Schéma ALE aléatoire pour les écoulements bifluides compressibles. Application à la simulation du déferlement.

Philippe Helluy (et Olivier Hurisse, Jonathan Jung)

IRMA Strasbourg, France

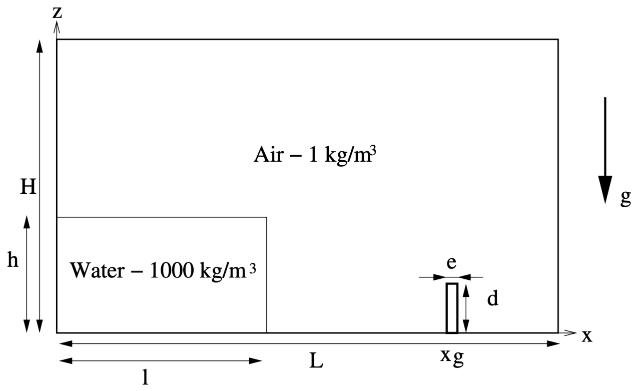
Séminaire LAMA, Chambéry, Janvier 2025

Outlines

Two-fluid model

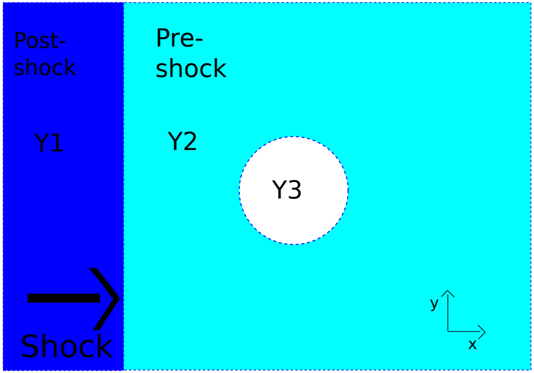
Physical problem I

Wave breaking



Physical problem II

Shock-bubble or shock-droplet interaction



Systems of conservation laws

The unknown is a vector $W(X,t) \in \mathbb{R}^m$ that depends on space $X \in \mathbb{R}^d$ $(d=2)$, time $t \in \mathbb{R}$ and satisfies

$$\partial_t W + \nabla_X \cdot F(W) = S(W).$$

The flux F is supposed to be hyperbolic in all the directions $N \in \mathbb{R}^d$:

$$A(W,N) := \nabla_W F(W) \cdot N$$

is diagonalizable with real eigenvalues.

- The solutions are complex: shock waves, non uniqueness, turbulence, *etc.*
- Needs of precise and robust numerical methods.
- Algorithms must be adapted to multicore computers.

Compressible two-fluid model

Vector of conservative variables $W = (\rho, \rho u, \rho v, \rho Q, \rho\varphi)^T$, where

- ρ is the density,
- $U = (u, v)^T$ is the velocity vector,
- Q is the total energy,
- φ the color function ($\varphi = 0$ in the liquid and $\varphi = 1$ in the gas).
- The internal energy is $e = Q - (u^2 + v^2)/2$.
- The pressure is defined by $p = p(\rho, e, \varphi)$.
- The flux and source are given by

$$F(W)\cdot N = (\rho U\cdot N, \rho(U\cdot N)U^T + pN^T, (\rho Q + p)U\cdot N, \rho\varphi U\cdot N)^T,$$

$$S(W) = (0, 0, -g, -\rho g v, 0)^T, \quad g = 9.81\mathrm{m/s}^2.$$

Diffusion of the color function

The color function is a solution of

$$\partial_t \varphi + U \cdot \nabla \varphi = 0,$$

thus

$$\forall (x,t) \quad \varphi(x,t) \in \{0\} \cup \{1\}.$$

However most numerical schemes will produce numerical diffusion and we have to interpolate the pressure law $p(\rho, e, \varphi)$ for

$$\varphi \in]0,1[.$$

Pressure law

For instance, we can consider a simple stiffened gas model for an air-water mixture

$$p(\rho, e, \varphi) = (\gamma(\varphi) - 1)\rho e - \gamma(\varphi)\pi(\varphi).$$

The gas corresponds to $\varphi = 1$:

$$\gamma(1) = \gamma_1 = 1.4, \quad \pi(1) = \pi_1 = 0 \text{ (perfect gas)}.$$

The liquid corresponds to $\varphi = 0$:

$$\gamma(0) = \gamma_2 = 3, \quad \pi(0) = \pi_2 = 8500 \times 10^5 \text{ Pa (stiffened gas)}.$$

We can use a linear interpolation of $1/(\gamma - 1)$ and $\gamma\pi/(\gamma - 1)$ for $0 < \varphi < 1$ [13].

Hyperbolicity

Let

$$\mathscr{W}_{ad}(\varphi)=\left\{W=(\rho,\rho u,\rho v,\rho Q,\rho\varphi)^T\in\mathbb{R}^m,\quad \rho>0,\quad p+\pi(\varphi)>0\right\},$$

and

$$\mathscr{W}_{ad}=\bigcup_{\varphi\in[0,1]}\mathscr{W}_{ad}(\varphi).$$

Let $c=\sqrt{\gamma(p+\pi)/\rho}$. The system is hyperbolic for $W\in\mathscr{W}_{ad}$ with eigenvalues $U\cdot N-c$, $U\cdot N$, $U\cdot N+c$. The pressure can be <0 (liquid tension).
For a given φ, the set $\mathscr{W}_{ad}(\varphi)$ is convex. But $\mathscr{W}_{ad}$ is not convex.

Riemann solver

First we consider the 1D framework $X = (x, y)^T$, $W = W(x, t)$, $N = (1, 0)^T$,

$$\partial_t W + \partial_x (F(W) \cdot N) = 0.$$

Let V_L and V_R be two constant states in $\mathscr{W}_{ad}$. We can prove that the Riemann problem

$$\partial_t V + \partial_x (F(V) \cdot N) = 0$$

$$V(x, 0) = \begin{cases} V_L \text{ if } x < 0, \\ V_R \text{ if } x \geq 0, \end{cases}$$

admits a unique global entropy solution, which is denoted by

$$R(V_L, V_R, x/t) = V(x, t) \in \mathscr{W}_{ad}.$$

The function R is called the Riemann solver. The negative pressures are not a problem.

Mesh

- We consider a 1D mesh made of cells $C_i =]x_{i-1/2}, x_{i+1/2}[$, $i \in \mathbb{Z}$. The size of cell C_i is $\Delta x_i = x_{i+1/2} - x_{i-1/2}$.
- We also consider time steps $\Delta t_n > 0$ satisfying a CFL condition and a sequence of times t_n satisfying $t_{n+1} = t_n + \Delta t_n$.
- The solution $W(x,t)$ is approximated in each cell by a constant value

$$W_i^n \simeq W(x, t_n), \quad x \in C_i^n.$$

Godunov scheme

The Godunov scheme reads

$$\Delta x_i \left(W_i^{n+1} - W_i^n \right) + \Delta t_n \left(F_{i+1/2}^n - F_{i-1/2}^n \right) = 0.$$

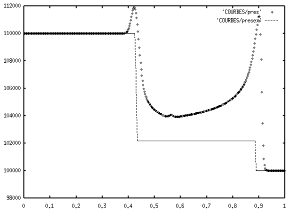
The numerical flux is defined from the Riemann solver $R(W_L, W_R, x/t)$ and

$$\begin{aligned} F_{i+1/2}^n &= F(W_{i+1/2}^n), \\ W_{i+1/2}^n &= R(W_i^n, W_{i+1}^n, 0). \end{aligned}$$

Construction: (1) exact resolution of interface Riemann problems. (2) averaging over the cells. In the convex case, CFL condition and entropy stability follow from Jensen inequality.

Pressure "oscillations"

Problem: $\mathscr{W}_{ad}$ is generally not convex. The Godunov scheme is not stable and may fail after only one time step [12].
Even when the computations are possible, the results are not accurate (spurious pressure "oscillations").



Better accuracy with the non-conservative scheme of Abgrall-Saurel [13], but with the same stability issue [11].

Possible cures

We can:

1. Construct another pressure law that ensures convexity of $\mathscr{W}_{ad}$. It's possible, we can discuss it during the lunch...
2. Construct another scheme that keeps W_i^n in $\mathscr{W}_{ad}$.

Random Interface Sampling

Lagrange and remap schemes

We consider the family of Lagrange plus remap schemes. The mesh is now moving within a time step. The cells depend on n

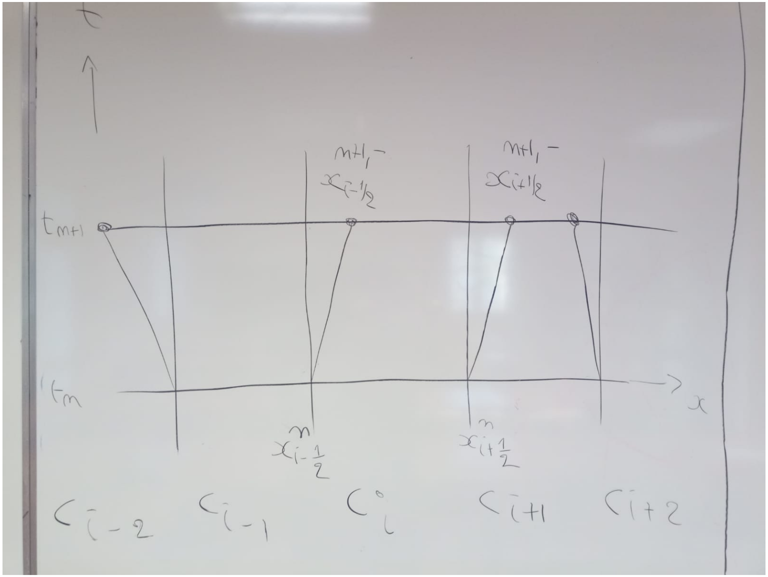
$$C_i^n =]x_{i-1/2}^n, x_{i+1/2}^n[, \quad \Delta x_i^n = x_{i+1/2}^n - x_{i-1/2}^n.$$

The cell boundary $x_{i+1/2}^n$ moves at the velocity $u_{i+1/2}^n$. Just before the remap step (time "$n+1,-$") the cell boundaries are given by

$$x_{i+1/2}^{n+1,-} = x_{i+1/2}^n + \Delta t_n u_{i+1/2}^n.$$

$$\Delta x_i^{n+1,-} = x_{i+1/2}^{n+1,-} - x_{i-1/2}^{n+1,-} = \Delta x_i^n + \Delta t_n (u_{i+1/2}^n - u_{i-1/2}^n).$$

Lagrange-remap mesh



Lagrange and remap schemes

Each time step of a Lagrange plus remap scheme is made of two stages.
In the first stage, we approximate the solution with a Lagrange scheme

$$\Delta x_i^{n+1,-} W_i^{n+1,-} - \Delta x_i^n W_i^n + \Delta t_n \left(F_{i+1/2}^n - F_{i-1/2}^n \right) = 0.$$

The Lagrange flux is defined from a Riemann solver $R(W_L, W_R, x/t)$ and

$$\begin{array}{rcl} F_{i+1/2}^n & = & F(W_{i+1/2}^n) - u_{i+1/2}^n W_{i+1/2}^n, \\ W_{i+1/2}^n & = & R(W_i^n, W_{i+1}^n, u_{i+1/2}^n), \end{array}$$

Conservative remap

The classic remap step consists in returning to the Euler grid with conservative averaging. We obtain

$$W_i^{n+1} = W_i^{n+1,-} - \frac{\Delta t_n}{\Delta x_i}\Big(\max(u_{i-1/2}^n, 0)(W_i^{n+1,-} - W_{i-1}^{n+1,-}) + \min(u_{i+1/2}^n, 0)(W_{i+1}^{n+1,-} - W_i^{n+1,-})\Big).$$

And we go back to the initial Euler grid

$$x_i^{n+1} = x_i^n, \quad C_i^{n+1} = C_i^n, \quad \Delta x_i^{n+1} = \Delta x_i^n.$$

The numerical results are similar to those obtained with the conservative scheme.

Non-conservative remap

Instead of averaging $\rho\varphi$, the last component of W, we average φ [2], which leads to

$$\varphi_i^{n+1} = \varphi_i^{n+1,-} - \frac{\Delta t_n}{\Delta x_i}\Big(\max(u_{i-1/2}^n, 0)(\varphi_i^{n+1,-} - \varphi_{i-1}^{n+1,-}) + \min(u_{i+1/2}^n, 0)(\varphi_{i+1}^{n+1,-} - \varphi_i^{n+1,-})\Big).$$

The resulting scheme is non-conservative. It preserves constant (u,p) states. The results are very similar to those obtained with the Abgrall-Saurel approach [13]. In the sequel, this scheme is called the BHRJ scheme.

Glimm remap (I)

We construct a sequence of random or pseudo-random numbers $\omega_n \in [0,1[$. According to this number we take [3]

$$W_i^{n+1} = W_{i-1}^{n+1,-} \text{ if } \omega_n < \frac{\Delta t_n}{\Delta x_i} \max(u_{i-1/2}^n, 0),$$

$$W_i^{n+1} = W_{i+1}^{n+1,-} \text{ if } \omega_n > 1 + \frac{\Delta t_n}{\Delta x_i} \min(u_{i+1/2}^n, 0),$$

$$W_i^n = W_i^{n+1,-} \text{ if } \frac{\Delta t_n}{\Delta x_i} \max(u_{i-1/2}^n, 0) \leq \omega_n \leq 1 + \frac{\Delta t_n}{\Delta x_i} \min(u_{i+1/2}^n, 0).$$

Glimm remap (II)

A good choice for the pseudo-random sequence ω_n is the (k_1, k_2) van der Corput sequence, computed by the following C algorithm

```
float corput(int n,int k1,int k2){
  float corput=0;
  float s=1;
  while(n>0){
    s/=k1;
    corput+=(k2*n%k1)%k1*s;
    n/=k1;}
  return corput;
}
```

In this algorithm, k_1 and k_2 are two relatively prime numbers and $k_1 > k_2 > 0$. In practice, we consider the $(5, 3)$ van der Corput sequence.

Glimm remap (III)

We recently discovered that the sequence

$$\omega_n = n\sqrt{2} \mod 1$$

also gives excellent results !

Glimm remap (IV)

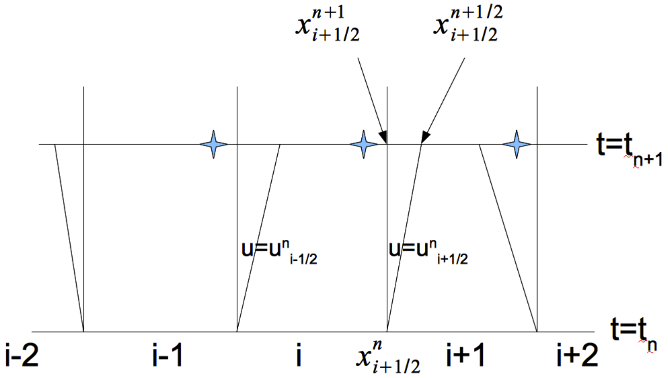


Figure: Example of Glimm remap. The stars correspond to the sampling points. In cells $i-1$ and i, we keep the values of the Lagrange cells. In cell $i+1$, we take the values of Lagrange cell $i+2$.

Lagrange interface velocity

We have to provide the interface velocities $u^n_{i+1/2}$.
In the resolution of the Riemann problem $R(W^n_i, W^n_{i+1}, x/t)$ we find four waves. The characteristic fields 2 and 3 are linearly degenerated and $\lambda_2(w) = \lambda_3(w) = u$, thus the velocity is constant across these waves. It corresponds to the interface velocity, which we denote by $u^*(W_i, W_{i+1})$. It is then natural to take

$$u^n_{i+1/2} = u^*(W^n_i, W^n_{i+1}).$$

Relaxation Riemann solver

For much faster numerical computations, we can use an approximate Riemann solver based on relaxation techniques [11].

- it is positive and handles vacuum.
- entropy dissipative.

Properties

- The constant (u, p) states are exactly preserved.
- The gas fraction is not smeared at all.
- It is possible to use any approximate Riemann solver in the Lagrange step.
- Statistically conservative.
- Convergence ?

Weak shock

The first test consists in a two-fluid shock tube. The stiffened gas parameters are

$$\gamma_W = 2, \quad \pi_W = 1,$$
$$\gamma_A = 1.4, \quad \pi_A = 0.$$

We take for the left and right initial data

$$(\rho_L, u_L, p_L, \varphi_L) = (2, 1/2, 2, 1),$$
$$(\rho_R, u_R, p_R, \varphi_R) = (1, 1/2, 1, 0).$$

We compare the non-conservative remap and the Glimm remap. The Riemann solver is the approximate VFRoe solver in the (ρ, u, p, φ) variables.

Convergence study

The convergence rate is approximately 0.6.

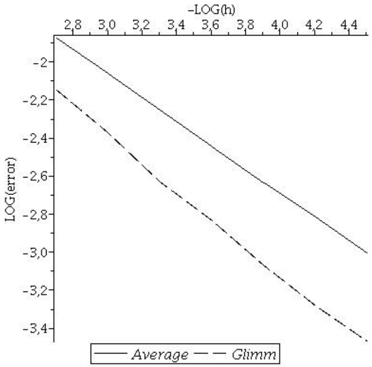


Figure: Convergence study: Glimm remap versus non-conservative averaging remap, weak shock.

Strong shock

Interaction between a shock of velocity $\sigma = 4$ and a contact of velocity $v = -1$.
The initial positions of the contact and the shock are chosen in such way that they meet together at the abscissa $x = 0$ at time $t = 1$. The EOS parameters are $\gamma_1 = 1.4$, $\pi_1 = 0, \gamma_2 = 2$, $\pi_2 = 7$.
The initial data are

$$\begin{array}{rcll} (\rho_L, u_L, p_L, \varphi_L) & = & (3.4884, 1.1333, 23.333, 1), & x < -4, \\ (\rho_M, u_M, p_M, \varphi_M) & = & (2, -1, 2, 1), & -4 \leq x \leq 1, \\ (\rho_R, u_R, p_R, \varphi_R) & = & (1, -1, 2, 0), & x > 1. \end{array}$$

After the interaction at time $t = 1$, the solution is simply given by the resolution of a two-fluid Riemann problem between states (L) and (R).

Strong shock

Similar phenomena in [4]

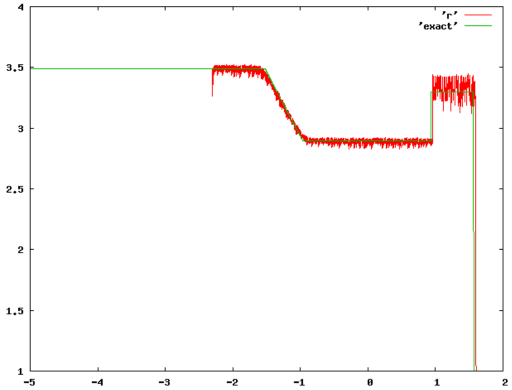


Figure: Glimm approach, density plot. TV explosion due to wall-heating effect propagation, strong shock.

Modified interface velocity

Simple remark: if one takes $u^n_{i+1/2} = 0$, we fall back on the classic Godunov scheme, which solves correctly the shock waves.
It is thus better to use the Glimm approach only at the interface,

$$u^n_{i+1/2} = \left\{ \begin{array}{c} u^*(W^n_i, W^n_{i+1}) \text{ if } \varphi^n_i \neq \varphi^n_{i+1}, \\ 0 \text{ if } \varphi^n_i = \varphi^n_{i+1}. \end{array} \right.$$

The scheme has the same properties as before and the “TV explosion” is removed in strong shocks.

Numerical results

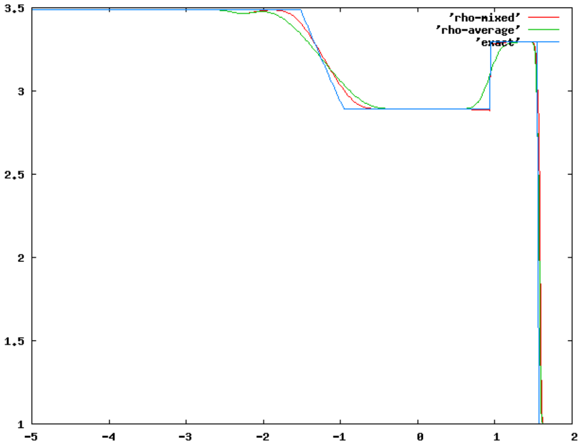


Figure: Density. Comparison of the modified Glimm and averaging remap schemes.

Convergence study

Better convergence rate !

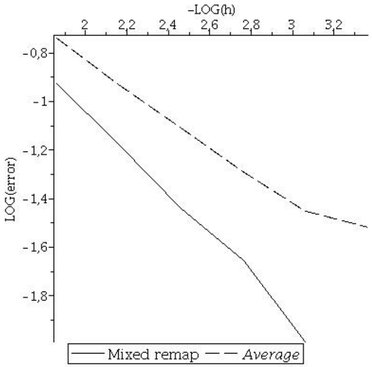


Figure: strong shock-interface interaction. Convergence study. modified Glimm remap and averaging remap.

Comparisons

Order of convergence of 1 on the density !

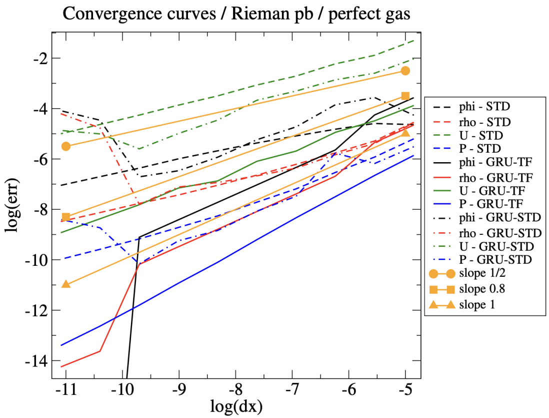


Figure: Gas-gas Riemann problem. Convergence study. Comparisons of several schemes.

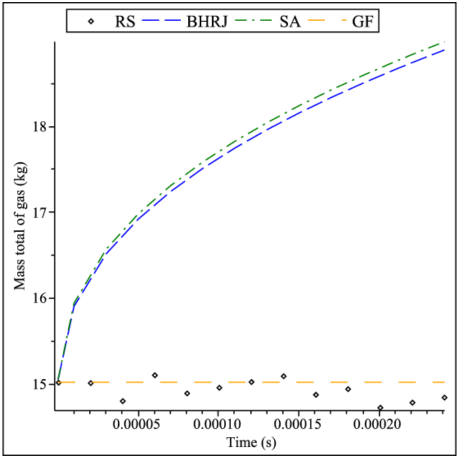
For rigorous convergence results see [6, 7, 8]

Conservation (I)

The numerical mass transfer between the two fluids should be zero.
We compare it for different schemes:

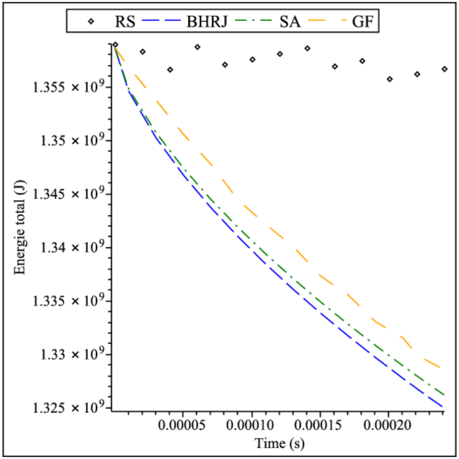
- Saurel-Abgrall scheme (SA) [13];
- A Lagrange and remap version of Saurel-Abgrall (BHRJ) [2];
- The "Ghost Fluid for the poor" (GF) scheme [1];
- The random scheme (RS).

Conservation (II)



Conservation (III)

The total energy should be exactly conserved. We compare the energy conservation property of the four same schemes



Two-dimensional computations

Dimensional splitting

In order to perform 2D computations, we can use dimensional splitting. For advancing a time step τ, we first numerically solve

$$\frac{W^* - W^n}{\tau} + \partial_x F^1(W^n) = 0,$$

and then

$$\frac{W^{n+1} - W^*}{\tau} + \partial_y F^2(W) = 0,$$

with the GRU scheme. But it's not fully general...

Unstructured GRU

Finite volume mesh $\mathscr{M}$. During the ALE step, the cell $K_i(t) \in \mathscr{M}$ depends on time.
Suppose that at time t_n the cell K_i is in phase $\alpha = \varphi_i^n \in \{0,1\}$ and has neighbors that are not in the same phase. There exists therefore at least one neighbor $K_j \in \mathscr{V}(K_i)$ such that

$$\beta = \varphi_j^n = 1 - \varphi_i^n = 1 - \alpha.$$

Let

$$\mathscr{V}_\alpha(K_i) = \{K_j \in \mathscr{V}(K_i), \varphi_j = \alpha = \varphi_i\},$$

$$\mathscr{V}_\beta(K_i) = \{K_j \in \mathscr{V}(K_i), \varphi_j = \beta\}.$$

The area of the cell K_i swept by the other phase is estimated by

$$\mathscr{A}_\beta = -\Delta t \sum_{K_j \in \mathscr{V}_\beta(K_i)} \min(U_{ij} \cdot N_{ij}, 0) s_{ij},$$

where U_{ij} is the ALE velocity chosen at the interface $K_i \mid K_j$.

Unstructured GRU

The area of the cell K_i swept by the displacement of the interface $K_i \mid K_j$ is given by

$$\mathscr{A}_{ij} = -\Delta t \min(U_{ij} \cdot N_{ij}, 0) s_{ij}.$$

Remember that $\mathscr{A}_{ij} = 0$ if $K_j \in \mathscr{V}_\alpha(K_i)$. At the end of the ALE step the cell K_i is entirely in the phase α. It is therefore natural to note

$$W_\alpha^* = W_i^{n+1,-}.$$

We can also compute the average value of the conservative variables coming only from the phase β

$$W_\beta^* = \frac{\sum_{K_j \in \mathscr{V}_\beta(K_i)} \mathscr{A}_{ij} W_j^{n+1,-}}{\sum_{K_j \in \mathscr{V}_\beta(K_i)} \mathscr{A}_{ij}}.$$

Unstructured GRU

We define the number between 0 and 1

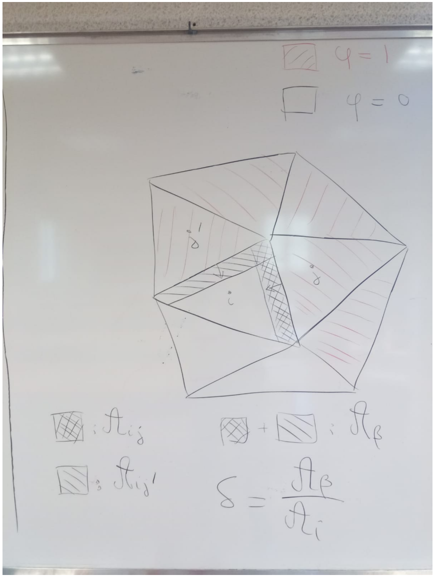
$$\delta = \frac{\sum_{K_j \in \mathscr{V}_\beta(K_i)} \mathscr{A}_{ij}}{\mathscr{A}_i},$$

where $\mathscr{A}_i$ is the area of cell K_i. This number is in $[0,1]$ because of the CFL condition. Taking a pseudo-random number ω_n in $[0,1]$, we then choose the value of W_i^{n+1} following the rule:

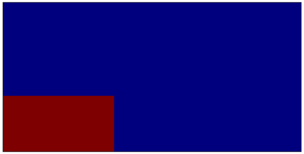
$$W_i^{n+1} = \begin{cases} W_\alpha^* & \text{if } \delta \le \omega, \\ W_\beta^* & \text{otherwise.} \end{cases}$$

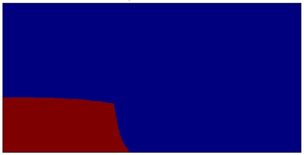
This procedure preserves constant states (U,p). It is (statistically) conservative and entropy dissipative. It also does not depend on the mesh numbering.

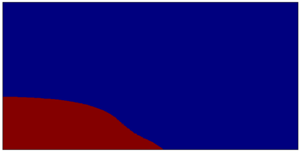
Unstructured GRU

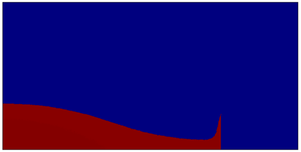


Dam break 1

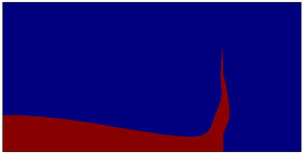

(a) $t = 0\ s.$

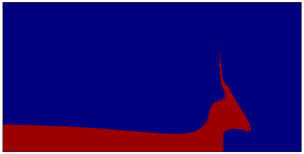

(b) $t = 0.091267\ s.$

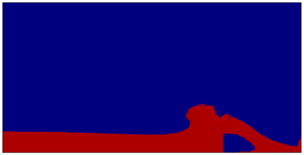

(c) $t = 0.182222\ s.$

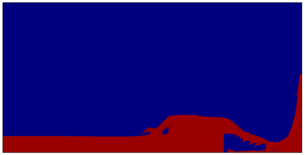

(d) $t = 0.363557\ s.$

Dam break 2

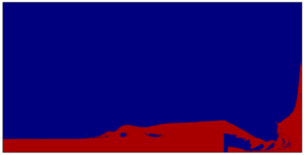

(e) $t = 0.545282\ s.$

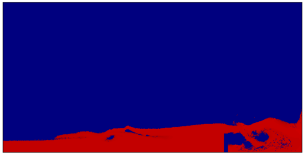

(f) $t = 0.726496\ s.$

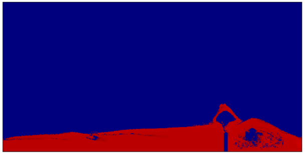

(g) $t = 0.906857\ s.$

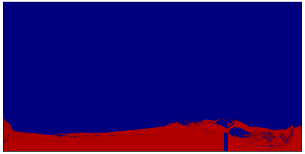

(h) $t = 1.087458\ s.$

Dam break 3


(i) $t = 1.267899\ s$.

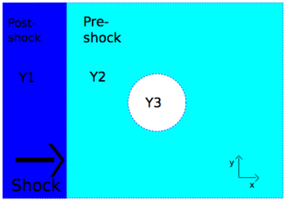

(j) $t = 1.4447929\ s$.


(k) $t = 1.628847\ s$.


(l) $t = 1.809698\ s$.

Shock-bubble interaction

We consider a shock that comes to a bubble at velocity $\sigma = 415 m.s^{-1}$ (see [KL10]).



The initial datas are:

Quantities	Y1	Y2	Y3
$\rho(kg.m^{-3})$	1.69	1.22	3.86
$u(m.s^{-1})$	113.5	0	0
$v(m.s^{-1})$	0	0	0
$p(Pa)$	1.6e5	1.0e5	1.0e5
φ	0	0	1
γ	1.4	1.4	1.249
π	0	0	0

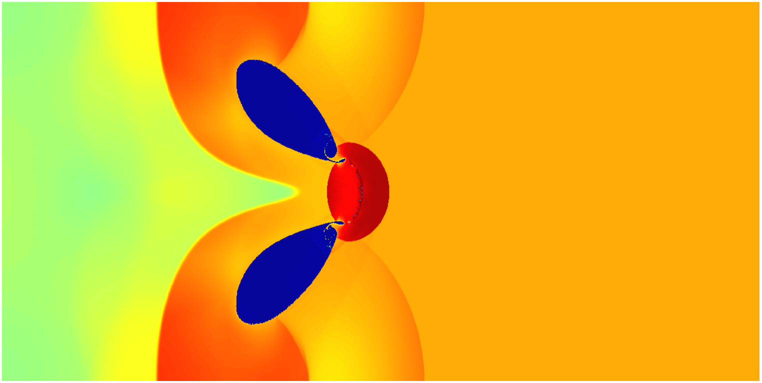
Animation

http://www.youtube.com/watch?v=c8hcqihJzbw

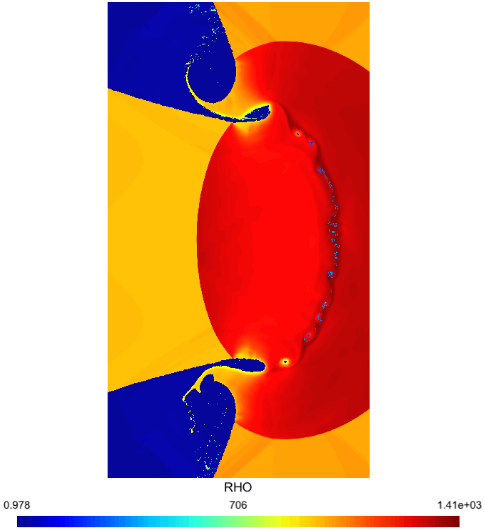
Numerical results

$t_{\mathrm{max}} = 0.45$ ms
Grid: 40,000×20,000 (4 billions unknowns for each time step)
GPU time: 30 h (10×NVIDIA K20)

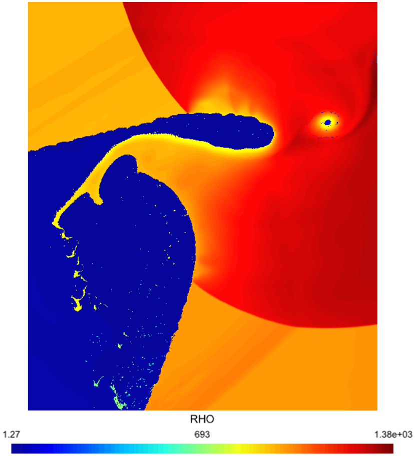
Density

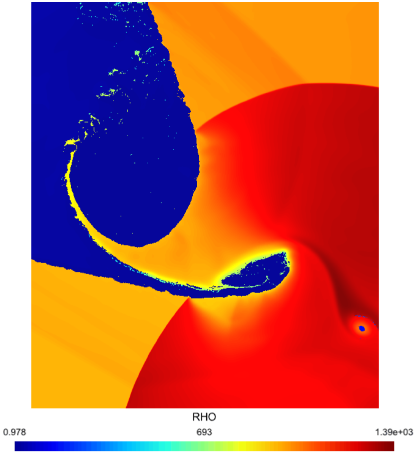


Zoom 1

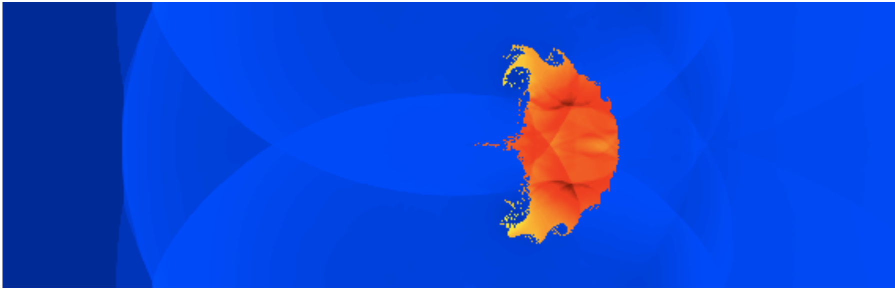


Zoom 2





Shock-droplet interaction (III)



Conclusion

- Random scheme for solving two-fluid compressible flows with non-convex hyperbolicity domain.
- Very simple !
- The random scheme enjoys interesting stability and conservation properties.
- It can be extended to unstructured meshes.
- It is well adapted to multicore computations.

Bibliography I

[1] Rémi Abgrall and Smadar Karni.
Ghost-fluids for the poor: a single fluid algorithm for multifluids, volume 140, 141 of *Internat. Ser. Numer. Math.*
Birkhäuser, Basel, 2001.

[2] Thomas Barberon, Philippe Helluy, and Sandra Rouy.
Practical computation of axisymmetrical multifluid flows.
Int. J. Finite Vol., 1(1):34, 2004.

[3] C. Chalons and F. Coquel.
Computing material fronts with lagrange-projection approach, 2010.
HYP2010 Proc. http://hal.archives-ouvertes.fr/hal-00548938/fr/.

[4] Phillip Colella.
Glimm's method for gas dynamics.
SIAM J. Sci. Statist. Comput., 3(1):76–110, 1982.

[5] J.-P. Croisille.
Contribution à l'étude théorique et à l'approximation par éléments finis du système hyperbolique de la dynamique des gaz multidimensionnelle et multiespèces.
PhD thesis, Université Paris VI, 1990.

[6] Thierry Gallouët and Olivier Hurisse.
Convergence of a multidimensional glimm-like scheme for the transport of fronts.
IMA Journal of Numerical Analysis, 42(4):2924–2958, 2022.

[7] Thierry Gallouët, Olivier Hurisse, and Samuel Kokh.
A random choice scheme for scalar advection.
International Journal for Numerical Methods in Fluids, 95(10):1656–1685, 2023.

[8] Thierry Gallouët, Olivier Hurisse, and Samuel Kokh.
Improving the accuracy of the Jin-Xin relaxation scheme.
PhD thesis, EDF R&D-dept. Mécanique des Fluides, Energies et Environnement, 2024.

Bibliography II

[9] A. Harten, P. D. Lax, C. D. Levermore, and W. J. Morokoff.
Convex entropies and hyperbolicity for general euler equations.
SIAM J. Numer. Anal., 35(6):2117–2127, 1998.

[10] P. Helluy and H. Mathis.
Pressure laws and fast legendre transform.
Math. Models Methods Appl. Sci., 21(4):745–775, 2011.

[11] J. Jung.
Schémas numériques adaptés aux accélérateurs multicœurs pour les écoulements bifluides.
PhD thesis, Université de Strasbourg, 2013.

[12] Siegfried Müller, Philippe Helluy, and Josef Ballmann.
Numerical simulation of a single bubble by compressible two-phase fluids.
Internat. J. Numer. Methods Fluids, 62(6):591–631, 2010.

[13] R. Saurel and R. Abgrall.
A simple method for compressible multifluid flows.
SIAM J. Sci. Comput., 21(3):1115–1145, 1999.

Convex mixture law

Extensive entropy of a single fluid

- The extensive entropy $S(V,E,M) \in \mathbb{R} \cup \{-\infty\}$ is a concave function of volume, energy and mass.
- We suppose that it is C^2 on its (convex) domain $\mathrm{dom}S = \{(V,E,M) \in \mathbb{R}^3, S(V,E,M) > -\infty\} \subset \{V,E,M \geq 0\}$.
- We suppose that it is Positively Homogeneous of degree 1 (PH1 in short): $S(\lambda V, \lambda E, \lambda M) = \lambda S(V,E,M)$, $\lambda \geq 0$.
- We define PH0 (or intensive) quantities $\rho = M/V$, $\tau = V/M$, $e = E/M$ and the intensive entropy $s(\tau,e) = S(\tau,e,1)$.
- We define the temperature $T = 1/\partial_e s(\tau,e)$, the pressure $p = T\partial_\tau s(\tau,e)$.

Lax entropy of a single fluid flow

We consider the Lax entropies for a single fluid flow [5, 9].
The pressure $p = p(\rho, e)$ is given by the previous construction (PH1 and concave extensive S, intensive s and $p = \partial_\tau s / \partial_e s$.)

Theorem
$(\rho, \rho U^T, \rho Q) \mapsto -\rho s(\tau, e)$ *is a Lax entropy of the single fluid Euler equations.*

Proof

- Lemma 1 [5]: $(x_0, x_1 \cdots x_n) \mapsto F(x_0, x_1 \cdots x_n)$ convex and PH1. $\mathrm{sgn} F'' = (0, 1, n-1)$ iff $(x_1 \cdots x_n) \mapsto F(1, x_1 \cdots x_n)$ is strictly convex.
- Lemma 2: if $Tds = de + pd\tau$ then Euler $\Rightarrow$additional conservation law $\partial_t(\rho s) + \nabla \cdot (\rho s) = 0$.
- $S(V, E, M) = VS(1, \rho e, \rho) = MS(\tau, e, 1) \Rightarrow \rho s(\tau, e) = S(1, \rho e, \rho)$ thus ρs is concave with respect to $(\rho, \rho e)$.
- Lemma 3: $(\rho, \rho U, \rho Q) \mapsto -\rho s(\tau, e)$ is strictly convex (if $T > 0$) and thus a Lax entropy.
- Mock's theorem $\Rightarrow$the Euler equations are hyperbolic on the convex domain of the Lax entropy.

Generalization

Two-fluid model with a general pressure law $p = p(\rho, e, \varphi)$

- We consider a concave PH1 function $(V, E, M, M_1) \in \mathbb{R}^4 \mapsto S(V, E, M, M_1) \in \mathbb{R} \cup \{-\infty\}$: the extensive entropy. C^2 on its (convex) domain $\mathrm{dom} S = \{(V, E, M, M_1) \in \mathbb{R}^4, S(V, E, M, M_1) > -\infty\}$.
- We define $\rho = M/V$, $\tau = V/M$, $e = E/M$, $\varphi = M_1/M$ and the specific entropy $s(\tau, e, \varphi) = S(\tau, e, 1, \varphi)$.
- We define the temperature $T = 1/\partial_e s$, the pressure $p = T\partial_\tau s$ and the potential $\lambda = T\partial_\varphi s$.

Theorem

$(\rho, \rho U^T, \rho Q, \rho\varphi) \mapsto -\rho s(\tau, e, \varphi)$ *is a Lax entropy of the two-fluid model.*

Proof

- Lemma 2': if $Tds = de + pd\tau + \lambda d\varphi$ then the two-fluid model $\Rightarrow$additional conservation law $\partial_t \rho s + \nabla \cdot (\rho s) = 0$.
- $S(V, E, M, M_1) = VS(1, \rho e, \rho, \rho\varphi) = MS(\tau, e, 1, \varphi) \Rightarrow \rho s(\tau, e, \varphi) = S(1, \rho e, \rho, \rho\varphi)$ thus ρs is concave with respect to $(\rho, \rho e, \rho\varphi)$.
- Lemma 3': $(\rho, \rho U, \rho Q, \rho\varphi) \mapsto -\rho s(\tau, e, \varphi)$ is strictly convex and thus a Lax entropy (if $T > 0$).
- Mock's theorem $\Rightarrow$the two-fluid model is hyperbolic on the convex domain of the Lax entropy.

Mixture pressure law

How to construct $S(V, E, M, M_1)$? Entropy optimization! [10]

$$S(V,E,M,M_1) = \sup_{V_1,E_1} S_1(V_1,E_1,M_1) + S_2(V-V_1,E-E_1,M-M_1).$$

- From its construction, S is concave and PH1.
- No optimization with respect to M_1: no phase transition.

What happens with a mixture of a perfect gas and a stiffened gas ?

$$S_1(\tau,e,1) = (\gamma_1 - 1)\ln\tau + \chi_1 \ln e,$$

$$S_2(\tau,e,1) = (\gamma_2 - 1)\ln\tau + \chi_2 \ln(e - \pi_2\tau).$$

S_1 and S_2 are extended by $-\infty$ for non-positive arguments of the logarithms.

Mixture pressure law

We introduce

$$\chi = \chi_1 \varphi + (1-\varphi)\chi_2, \quad \zeta = \frac{\chi_1 \varphi}{\chi_1 \varphi + (1-\varphi)\chi_2}, \quad \gamma = \zeta \gamma_1 + (1-\zeta)\gamma_2,$$

$$\delta = -\gamma_2 \pi_2, \quad r = (\delta + (\gamma-1)\rho e)^2 - 4\delta(\gamma_1 - 1)\zeta \rho e,$$
$$\alpha = \frac{\delta + (\gamma-1)\rho e - \sqrt{r}}{2\delta}.$$

Then, the entropy optimization procedure leads to

$$p = \frac{\partial_\tau s}{\partial_e s} = (\gamma-1)\rho e - \gamma(1-\alpha)\pi_2.$$

Pure phases

Pure gas $\varphi = 1$ then everything is OK

$$p = (\gamma_1 - 1)\rho e.$$

But when $\varphi = 0$ the liquid pressure is given by

$$p = \max((\gamma_2 - 1)\rho e - \gamma_2 \pi_2, 0).$$

We recover the stability of the Godunov scheme, but:

- pressureless model for

$$\varphi = 0, \quad \rho e \leq \frac{\gamma_2 \pi_2}{(\gamma_2 - 1)}.$$

- spurious oscillations are still here.

The Glimm strategy is more comfortable...

GPU (I)

A modern Graphics Processing Unit (GPU) is made of:

- Global memory (typically 1 Gb)
- Compute units (typically 27).

Each compute unit is made of:

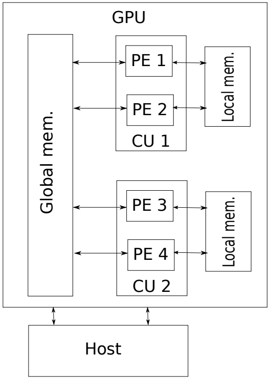
- Processing elements (typically 8).
- Local memory (typically 16 kb)

The same program can be executed on all the processing elements at the same time.

- All the processing elements have access to the global memory.
- The processing elements have only access to the local memory of their compute unit.
- If two processing elements write at the same location at the same time, only one wins...
- The access to the global memory is slow while the access to the local memory is fast.

GPU (II)

A (virtual) GPU with 2 Compute Units and 4 Processing Elements



OpenCL

- OpenCL means "Open Computing Language". It includes:
  - A library of C functions, called from the host, in order to drive the GPU.
  - A C-like language for writing the kernels that will be executed on the processing elements.
- Practically available since september 2009. The specification is managed by the Khronos Group (OpenGL).
- Virtually, it allows to have as many compute units (work-groups) and processing elements (work-items) as needed.
- The threads are sent to the GPU thanks to a mechanism of command queues on the real compute units and processing elements.
- Portable: the same program can run on a multicore CPU or a GPU.

Implementation of the splitting scheme

We organize the data in a (x, y) grid and for each time step:

- we associate a processor to each cell of the grid.
- we compute the fluxes balance in the x-direction for each cell of each row of the grid. A row (or a part of the row) is associated to one compute unit and one cell to one processor.
- subdomain strategy in order to retain data into the local cache memory. Covering of two cells between the subdomain (for the correctness of the boundary values).
- we transpose the grid (exchange x and y) with an optimized memory transfer algorithm.
- we compute the fluxes balance in the y-direction for each row of the transposed grid. Memory access are optimal.
- we transpose again the grid.

Speedup

	time (s)
AMD Phenom II x4 945 (1 core)	192
AMD Phenom II x4 945 (4 cores)	59
AMD Radeon HD5850	1.43
NVIDIA GTX 460	2.48
NVIDIA Geforce GTX470	0.93