u_h\|^2_{L^2(K)} = -\frac{1}{2}\int_{\partial K} u_h^2 \mathbf{c}.\mathbf{n}d\sigma + \int_K \mathbf{f^c}.\nabla u_h dx - \int_{\partial K} u_h \mathbf{f^c}.\mathbf{n}d\sigma. \tag{4.13}$$

Notons $\gamma_h = \bigcup_{K \in T_h} \partial K$ le squelette de $T_h$. Les deux termes de bord peuvent être signés. En effet, en sommant ces termes pour tout $K \in T_h$, nous pouvons les réécrire comme une somme sur les

arêtes du squelette du maillage :

$$-\sum_{K\in T_h}\int_{\partial K}(\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n}+u_h\mathbf{f^c}.\mathbf{n})d\sigma=-\sum_{e\in\gamma_h\setminus\partial\Omega}\int_e\left(\frac{1}{2}u_1^2\mathbf{c}.\mathbf{n_1}+u_1\mathbf{f_1^c}.\mathbf{n_1}+\frac{1}{2}u_2^2\mathbf{c}.\mathbf{n_2}+u_2\mathbf{f_2^c}.\mathbf{n_2}\right)dx$$
$$-\sum_{e\in\partial\Omega}\int_e(\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n}+u_h\mathbf{f^c}.\mathbf{n})dx, \quad (4.14)$$

où $K_1$ et $K_2$ sont les quadrangles tels que $K_1\cap K_2=e$, $u_1=u_{h|K_1}$ et $u_2=u_{h|K_2}$, et $\mathbf{n_1}$ la normale unitaire sortante de $K_1$ par $e$, $\mathbf{n_2}$ la normale unitaire sortante de $K_2$ par $e$. Soit le flux de Godunov entre les cellules $K_L$ et $K_R$ défini tel que :

$$\mathbf{f_h^G}=\mathbf{c}\left(\frac{1+signe(\mathbf{c}.\mathbf{n_L})}{2}u_L+\frac{1-signe(\mathbf{c}.\mathbf{n_R})}{2}u_R\right). \quad (4.15)$$

Alors $\mathbf{f^c}=\mathbf{f_h^G}-\mathbf{c}u_h$. Ainsi :

$$-\sum_{K\in T_h}\int_{\partial K}(\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n}+u_h\mathbf{f^c}.\mathbf{n})d\sigma=-\sum_{e\in\gamma_h\setminus\partial\Omega}\int_e\left(-\frac{1}{2}u_1^2\mathbf{c}.\mathbf{n_1}+u_1\mathbf{f_1^G}.\mathbf{n_1}-\frac{1}{2}u_2^2\mathbf{c}.\mathbf{n_2}+u_2\mathbf{f_2^G}.\mathbf{n_2}\right)dx$$
$$-\sum_{e\in\partial\Omega}\int_e(-\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n}+u_h\mathbf{f^G}.\mathbf{n})dx. \quad (4.16)$$

Notons $\mathbf{n_e}=\mathbf{n_1}=-\mathbf{n_2}$ la normale unitaire de l'arête $e$. Ici nous ferons le choix de $\mathbf{c}.\mathbf{n_e}>0$. Alors :

$$\begin{aligned}-\frac{1}{2}u_1^2\mathbf{c}.\mathbf{n_1}+\mathbf{f_1^G}.\mathbf{n_1}-\frac{1}{2}u_2^2\mathbf{c}.\mathbf{n_2}+u_2\mathbf{f_2^G}.\mathbf{n_2}&=\mathbf{c}.\mathbf{n_e}\left(-\frac{1}{2}u_1^2+u_1^2+\frac{1}{2}u_2^2-u_1u_2\right)\\&=\mathbf{c}.\mathbf{n_e}(u_1-u_2)^2>0.\end{aligned}$$

Ainsi :

$$-\sum_{e\in\gamma_h\setminus\partial\Omega}\int_e\left(-\frac{1}{2}u_1^2\mathbf{c}.\mathbf{n_1}+u_1\mathbf{f_1^G}.\mathbf{n_1}-\frac{1}{2}u_2^2\mathbf{c}.\mathbf{n_2}+u_2\mathbf{f_2^G}.\mathbf{n_2}\right)dx<0. \quad (4.17)$$

Il reste la somme des arêtes sur le bord de $\Omega$. Il existe plusieurs cas de figure selon les conditions aux limites utilisées. Le cas où il n'y a pas de flux est trivial. Pour le cas où la frontière est séparée en une partie entrante et une partie sortante, nous avons :

— Sur la frontière sortante, $\mathbf{c}.\mathbf{n_e} > 0$. Le flux prendra la valeur $\mathbf{c}u_h$ sur la frontière. Ainsi :

$$-\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n} + u_h^2\mathbf{c}.\mathbf{n} = \frac{1}{2}u_h^2\mathbf{c}.\mathbf{n} > 0. \tag{4.18}$$

D'où :

$$-\int_e(-\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n} + u_h\mathbf{f^G}.\mathbf{n})dx < 0. \tag{4.19}$$

— Sur la frontière entrante, $\mathbf{c}.\mathbf{n_e} < 0$. Le flux prendra donc la valeur d'entrée qui sera notée $\mathbf{c}u_e$. Alors :

$$-\left(-\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n_e} + u_h u_e\mathbf{c}.\mathbf{n_e}\right) = -\mathbf{c}.\mathbf{n_e}(\frac{1}{2}u_e^2 - \frac{1}{2}(u_e - u_h)^2) < -\mathbf{c}.\mathbf{n_e}\frac{1}{2}u_e^2. \tag{4.20}$$

Cette quantité est positive, mais il s'agit d'une contribution à l'énergie uniquement liée à ce qui est injecté dans le système physique. Ainsi en prenant $u_e = 0$, le terme est bien négatif.

Le terme sur la frontière de $\Omega$ et le terme sur $\gamma_h \backslash \partial\Omega$ sont bien tous deux négatifs. Donc :

$$-\sum_{K\in T_h}\int_{\partial K}(\frac{1}{2}u_h^2\mathbf{c}.\mathbf{n} + u_h\mathbf{f^c}.\mathbf{n})d\sigma < 0. \tag{4.21}$$

Or le terme $\int_K \mathbf{f^c}.\nabla u_h dx$ dans l'équation (4.13) n'est pas signé. C'est pourquoi nous allons utiliser la norme (4.2).

Ainsi appliquer l'opérateur $\partial^\alpha$ à (4.11) nous donne :

$$\partial_t\partial^\alpha u_h = -\partial^\alpha(\nabla.\mathbf{c}u_h)) - \partial^\alpha(\nabla.\mathbf{f^c}). \tag{4.22}$$

En multipliant l'équation (4.22) par $\partial^\alpha u_h$ et en intégrant, nous obtenons :

$$\int_K \frac{1}{2}\partial_t(\partial^\alpha u_h)^2 dx = -\int_K \partial^\alpha(\nabla.(\mathbf{c}u_h))\partial^\alpha u_h dx - \int_K \partial^\alpha(\nabla.\mathbf{f^c})\partial^\alpha u_h dx. \tag{4.23}$$

Donc en sommant sur les coefficients $\alpha$ nous avons finalement :

$$\int_K \frac{1}{2}\partial_t\left(\sum_\alpha a_\alpha(\partial^\alpha u_h)^2\right)dx = -\int_K \partial^\alpha(\nabla.(\mathbf{c}u_h))\partial^\alpha u_h dx - \int_K \sum_\alpha a_\alpha\partial^\alpha(\nabla.\mathbf{f^c})\partial^\alpha u_h dx. \tag{4.24}$$

D'où, en sommant l'équation (4.13) et l'équation (4.24), nous obtenons au final pour toute cellule $K$ :

$$\frac{1}{2}\partial_t\|u_h\|^2 = -\int_K \partial^\alpha(\nabla.(\mathbf{c}u_h))\partial^\alpha u_h dx - \frac{1}{2}\int_{\partial K} u_h^2 \mathbf{c}.\mathbf{n}d\sigma - \int_{\partial K} u_h \mathbf{f^c}.\mathbf{n}d\sigma + \int_K \mathbf{f^c}.\nabla u_h dx - \int_K \sum_\alpha a_\alpha \partial^\alpha(\nabla.\mathbf{f^c})\partial^\alpha u_h dx. \quad (4.25)$$

Nous souhaitons alors montrer que :

$$-\int_K \partial^\alpha(\nabla.(\mathbf{c}u_h))\partial^\alpha u_h dx + \int_K \mathbf{f^c}.\nabla u_h dx - \int_K \sum_\alpha a_\alpha \partial^\alpha(\nabla.\mathbf{f^c})\partial^\alpha u_h dx = 0. \quad (4.26)$$

Or $u_h$ est de degré $p$ en $x$ et en $y$, donc $\int_K \partial^\alpha(\nabla.(\mathbf{c}u_h))\partial^\alpha u_h dx = 0$. Cela nous donne finalement le critère suivant à respecter :

$$\int_K \mathbf{f^c}.\nabla u_h dx - \int_K \sum_\alpha a_\alpha \partial^\alpha(\nabla.\mathbf{f^c})\partial^\alpha u_h dx = 0. \quad (4.27)$$

Il nous suffit donc de trouver les coefficients $a_\alpha$ tels que le critère (4.27) est rempli pour tout $u_h$ et $\mathbf{f^c}$ sur toute cellule $K$. On se place alors dans $\hat{K}$. Le critère (4.27) devant être vrai pour tout $u_h$, il doit rester vrai pour un élément $l_i(x)l_j(y)$ de la base de $\hat{K}$. Nous prendrons $l_i$ (respectivement $l_j$) le polynôme de Lagrange interpolé aux $p+1$ points solution dans la direction $x$ (resp. $y$) et associé au point $x_i$ (resp. $x_j$).

De même, le critère (4.27) doit rester vrai en particulier pour un flux nul dans la direction $y$ (le cas où le flux dans la direction $x$ est nul est symétrique). Il s'agit d'une équation de transport linéaire et nous supposerons que $c_x > 0$. C'est pourquoi la correction de Godunov sera nulle sur le côté droit de la cellule. Donc le critère (4.27) reste vrai pour une correction $\begin{pmatrix} \hat{l}_1(x)l_j(y) \\ 0 \end{pmatrix}$, où $\hat{l}_1$ est le polynôme de Lagrange interpolé aux $p+2$ points flux et associé au point flux $\hat{x}_1$. Alors :

$$\nabla(l_i(x)l_j(y)) = \begin{pmatrix} l_i'(x)l_j(y) \\ l_i(x)l_j'(y) \end{pmatrix},\ \nabla.\begin{pmatrix} \hat{l}_1(x)l_j(y) \\ 0 \end{pmatrix} = \hat{l}_1'(x)l_j(y),$$

$$\partial^\alpha(l_i(x)l_j(y)) = l_i^{(\alpha_x)}(x)l_j^{(\alpha_y)}(y),\ \partial^\alpha\left(\nabla.\begin{pmatrix} \hat{l}_1(x)l_j(y) \\ 0 \end{pmatrix}\right) = \hat{l}_1^{(\alpha_x+1)}(x)l_j^{(\alpha_y)}(y).$$

En remplaçant dans l'équation (4.27), nous obtenons :

$$\int_{\hat{K}} l_i'(x)l_j(y)\hat{l}_1(x)l_j(y)dx - \int_{\hat{K}} \sum_\alpha a_\alpha l_i^{(\alpha_x)}(x)l_j^{(\alpha_y)}(y)\hat{l}_1^{(\alpha_x+1)}(x)l_j^{(\alpha_y)}(y)dx = 0.$$

Or quand $\alpha \neq (p,p)$, soit $\alpha_x \geq p+1$ soit $\alpha_y \geq p+1$. Et comme $l_i$ et $l_j$ sont de degré $p$,

soit $l_i^{(\alpha_x)}(x) = 0$ soit $l_j^{(\alpha_y)}(y) = 0$. Donc la somme se réduit à $\alpha = (p, p)$. Ainsi il nous suffit de déterminer le coefficient $a = a_{(p,p)}$ tel que :

$$\int_{\hat{K}} l_i'(x) l_j(y) \hat{l}_1(x) l_j(y) dx - a \int_{\hat{K}} l_i^{(p)}(x) l_j^{(p)}(y) \hat{l}_1^{(p+1)}(x) l_j^{(p)}(y) dx = 0 \tag{4.28}$$

$$\Longrightarrow \int_{-1}^{1} l_j^2 dy \int_{-1}^{1} \hat{l}_1 l_i' dx - a \int_{-1}^{1} (l_j^{(p)})^2 dy \int_{-1}^{1} l_i^{(p)} \hat{l}_1^{(p+1)} dx = 0. \tag{4.29}$$

Nous noterons dans la suite $I_j = \int_{\hat{K}} l_j^2 dy$ et $a_j$ le coefficient dominant de $l_j$. Alors $l_j^{(p)} = p! a_j$. Nous prendrons à partir de maintenant les $p$ points flux intérieurs dans la direction $x$ égaux aux points de Gauss-Legendre. Le polynôme $\hat{l}_1$ peut donc être exprimé tel que :

$$\hat{l}_1(x) = (-1)^p \frac{1}{2}(1 - x) L_p(x), \tag{4.30}$$

avec $L_p$ le polynôme de Legendre de degré $p$. Son coefficient dominant sera noté $c_p$. Il peut être déterminé grâce à la formule de Rodrigues :

$$\begin{aligned} L_p(x) &= \frac{1}{2^p} \frac{1}{p!} \frac{\partial^p}{\partial x^p} (x^2 - 1)^p \\ &= \frac{1}{2^p} \frac{1}{p!} \frac{\partial^p}{\partial x^p} (x^{2p} + ...) \\ &= \frac{1}{2^p} \frac{1}{p!} \frac{(2p)!}{p!} x^p + ... \\ &= \frac{1 \times 3 \times 5 \times ... \times (2p-1)}{p!} x^p + ... \end{aligned}$$

$L_p$ étant orthogonal à tout polynôme de degré inférieur à $p$, nous avons :

$$\begin{aligned} \int_{-1}^{1} \hat{l}_1 l_i' dx &= (-1)^{p+1} \frac{1}{2} \int_{-1}^{1} x L_p(x) l_i'(x) dx \\ &= (-1)^{p+1} \frac{1}{2} \int_{-1}^{1} L_p(x)(a_i p x^p + ...) dx \\ &= (-1)^{p+1} \frac{1}{2} \frac{a_i p}{c_p} \int_{-1}^{1} L_p(x) c_p x^p dx \\ &= (-1)^{p+1} \frac{1}{2} \frac{a_i p}{c_p} \left( \int_{-1}^{1} L_p(x)^2 dx - \int_{-1}^{1} L_p(x)(c_{p-1} x^{p-1} + ...) dx \right) \\ &= (-1)^{p+1} \frac{1}{2} \frac{2 p a_i}{(2p+1) c_p}, \end{aligned}$$

car $\int_{-1}^{1} L_p(x)^2 dx = \frac{2}{2p+1}$.

Nous avons aussi que $\hat{l}_1^{(p+1)} = (-1)^{p+1}\frac{1}{2}(p+1)!c_p$. Nous obtenons donc :

$$I_j(-1)^{p+1}\frac{1}{2}\frac{2pa_i}{(2p+1)c_p} - a \times 4(p!)^2 a_j^2(-1)^{p+1}\frac{1}{2}(p+1)!c_p p!a_j = 0. \tag{4.31}$$

En simplifiant, nous avons finalement l'expression suivante de $a$ :

$$a = \frac{p}{2(2p+1)(p!)^3(p+1)!c_p^2}\frac{I_j}{a_j^2} > 0, \tag{4.32}$$

car $I_j = \int_{-1}^{1} l_j^2 dy > 0$.

Cette preuve ne dépendant pas de la position des points solution, nous pouvons donc l'appliquer à la méthode SD et à la méthode FUSE.

# 5 Tests numériques

## 5.1 Présentation de HOPPS

HOPPS (High Order Performance Portable Solvers) est un code de calcul développé par le CERFACS. Il a été créé dans un premier temps pour déterminer si la librairie Kokkos pouvait être pertinente dans un contexte de codes CFD. Les résultats obtenus étant prometteurs, il a été décidé de continuer de le développer afin de simuler numériquement des cas et configurations physiques de la mécanique des fluides.

HOPPS se présente sous la forme d'un module Python, c'est-à-dire que les solveurs, les modèles, etc. peuvent être manipulés grâce à un script Python, indépendamment de l'architecture utilisée. Le coeur du module est quant à lui développé en C++. Il est construit autour de quatre grandes abstractions de base, représentées par des classes :

- Le maillage (Mesh) qui contient la définition du domaine de calcul (noeuds, éléments, bords...).
- Le domaine (Field) qui contient les valeurs des solutions. Cette structure dépend de la méthode numérique utilisée.
- Le solveur (Solver) exécute la méthode numérique utilisée.
- Le modèle (Model) définit le système physique à résoudre.

Pour l'instant, HOPPS comporte seulement quelques modèles (advection linéaire, advection-diffusion, Euler, Navier-Stokes) résolus numériquement avec les méthodes SD, FUSE et LBM (méthode Lattice-Boltzmann). Il n'utilise actuellement que des maillages hexahédriques et les seules conditions aux limites possibles sont des conditions périodiques.

Durant ce stage, le modèle d'advection linéaire a été ajouté à HOPPS en 1D et en 2D. De plus, la méthode FUSE en linéaire et non linéaire avec points fixes a été codée en 1D et en 2D.

## 5.2 Cas linéaire en 1D

Rappelons l'équation de transport linéaire 1D à résoudre dans cette section :

$$\partial_t u + \partial_x(cu) = 0 \text{ sur } \Omega \times [0, T], \tag{5.1}$$

où $c \in \mathbb{R}$ est la vitesse d'advection. La condition initiale sera notée :

$$u_0(x) = u(x, 0),\ x \in \Omega. \tag{5.2}$$

Alors la solution exacte de ce problème est :

$$u(x,t) = u_0(x - ct). \tag{5.3}$$

### 5.2.1 Condition initiale continue

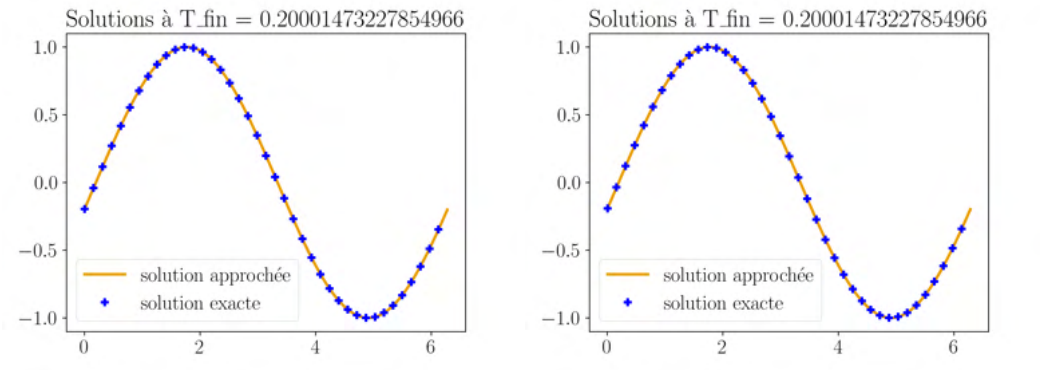
La méthode FUSE ainsi que la méthode SD standard ont été implémentées en Python afin de réaliser les tests numériques de cette section. Le schéma en temps utilisé est SSPRK3 [1].

Nous commençons par utiliser la condition initiale régulière suivante :

$$u_0(x) = \sin(x). \tag{5.4}$$

Nous prendrons $N_e = 200$ cellules, un ordre de $p = 2$, le nombre CFL égal à $cfl = 0.1$ et comme temps de fin $T_{fin} = 0.2$ s. Le domaine sur lequel nous souhaitons résoudre numériquement l'équation (5.2) sera $\Omega = [0, 2\pi]$ et les conditions aux bords seront périodiques. Enfin la vitesse d'advection sera $c = 1$ m.s$^{-1}$.

Les solutions obtenues pour la méthode SD standard et avec notre nouvelle méthode sont présentées sur le graphique 5.1.




(a) Méthode SD standard. (b) Méthode FUSE.

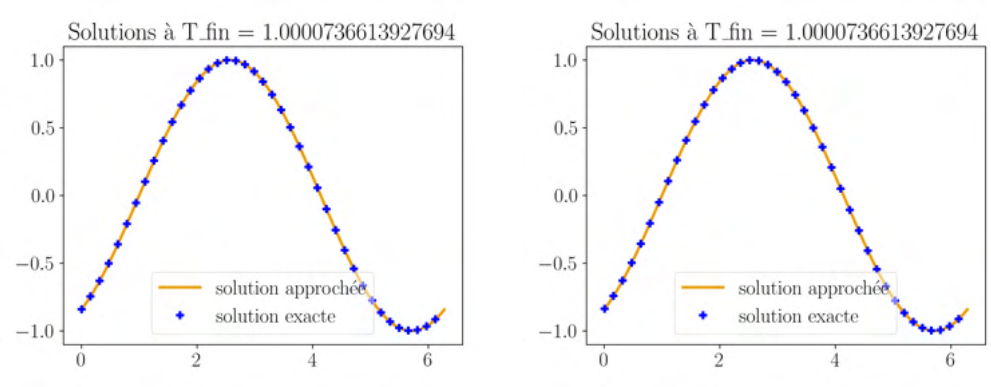
FIGURE 5.1 – Solutions obtenues pour les deux méthodes au temps $T_{fin} = 0.2$ s.

Nous pouvons observer dans les deux cas que le tracé de la solution exacte est superposé à celui de la solution approchée. Cela se confirme en regardant les erreurs commises dans le tableau 5.1.

| Erreur calculée | SD | FUSE |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.00012375407340802663 | 0.0001237559467524826 4 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 6.98227616634517e-05 | 6.982137484967676e-05 |

TABLE 5.1 – Erreurs commises pour les deux méthodes.

Les erreurs observées sont dans les deux cas sensiblement les mêmes. En changeant le temps de fin de la simulation, nous observons la même chose avec $T_{fin} = 1.0$ s sur le graphique 5.2 et le tableau 5.2.




(a) Méthode SD standard. (b) Méthode FUSE.

FIGURE 5.2 – Solutions obtenues pour les deux méthodes au temps $T_{fin} = 1.0$ s.

| Erreur | SD | FUSE |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.0006188560324894134 | 0.0006188579063737881 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.0003491540095723611 | 0.0003491509905454 4374 |

TABLE 5.2 – Erreurs commises pour les deux méthodes.

En calculant l'erreur $L^2$ entre la solution exacte et la solution calculée pour les deux méthodes et différents ordres $p$, nous obtenons les courbes de convergence présentées dans le graphique 5.3.

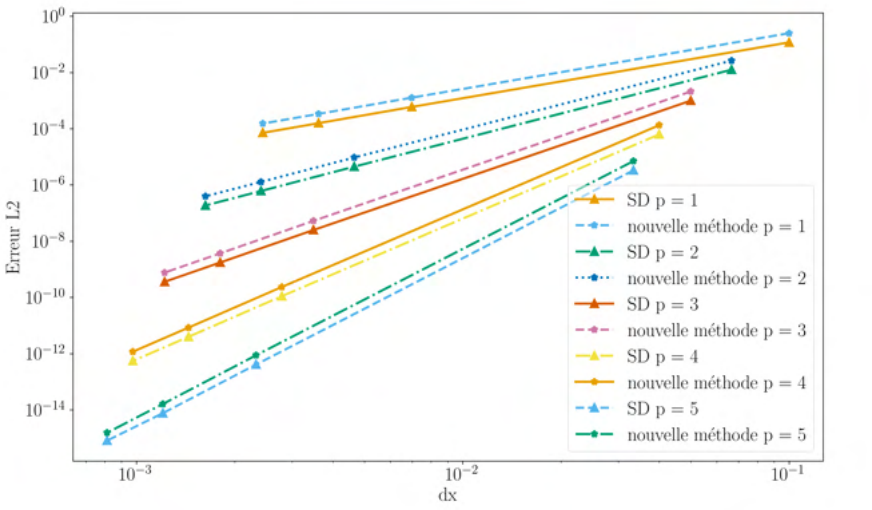



FIGURE 5.3 – Courbes de convergence pour l'équation de transport 1D avec une condition initiale régulière.

Les pentes des différentes courbes sont notées dans le tableau 5.3. A chaque ordre, nous pouvons observer que les pentes calculées pour chaque méthode correspondent à la pente théorique de $p+1$. Nous pouvons aussi noter que les courbes de convergence de la méthode FUSE sont légèrement au dessus de celles de la méthode SD standard, mais elles restent cependant du même ordre de grandeur.

| **Ordre polynomial $p$** | **Pente SD standard** | **Pente FUSE** | **Pente théorique** |
|---|---|---|---|
| 1 | 2.012232346766273 | 2.0122189701835658 | 2 |
| 2 | 3.0183691219219737 | 3.0183601377659666 | 3 |
| 3 | 4.024506496456029 | 4.024500759719532 | 4 |
| 4 | 5.030644405764189 | 5.030643163855714 | 5 |
| 5 | 5.712218777154561 | 5.989586210747027 | 6 |

TABLE 5.3 – Pentes des courbes d'erreur pour la méthode standard et la nouvelle méthode.

Avec cette condition initiale régulière, nous décidons de faire des tests de nombre CFL pour différents $p$ et différents $N_e$. Le critère de stabilité retenu sera que l'erreur $\|U_{ex} - U_{app}\|_{L^\infty(\Omega)}$ soit inférieure à $10^{-4}$ au bout de 1000 itérations. Les courbes 5.4 à 5.6 sont ainsi obtenues.

Nous observons que les nombres CFL calculés numériquement pour la méthode FUSE sont, de façon générale, légèrement plus petits que ceux calculés pour la méthode SD standard, ce qui est

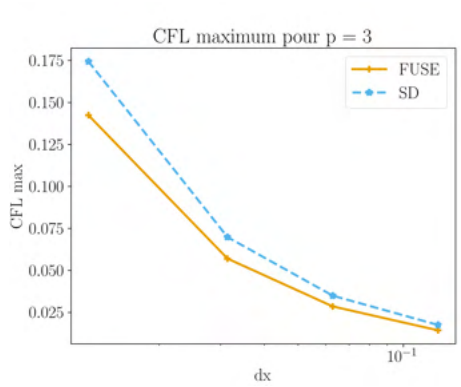



FIGURE 5.4 – Nombres CFL calculés pour les méthodes SD et FUSE avec $p = 3$.

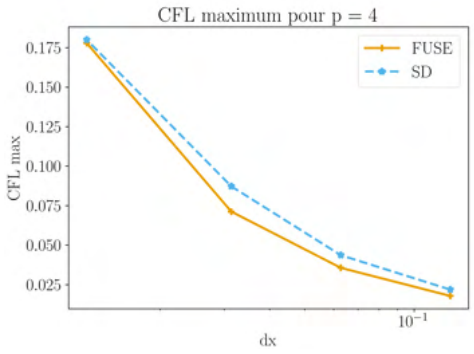



FIGURE 5.5 – Nombres CFL calculés pour les méthodes SD et FUSE avec $p = 4$.

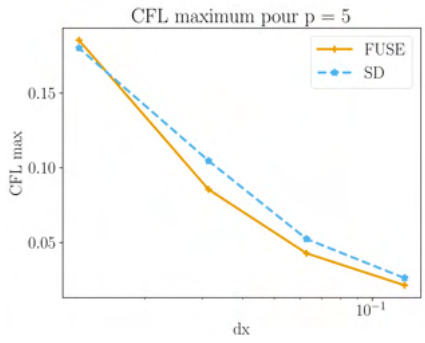



FIGURE 5.6 – Nombres CFL calculés pour les méthodes SD et FUSE avec $p = 5$.

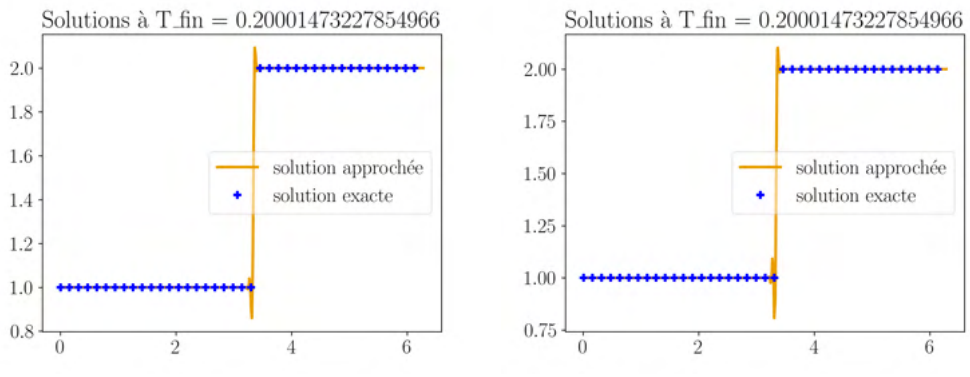
cohérent avec le fait que les erreurs commises par la méthode FUSE sont plus grandes que celles commises par la méthode SD.

### 5.2.2 Condition initiale discontinue

Nous souhaitons maintenant faire des tests en utilisant une condition initiale discontinue :

$$u_0(x) = \begin{cases} 1 \text{ si } x < \frac{|\Omega|}{2}, \\ 2 \text{ sinon.} \end{cases} \tag{5.5}$$

Nous gardons les mêmes paramètres que pour les figures 5.1, mais nous prenons cette fois des conditions de Dirichlet aux bords. Les graphiques 5.7 sont alors obtenus.




(a) Méthode SD standard.

(b) Méthode FUSE.

FIGURE 5.7 – Solutions obtenues pour les deux méthodes au temps $T_{fin} = 0.2$ s.

**Remarque 1** *Les oscillations observées au niveau des discontinuités correspondent au phénomène de Runge. En effet, nous utilisons une interpolation de Lagrange pour calculer la solution numérique. Pour atténuer les oscillations, nous pouvons utiliser des filtres pour capturer au mieux les discontinuités, comme celui présenté dans l'article [7] par exemple.*

Nous obtenons pour ce test les erreurs présentées dans le tableau 5.4.

| **Erreur calculée** | **SD** | **FUSE** |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.056510447983416146 | 0.06144398575267277 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.29650406161978915 | 0.3774852677684635 |

TABLE 5.4 – Erreurs commises pour les deux méthodes avec une condition initiale discontinue et un temps de fin $T_{fin} = 0.2$ s.

Nous pouvons observer dans les deux cas des oscillations de la solution approchée autour de la discontinuité, et le même ordre de grandeur au niveau des erreurs, bien qu'elles soient en peu plus grandes pour la méthode FUSE.

## 5.3 Cas scalaire non linéaire en 1D

### 5.3.1 Condition initiale continue

Afin de tester un cas scalaire non linéaire en 1D, nous allons résoudre numériquement l'équation de Burgers, qui est de la forme :

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0, \tag{5.6}$$

avec $f(u) = \frac{u^2}{2}$. Nous pouvons réécrire l'équation (5.6) sous forme non conservative :

$$\frac{\partial u}{\partial t} + a(u)\frac{\partial u}{\partial x} = 0, \tag{5.7}$$

où $a(u) = f'(u) = u$.

Nous utiliserons tout d'abord la condition initiale continue suivante :

$$u_0(x) = \sin(x). \tag{5.8}$$

Dans ce cas-là, la solution de l'équation (5.6) jusqu'à la date $T^* = \dfrac{-1}{\min\limits_{x_0 \in \mathbf{R}} \dfrac{d}{dx_0} a(u_0(x_0))} = 1$ sera :

$$u(x,t) = \sin(x_0), \tag{5.9}$$

avec $x_0$ qui vérifie :

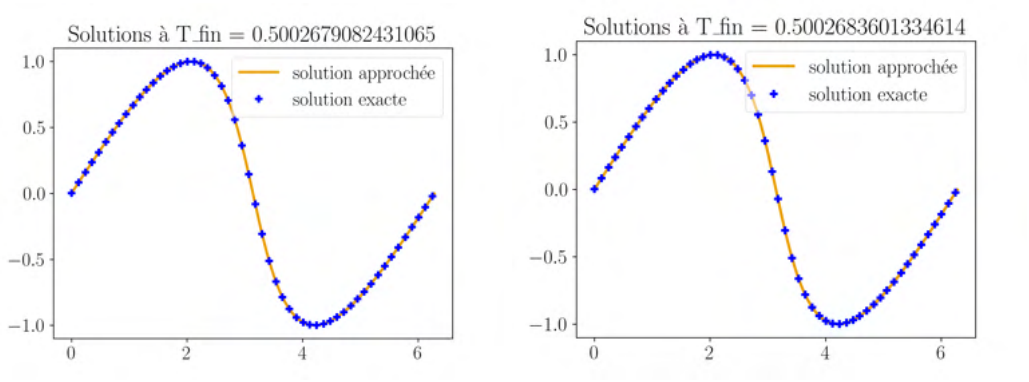
$$x_0 + \sin(x_0)t = x. \tag{5.10}$$

La méthode SD et la méthode FUSE avec adaptation des points solution ont été implémentées en Python pour les tests présentés dans cette partie. Le schéma d'intégration temporel utilisé est SSPRK3. Le domaine physique est $\Omega = [0, 2\pi]$, nous prendrons comme ordre $p = 3$, $N_e = 200$ éléments et un nombre CFL $cfl = 0.1$. Au temps $T_{fin} = 0.5$ s, nous obtenons les solutions numériques présentées dans les graphiques 5.8, ainsi que les erreurs notées dans le tableau 5.5.

| **Erreur calculée** | **SD** | **FUSE** |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.000216576844387351 | 0.0008262291953209899 |

TABLE 5.5 – Erreurs commises pour les deux méthodes pour la résolution de l'équation de Burgers avec une condition initiale continue.

Nous constatons à nouveau que les erreurs $L^2$ pour les deux méthodes sont du même ordre de




(a) Méthode SD standard. (b) Méthode FUSE.

FIGURE 5.8 – Solutions obtenues pour les deux méthodes au temps $T_{fin} = 0.5$ s.

grandeur, avec une erreur un peu plus grande pour la méthode FUSE par rapport à la méthode SD.

Pour ce test nous aurions aussi pu comparer les deux algorithmes FUSE présentés dans la partie 3.2.2 entre eux.

### 5.3.2 Problème de Riemann

A présent, nous souhaitons faire des tests pour la résolution d'un problème de Riemann, c'est-à-dire résoudre numériquement l'équation (5.6) avec une condition initiale qui sera de la forme :

$$u_0(x) = \begin{cases} u_L \text{ si } x < \frac{|\Omega|}{2}, \\ u_R \text{ sinon.} \end{cases} \tag{5.11}$$

Rappelons que dans le cas de l'équation de Burgers (où le flux $f(u) = \frac{u^2}{2}$ est convexe) :

— Si $u_L > u_R$, une onde de choc se forme. La solution exacte du problème sera donc :

$$u(x,t) = \begin{cases} u_L \text{ si } x < \sigma t, \\ u_R \text{ si } x > \sigma t. \end{cases} \tag{5.12}$$

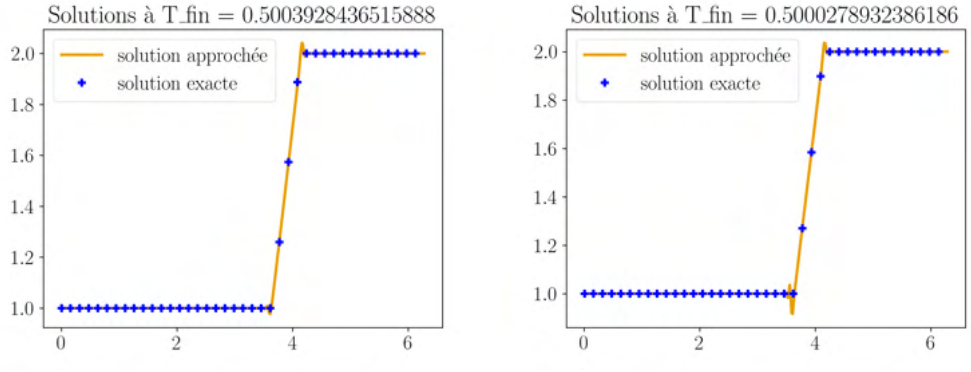
avec $\sigma$ donné par la relation de Rankine-Hugoniot : $-\sigma(f(u_R) - f(u_L)) + (u_R - u_L) = 0$.

— Si $u_L < u_R$, nous obtenons une onde de raréfaction. La solution du problème de Riemann

sera alors :

$$u(x,t) = \begin{cases} u_L \text{ si } x < f'(u_L)t + \frac{|\Omega|}{2}, \\ a^{-1}\left(\frac{x-\frac{|\Omega|}{2}}{t}\right) \text{ si } f'(u_L)t + \frac{|\Omega|}{2} < x < f'(u_R)t + \frac{|\Omega|}{2}, \\ u_R \text{ si } x > f'(u_R)t + \frac{|\Omega|}{2}. \end{cases} \tag{5.13}$$

Commençons par le cas $u_L < u_R$ :




(a) Méthode SD standard.

(b) Méthode FUSE.

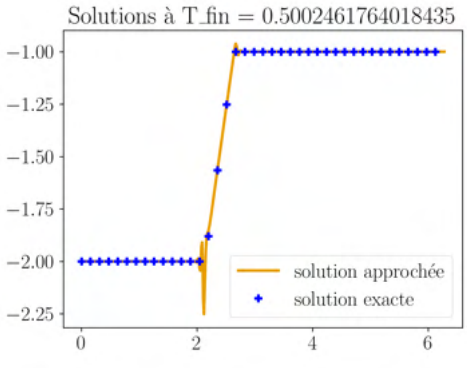
FIGURE 5.9 – Solutions pour les deux méthodes au temps $T_{fin} = 0.5$ s, avec $u_L = 1$ et $u_R = 2$.

| **Erreur calculée** | **SD** | **FUSE** |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.09703445344280012 | 0.1759846919409523 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.04143962289541436 | 0.0839102766051455 |

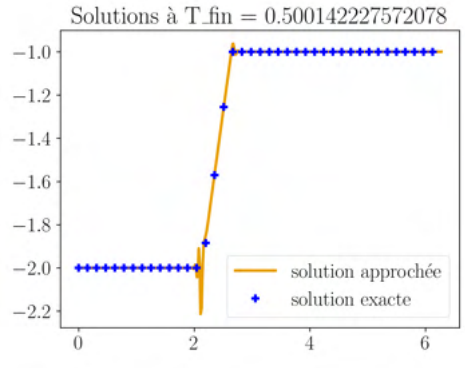
TABLE 5.6 – Erreurs commises pour les deux méthodes, avec $u_L = 1$ et $u_R = 2$.

| **Erreur calculée** | **SD** | **FUSE** |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.3907881028483392 | 0.3754874514921302 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.25111260836707316 | 0.21331038157672033 |

TABLE 5.7 – Erreurs commises pour les deux méthodes, avec $u_L = -2$ et $u_R = -1$.

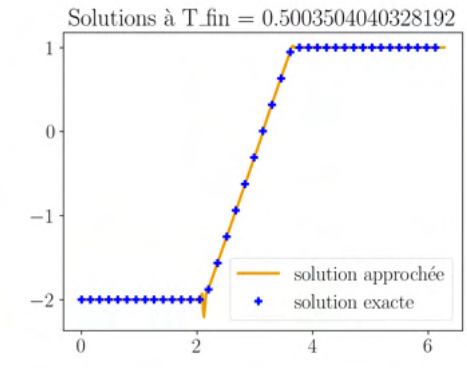



(a) Méthode SD standard.

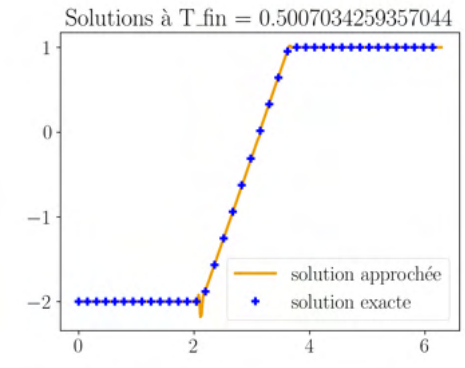



(b) Méthode FUSE.

FIGURE 5.10 – Solutions pour les deux méthodes au temps $T_{fin} = 0.5$ s, avec $u_L = -2$ et $u_R = -1$.




(a) Méthode SD standard.




(b) Méthode FUSE.

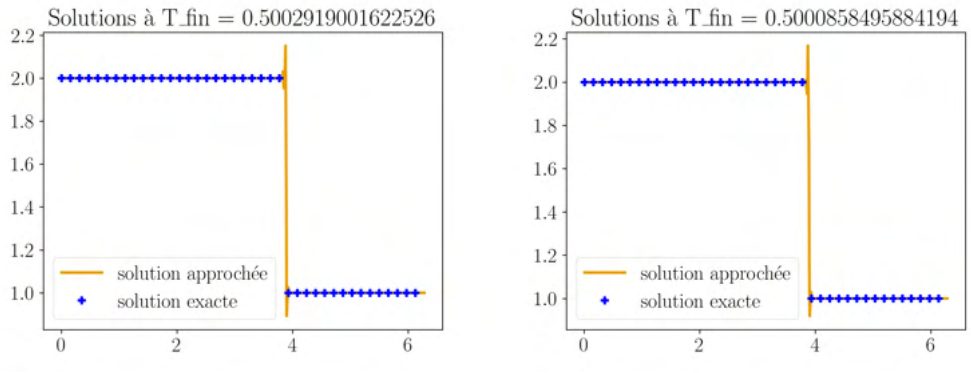
FIGURE 5.11 – Solutions pour les deux méthodes au temps $T_{fin} = 0.5$ s, avec $u_L = -2$ et $u_R = 1$.

| Erreur calculée | SD | FUSE |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.3412755073302644 | 0.3420729408260379 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.20506785652385684 | 0.18020683210521948 |

TABLE 5.8 – Erreurs commises pour les deux méthodes, avec $u_L = -2$ et $u_R = 1$.

Les trois tests présentés (graphiques 5.9 à 5.11 et tableaux 5.6 à 5.8) montrent des erreurs similaires pour les deux méthodes, qui sont confirmées par les graphiques : en effet, comme pour le transport linéaire d'une discontinuité, des oscillations apparaissent au début et à la fin de l'onde de détente, et ce pour les SD standard ou la méthode FUSE.

Passons maintenant au cas $u_L > u_R$ :




(a) Méthode SD standard. (b) Méthode FUSE.

FIGURE 5.12 – Solutions pour les deux méthodes au temps $T_{fin} = 0.5$ s, avec $U_L = 2$ et $U_R = 1$.

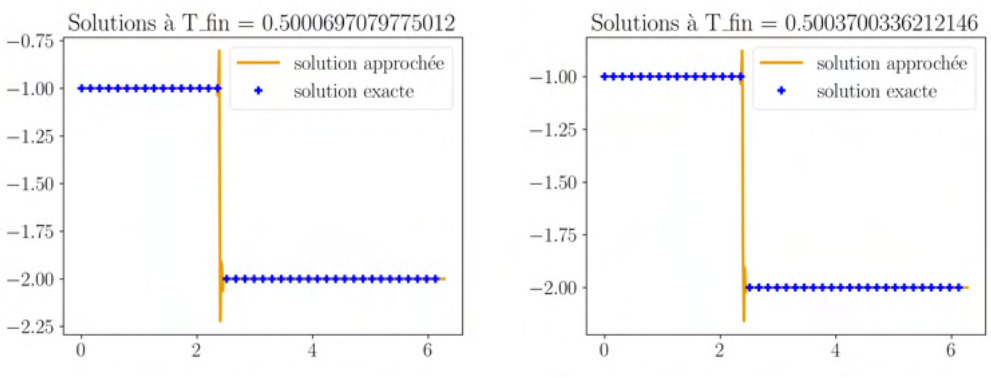
| Erreur calculée | SD | FUSE |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.29793274475712844 | 0.339785204026162 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.22869260545869796 | 0.2186648980745085 |

TABLE 5.9 – Erreurs commises pour les deux méthodes, avec $u_L = 2$ et $u_R = 1$.

| Erreur calculée | SD | FUSE |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.2973578514977129 | 0.339785204026162 |
| $\|u_{ex} - u_{app}\|_{L^\infty(\Omega)}$ | 0.22759868389194038 | 0.2186648980745085 |

TABLE 5.10 – Erreurs commises pour les deux méthodes, avec $u_L = -1$ et $u_R = -2$.

Comme pour les cas d'ondes de détente, nous observons, grâce aux tableaux 5.9 - 5.10 et aux graphiques 5.12 - 5.13, aussi des erreurs similaires pour les deux méthodes et des oscillations au




(a) Méthode SD standard. (b) Méthode FUSE.

FIGURE 5.13 – Solutions pour les deux méthodes au temps $T_{fin} = 0.5$ s, avec $u_L = -1$ et $u_R = -2$.

niveau des chocs à peu près semblables tant pour le SD que pour la méthode FUSE.

## 5.4 Cas linéaire en 2D

Commençons les tests en 2D par l'équation de transport linéaire, définie telle que :

$$\partial_t u + \nabla.\mathbf{f}(u) = 0 \text{ sur } \Omega \times [0, T], \tag{5.14}$$

avec $\Omega \subset \mathbb{R}^2$ le domaine physique et $\mathbf{f}(u) = \mathbf{c}u$ le flux, où $\mathbf{c} = \begin{pmatrix} c_x \\ c_y \end{pmatrix}$ est la vitesse d'advection.

Notons la condition initiale du problème $u_0$ telle que :

$$u_0(\mathbf{x}) = u(\mathbf{x}, 0), \ \mathbf{x} \in \Omega. \tag{5.15}$$

Alors la solution exacte de (5.14) sera :

$$u(\mathbf{x}, t) = u_0(\mathbf{x} - \mathbf{c}t). \tag{5.16}$$

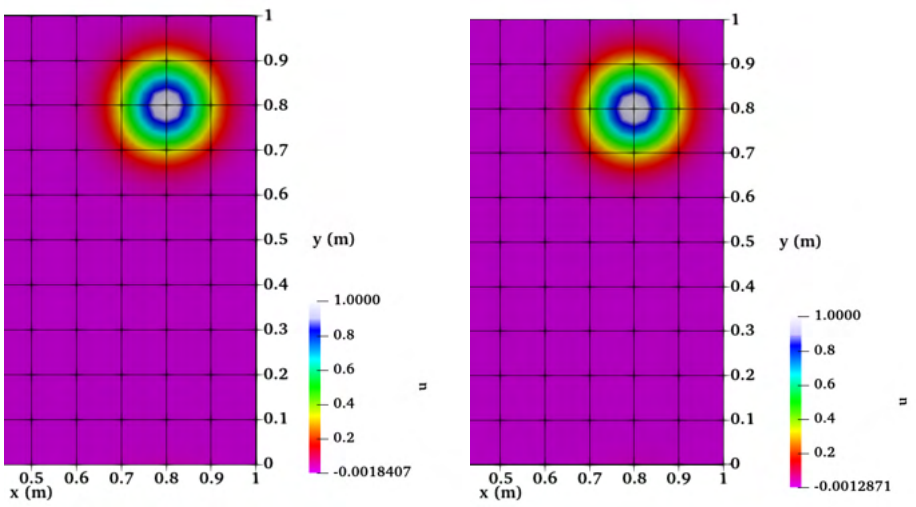
Afin d'effectuer les tests ci-dessous, la méthode FUSE a été implémentée dans HOPPS (cf. partie 5.1), ainsi que le modèle d'advection linéaire (5.14). Nous utiliserons pour les tests présentés dans cette section le schéma d'intégration temporelle SSPRK3. De plus, le solveur de Riemann utilisé pour les SD standards sera le schéma décentré, aussi appelé schéma "upwind".

Dans un premier temps, nous ferons un test où les vitesses d'advection dans les directions $x$ et $y$ seront $c_x = c_y = 1$ m.s$^{-1}$, le domaine de calcul sera $\Omega = [0, L]^2$ avec $L = 1$ m. Le nombre de

cellules sera $N_e = N^2$ avec $N = 10$ et le nombre CFL sera $cfl = 0.2$. Nous utiliserons la condition initiale suivante :

$$u_0\left({}^t(x,y)\right) = \exp\left(-\frac{\left(x-\frac{L}{2}\right)^2 + \left(y-\frac{L}{2}\right)^2}{\sigma^2}\right), \tag{5.17}$$

avec $\sigma = 0,1$. A l'instant $t = 0,3$ s, les solutions approchées avec les méthodes SD et FUSE sont présentées sur la figure 5.14. Les erreurs $L^2$ pour chaque méthode sont notées dans le tableau 5.11.




(a) Méthode SD. (b) Méthode FUSE.

FIGURE 5.14 – Solution au temps $t = 0.000172461$ s obtenue avec les différentes méthodes.

Nous pouvons observer que l'erreur commise par la méthode FUSE est un peu plus grande que celle commise par la méthode SD standard, ce que nous avions déjà observé en 1D.

| **Erreur** | **SD** | **FUSE** |
|---|---|---|
| $\|u_{ex} - u_{app}\|_{L^2(\Omega)}$ | 0.0010072067199307648 | 0.001098513187613855 |

TABLE 5.11 – Erreurs $L^2$ obtenues à $t = 0.3$ s pour les différentes méthodes numériques.

Nous souhaitons maintenant voir si, de façon similaire à ce qui a été fait en 1D, nous obtenons à nouveau des courbes de convergence de pente $p + 1$ pour différents $p$. Ainsi, en faisant varier $N$ nous obtenons les courbes présentées sur la figure 5.15.

Nous observons à nouveau que les erreurs commises par la méthode FUSE sont plus grandes

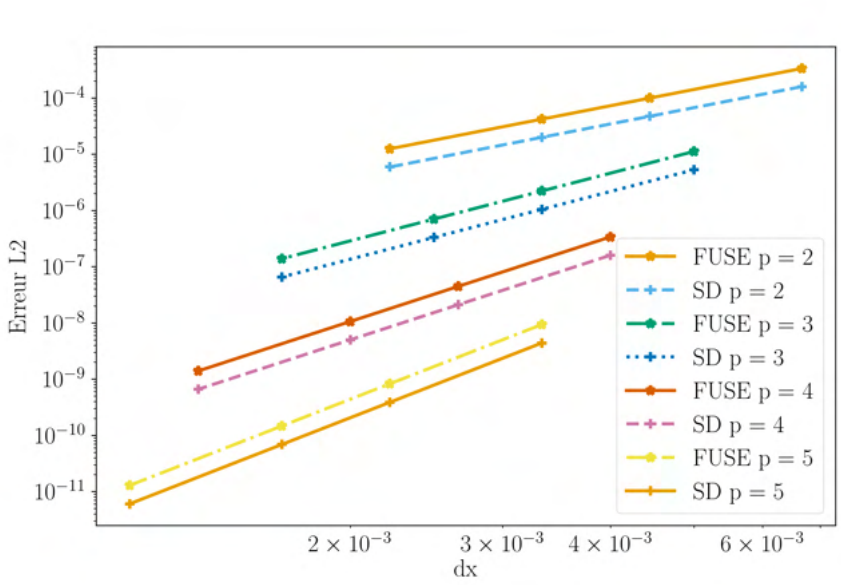



FIGURE 5.15 – Courbes de convergence de l'erreur $L^2$ des méthodes SD et FUSE pour la résolution de l'équation de transport linéaire 2D.

| **Ordre polynomial $p$** | **Pente SD** | **Pente FUSE** | **Pente théorique** |
|---|---|---|---|
| 2 | 2.996000824302383 | 2.9987169114878056 | 3 |
| 3 | 3.998522303898951 | 3.9985872560963345 | 4 |
| 4 | 4.993531009661508 | 4.998535235697732 | 5 |
| 5 | 5.997849683027483 | 5.998509940282256 | 6 |

TABLE 5.12 – Pentes des courbes d'erreur $L^2$ pour la méthode SD et la méthode FUSE.

que celles commises par la méthode SD. Cependant elles restent du même ordre de grandeur, et l'ordre de convergence reste en $p+1$, comme nous le montre le tableau 5.12.

## 5.5 Cas d'un système en 2D : les équations d'Euler

Dans cette section nous nous penchons sur le cas d'un système hyperbolique en 2D : les équations d'Euler, définies telles que :

$$\begin{cases} \partial_t \rho + \nabla.(\rho \mathbf{V}) = 0, \\ \partial_t (\rho \mathbf{V}) + \nabla.(\rho \mathbf{V} \otimes \mathbf{V}) + \nabla p = 0, \\ \partial_t (\rho E) + \nabla.(\rho E \mathbf{V} + p \mathbf{V}) = 0, \end{cases} \tag{5.18}$$

avec $\rho$ la densité du fluide (en kg.m$^{-3}$), $\mathbf{V} = \begin{pmatrix} u \\ v \end{pmatrix}$ sa vitesse (en m.s$^{-1}$), $p$ la pression (en Pa), $E$ l'énergie totale par unité de masse (en J.kg$^{-1}$) telle que $E = \frac{p}{\rho(\gamma-1)} + \frac{\|\mathbf{V}\|^2}{2}$, où $\gamma$ est le rapport des chaleurs spécifiques à pression et volume constant. En réécrivant le système (5.18) sous la forme (2.17), nous avons :

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \ \mathbf{E}(\mathbf{U}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho(E+p)u \end{pmatrix}, \text{ et } \mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho(E+p)v \end{pmatrix}.$$

Afin de vérifier si notre méthode fonctionne pour ce système, nous allons considérer le test d'une COnvection de VOrtex (COVO) sur le domaine $\Omega = [0, L]^2$, avec $L = 0.1$ m. Soit la fonction suivante :

$$\psi(x, y) = \Gamma \exp\left(-\frac{r^2}{2}\right), \tag{5.19}$$

avec :

$$r = \frac{\sqrt{(x - x_c)^2 + (y - y_c)^2}}{R_c}, \tag{5.20}$$

où $\Gamma$ est la force du vortex, $R_c$ son rayon et $(x_c, y_c)$ son centre. Alors la condition initiale pour la vitesse $\mathbf{V}_0 = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix}$ sera :

$$u_0 = U_\infty + \partial_y \psi \text{ et } v_0 = -\partial_x \psi, \tag{5.21}$$

avec $U_\infty = M_\infty \sqrt{\gamma R_{gaz} T_\infty}$ la vitesse du fluide sans perturbation, $R_{gaz}$ la constante du gaz et $M_\infty$ le nombre de Mach. De manière analogue $T_\infty$, $\rho_\infty$ et $p_\infty$ seront respectivement la température, densité et pression du fluide sans perturbation. Les conditions initiales pour ces quantités seront :

$$T_0 = T_\infty - \frac{\Gamma^2 \exp\left(-\frac{r^2}{R_c^2}\right)}{2C_p R_c^2}, \tag{5.22}$$

$$\rho_0 = \rho_\infty \left(\frac{T_0}{T_\infty}\right)^{\frac{1}{\gamma-1}}, \tag{5.23}$$

$$p_0 = \rho_0 R_{gaz} T_0. \tag{5.24}$$

Les valeurs des constantes utilisées dans les conditions initiales (5.21)-(5.24) sont présentées dans le tableau 5.13.

| **Constantes** | **Valeur** |
|---|---|
| $\gamma$ (-) | 1,4 |
| $R_{gaz}$ (J.kg$^{-1}$.K) | 287 |
| $C_p$ (J.kg$^{-1}$.K) | $\frac{\gamma R_{gaz}}{\gamma-1}$ |
| $M_\infty$ (-) | 0,5 |
| $p_\infty$ (Pa) | 101300 |
| $T_\infty$ (K) | 300 |
| $\rho_\infty$ (kg.m$^{-3}$) | 1,17170407 |
| $R_c$ (m) | $\frac{L}{20}$ |
| $x_c$ (m) | $\frac{L}{2}$ |
| $y_c$ (m) | $\frac{L}{2}$ |
| $\Gamma$ (m$^2$.s$^{-1}$) | $0,04 U_\infty R_c \sqrt{e}$ |

TABLE 5.13 – Tableau des valeurs des différentes constantes utilisées à l'initialisation du COVO.

Notons donc la condition initiale :

$$\mathbf{U_0} = \begin{pmatrix} \rho_0 \\ \rho_0 u_0 \\ \rho_0 v_0 \\ \rho_0 E_0 \end{pmatrix}. \tag{5.25}$$

Alors la solution exacte de ce problème sera :

$$\mathbf{U}(\mathbf{x}, t) = \mathbf{U_0}(\mathbf{x} - \mathbf{c}_\infty t), \tag{5.26}$$

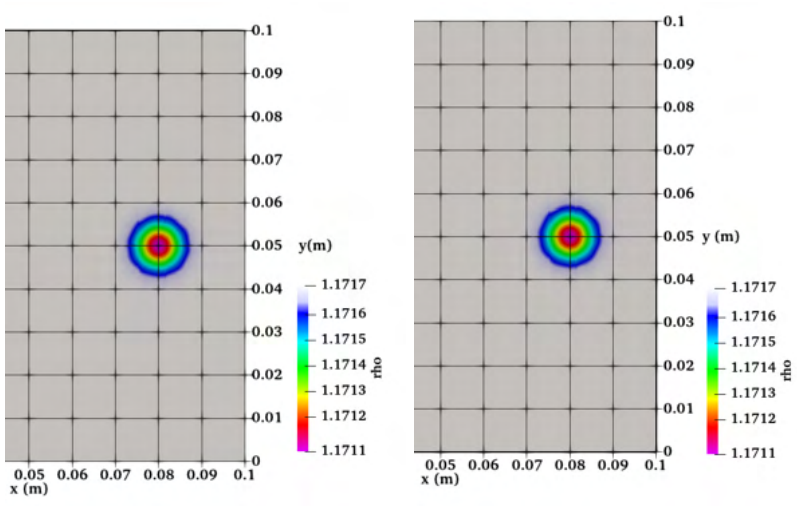
avec $\mathbf{c}_\infty = \begin{pmatrix} U_\infty \\ 0 \end{pmatrix}$.

Dans un premier temps, nous prendrons comme ordre $p = 4$, comme nombre de cellules $N_e = N^2$ avec $N = 16$, comme nombre CFL $cfl = 0.2$ et le schéma d'intégration temporelle RK54. Nous obtenons donc au temps $t = 0.000172461$ s les solutions présentées sur la figure 5.16 pour la densité $\rho$.

| **Erreur** | **SD** | **FUSE** |
|---|---|---|
| $\|\rho_{ex} - \rho_{app}\|_{L^2(\Omega)}$ | 2.0638254376962932e-07 | 2.076265289437885e-07 |

TABLE 5.14 – Erreurs $L^2$ obtenues pour la variable $\rho$ à $t = 0.000172461$ s pour les différentes méthodes numériques.

Nous observons à nouveau grâce aux erreurs $L^2$ données dans le tableau 5.14 que les solutions numériques calculées avec les deux méthodes sont similaires, avec une erreur un petit peu plus grande pour la méthode FUSE, ce que nous avions déjà remarqué pour les cas d'advection linéaire.




(a) Méthode SD. (b) Méthode FUSE.

FIGURE 5.16 – Densité $\rho$ au temps $t = 0.000172461$ s obtenue avec les différentes méthodes.

Nous souhaitons donc faire à nouveau les courbes de convergence pour plusieurs ordres $p$. Nous obtenons ainsi le graphique 5.17 et les pentes des courbes sont notées dans le tableau 5.15.

| **Ordre polynomial $p$** | **Pente SD** | **Pente FUSE** | **Pente théorique** |
|---|---|---|---|
| 2 | 2.995668678345321 | 2.9940116980981473 | 3 |
| 3 | 3.994490399203938 | 3.993475252437675 | 4 |
| 4 | 4.993933309211478 | 4.993231994444279 | 5 |
| 5 | 5.866559387453854 | 5.964326894123795 | 6 |

TABLE 5.15 – Pentes des courbes d'erreur $L^2$ pour la méthode SD et la méthode FUSE.

Comme pour le modèle d'advection linéaire en 1D et 2D, nous obtenons bien un ordre de convergence de $p+1$ pour les deux méthodes. Nous pouvons aussi observer que, à l'instar de ce qui a été obtenu pour le transport linéaire, l'erreur commise par la méthode FUSE est plus grande que celle commise par la méthode SD standard.

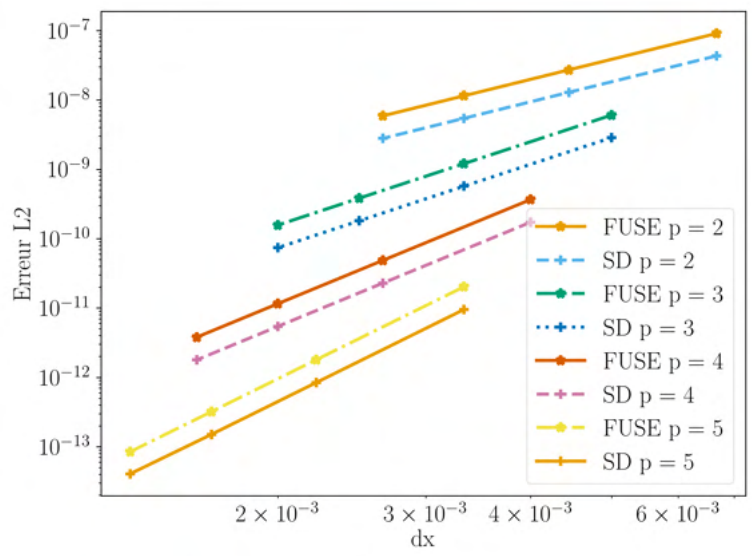



FIGURE 5.17 – Courbes de convergence de l'erreur $L^2$ pour la densité $\rho$ des méthodes SD et FUSE pour la résolution des équations d'Euler 2D.

# 6 Analyse de performance

## 6.1 Complexité algorithmique

Nous souhaitons quantifier le gain en termes d'opérations de la méthode FUSE par rapport à la méthode SD pour le cas d'une équation de transport linéaire en 1D.

Soit $N_e$ le nombre de cellules du maillage, $N_{SP} = p+1$ le nombre de points solution par cellule et $N_{FP} = p+2$ le nombre de points flux. A un temps $t_n$ donné, l'étape d'extrapolation coûte $N_e N_{FP} N_{SP}$ opérations pour la méthode SD standard, tandis qu'il y a seulement $N_e N_{SP}$ opérations pour la méthode FUSE. Appliquer les conditions aux limites coûte 2 opérations pour chaque méthode. L'étape de calcul du flux aux points intérieurs vaut $N_e(N_{FP}-2)$ pour les SD standard, mais elle coûte $N_e N_{FP}$ pour la méthode FUSE car nous n'utilisons pas de solveur de Riemann donc nous pouvons aussi calculer le flux aux points au bord de chaque cellule. Soit $c_{Riemann}$ la complexité du solveur de Riemann utilisé, alors cette étape vaut $N_e c_{Riemann}$ pour la méthode SD. Cependant, comme expliqué précédemment, il n'y a pas besoin de cette étape pour la méthode FUSE. L'évaluation de la dérivée aux points solution coûte $N_e N_{SP} N_{FP}$ opérations dans les deux cas. Enfin, en notant $c_{RK}$ la complexité du schéma d'intégration temporelle choisi, alors l'étape d'intégration temporelle coûte $N_e N_{SP} c_{RK}$ opérations. Au total, en notant $N_{iter}$ le nombre d'itérations en temps :

— Pour la méthode SD :

$$\begin{aligned} &N_{iter}\left(N_e N_{FP} N_{SP} + 2 + N_e(N_{FP}-2) + N_e c_{Riemann} + N_e N_{SP} N_{FP} + N_e N_{SP} c_{RK}\right)\mathcal{O}(1) \\ &\quad = \left(1 + \frac{2}{N_e N_{FP} N_{SP}} + \frac{N_{FP} - 2 + c_{Riemann}}{N_{FP} N_{SP}} + \frac{c_{RK}}{N_{FP}}\right)\mathcal{O}(N_{iter} N_e N_{FP} N_{SP}). \end{aligned}$$

— Pour la méthode FUSE :

$$\begin{aligned} &N_{iter}\left(N_e N_{SP} + 2 + N_e N_{FP} + N_e N_{SP} N_{FP} + N_e N_{SP} c_{RK}\right)\mathcal{O}(1) \\ &= \left(\frac{1}{N_{FP}} + \frac{2}{N_e N_{FP} N_{SP}} + \frac{1}{N_{SP}} + 1 + \frac{c_{RK}}{N_{FP}}\right)\mathcal{O}(N_{iter} N_e N_{FP} N_{SP}). \end{aligned}$$

Nous pouvons donc noter que les deux algorithmes ont une complexité en $\mathcal{O}(N_{iter} N_e N_{FP} N_{SP})$. Cependant la constante pour la méthode FUSE est plus petite que celle pour la méthode SD.

Nous souhaitons à présent comparer le nombre d'opérations pour les différents algorithmes présentés pour une équation non linéaire 1D. La méthode SD reste la même qu'en linéaire. pour l'algorithme FUSE où la position des points flux évolue, nous noterons $N^n_{FUSE}$ le nombre de cellules où, à un temps $t_n$ donné, les points solution sont collocalisés avec certains points flux, et $N^n_{SD}$ le

nombre de cellules où les points solution sont les points de Gauss-Tchebychev. Ainsi, pour tout $t_n$, $N_{SD}^n + N_{FUSE}^n = N_e$.

A un temps $t_n$ donné, l'étape d'extrapolation coûte $N_e N_{SP} + N_e N_{SP} = 2N_e N_{SP}$ pour l'algorithme avec points fixés, tandis qu'elle en coûte $N_{FUSE}^n N_{SP} + N_{SD}^n N_{SP} N_{FP}$ pour l'algorithme avec points adaptatifs. L'application des conditions aux limites coûte 2 opérations pour les deux algorithmes. Le calcul du flux aux points intérieurs vaut $N_e(N_{FP} - 2)$ opérations pour l'algorithme avec points fixés, mais elle en vaut $N_{FUSE}^n N_{FP} + N_{SD}^n (N_{FP} - 2)$ pour le deuxième algorithme. Pour l'étape du solveur de Riemann, nous avons, comme pour la méthode SD usuelle, $N_e c_{Riemann}$ opérations pour l'algorithme avec points fixés, mais nous en avons seulement $(N_{SD}^n + 1)c_{Riemann}$ pour l'algorithme avec points adaptatifs. Ce dernier contient une étape supplémentaire où $a(u)$ est calculé à chaque interface afin d'adapter la position des points solution pour l'itération suivante. Cette étape vaut donc $N_e + 1$ opérations. Enfin, les étapes de dérivation du flux aux points solution et l'intégration temporelle valent $N_e N_{FP} N_{SP} + N_e N_{SP} c_{RK}$ opérations pour les deux algorithmes. Tout cela nous donne finalement :

— Pour l'algorithme avec points fixés :

$$\begin{aligned} &N_{iter}\left(2N_e N_{SP} + 2 + N_e(N_{FP} - 2) + N_e c_{Riemann} + N_e N_{SP} N_{FP} + N_e N_{SP} c_{RK}\right)\mathcal{O}(1) \\ &= \left(\frac{2}{N_{FP}} + \frac{2}{N_e N_{FP} N_{SP}} + \frac{N_{FP} - 2 + c_{Riemann}}{N_{FP} N_{SP}} + 1 + \frac{c_{RK}}{N_{FP}}\right)\mathcal{O}(N_{iter} N_e N_{FP} N_{SP}). \end{aligned}$$

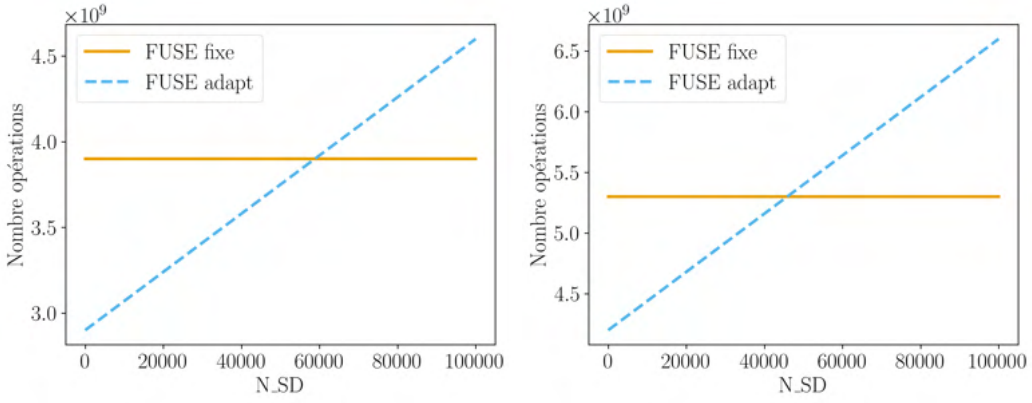
— Pour l'algorithme avec points adaptatifs :

$$\begin{aligned} N_{iter}\,&(N_{FUSE}^n N_{SP} + N_{SD}^n N_{SP} N_{FP} + 2 + N_{FUSE}^n N_{FP} + N_{SD}^n (N_{FP} - 2) \\ &+ (N_{SD}^n + 1)c_{Riemann} + N_e + 1 + N_e N_{SP} N_{FP} + N_e N_{SP} c_{RK})\,\mathcal{O}(1), \\ &= \left(1 + \frac{1}{N_{SP}} + \frac{c_{RK} + 1}{N_{FP}} + \frac{1}{N_{SP} N_{FP}} + \frac{3 + c_{Riemann}}{N_e N_{FP} N_{SP}}\right. \\ &\left. + \frac{N_{SD}^n}{N_e}\left(1 - \frac{1}{N_{FP}} + \frac{c_{Riemann} - 2}{N_{SP} N_{FP}}\right)\right)\mathcal{O}(N_{iter} N_e N_{FP} N_{SP}). \end{aligned}$$

Ainsi, en admettant que $N_{SD}^n$ reste constant pour chaque itération, l'algorithme avec points adaptatifs vaut plus d'opérations que l'algorithme avec points fixés lorsque :

$$N_{SD}^n = N_{SD} > \frac{N_e N_{SP} + N_e c_{Riemann} - 2N_e - c_{Riemann} - N_e - 1}{N_{SP} N_{FP} - N_{SP} - 2 + c_{Riemann}} = N_{SD}^{max}. \tag{6.1}$$

A titre d'illustation, fixons $N_{iter} = 1000$, $c_{RK} = 3$, $c_{Riemann} = 10$ et $N_e = 100000$. En faisant varier $N_{SD}$, nous obtenons les graphiques 6.1 pour $p = 2$ et $p = 3$.




(a) $p = 2$. (b) $p = 3$.

FIGURE 6.1 – Nombre d'opérations des deux algorithmes en fonction de $N_{SD}$.

Pour l'exemple avec $p = 2$, $N_{SD}^{max} = 58822$, et pour $p = 3$, $N_{SD}^{max} = 45832$. Nous pouvons donc observer qu'en augmentant $p$ nous faisons baisser $N_{SD}^{max}$. En conséquence, l'algorithme avec points adaptatifs est plus intéressant en terme de coût de calcul pour des cas avec peu de variations dans le sens du flux. Par exemple, pour le cas de l'équation de Burgers avec la condition initiale (5.8) sur le domaine $[0, 2\pi]$, il est plus intéressant d'utiliser l'algorithme 2 car il n'y a un changement de signe de $a(u) = u$ qu'à $x = \pi$.

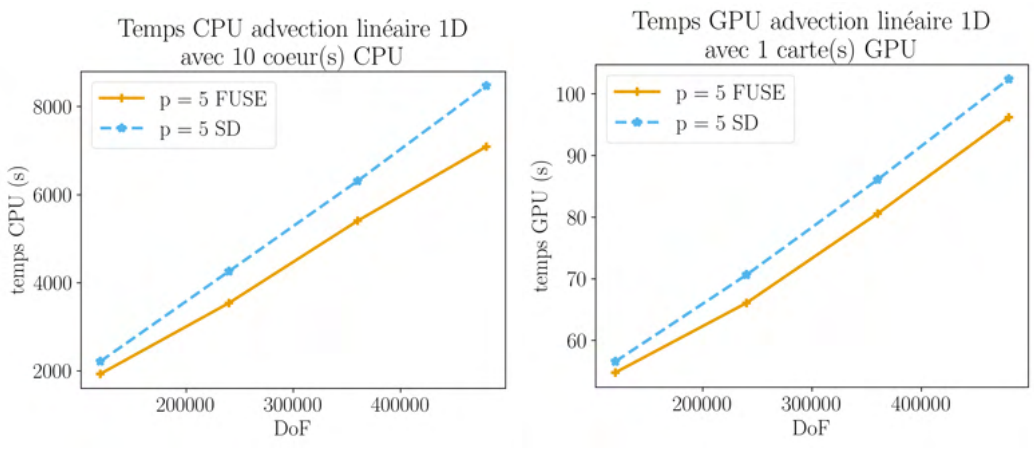
## 6.2 Tests de performance

### 6.2.1 Tests en 1D avec HOPPS

Nous souhaitons maintenant comparer la performance des deux méthodes, mais aussi comparer les performances obtenues sur différentes architectures matérielles. Pour ce faire, nous allons utiliser un des supercalculateurs du CERFACS, nommé Kraken, qui dipose de noeuds composés de coeurs CPU, mais aussi de cartes GPU Nvidia A30. Nous ferons des tests avec ce qui a été implémenté dans HOPPS.

Commençons par des tests pour le modèle d'advection linéaire en 1D sur le domaine $\Omega = [0, 6\pi]$ et pour un ordre $p = 5$. Les temps CPU pour 10 coeurs CPU et pour 1 carte GPU sont présentés dans le graphique 6.2 pour différents nombres de degrés de liberté, c'est-à-dire $N_{DoF} = N_e \times (p+1)$ en 1D.

Nous pouvons noter dans un premier temps que, en GPU comme en CPU, la méthode FUSE est plus rapide que la méthode SD, ce qui est cohérent avec les complexités algorithmiques calculées dans le paragraphe 6.1. En notant $T_{SD}$ le temps de calcul pour la méthode SD et $T_{FUSE}$ le temps




(a) Temps CPU pour 10 coeurs CPU.

(b) Temps GPU pour 1 carte GPU.

FIGURE 6.2 – Temps CPU/GPU pour la résolution numérique avec la méthode FUSE et la méthode SD du modèle de transport linéaire 1D avec $p = 5$ pour différents nombres de degrés de liberté.

| $N_{DoF}$ | $\frac{T_{SD}-T_{FUSE}}{T_{SD}}$ GPU | $\frac{T_{SD}-T_{FUSE}}{T_{SD}}$ CPU |
|---|---|---|
| 120000 | 0.03157929 | 0.1280321 |
| 240000 | 0.0644844 | 0.16905409 |
| 360000 | 0.06384301 | 0.14354575 |
| 480000 | 0.0606742 | 0.16304394 |

TABLE 6.1 – Différence relative de temps GPU et CPU entre la méthode FUSE et la méthode SD pour le modèle d'advection linéaire 1D pour différents nombres de degrés de liberté $N_{DoF}$.

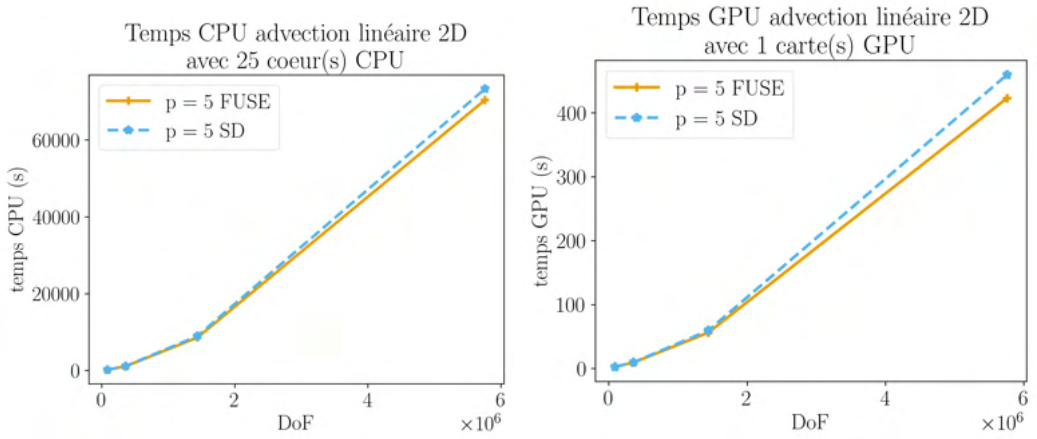
de calcul pour la méthode FUSE, le tableau 6.1 réunit les différences relatives entre le temps CPU (ou GPU) de la méthode FUSE et de la méthode SD par rapport au temps de la méthode SD. Nous pouvons ainsi observer un gain d'environ 6% du temps pour la méthode FUSE par rapport à la méthode SD sur carte GPU. Pour le calcul sur CPU, nous gagnons même de 12% à 16% de temps CPU. Une étude de scalabilité aurait potentiellement pu expliquer cette différence de gain de temps de calcul entre CPU et GPU.

Nous pouvons aussi observer que les temps de calcul pour 1 carte GPU sont nettement plus petits que ceux obtenus avec 10 coeurs CPU.

### 6.2.2 Tests en 2D avec HOPPS

Nous souhaitons effectuer le même type de tests que ceux présentés dans la partie 6.2.1 pour des cas 2D. Nous commençons par le modèle d'advection linéaire 2D sur le domaine $\Omega = [0, 1]^2$ avec un ordre $p = 5$. Nous présentons dans le graphique 6.3 le temps CPU et GPU pour différents nombres

de degrés de liberté $N_{DoF} = N_e(p+1)^2$.




(a) Temps CPU pour 25 coeurs CPU.

(b) Temps GPU pour 1 carte GPU.

FIGURE 6.3 – Temps CPU/GPU pour la résolution numérique avec la méthode FUSE et la méthode SD du modèle de transport linéaire 2D avec $p = 5$ pour différents nombres de degrés de liberté.

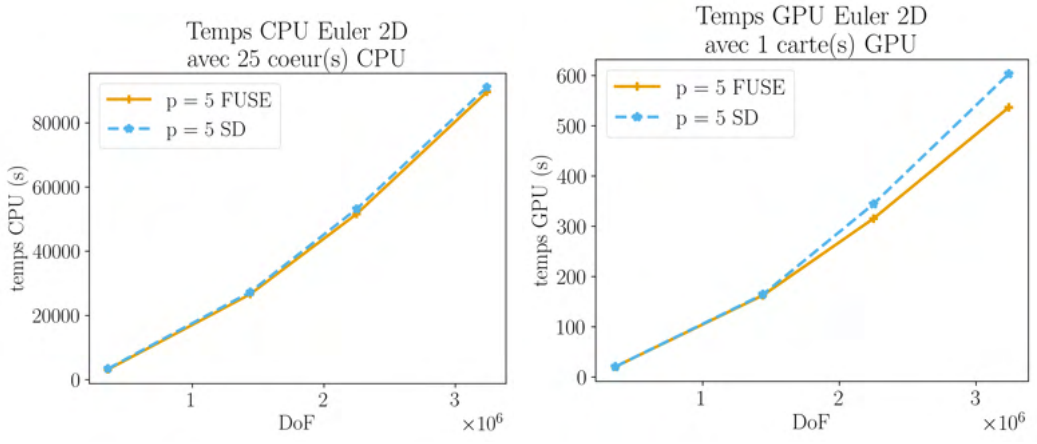
Nous observons à nouveau que, sur GPU comme sur CPU, le temps de calcul pour la méthode FUSE est plus petit que celui de la méthode SD standard. Grâce au tableau 6.2 qu'à partir de $10^5$ degrés de liberté, nous gagnons 3 à 7% de temps de calcul sur GPU et environ 4% sur CPU.

| $N_{DoF}$ | $\frac{T_{SD}-T_{FUSE}}{T_{SD}}$ GPU | $\frac{T_{SD}-T_{FUSE}}{T_{SD}}$ CPU |
|---|---|---|
| 90000 | -0.08754912 | 0.00465952 |
| 360000 | 0.03756522 | 0.04175313 |
| 1440000 | 0.05422712 | 0.04469892 |
| 5760000 | 0.07954531 | 0.03966221 |

TABLE 6.2 – Différence relative de temps GPU et CPU entre la méthode FUSE et la méthode SD pour le modèle d'advection linéaire 2D pour différents nombres de degrés de liberté $N_{DoF}$.

Regardons enfin les temps CPU et GPU pour la résolution des équation d'Euler en 2D avec la méthode SD et la méthode FUSE (avec points fixes). Nous obtenons le graphique 6.4 (a) pour les calculs lancés sur 25 coeurs CPU et le graphique 6.4 (b) pour ceux lancés sur 1 carte GPU.

Nous pouvons faire les mêmes observations que pour les cas linéaires : la méthode FUSE est plus rapide que la méthode SD, même si nous pouvons voir grâce au tableau 6.3 que le gain de temps est moins conséquent sur CPU que sur GPU.




(a) Temps CPU pour 25 coeurs CPU. (b) Temps GPU pour 1 carte GPU.

FIGURE 6.4 – Temps CPU/GPU pour la résolution numérique avec la méthode FUSE et la méthode SD des équations d'Euler 2D avec $p = 5$ pour différents nombres de degrés de liberté.

| $N_{DoF}$ | $\frac{T_{SD}-T_{FUSE}}{T_{SD}}$ GPU | $\frac{T_{SD}-T_{FUSE}}{T_{SD}}$ CPU |
|---|---|---|
| 360000 | 0.00524574 | 0.07538979 |
| 1440000 | 0.00980509 | 0.02490826 |
| 2250000 | 0.0842039 | 0.02904119 |
| 3240000 | 0.11064584 | 0.01663689 |

TABLE 6.3 – Différence relative de temps GPU et CPU entre la méthode FUSE et la méthode SD pour les équations d'Euler 2D pour différents nombres de degrés de liberté $N_{DoF}$.

# 7 Conclusions et perspectives

Pendant ce stage, nous avons adapté la méthode FUSE au formalisme des Différences Spectrales, en collocalisant les points solution avec certains points flux. Nous avons montré qu'elle était stable en 1D et en 2D avec maillages hexahédriques. Elle a été implémentée pour des cas linéaires et non linéaires 1D et 2D dans HOPPS. Les tests numériques effectués ont montré que, bien que les erreurs commises par la méthode FUSE soient un peu plus grandes que celles de la méthode SD standard, elles restent du même ordre de grandeur et l'ordre de convergence en $p+1$ est conservé. Une étude de la complexité algorithmique des deux méthodes a montré que la méthode FUSE a un coût plus bas en termes d'opérations que la méthode SD. Ceci a été confirmé par les temps de calcul CPU et GPU en 1D et en 2D, qui se sont avérés plus rapides pour la méthode FUSE que pour la méthode SD.

Les deux algorithmes proposés en scalaire n'ont pas pu être comparés numériquement pendant ce stage, l'algorithme avec points adaptatifs pourrait donc être, à l'avenir, codé dans HOPPS. De plus, nous pourrions faire des tests en 3D pour des maillages hexahédriques. De même, la méthode pourrait être adaptée pour les cas d'équations avec un flux visqueux (notamment les équations de Navier-Stokes) afin de réaliser des cas tests sur des configurations physiques plus réalistes. Enfin, la structure tensorisée des maillages hexahédriques nous a permis de généraliser la méthode 1D au cas 2D assez facilement. Une adaptation aux maillages triangulaires pourrait donc être une autre piste de travail à explorer.

# Références


[1] Marlon Mesquita Lopes Cabreira et Renan de S Teixeira. « Strong Stability Preserving Runge-Kutta Methods Applied to Advection-Diffusion Problem ». In : *XLI Ibero-Latin American Congress on Computational Methods in Engineering*. T. 2. 02. 2020.

[2] Jan S Hesthaven et Tim Warburton. *Nodal discontinuous Galerkin methods : algorithms, analysis, and applications*. Springer Science & Business Media, 2007.

[3] Hung T Huynh. « A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods ». In : *18th AIAA computational fluid dynamics conference*. 2007, p. 4079.

[4] Antony Jameson. « A proof of the stability of the spectral difference method for all orders of accuracy ». In : *Journal of Scientific Computing* 45 (1-3 oct. 2010), p. 348-358. issn : 08857474. doi : 10.1007/s10915-009-9339-4.

[5] Yen Liu, Marcel Vinokur et Zhi Jian Wang. « Spectral difference method for unstructured grids I : Basic formulation ». In : *Journal of Computational Physics* 216.2 (2006), p. 780-801.

[6] Thomas Marchal. *Extension of the Spectral Difference method to combustion*. Avr. 2022.

[7] Nadir-Alexandre Messaï, Guillaume Daviller et Jean-François Boussuge. « Artificial viscosity-based shock capturing scheme for the Spectral Difference method on simplicial elements ». In : *Journal of Computational Physics* 504 (mai 2024), p. 112864. issn : 00219991. doi : 10.1016/j.jcp.2024.112864.

[8] Yulong Pan et Per-Olof Persson. « A Face-Upwinded Spectral Element Method ». In : (juin 2023).

[9] Yushi Sun, Z. J. Wang et Yen Liu. « High-Order Multidomain Spectral Difference Method for the Navier-Stokes Equations on Unstructured Hexahedral Grids ». In : *Communications in Computational Physics* 2 (2 avr. 2006), p. 310-333.

[10] Christian R. Trott et al. « Kokkos 3 : Programming Model Extensions for the Exascale Era ». In : *IEEE Transactions on Parallel and Distributed Systems* 33.4 (2022), p. 805-817. doi : 10.1109/TPDS.2021.3097283.

[11] Meilin Yu, Z. J. Wang et Yen Liu. « On the accuracy and efficiency of discontinuous Galerkin, spectral difference and correction procedure via reconstruction methods ». In : *Journal of Computational Physics* 259 (fév. 2014), p. 70-95. issn : 10902716. doi : 10.1016/j.jcp.2013.11.023.