CentraleSupelec – ST7

Simulation à Haute Performance pour la réduction d'empreintes

DE LA RECHERCHE À L'INDUSTRIE

Compléments d'informations sur les projets CEA-DAM

9 février 2022

Stéphane Jaouen

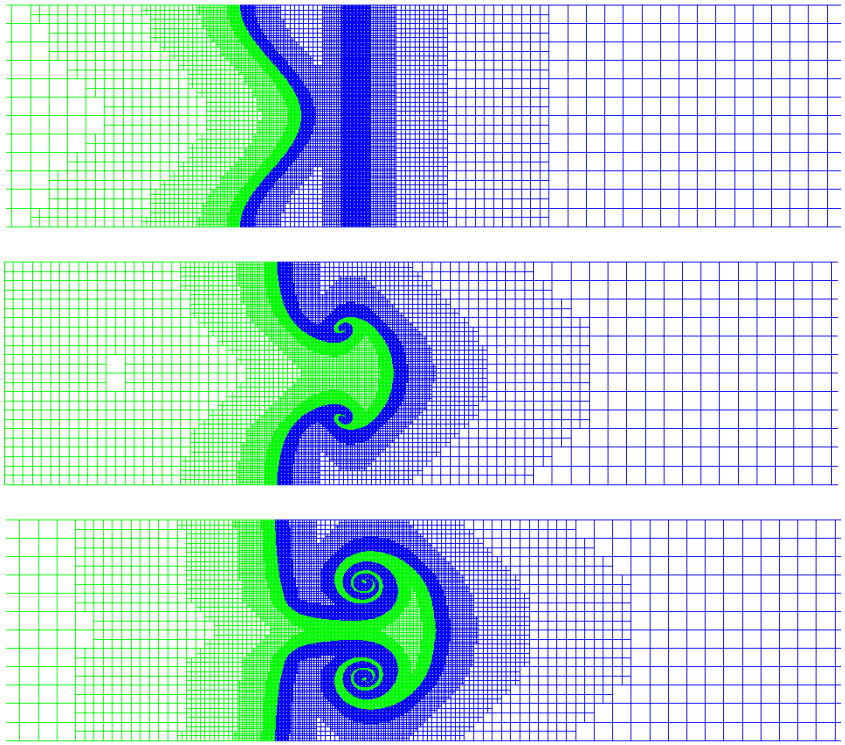
Grands codes du CEA-DAM : plusieurs approches

- Codes lagrangiens : les nœuds du maillage se déplacent à la vitesse matière
  - Excellent rapport précision / coût de calcul
  - Très bien adapté à la prise en compte de plusieurs matériaux (pas de mailles mixtes)
  - Potentiellement fragiles (déformations de maillage trop importantes, surfaces de glissement, …)
- Codes eulériens : le maillage reste fixe au cours du temps
  - Plus robustes
  - Coût calcul et encombrement mémoire beaucoup plus importants
  - Traitement délicat des mailles mixtes, des conditions aux limites, …

Deux variantes palliant en partie les défauts majeurs de ces approches

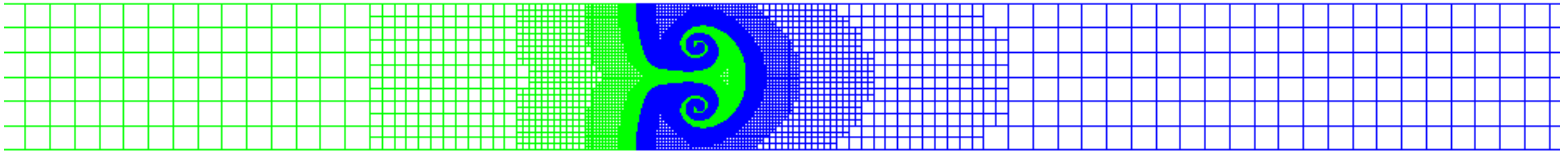
- Méthode ALE (*Arbitrary Lagrangian Eulerian*)
  - Améliore la robustesse des codes lagrangiens (vitesses de nœuds a priori différentes de la vitesse matière)
  - Contrôle possible de la régularité du maillage
- Méthode AMR (*Adaptative Mesh Refinement*)
  - Modification dynamique du maillage suivant des critères *adhoc*
  - Contrôle du coût et de la précision d'une simulation.
  - Limitation de l'encombrement mémoire

Exemple : cas-test de *Stony-Brook*
(instabilité de Richtmyer-Meshkov)



Raffinement adaptatif suivant les zones d'intérêt de l'écoulement

AMR



Euler

ARMEN : quelques caractéristiques

- AMR « tree-based » sur grille cartésienne, facteurs de raffinement 2 ou 3
  - 1D plan / cylindrique / sphérique, 2D plan / axisymétrique, 3D
  - Trois niveaux de parallélisme :
    - Macro : décomposition de domaine MPI avec équilibrage dynamique de charge
    - Méso : multi-threading OpenMP
    - Micro : vectorisation Single Instruction Multiple Data (SIMD)

- Solveurs Hydro Multi-matériaux, basés sur l'approche « Lagrange-projection »
  - Phase Lagrange multi-matériaux miscibles (peu utilisée) / non miscibles:
    - Schémas centrés de type Godunov : solveur acoustique, GAD (2nd ordre), GAIA & GoHy (ordre très élevé)
    - Schémas décalés : BBC (2nd ordre), SHORK (ordre très élevé)
  - Phase de projection
    - « upwind », vanleer (2nd ordre), PPM (3ème ordre), reconstruction polynômiale (ordre très élevé)
    - Reconstruction d'interface de type Volume Of Fluid (VOF) en 1D / 2D / 3D : Youngs, ELVIRA
    - Schémas anti-dissipatifs multi-matériaux (VoFire)
  - Splitting directionnel : phase Lagrange (optionnel), projection (obligatoire)
  - Traitement de type « frontière immergée » pour les obstacles fixes (CL de type « mur » sur géométries complexes)

- APIs C++, C# et Python
  - Initialisation géométrique 2D / 3D (composition de scènes)
  - Plusieurs points d'entrée « modèles utilisateurs » : initialisation, EOSs, termes sources, diagnostics, etc.

Intérêt lié à l'approche Eulérienne : grille régulière

Mise en œuvre relativement simple de méthodes numériques avancées très précises

- Quelques aspects liés à cette approche (focus sur l'hydrodynamique multi-matériaux)
  - Formalisme Lagrange-projection (équations d'Euler compressible)
  - Exemple d'un solveur d'ordre élevé
  - Traitement des mailles mixtes
  - Prise en compte des conditions aux limites sur des frontières complexes (principe et exemples)
  - Prise en compte du parallélisme « décomposition de domaines » sur maillage dynamique
  - Stratégie (recommandée) de pilotage de la grille AMR
- Premiers pas
  - Description du script Python de définition du cas et d'exécution
  - Description du listing d'exécution
- Description des scripts Python utilisés pour les projets
  - Caractérisation d'une source à l'échelle urbaine avec prise en compte du bâti
  - Mise en place d'un réseau de détection d'ondes infrasonores

ARMEN

Partie 1

QUELQUES ASPECTS SUR LES SCHÉMAS HYDRO AMR

FORMALISME LAGRANGE - PROJECTION

Intérêt : découplage physique / transport → mise en œuvre de méthodes numériques adaptées à chaque étape

- Traitements spécifiques sur les interfaces et les mailles mixtes plus aisés
- Pas (ou peu) de diffusion numérique au voisinage des interfaces
- Solveurs plus simples : dans la phase Lagrange les vitesses d'ondes « marchent » par paires ($\pm\lambda_i$)

Grâce au splitting directionnel on se ramène à l'étude de systèmes 1D

- Equations d'Euler – coordonnées (x, t)

$$\begin{cases} \partial_t \rho + \partial_x(\rho u) & = & 0, \\ \partial_t(\rho u) + \partial_x(\rho u^2 + p) & = & 0, \\ \partial_t(\rho e) + \partial_x(\rho e u + p u) & = & 0. \end{cases} \quad (1)$$

- Formulation lagrangienne – coordonnées (X, t)

Le changement de coordonnées est défini par $\begin{cases} \partial_t x = u(X,t), \\ x(X,0) = X \end{cases}$ (2)

Avec $\rho_0 = \rho(X,0)$, le système (1) se réécrit $\begin{cases} \partial_t(\rho_0 \tau) - \partial_X u & = & 0, \\ \partial_t(\rho_0 u) + \partial_X p & = & 0, \\ \partial_t(\rho_0 e) + \partial_X(pu) & = & 0. \end{cases}$ (3)

- Schémas de type Volumes Finis - Approche Lagrange-projection

1 – La phase lagrangienne

Les systèmes (2)-(3) sont intégrés en temps entre t^n et t^{n+1} avec $\rho_0 = \rho(x, t^n)$ et $X = x$

On obtient les variables conservatives à t^{n+1} sur une grille non uniforme $\{x^{n+1}\}_j$ solution de (2)

2 – Phase de projection

Ces variables sont projetées de manière conservative sur la grille initiale $\{x\}_j$

On se focalise sur les schémas centrés

Soit $U = (\rho_0\tau, \rho_0 u, \rho_0 e)^t$ et $F(U) = (-u, p, pu)^t$. Le système (3) se réécrit $\partial_t U + \partial_X F(U) = 0$

Intégrer (3) sur $[X_{i-\frac{1}{2}}; X_{i+\frac{1}{2}}] \times [t^n; t^{n+1}]$ conduit au schéma VF *exact* :

$$\frac{\overline{U}_i^{n+1} - \overline{U}_i^n}{\Delta t} + \frac{F^*_{i+\frac{1}{2}} - F^*_{i-\frac{1}{2}}}{\Delta X} = 0 \quad \text{où} \quad \begin{cases} \overline{U}_i^n = \dfrac{1}{\Delta X}\displaystyle\int_{X_{i-\frac{1}{2}}}^{X_{i+\frac{1}{2}}} U(x, t^n)dx, \\ F^*_{i+\frac{1}{2}} = \dfrac{1}{\Delta t}\displaystyle\int_{t^n}^{t^{n+1}} F(U)(X_{i+\frac{1}{2}}, \theta)d\theta. \end{cases}$$

- Il reste à calculer F* : solution approchée, solveur de Riemann

Quelques solveurs de Riemann implémentés:

- « acoustique » de Godunov : d'ordre 1 en espace, entropique
- GAD : d'ordre 2 en espace
- GAIA : d'ordre arbitrairement élevé en espace et en temps dans la limite *linéaire* (17 maxi en pratique)
- GoHy : de l'ordre 1 à l'ordre 6 en espace et en temps en non linéaire

Principe basé sur les invariants de Riemann : $J^{\pm} = u \pm \int (\rho c)^{-1} dp$

Dans la limite linéaire, les invariants de Riemann sont solutions de deux équations d'advection découplées : $\partial_t J^{\pm} \pm c\partial_X J^{\pm} = 0$

Or, *sur grille régulière*, il existe une formule de récurrence donnant le flux d'advection à un ordre N connaissant celui à l'ordre N-1(**)

- ***Etape 1 : Formule de récurrence pour calculer $J^{\pm}$ à l'ordre voulu en espace et en temps***

(a) On définit les invariants de Riemann linéarisés $J_j^{\pm,n} = u_j^n \pm (\rho c)_j^{-1} p_j^n$ et $\tilde{J}_{j,k}^{\pm,n} = u_k^n \pm (\rho c)_j^{-1} p_k^n$

(b) On initialise la relation de récurrence avec les flux *upwind* du 1er ordre $J_{j+\frac{1}{2}}^{+,1} = J_j^{+,n}$ et $J_{j+\frac{1}{2}}^{-,1} = J_{j+1}^{-,n}$.

(c) Calcul de $J_{j+\frac{1}{2}}^{\pm}$ à l'ordre N en espace et en temps par la formule de récurrence

$$\begin{cases} J_{j+\frac{1}{2}}^{+,N} = J_{j+\frac{1}{2}}^{+,N-1} - \dfrac{1}{N!}\left(\displaystyle\prod_{\substack{i=-m \\ i\neq 0}}^{M} (\nu_j + i)\right)\left(\displaystyle\sum_{k=0}^{N-1} (-1)^{k+N} C_{N-1}^k \tilde{J}_{j,j+m-k}^{+,n}\right), \\ J_{j+\frac{1}{2}}^{-,N} = J_{j+\frac{1}{2}}^{-,N-1} - \dfrac{1}{N!}\left(\displaystyle\prod_{\substack{i=-M \\ i\neq 0}}^{m} (\nu_{j+1} + i)\right)\left(\displaystyle\sum_{k=0}^{N-1} (-1)^{k+N} C_{N-1}^k \tilde{J}_{j+1,j+M-k+1}^{-,n}\right), \end{cases} \quad (1)$$

avec $\nu_j = c_j \dfrac{\Delta t}{\Delta X}$, $m = E(\dfrac{N}{2})$, $M = E(\dfrac{N-1}{2})$ et $C_n^p = \dfrac{n!}{p!(n-p)!}$

- ***Etape 2 : Raccord de ces invariants d'ordre élevé à ceux d'ordre 1 via un limiteur (contrôle des oscillations aux discontinuités)***

$$J_{j+\frac{1}{2}}^{\pm,lim} = J_{j+\frac{1}{2}}^{\pm,1} + \phi_{j+\frac{1}{2}}^{\pm}\left(J_{j+\frac{1}{2}}^{\pm,N} - J_{j+\frac{1}{2}}^{\pm,1}\right).$$

- ***Etape 3 : Les flux d'ordre élevé sont alors obtenus en inversant le système linéaire 2x2 suivant***

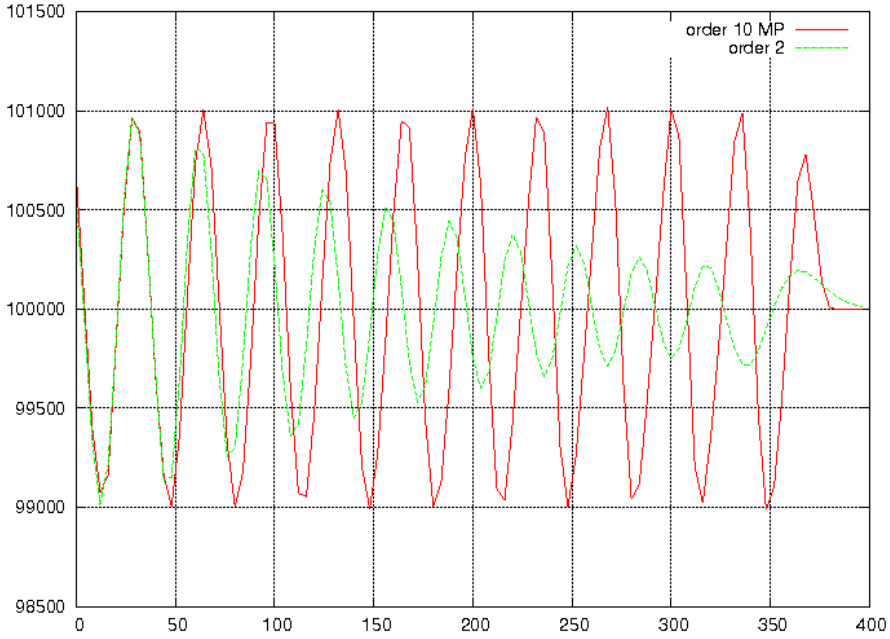
$$\begin{cases} J_{j+\frac{1}{2}}^{+,lim} = u_{j+\frac{1}{2}}^{*,lim} + (\rho c)_j^{-1} p_{j+\frac{1}{2}}^{*,lim}, \\ J_{j+\frac{1}{2}}^{-,lim} = u_{j+\frac{1}{2}}^{*,lim} - (\rho c)_{j+1}^{-1} p_{j+\frac{1}{2}}^{*,lim}. \end{cases}$$

(*) Del Pino & Jourdren, CRAS (2006) : « Godunov Acoustic Invariant Advection"

(**) Daru & Tenaud, JCP (2004)

- Par construction on retrouve le solveur acoustique de Godunov à l'ordre 1
- L'ordre élevé n'est effectif que dans la limite linéaire des équations d'Euler
- En pratique on est limité à l'ordre 17 du fait de la représentation des réels en 64 bits ***(17! # 3.55e+14)***
- Type de schéma très utile pour nos projets

Propagation d'infrasons sur de très longues distances



- Les coordonnées eulériennes, qui évoluent suivant (2), sont également intégrées en temps

$$x_{i+\frac{1}{2}}^{n+1} - x_{i+\frac{1}{2}} = \int_{t^n}^{t^{n+1}} u(X_{i+\frac{1}{2}}, \theta) d\theta = u_{i+\frac{1}{2}}^{*} \Delta t, \tag{4}$$

où $\{x_{i+\frac{1}{2}}\}_i = \{X_{i+\frac{1}{2}}\}_i$ est la grille Eulérienne fixe (régulière).

Puisque u^* est calculé à l'ordre voulu, les positions des faces $x_{i+\frac{1}{2}}^{n+1}$ du maillage mobile le sont aussi à t^{n+1}.

- Lien entre les variables conservatives lagrangiennes et eulériennes à t^{n+1}

Pour $\phi \in \{1, u, e\}$ le changement de variable $(x,t) \to (X,t)$ conduit à la formule *exacte* suivante

$$\begin{aligned} \frac{1}{\Delta x_i^{n+1}} \int_{x_{i-\frac{1}{2}}^{n+1}}^{x_{i+\frac{1}{2}}^{n+1}} (\rho\phi)(s, t^{n+1}) ds &= \frac{1}{\Delta x_i^{n+1}} \int_{X_{i-\frac{1}{2}}}^{X_{i+\frac{1}{2}}} (\rho_0\phi)(s, t^{n+1}) ds, \\ &= \frac{\Delta X}{\Delta x_i^{n+1}} \overline{(\rho_0\phi)}_i^{n+1}, \\ &= \frac{\overline{(\rho_0\phi)}_i^{n+1}}{\overline{(\rho_0\tau)}_i^{n+1}}. \end{aligned} \tag{5}$$

PHASE DE PROJECTION

- Les variables conservatives $\rho\Phi$ pour $\rho\Phi = \{1, u, e\}$ sont projetées sur la grille régulière initiale

$$\overline{(\rho\phi)}_i^{n+1} \stackrel{\text{def}}{=} \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (\rho\phi)(x, t^{n+1})dx = \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i-\frac{1}{2}}^{n+1}} (\rho\phi) + \underbrace{\frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}^{n+1}}^{x_{i+\frac{1}{2}}^{n+1}} (\rho\phi)}_{(\mathcal{I})} + \frac{1}{\Delta x}\int_{x_{i+\frac{1}{2}}^{n+1}}^{x_{i+\frac{1}{2}}} (\rho\phi).$$

On utilise le changement de variable (x, t) → (X, t) dans *($\mathcal{I}$)* et on introduit le flux ***exact*** de projection

$$(\rho\phi)^*_{i+\frac{1}{2}} \stackrel{\text{def}}{=} \frac{1}{x_{i+\frac{1}{2}}^{n+1} - x_{i+\frac{1}{2}}}\int_{x_{i+\frac{1}{2}}}^{x_{i+\frac{1}{2}}^{n+1}} (\rho\phi)(x, t^{n+1})dx. \qquad (6)$$

La phase de projection se réécrit sous la forme suivante, plus courante, en utilisant (4)

$$\overline{(\rho\phi)}_i^{n+1} = \overline{(\rho_0\phi)}_i^{n+1} - \frac{\Delta t}{\Delta x}\left[u^*_{i+\frac{1}{2}}(\rho\phi)^*_{i+\frac{1}{2}} - u^*_{i-\frac{1}{2}}(\rho\phi)^*_{i-\frac{1}{2}}\right].$$

- Evaluation d'ordre élevé des flux de projection (6)

P^Φ: polynôme de reconstruction d'ordre *k* de *($\rho\Phi$)*, décentré amont, sur la grille non uniforme

$$(\rho\phi)^*_{i+\frac{1}{2}} = \frac{1}{x_{i+\frac{1}{2}}^{n+1} - x_{i+\frac{1}{2}}}\int_{x_{i+\frac{1}{2}}}^{x_{i+\frac{1}{2}}^{n+1}} P^\phi_\eta(x)dx \quad \text{où} \quad \eta = \begin{cases} i & \text{si} \quad u^*_{i+\frac{1}{2}}\Delta t \geq 0, \\ i+1 & \text{sinon.} \end{cases}$$

P^Φ se calcule en utilisant (5) et les techniques classiques de reconstruction sur stencils centrés

L'intégrale est calculée analytiquement ou à l'aide de quadratures de Gauss.

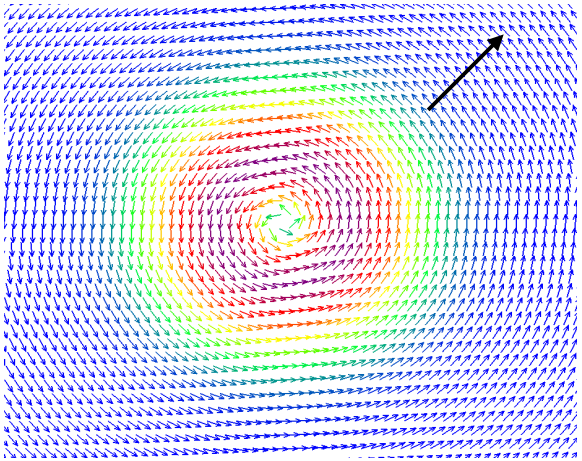
Schéma Lagrange-projection d'ordre élevé par direction
+
Reconstruction transverse des flux
+
Splitting directionnel ***d'ordre élevé***

} Schéma multi-dimensionnel d'ordre élevé(*)

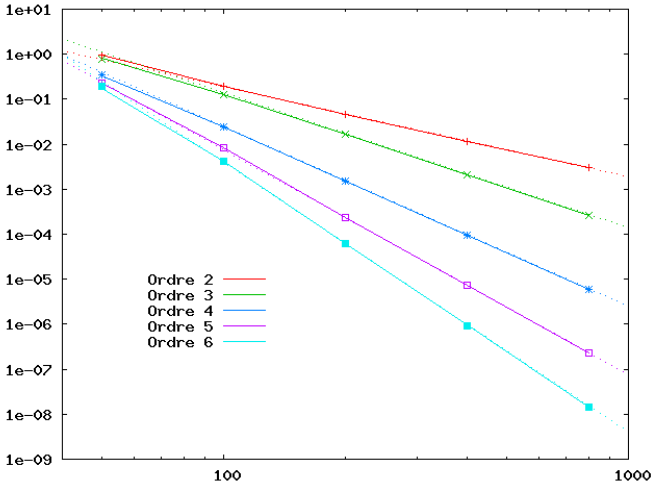
- Exemple : advection d'un vortex 2D avec solution analytique

$$\text{Condition initiale, avec } \gamma = 1.4,\ \beta = 5 \text{ et } r = \sqrt{x^2 + y^2}$$

$$\rho(x,y) = \left(1 - \frac{(\gamma-1)\beta^2}{8\gamma\pi^2} e^{1-r^2}\right)^{\frac{1}{\gamma-1}} \quad , \ p(x,y) = \rho(x,y)^{\gamma} \quad , \ \vec{u}(x,y) = \begin{pmatrix}1\\1\end{pmatrix} + \frac{\beta}{2\pi} e^{\frac{1-r^2}{2}} \begin{pmatrix}-y\\x\end{pmatrix}.$$

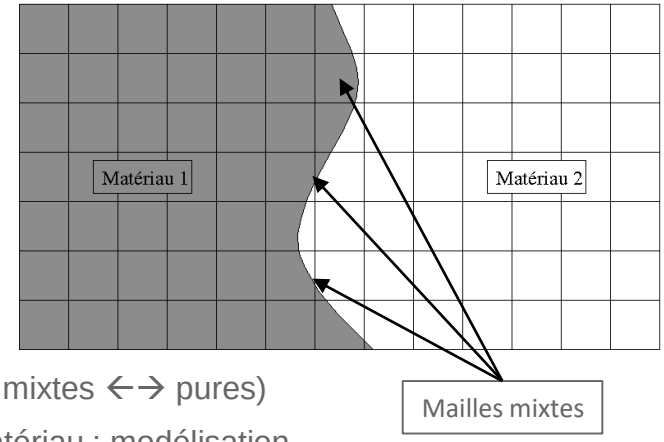


Norme L1 de l'erreur en espace ***et*** en temps
à t = 1 s en fonction du nombre de mailles



(*) Duboc, Enaux, Jaouen, Jourdren, Wolff, note CRAS (2010)
Wolff, thèse de doctorat effectuée au sein du labo (2011)

MAILLES MIXTES



Phase Lagrange

Modèles non miscibles :

- On opère sur le fluide moyen
- Conservation de la masse : pas de changement de statut (mailles mixtes ←→ pures)
- Hypothèses pour la répartition des accroissements sur chaque matériau : modélisation

Modèles miscibles :

- Une phase Lagrangienne par matériau miscible (on se ramène donc au cas mono-matériau)
- Terme de couplage (force de traînée)

Phase de projection

On opère matériau par matériau

Algorithmes de reconstruction d'interface : Youngs, ELVIRA, …

- Adaptés aux modèles non miscibles

Algorithmes non dissipatifs (VoFire(*) : schéma décentré aval sous contraintes de stabilité)

- Adaptés aux zones de mélange

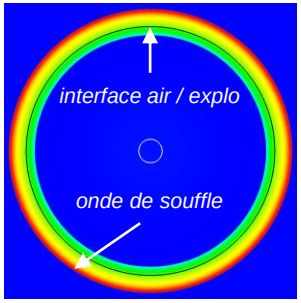
(*) Lagoutière, thèse de doctorat effectuée au sein du labo (2000), et publications suivantes …

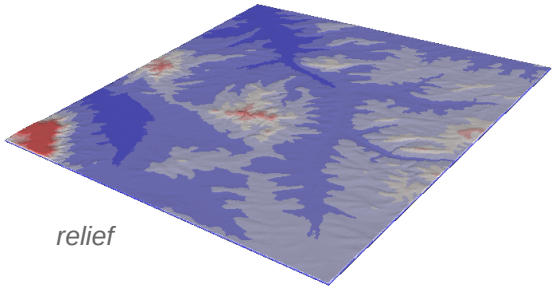
CL SUR FRONTIÈRES A GÉOMÉTRIE COMPLEXE

→ Prise en compte de frontières de type « corps rigides » à géométrie complexe

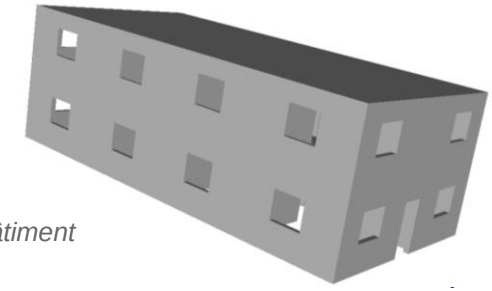
- Relativement *simples* à implémenter dans un code Lagrangien
- Potentiellement très délicates à implémenter dans un code Eulérien
- Nécessaire pour nombres d'applications

Exemple :





relief



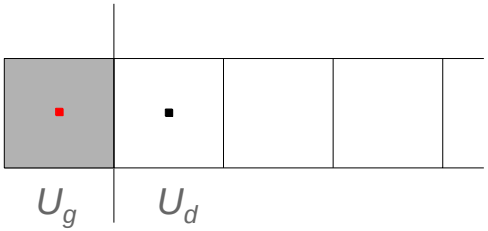
bâtiment

→ Implémentation d'une méthode de type ***frontière immergée*** : « Ghost-Cell Method » (GCM)
→ Implémentation d'une méthode de type « cut-cell » conservative

CL SUR FRONTIÈRES A GÉOMÉTRIE COMPLEXE

- Surcharge des quantités conservatives dans le corps rigide par des valeurs *adhoc* de telle sorte que le schéma approche le mieux possible la condition aux limites

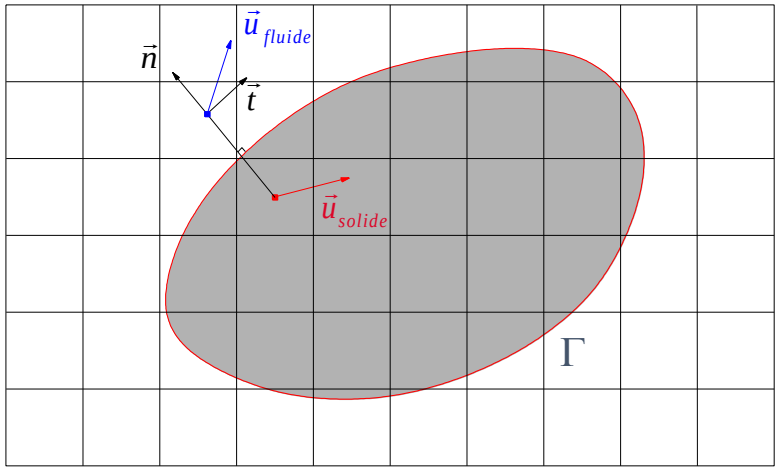
Exemple trivial : implémentation d'une CL de type mur en 1D pour le solveur acoustique de Godunov.



$$u^* = \frac{(\rho c)_g u_g + (\rho c)_d u_d}{(\rho c)_g + (\rho c)_d} + \frac{p_g - p_d}{(\rho c)_g + (\rho c)_d}$$

$$\begin{cases} u_g & = & -u_d, \\ p_g & = & p_d, \\ (\rho c)_g & = & (\rho c)_d \end{cases} \quad \Rightarrow \quad (\vec{u}.\vec{n})_\Gamma = u^* = 0$$

➔ *Généralisation en 2D/3D aux frontières quelconques.*



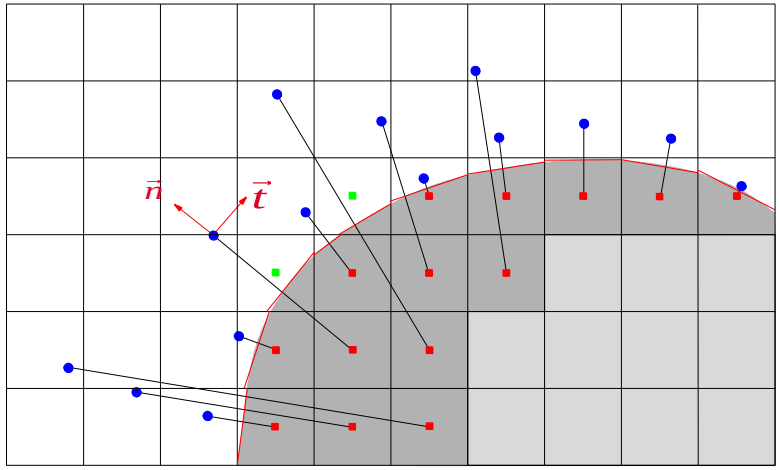
$$\vec{u}_{fluide} = (\vec{u}.\vec{n})\,\vec{n} + (\vec{u}.\vec{t})\,\vec{t}$$

$$\vec{u}_{solide} = -(\vec{u}.\vec{n})\,\vec{n} + (\vec{u}.\vec{t})\,\vec{t}$$

$$U^{thermo}_{solide} = U^{thermo}_{fluide}$$

CL SUR FRONTIÈRES A GÉOMÉTRIE COMPLEXE

- IBM-GCM : algorithme sur un balayage directionnel.
  - Sauvegarde des volumes partiels et des fractions de présence.
  - Recherche des mailles « fantômes » à surcharger.
  - Calcul des normales (Youngs / Elvira) et positionnement des interfaces.
  - Recherche des mailles « images » (calcul de distances).
  - Affectation dans les mailles « fantômes » des quantités conservatives (condition de type « miroir »).
  - Schéma Lagrange + projection dans la direction considérée : ***sans aucune modification***.
  - Réaffectation des volumes partiels et des fractions de présence : ***perte locale de conservativité***.



$$\vec{u}_{fluide} = (\vec{u}.\vec{n})\,\vec{n} + (\vec{u}.\vec{t})\,\vec{t}$$

$$\vec{u}_{solide} = -(\vec{u}.\vec{n})\,\vec{n} + (\vec{u}.\vec{t})\,\vec{t}$$

$$U^{thermo}_{solide} = U^{thermo}_{fluide}$$

CONDITIONS AUX LIMITES

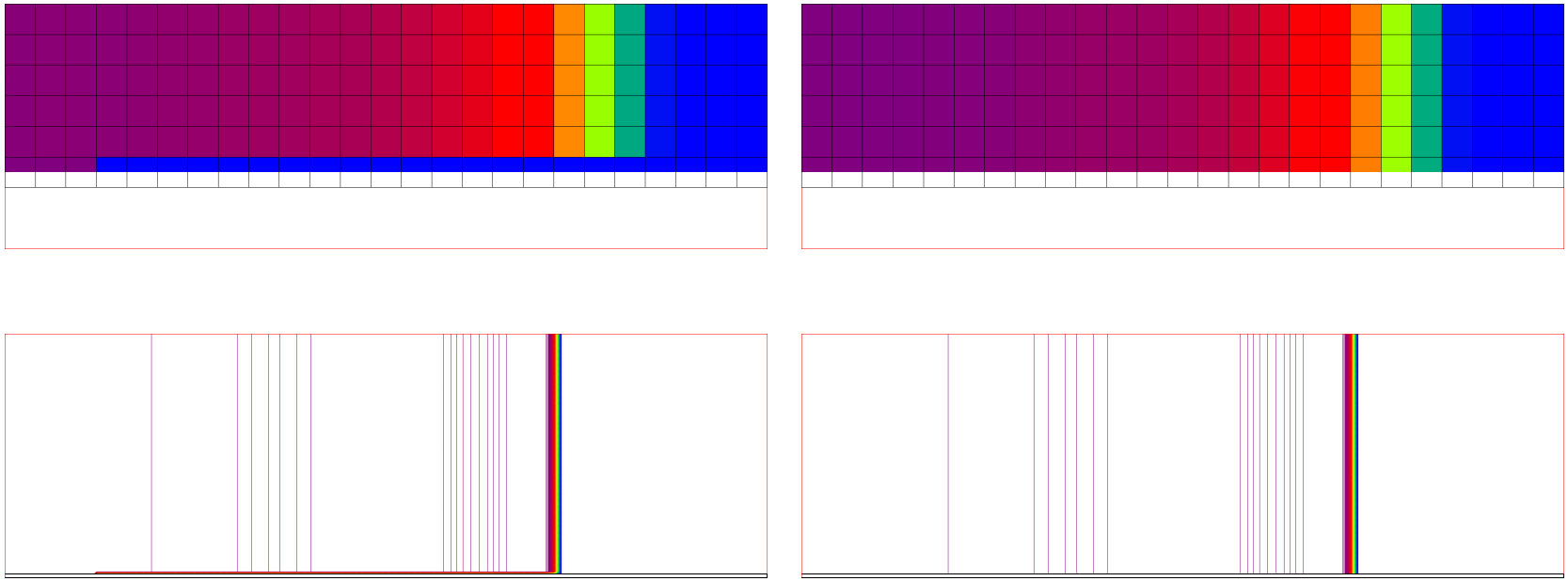
- Un exemple : propagation d'un choc plan à Mach 10

Traitement mailles mixtes / solide de densité infinie

Traitement frontière immergée (IBM-GCM)

Maillage grossier

Maillage fin



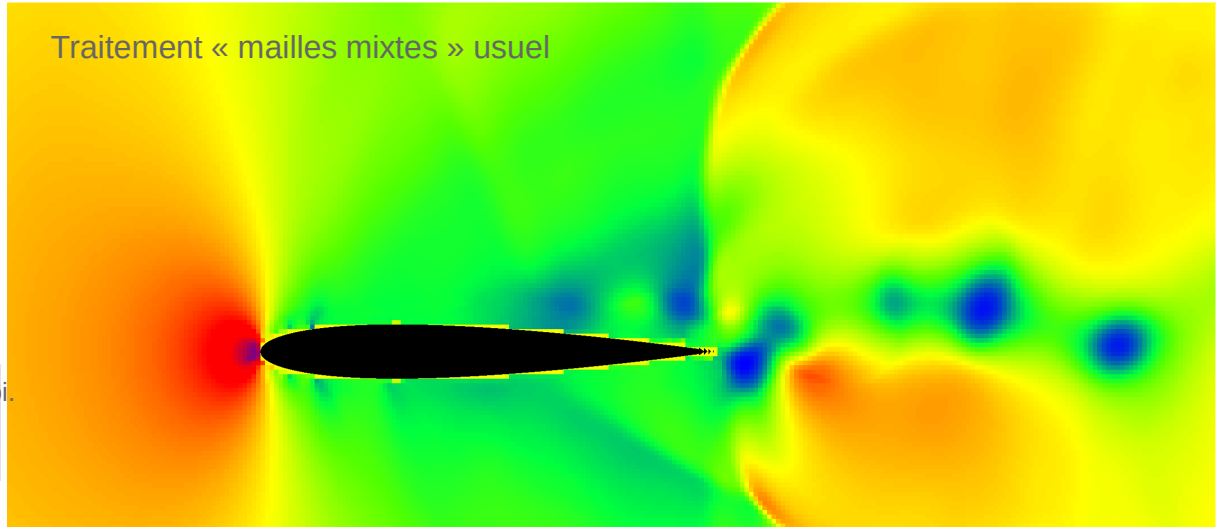
- Traitement mailles mixtes usuel : couche limite numérique au niveau de la paroi.
- IBM-GCM : planéité du front de choc respectée.

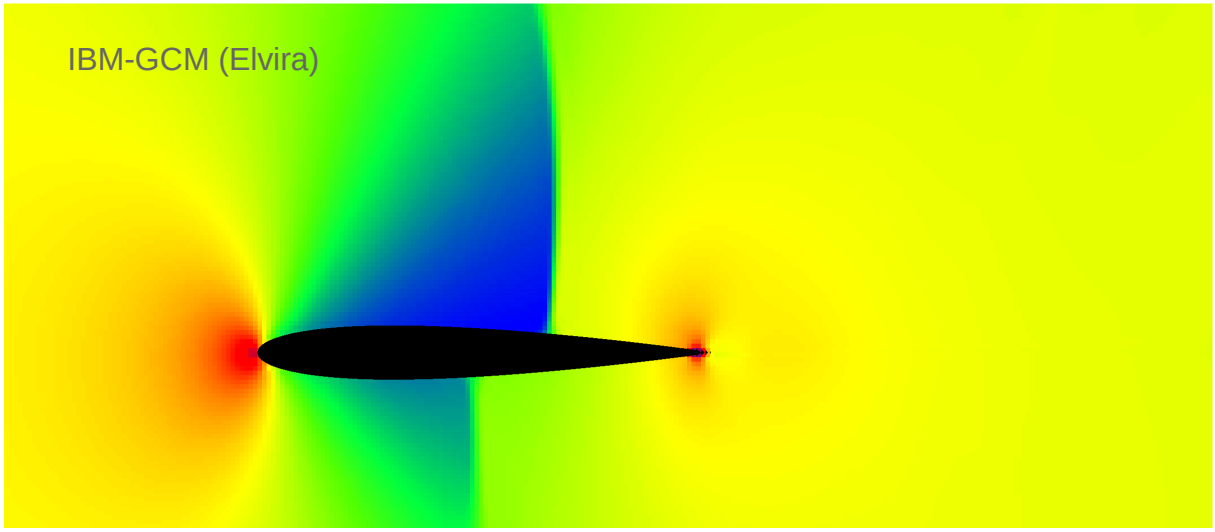
CONDITIONS AUX LIMITES

- NACA0012

$$\begin{cases} M_\infty = 0.8 \\ \alpha = 1.25^\circ \end{cases}$$

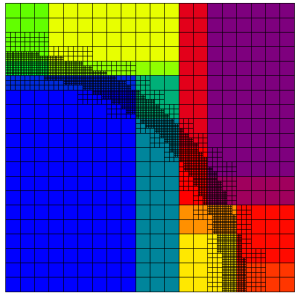
- Couche limite numérique au voisinage de la paroi.
- Discrétisation du profil en marches d'escalier.
  ⇨ Pas de régime stationnaire.





⇨ Régime stationnaire *quasiment* atteint.
(Traitement spécifique nécessaire pour le bord de fuite non intégré au code AMR).

PARALLÉLISME MPI

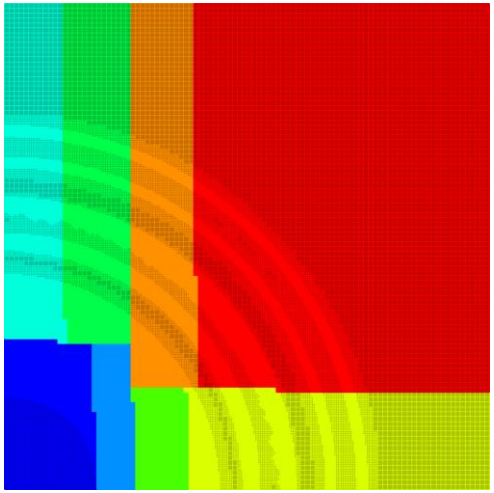


- Décomposition de domaines MPI

  Difficulté : tous les sous-domaines doivent avoir à peu près la même charge de calculs à effectuer

  - Méthode simple à équilibrer sur maillage régulier
  - Algorithmes d'équilibrage de charge nécessaires sur maillage AMR dynamique

- Un exemple



- Equilibrage dynamique de charge avec algorithme ORP (Orthogonal Recursive Partition)
- Echange de mailles de niveau 0 et leurs descendantes
- Paramétrage de l'équilibrage dynamique de charge via le bloc parallelisme dans le fichier de données (algorithme de partitionnement, fréquence, seuil de déclenchement, …)

- Autres mécanismes de parallélisme implémentés :
  - Multi-threading OpenMP
  - Vectorisation SIMD (Single Instruction Multiple Data)

CRITÈRES DE RAFFINEMENT

- De très nombreux critères
  - Basés sur les gradients de pression, densité, vorticité,…
  - Raffinement / déraffinement sur les frontières (par couple de matériaux)
  - Raffinement / déraffinement de zone
  - Modèles utilisateurs
  - …

- Stratégie recommendée pour le pilotage de la grille AMR

  i. Mener une simulation au niveau le plus fin (calcul Euler, généralement possible, au moins en 2D)
  ii. Mettre en place les critères de raffinement
  iii. Piloter le raffinement / déraffinement de telle sorte que l'on *colle* au plus près à la simulation « tout Euler »

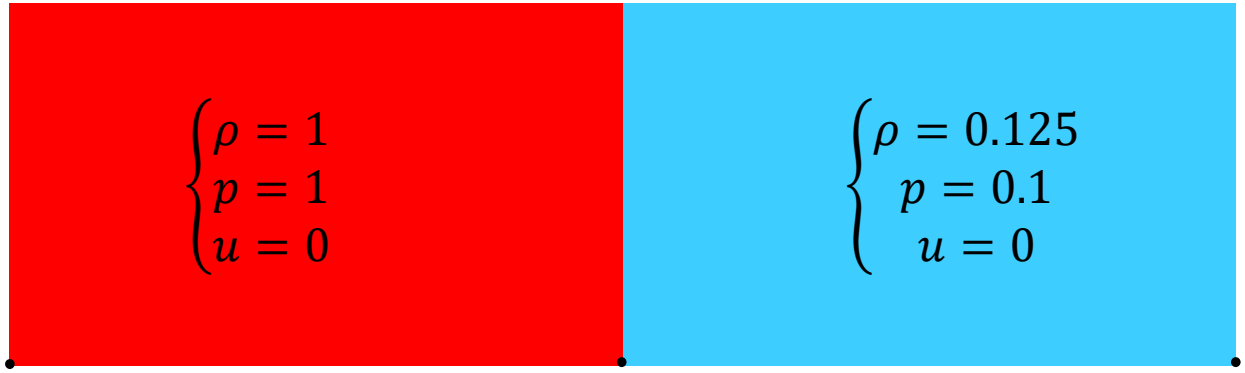
Idée générale : déraffiner sans « casser » le résultat de référence

Pour les cas 3D, on conseille de régler le pilotage de la grille AMR sur un cas 2D axi proche du cas 3D

Partie 2 PRISE EN MAIN DU CODE

Tube à choc de Sod 1D

Discontinuité initiale entre deux gaz parfaits (γ=1.4)



x = 0 x = 0.5 x = 1

Solution : onde de détente et onde choc se propageant respectivement à gauche et à droite d'une discontinuité de contact

Fichier de données Python et Exécution : tube à choc de Sod 1D

```python
#!/usr/bin/env python
##############################################################################
# Copyright ou (C) ou Copr. Commissariat a Energie Atomique et aux Energies   #
# Alternatives, 2020.                                                         #
#                                                                             #
# Ce logiciel est la propriete intellectuelle entiere du CEA.                 #
# Vous pouvez utiliser ce programme uniquement sur des machines du CEA a la   #
# condition que les responsables du programme vous en ont donne               #
# l'autorisation et l'acces.                                                  #
# Il est interdit de copier ce programme et ses sources sur des machines      #
# non administrees par le CEA.                                                #
#                                                                             #
# Le fait que vous puissiez acceder a cet en-tete signifie que vous en avez   #
# pris connaissance et que vous en acceptez les termes.                       #
##############################################################################
# Python : donnees traduites par DataMoira

from hera import *

materiau["defaut"] = Mat( gamma=1.4 )
materiau["defaut"].sorties(bilan_scalaire="qmvt[0]/energie_interne/rho/energie_cinetique")

materiau["haut"] = Mat( gamma=1.4 )
materiau["haut"]  .ini( pmat=1, rho=1 )
materiau["haut"]  .frt( [(0, 0), (0.5, 0), (0.5, 0.1), (0, 0.1), (0, 0)] )

materiau["bas"] = Mat( gamma=1.4 )
materiau["bas"]  .ini( pmat=0.1, rho=0.125 )
materiau["bas"]  .frt( [(0.5, 0), (1, 0), (1, 0.1), (0.5, 0.1), (0.5, 0)] )

schema["Hydro"] = Hydro(
    ordre               = (2, 2),
    cfl_godunov         = 0.8,
    lagrange_schema     = "gad",
    projection_schema   = "vanleer1",

    calcul_dt_vson_moy  = 0,
    meth_press_vson_moy = "rhoc2",
    repart_energie      = "rhoc2",
    ecin_moyen          = "energie_interne",
    relaxation          = 1,

    cond_lim            = ("mur", "mur"),
)

cycle_temps(
    cycle_max  = 5000,
    date_max   = 0.14,
)
```

Important :
prenez bien connaissance de ce banner

- Déclaration des matériaux
  - Nature du matériau (hydro, corps rigide, …)
  - Equation d'état
  - Etat initial
  - Géométrie
  - …
- Configuration du solveur hydro
  - Ordre des schémas
  - Schéma « Phase Lagrange »
  - Schéma « Phase de projection »
  - Traitement des mailles mixtes
  - Conditions aux limites
  - …
- Gestion du temps
  - Temps max. de la simulation
  - Nombre d'itération maximum
  - …

Fichier de données Python et Exécution : tube à choc de Sod 1D

```
sorties (
    bilan_scalaire   = "qmvt[0]/rho/energie_tot",
    coupe1d_nbseg    = 1,
    coupe1d_scalaire = "rho",
    coupe1d_segment  =  ([[0.0], [1.0]],),
    critere_coupe1d  = 1,
    critere_prot     = 1,
    rycoupe1d        = 0.01,
    ryprot           = 0.13,
)

maillage (
    nombre_mailles = (100,),
    min_coord      = (0, ),
    max_coord      = (1, ),
    recouvrement   = 2,
)

geom (
    dimension = 1,
    type      = 0,
)

simu (
    nom              = "SOD",
    liste_schemas    = ["Hydro"],
    liste_materiaux  = ["haut", "bas"]
)

## ========================================================== Simulation

launch_run( "--hercule_prot", cpu = ParallelParam(nomp = 2) )
```

- Sorties
  - Bilans globaux
  - Coupes 1D, stations, …
  - Critère et rythme des sorties
  - Protection-reprise
- Caractéristiques du maillage
  - Bornes du domaine de calcul
  - Nombre de mailles par direction
  - …
- Géométrie
  - Dimension, type (plan, cylindrique, sphérique)
- Définition du cas
  - Nom
  - Liste des schémas
  - Liste des matériaux
- Exécution

Description du listing d'exécution

```
1 arguments :
  SOD.py

 ____________________ Radical - SOD

__executable__    hrun
__script_python__ SOD.py

def simu( cas de nom- SOD)

repertoire de travail du cas- |/ccc/dsku/nfs-server/user/cont001/ocre/jaouens/CA
S_SOD|

Arguments de HERA : <-opr -host-haswell  -omp-2  --hercule_prot>

Arguments de HERA : <-opr -host-haswell  -omp-2  --hercule_prot>


PYTHONPATH-/ccc/dsku/nfs-server/user/cont001/ocre/jaouens/CAS_SOD; export PYTHON
PATH

############ INFORMATIONS OPENMP
### Variable OMP_NUM_THREADS deja definie (-1)
### Nouvelle valeur de OMP_NUM_THREADS (definie par -omp-2) : 2
############
NO ENA_FPE
Option(s) passÃ©e(s) au launcher : vide

############ INFORMATION PARTITION ################
#   Execution sur la partition : haswell
#################################################

Commande lancement armen :  /ccc/work/cont001/armen/armen/opt/armen-1.0.0/bin/ar
men.exe  --hercule_prot
***************************
****   2 threads OpenMP ***
***************************
OpenMP   ACTIF pour mesh
version OpenMP : 201611

HERA_LD_PATH - .
SIMU CEA [ARMEN CEA] 3.1.76/13.0.153/2.24.5/4.1.4/1.0.30/17.8.9/3.0.5/1.0.22/2.6
.0/1.3.17/8.7.17/5.12.23/6.1.47/1.3.19/4.1.10/2.6.1/1.0.3/2.2.4/2.1.6/4.3.56/6.9
.24/8.5.3/1.2.29/1.0.2/3.0.3/5.17.8 (Feb 17 2020)
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------

CODE : ARMEN - RELEASE : 1.0.0 - BUILD 54

Date de gÃ©nÃ©ration de l'exÃ©cutable : Thu Feb 13 14:41:08 2020

pid : 32696, hostname : r1login

Repertoires :
        - courant                       : /ccc/dsku/nfs-server/user/cont001/ocre
/jaouens/CAS_SOD
        - pour fichiers de verification : /tmp/jaouens/CAS_SOD/CHECK/VALEURS/
        - pour fichiers de divergence   : /tmp/jaouens/CAS_SOD/CHECK/DIVERGENCE/
        - pour fichiers piles d'appels  : /tmp/jaouens/CAS_SOD/STACK_TRACE/
        - pour protections internes     : /tmp/jaouens/proc32696_0/20201702-16:5
2/PROT_AMR


Effacements fichiers exÃ©cution prÃ©cÃ©dente... : terminÃ©e.

SchÃ©ma utilisÃ© : Hydro.
```

création du répertoire CAS_SOD

exécutable

répertoires utilisés

```
Lecture dans le bloc sorties

          critere_prot                              : 1

Lecture dans le bloc geom

          dimension                                 : 1
          type                                      : 0

Lecture dans le bloc maillage

          recouvrement                              : 2

Lecture dans le bloc maillage

          nombre_mailles[0]                         : 100
          min_coord[0]                              : 0
          max_coord[0]                              : 1

Lecture dans le bloc cycle_temps

          date_max                                  : 0.14
          cycle_max                                 : 5000
Facteur de raffinement en dimension 0 (direction x nul remplacÃ© par : 2

Dimension 1 (dernier axe x), raffinement 2.
Pas de conditions limites spÃ©cifiÃ©es, murs par dÃ©faut.
Condition limite de type mur (dÃ©faut) gauche en direction x.
Condition limite de type mur (dÃ©faut) droite en direction x.

relevÃ© de frontiÃ"re non implÃ©mentÃ© en 1D

Lecture dans le bloc sorties (Coupe1D)

          critere_coupe1d                           : 1
          coupe1d_scalaire                          : rho
          coupe1d_nbseg                             : 1
          coupe1d_segment[0][0][0]                  : 0
          coupe1d_segment[0][1][0]                  : 1

Lecture dans le bloc sorties (Bilans)

          bilan_scalaire                            : qmvt[0]/rho/energie_tot

Lecture dans le bloc sorties (Bilans/MatÃ©riaux)

          bilan_scalaire (haut)                     : qmvt[0]

                                                      energie_interne

                                                      rho

                                                      energie_cinetique

          bilan_scalaire (bas)                      : qmvt[0]

                                                      energie_interne

                                                      rho

                                                      energie_cinetique


Conditions Limites :


Lecture dans le bloc Hydro

          cond_lim[0]                               : mur
```

lecture des mots-clés

```
          cond_lim[1]                                  : mur

modÃ"les statiques disponibles :

Nom modÃ"le :
StiffenedGasVE
Nom modÃ"le :
StiffenedGasVT
Nom modÃ"le :
MieGruneisen
Nom modÃ"le :
MieGruneisenT
Nom modÃ"le :
mixing
Nom modÃ"le :
PythonUserModel
Nom modÃ"le :
CSharpUserModel

modÃ"les en chargement dynamique :


Lecture dans le bloc Hydro

          pmax_gradient_shlieren                       : 0
          pmax                                         : 0
          cmax                                         : 0
          elim_eclat                                   : 0
          elim_eclat_avec_voisin                       : 0
          elim_eclat_avec_voisin_dilat_autre_mat       : 0

Lecture dans le bloc Hydro

          calcul_dt_vson_moy                           : 0
          repart_energie                               : rhoc2
          meth_press_vson_moy                          : rhoc2
          lagrange_schema                              : gad
          projection_schema                            : vanleer1
          ecin_moyen                                   : energie_interne
          relaxation                                   : 1

Table de marche : init
          cfl_godunov                                  : 0.8
          cfl_godunov                                  : 0.8
          ordre phase lagrange                         : 2
          ordre phase projection                       : 2
          mÃ©thode lagrange ordre 2                     :
          mÃ©thode projection ordre 2                   : vanleer1
          hydro_figee                                  : 0
          pression_figee                               : 0

Table de marche : init

Lecture dans le bloc Hydro

          rycoupe1d                                    : 0.01

Table de marche : init
          ryprot                                       : 0.13

Construction maillage niveau0 ...
Traitement domaine : haut

Traitement domaine : bas
debut pre processing niveau : 0


fin pre processing
```

```
Initialisation fractions de presences jusqu'au niveau 0 .

Lecture dans le bloc

          gamma[1]                                     : 1.4
          gamma[2]                                     : 1.4
          gamma[1]                                     : 1.4
          gamma[2]                                     : 1.4

Lecture des grandeurs partielles mÃ©ca fluide ...
Fin initialisation


Cycle Progression DateFinCycle      DeltaTemps      S rN  ME% NbMailles
-------------------------------------------------------------------------
    0   0.0%  0.000000000000e+00 0.000000000000e+00 G   0        100
------------------------------------------------------------------------- Coupe(s)
 1D  (P0.)
    1   3.2%  4.507489358552e-03 4.507489358552e-03 G   0        100
Protection interne :
 - phase configuration
 - phase Ã©criture grandeurs
 - libÃ©ration tableaux, structures AMR etc ..
 - redistribution des formats mÃ©moires des tableaux AMR


------------------------------
MatÃ©riau : haut : taux_occupation_chunk = 50.0 %
MatÃ©riau : haut : taux_occupation_mat   = 51.0 %
MatÃ©riau : bas  : taux_occupation_chunk = 100.0 %
MatÃ©riau : bas  : taux_occupation_mat   = 50.0 %
------------------------------

Changement taux_occupation_chunk : ancienne valeur : 30% nouvelle_valeur : 10%

Changement : haut --> direct
Changement : bas  --> direct
 - allocation et construction tableaux, structures AMR etc ..
 - diminution de la taille des piles internes
 - relecture phase configuration
 - relecture phase lecture grandeurs
 - fin

    2   5.7%  8.020748044773e-03 3.513258686221e-03 G   0        100 (Prot Inter
ne)
    3   8.0%  1.115172928426e-02 3.130981239489e-03 G   0        100 Coupe(s) 1D

    4  10.0%  1.401771154786e-02 2.865982263602e-03 G   0        100
    5  11.9%  1.669810659815e-02 2.680395050283e-03 G   0        100
Protection interne :
 - phase configuration
 - phase Ã©criture grandeurs
 - libÃ©ration tableaux, structures AMR etc ..
 - redistribution des formats mÃ©moires des tableaux AMR


------------------------------
MatÃ©riau : haut : taux_occupation_mat   = 51.9 %
MatÃ©riau : bas  : taux_occupation_mat   = 49.0 %
------------------------------

 - allocation et construction tableaux, structures AMR etc ..
 - diminution de la taille des piles internes
 - relecture phase configuration
 - relecture phase lecture grandeurs
 - fin

    6  13.8%  1.925825436194e-02 2.560147763797e-03 G   0        100 (Prot Inter
```

temps en fin de cycle

pas du temps du cycle courant

pourcentage du temps simulé jusqu'à la date max.

schéma utilisé

niveau AMR max. lors du cycle

numéro du cycle

nombre de mailles

```
ne)
    7  15.5%  2.176315709705e-02 2.504902735102e-03 G   0        100 Coupe(s) 1D

    8  17.3%  2.422780405699e-02 2.464646959942e-03 G   0        100
    9  19.0%  2.665298526591e-02 2.425181208922e-03 G   0        100
   10  20.8%  2.905569006089e-02 2.402704794975e-03 G   0        100
   11  22.5%  3.144318806349e-02 2.387498002603e-03 G   0        100 Coupe(s) 1D

Protection interne :
 - phase configuration
 - phase Ã©criture grandeurs
 - libÃ©ration tableaux, structures AMR etc ..
 - redistribution des formats mÃ©moires des tableaux AMR


-------------------------------
MatÃ©riau : haut : taux_occupation_mat   = 52.9 %
MatÃ©riau : bas  : taux_occupation_mat   = 48.1 %
-------------------------------

 - allocation et construction tableaux, structures AMR etc ..
 - diminution de la taille des piles internes
 - relecture phase configuration
 - relecture phase lecture grandeurs
 - fin

   12  24.2%  3.382118233072e-02 2.377994267235e-03 G   0        100 (Prot Inter
ne)
```

protection interne + gestion des tableaux compressés (chunk)

indications lors de sorties et protections-reprises

```
 - phase Ã©criture grandeurs
 - libÃ©ration tableaux, structures AMR etc ..
 - redistribution des formats mÃ©moires des tableaux AMR


------------------------------
MatÃ©riau : haut : taux_occupation_mat   = 59.6 %
MatÃ©riau : bas  : taux_occupation_mat   = 41.3 %
------------------------------

 - allocation et construction tableaux, structures AMR etc ..
 - relecture phase configuration
 - relecture phase lecture grandeurs
 - fin

   41  73.8%  1.032886566997e-01 2.412813171243e-03 G   0        100 (Prot Inter
ne)
   42  75.5%  1.057016755502e-01 2.413018850455e-03 G   0        100
   43  77.2%  1.081148154723e-01 2.413139922120e-03 G   0        100
   44  78.9%  1.105287369551e-01 2.413921482808e-03 G   0        100 Coupe(s) 1D

   45  80.7%  1.129430354383e-01 2.414298483187e-03 G   0        100
   46  82.4%  1.153575033538e-01 2.414467915491e-03 G   0        100
Protection interne :
 - phase configuration
 - phase Ã©criture grandeurs
 - libÃ©ration tableaux, structures AMR etc ..
 - redistribution des formats mÃ©moires des tableaux AMR


------------------------------
MatÃ©riau : haut : taux_occupation_mat   = 60.6 %
MatÃ©riau : bas  : taux_occupation_mat   = 40.4 %
------------------------------

 - allocation et construction tableaux, structures AMR etc ..
 - relecture phase configuration
 - relecture phase lecture grandeurs
 - fin

   47  84.1%  1.177720614279e-01 2.414558074089e-03 G   0        100 (Prot Inter
ne)
   48  85.8%  1.201866831172e-01 2.414621689283e-03 G   0        100 Coupe(s) 1D

   49  87.6%  1.226018762900e-01 2.415193172844e-03 G   0        100
   50  89.3%  1.250172441950e-01 2.415367904964e-03 G   0        100
   51  91.0%  1.274326988816e-01 2.415454686601e-03 G   0        100
   52  92.7%  1.298482005196e-01 2.415501638011e-03 G   0        100
Protection interne :
 - phase configuration
 - phase Ã©criture grandeurs
 - libÃ©ration tableaux, structures AMR etc ..
 - redistribution des formats mÃ©moires des tableaux AMR


------------------------------
MatÃ©riau : haut : taux_occupation_mat   = 62.5 %
MatÃ©riau : bas  : taux_occupation_mat   = 38.5 %
------------------------------

 - allocation et construction tableaux, structures AMR etc ..
 - relecture phase configuration
 - relecture phase lecture grandeurs
 - fin

   53  94.5%  1.322641235622e-01 2.415923042620e-03 G   0        100 (Prot Inter
ne) Coupe(s) 1D  (P1.)
   54  96.2%  1.346801858586e-01 2.416062296335e-03 G   0        100
   55  97.9%  1.370963153183e-01 2.416129459772e-03 G   0        100
```

```
   56  99.7%  1.395124804559e-01 2.416165137530e-03 G   0        100
   57 101.4%  1.419286764502e-01 2.416195994345e-03 G   0        100 Coupe(s) 1D
 ----------------------------------------
masse_globale_init  = 5.62500000000000e-01
masse_globale_final = 5.62500000000000e-01
delta = 4.44e-14%

etot_globale_init  = 1.37500000000000e+00
etot_globale_final = 1.37500000000000e+00
delta = 1.11e-14%
----------------------------------------


Fermeture fichier : nombre_de_mailles
Fermeture fichier : pas_de_temps




Statistiques de la simulation numÃ©rique
----------------------------------------

Nb si tout au raffinement maxi  : 100.

Nb cycles de travail                : 58,
Mailles totales Ã M-^W nb cycles      :         5700,
      dont utiles (sans enfant) :          5700,
   d'oÃ¹ efficacitÃ© : 100.00 %.
   Ratio mailles utiles / mailles allouÃ©es : 86.96 %.

Nombre mailles / niveau :
    niveau 0 : 5700 (100.00 %)


Nombre de matÃ©riaux : 3
Nombre de domaines  : 2

Usage mÃ©moire : 8 octets par rÃ©el (simple prÃ©cision),
     physique : 3498355712 rÃ©els (26 Go),
   adressable : 2305843009213693951 rÃ©els (16383 Po),
     utilisÃ©e :   41943040 rÃ©els (40.00 Mo),

Statistiques d'utilisation des ArrayStack internes :

Nombre max de tableaux utilisÃ©s simultanÃ©ment dans les ArrayStack : 25
Nombre max de ArrayStack                  : 1
Nombre d'allocations dans les ArrayStack : 43388
Nombre d'Ã©checs dans les ArrayStack      : 0

Taille des ArrayStack (min / max) : ( 0.25 / 16.00 ) Mo

Nombre maximum simultanÃ© de tableaux temporaires aux maillages : 19
Nombre de tableaux aux maillages recyclÃ©s : 2755
Nombre moyen de tableaux aux maillages recyclÃ©s par cycle : 47.50

Usage mÃ©moire statistique :
      % phys. : 1.20 %,
      % disp. : 0.00 %,
   par maille : 52428.80 rÃ©els (409 Ko).

Temps de travail total : 0.51 s,
      dont utilisateur : 0.33 s,
          dont systÃ¨me : 0.05 s

Temps de travail total hors initialisation: 0.41 s,
      dont utilisateur : 0.28 s,
          dont systÃ¨me : 0.03 s.
```

```
Statistiques de la simulation numÃ©rique
----------------------------------------

Nb si tout au raffinement maxi  : 100.

Nb cycles de travail                : 58,
Mailles totales ÃM-^W nb cycles      :          5700,
       dont utiles (sans enfant) :          5700,
   d'oÃ¹ efficacitÃ© : 100.00 %.
   Ratio mailles utiles / mailles allouÃ©es : 86.96 %.

Nombre mailles / niveau :
    niveau 0 : 5700 (100.00 %)


Nombre de matÃ©riaux : 3
Nombre de domaines  : 2

Usage mÃ©moire : 8 octets par rÃ©el (simple prÃ©cision),
     physique : 16871356416 rÃ©els (125 Go),
   adressable : 2305843009213693951 rÃ©els (16383 Po),
     utilisÃ©e :   41943040 rÃ©els (40.00 Mo),

Statistiques d'utilisation des ArrayStack internes :

Nombre max de tableaux utilisÃ©s simultanÃ©ment dans les ArrayStack : 25
Nombre max de ArrayStack                    : 1
Nombre d'allocations dans les ArrayStack : 43388
Nombre d'Ã©checs dans les ArrayStack      : 0

Taille des ArrayStack (min / max) : ( 0.25 / 16.00 ) Mo

Nombre maximum simultanÃ© de tableaux temporaires aux maillages : 19
Nombre de tableaux aux maillages recyclÃ©s : 2755
Nombre moyen de tableaux aux maillages recyclÃ©s par cycle : 47.50

Usage mÃ©moire statistique :
      % phys. : 0.25 %,
      % disp. : 0.00 %,
   par maille : 52428.80 rÃ©els (409 Ko).

Temps de travail total : 0.47 s,
      dont utilisateur : 0.30 s,
          dont systÃ¨me : 0.03 s

Temps de travail total hors initialisation: 0.38 s,
      dont utilisateur : 0.24 s,
          dont systÃ¨me : 0.02 s.

Temps de travail par maille : 81.50 Âµs / 163.01 Âµs (par coeur),
           dont utilisateur : 50.91 Âµs / 101.83 Âµs (par coeur),
           dont utilisateur hors initialisation : 41.96 Âµs / 83.91 Âµs (par coe
ur),
               dont systÃ¨me : 5.18 Âµs / 10.35 Âµs (par coeur)

Taille plus petite maille : 1.00e-02 (si MKS : 1.00e+01 mm)
si MKS : nb mailles / millimÃ¨tre : 0.1

Date max atteinte : 1.40e-01

.
```

statistiques maillage

statistiques d'utilisation mémoire

statistiques sur le temps consommé

raison de l'arrêt de l'exécution

Répertoire du cas

- Contenu du répertoire du cas `CAS_SOD`

`$ls -al CAS_SOD :`

```
(c-inti)[jaouens@r1login CAS_SOD] $ ll
total 20
-rw-r----- 1 jaouens ocre   90 Feb 17 16:52 bas.frt
-rw-r----- 1 jaouens ocre 1479 Feb 17 16:52 donnees
-rw-r----- 1 jaouens ocre   88 Feb 17 16:52 haut.frt
drwxr----- 6 jaouens ocre 4096 Feb 17 16:52 POST1D
drwxrwx--- 2 jaouens ocre 4096 Feb 17 16:52 PROT_AMR
-rw-r----- 1 jaouens ocre   38 Feb 17 16:52 temps_execution.dat
-rw-rw---- 1 jaouens ocre   43 Feb 17 16:52 TEMPS_PROT
-rw-rw---- 1 jaouens ocre   35 Feb 17 16:52 T_OK
(c-inti)[jaouens@r1login CAS_SOD] $
```

données
fichier de données ASCII généré

***.frt**
fichier de contour des matériaux (.frt())

TEMPS_PROT
temps des protections-reprises

POST1D
répertoire des sorties 1D (particules traceuses, stations, bilans, …)

PROT_AMR
répertoire des protections/reprises

T_OK
Généré lorsque date_max atteinte

```
$ module load grace
$xmgrace CAS_SOD/*.frt
```

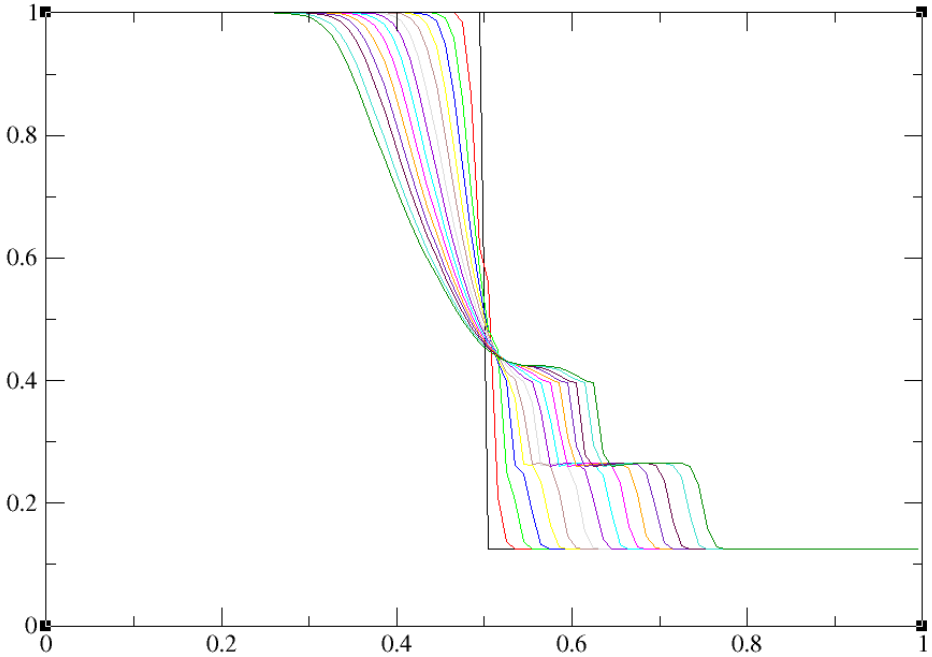
Dépouillement du cas 1D

```
(c-inti)[jaouens@r1login CAS_SOD] $ ll
total 20
-rw-r----- 1 jaouens ocre   90 Feb 17 16:52 bas.frt
-rw-r----- 1 jaouens ocre 1479 Feb 17 16:52 donnees
-rw-r----- 1 jaouens ocre   88 Feb 17 16:52 haut.frt
drwxr----- 6 jaouens ocre 4096 Feb 17 16:52 POST1D
drwxrwx--- 2 jaouens ocre 4096 Feb 17 16:52 PROT_AMR
-rw-r----- 1 jaouens ocre   38 Feb 17 16:52 temps_execution.dat
-rw-rw---- 1 jaouens ocre   43 Feb 17 16:52 TEMPS_PROT
-rw-rw---- 1 jaouens ocre   35 Feb 17 16:52 T_OK
(c-inti)[jaouens@r1login CAS_SOD] $ cd POST1D/
(c-inti)[jaouens@r1login POST1D] $ ll
total 16
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 COUPES
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 EVOL
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 RELEVE_Frontiere
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 TE
(c-inti)[jaouens@r1login POST1D] $ cd COUPES/
(c-inti)[jaouens@r1login COUPES] $ ll
total 124
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00000
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00001
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00002
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00003
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00004
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00005
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00006
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00007
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00008
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00009
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00010
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00011
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00012
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00013
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00014
-rw-rw---- 1 jaouens ocre  255 Feb 17 16:52 TEMPS_COUPE1D
(c-inti)[jaouens@r1login COUPES] $
```

```
$ xmgrace rho_0_sort000*
```

Dépouillement du cas 1D

```
(c-inti)[jaouens@r1login CAS_SOD] $ ll
total 20
-rw-r----- 1 jaouens ocre   90 Feb 17 16:52 bas.frt
-rw-r----- 1 jaouens ocre 1479 Feb 17 16:52 donnees
-rw-r----- 1 jaouens ocre   88 Feb 17 16:52 haut.frt
drwxr----- 6 jaouens ocre 4096 Feb 17 16:52 POST1D
drwxrwx--- 2 jaouens ocre 4096 Feb 17 16:52 PROT_AMR
-rw-r----- 1 jaouens ocre   38 Feb 17 16:52 temps_execution.dat
-rw-rw---- 1 jaouens ocre   43 Feb 17 16:52 TEMPS_PROT
-rw-rw---- 1 jaouens ocre   35 Feb 17 16:52 T_OK
(c-inti)[jaouens@r1login CAS_SOD] $ cd POST1D/
(c-inti)[jaouens@r1login POST1D] $ ll
total 16
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 COUPES
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 EVOL
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 RELEVE_Frontiere
drwxr----- 2 jaouens ocre 4096 Feb 17 16:52 TE
(c-inti)[jaouens@r1login POST1D] $ cd COUPES/
(c-inti)[jaouens@r1login COUPES] $ ll
total 124
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00000
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00001
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00002
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00003
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00004
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00005
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00006
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00007
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00008
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00009
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00010
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00011
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00012
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00013
-rw-rw---- 1 jaouens ocre 4231 Feb 17 16:52 rho_0_sort00014
-rw-rw---- 1 jaouens ocre  255 Feb 17 16:52 TEMPS_COUPE1D
(c-inti)[jaouens@r1login COUPES] $
```




```
$ xmgrace rho_0_sort000*
```

Options d'exécution (non exhaustif)

- Protection/reprise
  - `-lp` : repartir de la dernière protection (PROT_AMR/num-lp)
  - `-p 3` : repartir de la protection numéro 3 (si existante)
  - `-tp` : si aucune protection, reprend à t = 0 sinon repart de la dernière protection
  - `--hercule_prot` : protection/reprise en mode Hercule
- Suppression de fichiers
  - `-d` : détruit les sorties UCD d'une précédente exécution
  - `-da` : detruit toutes les sorties d'une précédente exécution
- Statistiques (consommation mémoire, grind time)
  - `--print_stats` : affichage du coût du cycle et de la consommation mémoire par nœud

Configuration

- « Sourcer » la version 1.0.0 d'ARMEN

```
$ module purge
$ module load /ccc/work/cont003/armen/armen/opt/armen-1.1.0/share/armen/1.1.0
```

- Vérification

```
$ which hrun                ← commande utilisée pour le lancement d'ARMEN
> /ccc/work/cont003/armen/armen/opt/armen-1.0.0/bin/hrun
```

- Récupérer le script python du cas

```
$ cd $CCCWORKDIR
$ mkdir TEST-SOD
$ cp -r /ccc/work/cont003/ocre/jaouens03/CentraleSupélec/SOD.py TEST-SOD
$ cd TEST-SOD
$ python3 SOD.py ← commande de lancement du cas, appelle « hrun »
```

→ **Pour la prochaine séance : se connecter à inti et refaire l'exercice.**

Partie 3

ST7 : PROJETS CEA-DAM

Contexte

- ***HPC : au service de nombreuses missions du CEA-DAM***

  → notamment dans le domaine de la sécurité et de la surveillance de l'environnement :
  - surveillance du respect des traités internationaux (TICE) ;
  - lutte contre la prolifération ;
  - activités sismiques, alertes aux tsunamis (CENALT);
  - etc.

- ***HPC : au cœur des missions de notre département***
  - supercalculateurs et leurs servitudes
  - codes massivement parallèles et leurs environnements

Dans ce cadre, nous avons imaginé deux exercices répondant aux objectifs de la ST7 qui utiliseront les moyens de calculs du centre DIF (machine + code HPC)

1. Caractérisation d'une source à l'échelle urbaine avec prise en compte du bâti
2. Mise en place d'un réseau de détection d'ondes infrasonores

Note: il s'agit de cas d'écoles, complètement imaginaires, d'ailleurs pas tout à fait réalistes, mais représentatifs des applications et adaptés aux objectifs de la ST7

Outils mis à disposition

Pour mener à bien ces études, on utilisera :

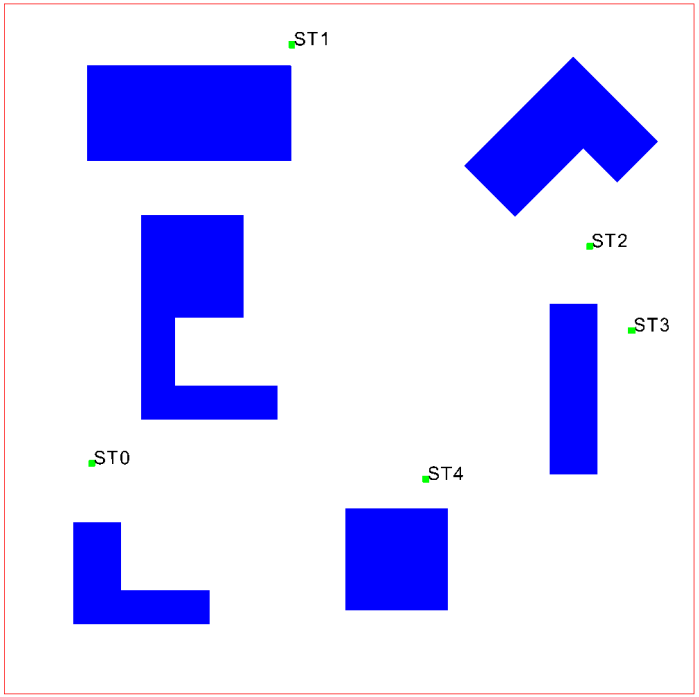
- Le code ***ARMEN***
  - Code eulérien multi matériaux d'hydrodynamique compressible sur grille cartésienne 1D-2D-3D
  - Raffinement adaptatif de maillage (AMR tree-based)
  - Traitement de type « frontière immergée » pour les obstacles fixes (topographie, bâti)
  - Schémas numériques d'ordre très élevé (adaptés aux problèmes de propagation d'ondes sur de grandes distances)
  - Scripting Python du jeu de données et du lancement des calculs (facilite les études paramétriques)
  - Modèle utilisateur C# pour surcharger des grandeurs (prise en compte du vent)
  - Massivement parallèle, avec trois niveaux de parallélisme :
    - Macro : décomposition de domaine ***MPI*** avec équilibrage dynamique de charge
    - Méso : multithreading ***OpenMP***
    - Micro : vectorisation Single Instruction Multiple Data (***SIMD***)
- Le calculateur ***INTI***
  - De nombreuses partitions. Nous utiliserons vraisemblablement :
    - La partition sklb : 32 nœuds de 48 cœurs (soit 1536 cœurs)
    - La partition sandy : 201 nœuds de 16 cœurs (soit 3216 cœurs)
  - Attention : ressources à partager entre tous les utilisateurs du calculateur…

Caractérisation d'une source à l'échelle urbaine avec prise en compte du bâti

▶ Caractérisation d'une source à l'échelle urbaine avec prise en compte du bâti

(Exercice fictif, adapté du problème dit du « sniper »)

→ **Objectif: localiser et déterminer la puissance d'une explosion sur la base de relevés enregistrés par des stations de mesure**



Pour cet exercice, seront connus :

- La position des capteurs de pression
- Les signaux enregistrés par ces capteurs

→ *Mise au point d'une boucle d'optimisation faisant appel au code de simulation, en faisant varier la position et la puissance de la source*

- Première étape : cas d'exercice en 2D plan
- Deuxième étape : cas plus réaliste en 3D, avec topographie

Description du script Python *SouffleBati2D.py*


```
#! /usr/bin/env python
# -*- coding:utf-8 -*-
##############################################################################
# Copyright ou (C) ou Copr. Commissariat a Energie Atomique et aux Energies   #
# Alternatives, 2020.                                                         #
#                                                                             #
# Ce logiciel est la propriete intellectuelle entiere du CEA.                 #
# Vous pouvez utiliser ce programme uniquement sur des machines du CEA a la   #
# condition que les responsables du programme vous en ont donne               #
# l'autorisation et l'acces.                                                  #
# Il est interdit de copier ce programme et ses sources sur des machines      #
# non administrees par le CEA.                                                #
#                                                                             #
# Le fait que vous puissiez acceder a cet en-tete signifie que vous en avez   #
# pris connaissance et que vous en acceptez les termes.                       #
##############################################################################

import math, os

from hera import *
from utilitaires_hera import *
from validation_hera import *
```

```python
# ----------------------------------------------------
# Ecriture d'une liste de points (x,y,z) dans un fichier
# ----------------------------------------------------

def write_file(dim, file, lst, mode = "w"):
    print "\nEcriture du fichier :",file
    f=open(file,mode)
    if (dim == 1):
        for l in lst:
            f.write(str(l[0])+"\n")
    elif (dim == 2):
        for l in lst:
            f.write(str(l[0])+" "+str(l[1])+"\n")
    elif (dim == 3):
        for l in lst:
            f.write(str(l[0])+" "+str(l[1])+" "+str(l[2])+"\n")
    f.close()

# ----------------------------------------------------
# Fonction cercle
# ----------------------------------------------------

def cercle(R , theta_deb, theta_fin, theta_nb, xcentre = 0., ycentre = 0.) :

    liste_frt = []

    theta_pas = math.fabs(theta_fin - theta_deb) / theta_nb

    for i in range(theta_nb + 1) :

        theta_deg = theta_deb + theta_pas * i
        theta_rad = math.pi * theta_deg / 180.

        x = xcentre + R * math.cos(theta_rad)
        y = ycentre + R * math.sin(theta_rad)

        if math.fabs(x) < 1.e-15 : x = 0.
        if math.fabs(y) < 1.e-15 : y = 0.

        liste_frt.append( (x,y) )

    return liste_frt
```

Description du script Python *SouffleBati2D.py*

```
##########################################################################################
##########################################################################################
# LA DEFINITION DU CAS
##########################################################################################
##########################################################################################

def CAS_SB2D(nom_simu, amr, pas, M_tnt, tps_f, XC, YC):

    Lx = 100.0
    Ly = 100.0
    Lz = 100.0

    xmin = 0.0          # Borne xmin du domaine
    ymin = 0.0          # Borne ymin du domaine
    zmin = 0.0          # Borne zmin du domaine

    xmax = xmin + Lx    # Borne xmax du domaine
    ymax = ymin + Ly    # Borne ymax du domaine
    zmax = zmin + Lz    # Borne zmax du domaine

    raff = 3            # Facteur de raffinement
    nmax = 3            # Nombre max de niveaux de raffinement
    if amr == 0: nmax = 0

    # Fonction calculant le nombre de mailles au niveau le plus grossier, connaissant
    # le pas d'espace au niveau le plus fin et les bornes du domaine
    # Pour "tomber sur un entier, on autorise une légère mofication des bornes du domaine :
    # modif = "max"    : modification de la borne max autorisée
    # modif = "min"    : modification de la borne min autorisée
    # modif = "minmax" : modification des bornes min et max autorisées

    [nx, xmin, xmax] = calculerNombreDeMailles2(pas, xmin, xmax, nmax, raff, modif="max")
    [ny, ymin, ymax] = calculerNombreDeMailles2(pas, ymin, ymax, nmax, raff, modif="max")
    [nz, zmin, zmax] = calculerNombreDeMailles2(pas, zmin, zmax, nmax, raff, modif="max")

    fact = int(math.pow(raff, nmax))

    # Affichage dans le listing, par direction :
    # - des bornes du domaine ;
    # - du nombre de mailles ;
    # - de la résolution spatiale.
    print "xmin = %e -> xmax = %e, nx = %d, dx = %e" % (xmin, xmax, nx, (xmax - xmin) / (nx * fact))
    print "ymin = %e -> ymax = %e, ny = %d, dy = %e" % (ymin, ymax, ny, (ymax - ymin) / (ny * fact))
    print "zmin = %e -> zmax = %e, nz = %d, dz = %e" % (zmin, zmax, nz, (zmax - zmin) / (nz * fact))
```

Description du script Python *SouffleBati2D.py*

```
##########################################################################
#                                                                        #
#                Définitions des materiaux                               #
#                                                                        #
##########################################################################

    # Nature des matériaux, EOS, etc.

    materiau["source"] = Mat(gamma                  = 3.0,    # Gaz parfait, gamma = 3
                             corps_rigide           = 0,      # Pas de traitement "corps rigide"
                             ordre_phase_lagrange   = 1,      # Schéma d'ordre 1 en phase Lagrange
                             ordre_phase_projection = 1,      # Schéma d'ordre 1 en phase de projection
                             elim_eclat_avec_voisin = 0.9,    # Critère d'élimination d'éclats qui feraint
                             frac_voisin_eclat      = 5.e-2)  # chuter le pas de temps

    materiau["air"]    = Mat(gamma        = 1.4,              # Gaz parfait, gamma = 7/5
                             corps_rigide = 0)                 # Pas de traitement "corps rigide"

    materiau["bati"]   = Mat(gamma                  = 1.4,    # Gaz parfait, gamma = 7/5
                             corps_rigide           = 1,      # Traitement "corps rigide" pour le bâti
                             ordre_phase_lagrange   = 1,      # Schéma d'ordre 1 en phase Lagrange
                             ordre_phase_projection = 1)      # Schéma d'ordre 1 en phase de projection

    materiau["bati2"] = Mat() ; materiau["bati2"].equiv("bati") # "bati2" -> sera fusionné "bâti" après l'init
    materiau["bati3"] = Mat() ; materiau["bati3"].equiv("bati") # "bati3" -> sera fusionné "bâti" après l'init
    materiau["bati4"] = Mat() ; materiau["bati4"].equiv("bati") # "bati4" -> sera fusionné "bâti" après l'init
    materiau["bati5"] = Mat() ; materiau["bati5"].equiv("bati") # "bati5" -> sera fusionné "bâti" après l'init
    materiau["bati6"] = Mat() ; materiau["bati6"].equiv("bati") # "bati6" -> sera fusionné "bâti" après l'init

    ################################################################

    # Etat initial

    rho0 = 1.29   ; p0 = 1e5
    rho1 = 1200.0 ; p1 = 60e8;

    R_source = math.pow(( 3.0 * M_tnt ) / (4.0 * math.pi * rho1), 1. / 3.) # Rayon de la source
    print "Rayon de la source : ", R_source

    materiau["source"].ini(rho=rho1,  pmat=p1, vitesse=(0., 0.)) # Conditions initiales sur chaque matériau
    materiau["air"]   .ini(rho=rho0,  pmat=p0, vitesse=(0., 0.))
    materiau["bati"]  .ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))
    materiau["bati2"] .ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))
    materiau["bati3"] .ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))
    materiau["bati4"] .ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))
    materiau["bati5"] .ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))
    materiau["bati6"] .ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))

    ################################################################
```

Description du script Python *SouffleBati2D.py*

```
# Initialisation géométrique

# LA SOURCE
SOURCE = cercle(R_source, 0., 360., 3600, XC, YC)  # Liste de points décrivant la frontière SOURCE
materiau["source"].frt(SOURCE)                     # Définition de la frontière du matériau "source"

# L'AIR
# Remarque : l'air est défini sur tout le domaine. Il devra être défini en 1er dans la liste des
# matériaux. Les autres viendront se "superposer" dessus
AIR = [ (xmin, ymin), (xmax, ymin) , (xmax, ymax), (xmin, ymax) ]
materiau["air"].frt( AIR )

# LE BATI

####### Batiment 1
x0 = 10.08 ; y0 = 10.223
x1 = 30.08 ; y1 = 10.223
x2 = 30.08 ; y2 = 15.223
x3 = 17.08 ; y3 = 15.223
x4 = 17.08 ; y4 = 25.223
x5 = 10.08 ; y5 = 25.223

Bat1    = [ (x0, y0), (x1, y1) , (x2, y2), (x3,y3), (x4, y4), (x5, y5) ]
materiau["bati"].frt( Bat1 )

####### Batiment 2
x0 = 50.08 ; y0 = 12.223
x1 = 65.08 ; y1 = 12.223
x2 = 65.08 ; y2 = 27.223
x3 = 50.08 ; y3 = 27.223

Bat2    = [ (x0, y0), (x1, y1) , (x2, y2), (x3,y3) ]
materiau["bati2"].frt( Bat2 )

####### Batiment 3
x0 = 80.08 ; y0 = 32.223
x1 = 87.08 ; y1 = 32.223
x2 = 87.08 ; y2 = 57.223
x3 = 80.08 ; y3 = 57.223

Bat3    = [ (x0, y0), (x1, y1) , (x2, y2), (x3,y3) ]
materiau["bati3"].frt( Bat3 )

####### Batiment 4
```

Etc…

Description du script Python *SouffleBati2D.py*

```
##############################################################################
#                                                                            #
#                Définitions du schema numerique                             #
#                                                                            #
##############################################################################

    LAGRANGE  = 3 # ordre du schéma dans la phase Lagrange
    ADVECTION = 3 # ordre du schéma dans la phase de projection

    if LAGRANGE == 1: # Schéma Lagrange d'ordre 1
        lag_sch = "acoustique"
        lag_lim = "aucun"
    elif LAGRANGE==2 : # Schéma Lagrange d'ordre 2 avec limiteur
        lag_sch = "gad"
        lag_lim = "superbee"
    else:              # Schéma Lagrange d'ordre > 2 avec limiteur
        lag_sch = "gaia"
        lag_lim = "tvd"

    if ADVECTION == 1: # Schéma Projection d'ordre 1
        proj_sch = "upwind"
        proj_lim = "aucun"
    else:              # Schéma Projection d'ordre > 2 avec limiteur
        proj_sch = "reconstruction_polynomiale"
        proj_lim = "tvd"
```

Description du script Python *SouffleBati2D.py*

```
schema["Hydro"] = Hydro( # Bloc "Schéma Hydro"

    frontiere_rigide = 1,           # Activation du traitement "Frontières immergées" pour les corps rigides

    ordre = (LAGRANGE, ADVECTION),  # Ordre des schemas pour la phase lagrange et la phase de projection

    lagrange_schema     = lag_sch,  # Schema hydro pour la phase lagrange
    lagrange_limiteur   = lag_lim,  # Limiteur pour la phase lagrange
    projection_schema   = proj_sch, # Schema hydro pour la phase de projection
    projection_limiteur = proj_lim, # Limiteur pour la phase de projection

    lagrange_mixte_ordre1            = 1,  # schema lagrange      a l ordre 1 sur les mailles mixtes
    pente_mixte_ordre1               = 1,  # schema de projection a l ordre 1 sur les mailles mixtes
    lagrange_transition_amr_ordre1   = 1,  # schema lagrange      a l ordre 1 sur les transitions de niveau
    projection_transition_amr_ordre1 = 1,  # schema de projection a l ordre 1 sur les transitions de niveau

    raff_crit              = "shlieren",   # critère de raffinement (gradients de pression, shlieren)
    severite_normalise_mat = 1,            # normalisation de la severite par materiau activée (0 desactivée)

    cond_lim = ("flux", "flux", "flux", "flux", "flux", "flux"), # CL de type flux sortants

    lag_croix   = 0,   # Stratégie de splitting Lagrange + projection
    cfl_godunov = 0.7, # Condition de Courant-Friedricks-Levy

    meth_press_vson_moy = "pond_vol",          # Modèle de mailles mixtes : pression moyenne
    repart_volume       = "volume",            # Modèle de mailles mixtes
    repart_energie      = "masse",             # Modèle de mailles mixtes
    ecin_moyen          = "vitesse_partielle", # Modèle de mailles mixtes
    relaxation             = 1,                # Relaxation des pressions activée dans les mailles mixtes
    option_energie_interne = 1,                # Modèle de mailles mixtes

    calcul_dt_vson_moy      = 0, # Calcul du pas de temps à partir de la vitesse moyenne
    elim_vson_neg           = 1, # Suppression du matériau si la vitesse du son est négative
    rho_min_dilat_autre_mat = 1, # Dilatation des autres matériaux ds la mille si on en élimine un
)

geom(dimension = 2, type = 0)
```

Description du script Python *SouffleBati2D.py*

```
# Position des stations
x0 = 12.82 ; y0 = 33.83
x1 = 42.21 ; y1 = 95.27
x2 = 85.95 ; y2 = 65.66
x3 = 92.13 ; y3 = 53.32
x4 = 61.87 ; y4 = 31.51

station_coord = [ (x0, y0), (x1, y1), (x2, y2), (x3, y3), (x4, y4) ]

liste_station_coord = station_coord
fic_station = "Stations"+nom_simu+".dat"

REP_CAS=os.path.join(os.getcwd(),"CAS_"+nom_simu)
if (not os.path.exists(REP_CAS)) : os.makedirs(REP_CAS)

station_coord_fic=os.path.join(REP_CAS,fic_station)

# Ecriture des stations
write_file(2,station_coord_fic,liste_station_coord)
liste_station_scalaire = ("pmat",) # Choix du scalaire enregistré par la station

if amr == 1 : sseverite = 1 # Sortie de la severite
else        : sseverite = 0 # pas de sortie de la severite

dico_rydep     = {0.0 : 0.001,} # Table de marche du rythme des sorties
dico_ryprot    = {0.0 : 0.1,}   # Table de marche du rythme des protections
dico_rystation = {0.0 : 1.e-6,} # Table de marche du rythme des sorties station

# critere_xxx => 0 = pas de sorties xxx
#                1 = sortie xxx avec un rythme en temps     (a associer ryxxx)
#                2 = sortie xxx avec un rythme en cycles    (a associer ryxxx_cycles)
#                3 = sortie xxx avec un rythme en temps cpu (a associer ryxxx_cpu)
# xxx vaut pour : prot, hdep
sorties(
        critere_hdep = 1, # Sorties Hercule sur criteres de date
        critere_prot = 1, # Protections sur criteres de date

        ryhdep = dico_rydep, # Rythme des sorties
        ryprot = dico_ryprot,# Rythme des protections

        protection_nb_fichiers = 2,         # Nombre de protections que l'on garde (2 dernières)
        sortie_severite        = sseverite, # sortie de la severite suivant le cas (simu amr ou sans amr)

        rystation          = dico_rystation,         # Rythme des sorties station
        station_scalaire   = liste_station_scalaire, # scalaires à enregistrer par les stations
        station_coord_fich = station_coord_fic,      # coordonnées des stations
        )
```

Description du script Python *SouffleBati2D.py*

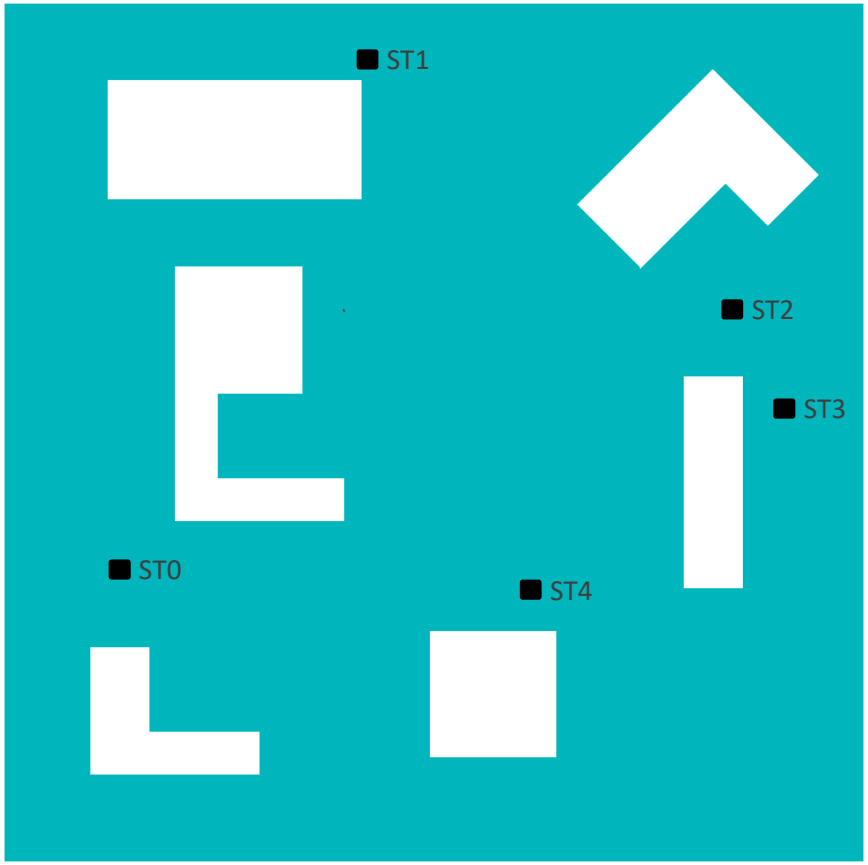
```
cycle_temps(date_max = tps_f,  # Date max de la simulation
            dt_stop  = 1.e-14) # Arrêt de la simulation si le pas de temps est inférieur à dt_stop

 raffinement(fact_raff      = raff,    # facteur de raffinement (par direction)
            niv_max        = nmax,     # nombre max de niveaux AMR
            extra          = 3,        # zone tampon (en mailles) autour d une frontiere
            extra_frt      = {"air/bati": 3, "source/bati" : 3, "air/source" : 3}, # idem
            propag_sev     = 3,        # zone tampon autour d une maille raffinable par le critere de severite
            seuil_raffine  = 0.05,     # seuil (severite) au-dela duquel on va raffiner
            seuil_regroupe = 0.0005,   # seuil (severite) en-deca duquel on va regrouper
            freq_raffine   = 4,        # on regarde tous les 4 cycles si on doit raffiner
            freq_regroupe  = 2,        # on regarde tous les 2 * freq_raffine cycles (= 8) si on doit regrouper
            niv_min_frt    = {"air/bati": nmax, "source/bati" : nmax, "air/source" : nmax}, # niv min par frt
            )

maillage(nombre_mailles = (nx, ny, nz),       # Nombre de mailles par direction
         min_coord      = (xmin, ymin, zmin), # Bornes min du domaine
         max_coord      = (xmax, ymax, zmax), # Bornes max du domaine
         recouvrement   = 3,                  # Nombre de mailles "fantômes"
         )

interface(mosso=0) # Pas d'algorithme mosso sur les mailles à plus de deux matériaux

#########################################################################

# Création du cas (nouveau répertoire CAS_nom_simu)
simu(nom            = nom_simu,
     liste_schemas  = ["Hydro"],
     liste_materiaux= ("air", "bati", "bati2", "bati3", "bati4", "bati5", "bati6", "source")
     )

##################################################################################
##################################################################################
##################################################################################
```

```
##########################################################################################
##########################################################################################
##########################################################################################

TPS_F = 0.2001 # Temps final (en s)
PAS   = 0.1    # Résolution spatiale la plus fine (en m)
AMR   = 1      # Activation de l'AMR

# Les paramètres de l'étude :
M_TNT = 10.00  # masse d'explosif (en kg)
XC    = 40     # Position X de la source dans cet exemple
YC    = 65     # Position Y de la source dans cet exemple

# Nom du cas
radical  = "SB_2D_" + str(PAS*1e2) + "_cm_"
radical += str(XC) + "_" + str(YC) + "_"
radical += str(M_TNT) + "kg_"
NOM      = radical

#Création et exécution ( ici sur 4 ss domaines MPI avec 4 threads OpenMP (16 coeurs)
CAS_SB2D(NOM, AMR, PAS, M_TNT, TPS_F, XC, YC)

launch_run("--print_stats ", cpu = ParallelParam(npart = 4, nomp = 4))
```

-np 4
(décomposition de domaine MPI)

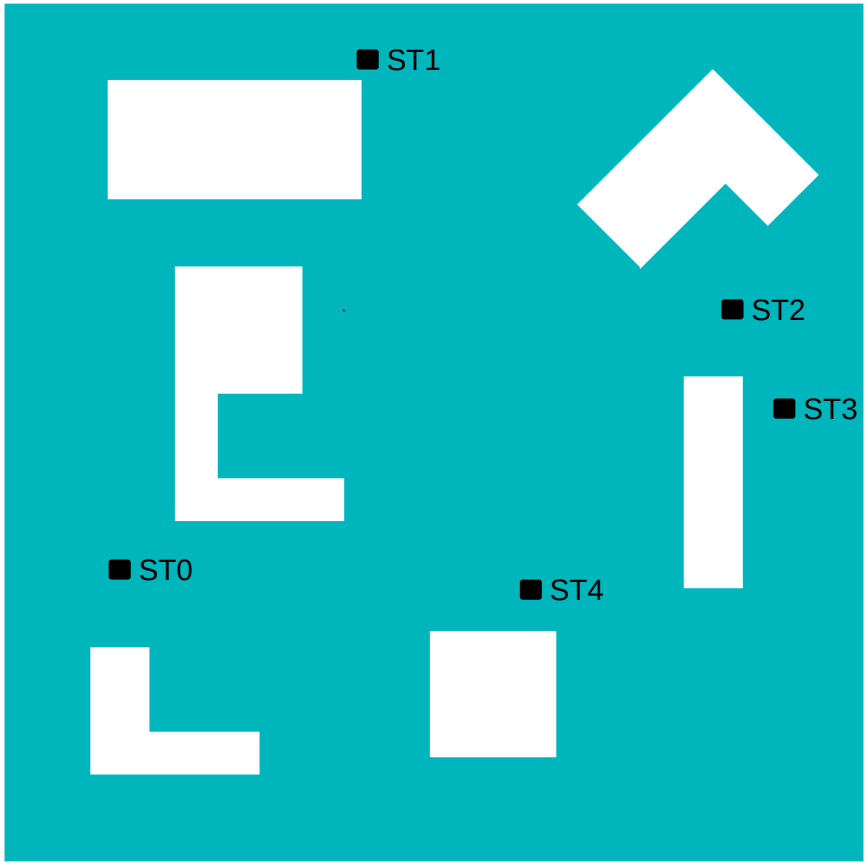
-omp=4
(multithreading OpenMP)

▶ Exemple d'une simulation 2D pour une source donnée



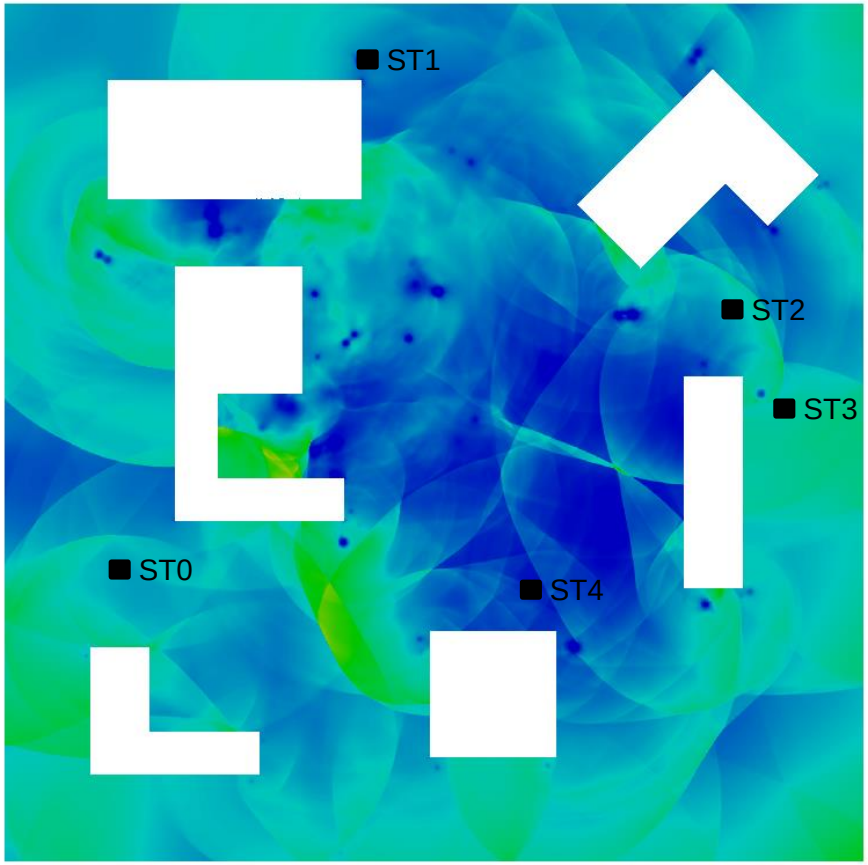
- Onde de souffle
- Propagation d'une onde de choc dans l'air
- Interaction avec le bâti
- Réflexions multiples
- Interactions d'ondes

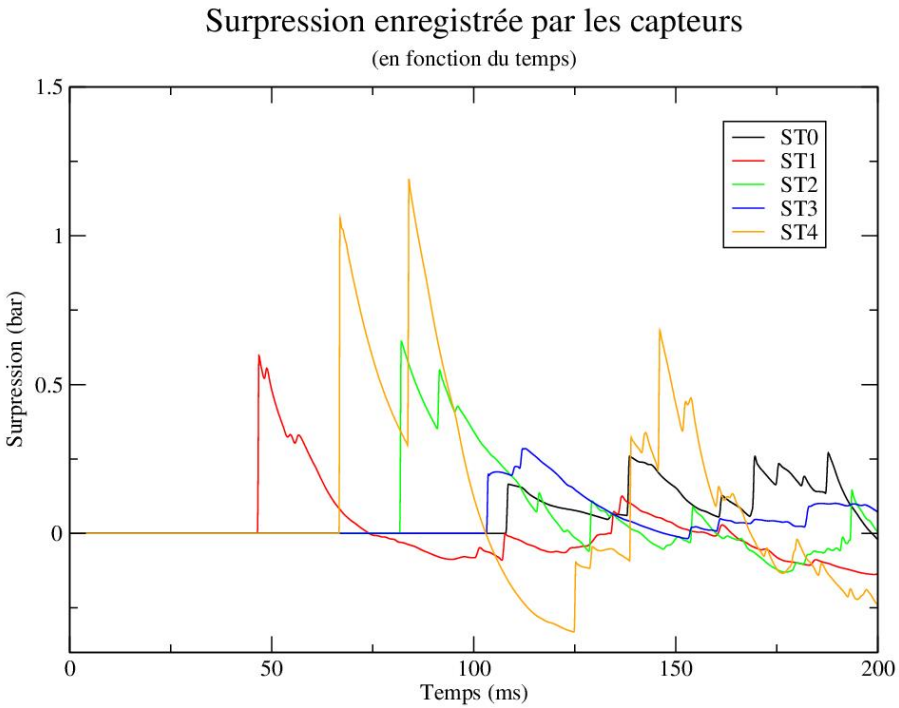
▶ Exemple d'une simulation 2D pour une source donnée



- Onde de souffle
- Propagation d'une onde de choc dans l'air
- Interaction avec le bâti
- Réflexions multiples
- Interactions d'ondes

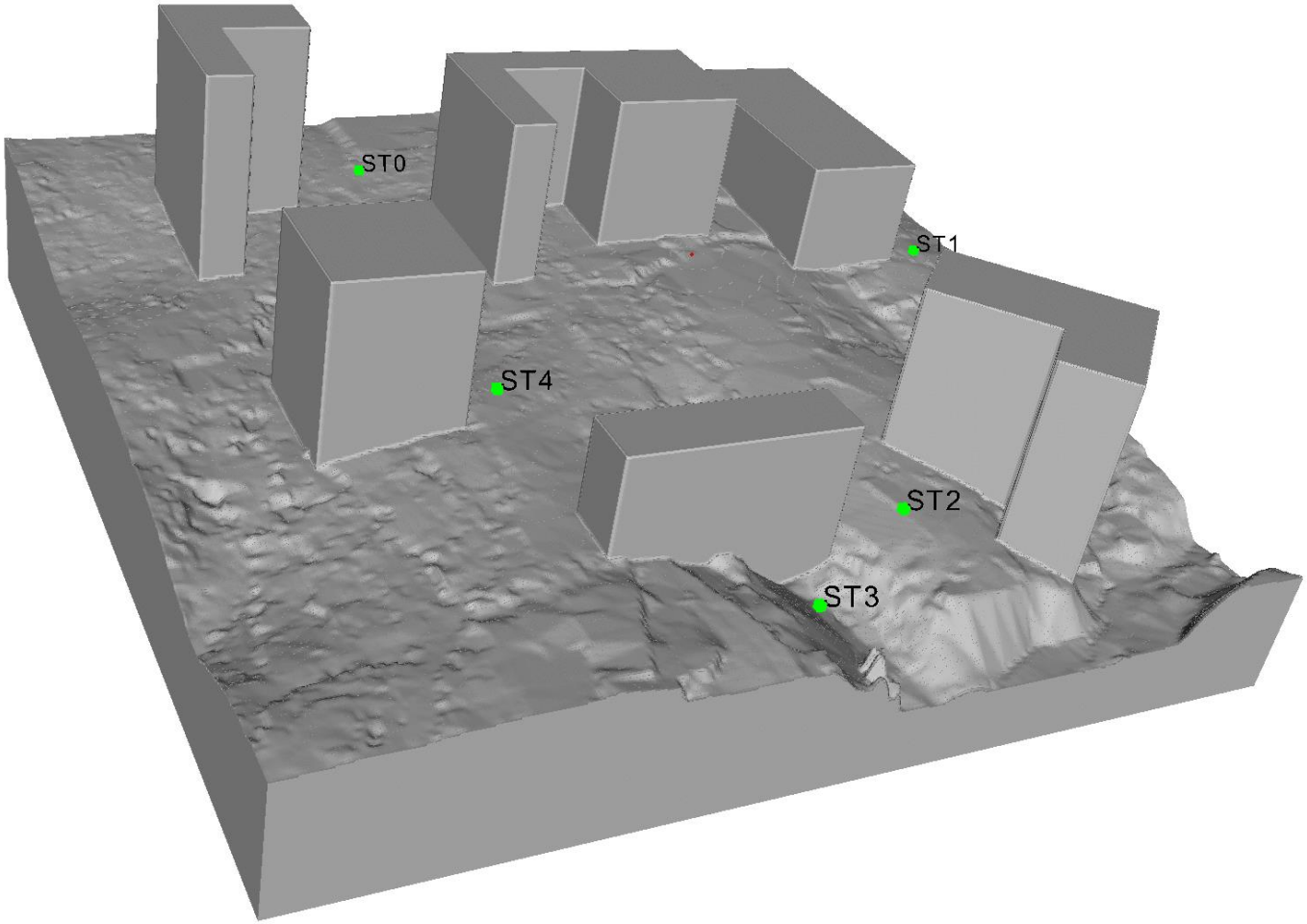
▶ Exemple d'une simulation 2D pour une source donnée





```
#################################################################

# Initialisation géométrique

# LA SOURCE
GeoSource = Sphere(R_source) + Vect(XC, YC, 11)
materiau["source"].obj(GeoSource, shell_discret=1)

# L'AIR
p1 = Parallelepipede(xmax-xmin, ymax-ymin, zmax-zmin) + Vect(0.5 * (xmin + xmax), 0.5 * (ymin + ymax), 0.5 * (zmin + zmax))
GeoAir = p1 - GeoSource
materiau["air"].obj(GeoAir, shell_discret=1)

# LA TOOPOGRAPHIE
GeoRelief = Topo3D(relief)
materiau["relief"].obj(GeoRelief, shell_discret=1)

# LE BATI

####### Batiment 1
x0 = 10.08 ; y0 = 10.223
x1 = 30.08 ; y1 = 10.223
x2 = 30.08 ; y2 = 15.223
x3 = 17.08 ; y3 = 15.223
x4 = 17.08 ; y4 = 25.223
x5 = 10.08 ; y5 = 25.223

Bat1    = [ (x0, y0), (x1, y1) , (x2, y2), (x3,y3), (x4, y4), (x5, y5) ]
Bat1    = Polygon( Bat1 )
vec_dir = Vect(0.0,0.0,30.0)
Bat1    = Extrude( Bat1 , vec_dir, 1)
materiau["bati1"].obj( Bat1, shell_discret=1)

####### Batiment 2
x0 = 50.08 ; y0 = 12.223
x1 = 65.08 ; y1 = 12.223
x2 = 65.08 ; y2 = 27.223
x3 = 50.08 ; y3 = 27.223

Bat2    = [ (x0, y0), (x1, y1) , (x2, y2), (x3,y3) ]
Bat2    = Polygon( Bat2 )
vec_dir = Vect(0.0,0.0,25.0)
Bat2    = Extrude( Bat2 , vec_dir, 1)
materiau["bati2"].obj( Bat2 )
```

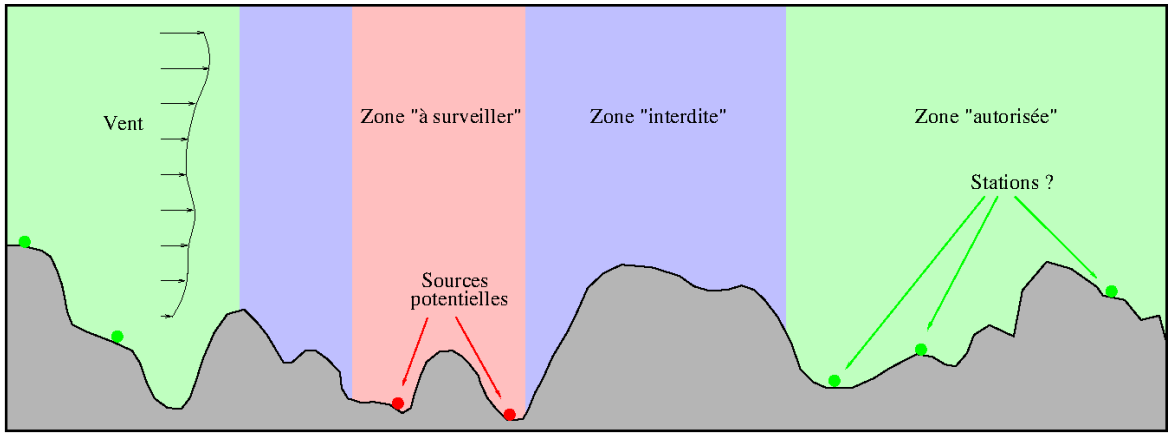
▶ Exemple de scène 3D avec topographie et bâti



Mise en place d'un réseau de détection d'ondes infrasonores

▶ Mise en place d'un réseau de détection d'ondes infrasonores (exercice fictif, idéalisé)

→ **Objectif: judicieusement positionner des capteurs pour détecter d'hypothétiques expérimentations au sein d'une zone à surveiller, avec prise en compte de la topographie et de la météo (vent)**



Pour cet exercice, seront connues :

- Les caractéristiques de la source (signal, positions possibles)
- La topographie
- Les zones « interdites » (territoire étranger, accès difficile, …), les zones « autorisées »

→ *Mise au point d'une boucle d'optimisation faisant appel au code de simulation, en faisant varier la position des capteurs et les cartes de vent*

- Première étape : cas d'exercice en 2D plan
- Deuxième étape : cas plus réaliste en 3D

Description du script Python *Infrason_relief_meteo_2D.py*

```python
#! /usr/bin/env python
# -*- coding:utf-8 -*-
##############################################################################
# Copyright ou (C) ou Copr. Commissariat a Energie Atomique et aux Energies   #
# Alternatives, 2020.                                                         #
#                                                                             #
# Ce logiciel est la propriete intellectuelle entiere du CEA.                 #
# Vous pouvez utiliser ce programme uniquement sur des machines du CEA a la   #
# condition que les responsables du programme vous en ont donne               #
# l'autorisation et l'acces.                                                  #
# Il est interdit de copier ce programme et ses sources sur des machines      #
# non administrees par le CEA.                                                #
#                                                                             #
# Le fait que vous puissiez acceder a cet en-tete signifie que vous en avez   #
# pris connaissance et que vous en acceptez les termes.                       #
##############################################################################

import math, os, sys, os.path

from utilitaires_hera import *
from hera import *
from validation_hera import *

script_path = os.path.dirname(os.path.realpath(__file__))
fic_CS = script_path + "/source_et_meteo.cs"
relief = script_path + "/topoIS_2D_ymax.dat"


def modele_utilisateur(gamma = 0, frequence = 0, xmin = 0, xmax = 0, XC = 0, YC = 0):

    dict={'gamma_parse':str(gamma),
          'nu_parse':str(frequence),
          'xmin_parse':str(xmin),
          'xmax_parse':str(xmax),
          'XC_parse':str(XC),
          'YC_parse':str(YC),}

    fd         = open(fic_CS, 'r')
    mon_modele = fd.readlines()
    fd.close()

    for (k,line) in enumerate(mon_modele):
        for nom,val in dict.iteritems():
            mon_modele[k] = mon_modele[k].replace(nom,val)

    txt = ""
    for line in mon_modele :
        txt += line
    UserAppendix("CSHARP.cs",txt)
```

Description du script Python *Infrason_relief_meteo_2D.py*

```
##########################################################################################
##########################################################################################
# LA DEFINITION DU CAS
##########################################################################################
##########################################################################################

def CAS_Infrason(nom_simu, pas, frequence, tps_f, XC, YC):

    Lx = 40e3 ; xmin = -10e3 ; xmax = xmin + Lx
    Ly = 11e3 ; ymin =   1e3 ; ymax = ymin + Ly

    raff = 3 ; nmax = 2

    [nx, xmin, xmax] = calculerNombreDeMailles2(pas, xmin, xmax, nmax, raff, modif="minmax")
    [ny, ymin, ymax] = calculerNombreDeMailles2(pas, ymin, ymax, nmax, raff, modif="minmax")

    fact = int(math.pow(raff, nmax))

    print "xmin = %e -> xmax = %e, nx = %d, dx = %e" % (xmin, xmax, nx, (xmax - xmin) / (nx * fact))
    print "ymin = %e -> ymax = %e, ny = %d, dy = %e" % (ymin, ymax, ny, (ymax - ymin) / (ny * fact))

##############################################################################
#                                                                            #
#                Définitions des materiaux                                   #
#                                                                            #
##############################################################################

    gamma = 1.4
    materiau["air"]    = Mat(gamma        = gamma,
                             corps_rigide = 0,
                             extensions   = "CSharpUserModel")

    materiau["relief"] = Mat(gamma                  = gamma,
                             corps_rigide           = 1,
                             ordre_phase_lagrange   = 1,
                             ordre_phase_projection = 1)

    ##################################################################

    c2     = 343. * 343.
    p0     = 1e5
    rho0   = gamma * p0 / c2
    materiau["air"]   .ini(rho=rho0,  pmat=p0, vitesse=(0., 0.))
    materiau["relief"].ini(rho=1.e10, pmat=p0, vitesse=(0., 0.))

    ##################################################################
```

Description du script Python *Infrason_relief_meteo_2D.py*

```
    ##################################################################

    # Initialisation géométrique

    # L'AIR
    GeoAir = [ (xmin, ymin), (xmax, ymin) , (xmax, ymax), (xmin, ymax) ]
    materiau["air"].frt(GeoAir)

    # LA TOOPOGRAPHIE
    f = open(relief);
    GeoRelief = [(xmax,ymin), (xmin, ymin)]
    for line in f:
        pair = line.split()
        if len(pair) == 3:
            x = float(pair[0])
            z = float(pair[2])
            GeoRelief.append( (x, z) )
    f.close()
    materiau["relief"].frt(GeoRelief)

##############################################################################
#                                                                            #
#                Définitions du schema numerique                             #
#                                                                            #
##############################################################################

    LAGRANGE  = 5; ADVECTION = 5

    if LAGRANGE == 1:
        lag_sch = "acoustique"
        lag_lim = "aucun"
    elif LAGRANGE==2 :
        lag_sch = "gad"
        lag_lim = "superbee"
    else:
        lag_sch = "gaia"
        lag_lim = "mp"

    if ADVECTION == 1:
        proj_sch = "upwind"
        proj_lim = "aucun"
    else:
        proj_sch = "reconstruction_polynomiale"
        proj_lim = "mp"
```

Description du script Python
Infrason_relief_meteo_2D.py

```
schema["Hydro"] = Hydro(

    frontiere_rigide = 1,                  # Traitement "Frontieres Immergees" pour le relief

    ordre = (LAGRANGE, ADVECTION),         # Ordre des schemas pour la phase lagrange et la phase de projectio


    lagrange_schema     = lag_sch,         # Schema hydro pour la phase lagrange
    lagrange_limiteur   = lag_lim,         # Limiteur pour la phase lagrange
    projection_schema   = proj_sch,        # Schema hydro pour la phase de projection
    projection_limiteur = proj_lim,        # Limiteur pour la phase de projection

    lagrange_mixte_ordre1            = 1,  # schema lagrange      a l ordre 1 sur les mailles mixtes
    pente_mixte_ordre1               = 1,  # schema de projection a l ordre 1 sur les mailles mixtes
    lagrange_transition_amr_ordre1   = 1,  # schema lagrange a l ordre 1 sur les transitions de niveau
    projection_transition_amr_ordre1 = 1,  # schema de projection a l ordre 1 sur les transitions de niveau

    lag_croix   = 0,
    cfl_godunov = 0.7,

    meth_press_vson_moy    = "pond_vol",
    repart_volume          = "volume",
    repart_energie         = "masse",
    ecin_moyen             = "vitesse_partielle",
    option_energie_interne = 1,

    calcul_dt_vson_moy      = 0,
    elim_vson_neg           = 1,
    rho_min_dilat_autre_mat = 1,

    raff_crit = "neant",
    cond_lim  = ("flux", "flux", "flux", "flux", ),
)

geom(dimension = 2, type = 0)
```

Description du script Python *Infrason_relief_meteo_2D.py*

```
##############################################################################
#                                                                            #
#                Définitions des stations                                    #
#                                                                            #
##############################################################################

    x0 = -7787.9992676;  y0 = 1826.66028213 + 2 * pas;
    x1 =  112.00073242;  y1 = 2946.86843264 + 2 * pas;
    x2 =  6162.0007324;  y2 = 3190.52332544 + 2 * pas;
    x3 =  10012.000732;  y3 = 2288.12351821 + 2 * pas;
    x4 =  12412.000732;  y4 = 4121.12426601 + 2 * pas;
    x5 =  13112.000732;  y5 = 4032.95509588 + 2 * pas;
    x6 =  16612.000732;  y6 = 3698.06999683 + 2 * pas;
    x7 =  17812.000732;  y7 = 4073.859311   + 2 * pas;

    station_coord = [ (x0, y0), (x1, y1), (x2, y2), (x3, y3), (x4, y4), (x5, y5), (x6, y6) , (x7, y7)  ]

    liste_station_coord = station_coord
    fic_station         = "Stations"+nom_simu+".dat"

    REP_CAS=os.path.join(os.getcwd(),"CAS_"+nom_simu)
    if (not os.path.exists(REP_CAS)) : os.makedirs(REP_CAS)

    station_coord_fic = os.path.join(REP_CAS,fic_station)

    # Ecriture des stations
    write_file(2,station_coord_fic,liste_station_coord)
    liste_station_scalaire = ("pmat",)

#    dico_rydep     = {0.0 : 5e-1,}
    dico_rydep     = {0.0 : 20,}
    dico_ryprot    = {0.0 : 5,}
    dico_rystation = {0.0 : 1.e-2,}

    sorties(critere_hdep = 1,
            critere_prot = 1,

            ryhdep = dico_rydep,
            ryprot = dico_ryprot,

            protection_nb_fichiers = 2,

            rystation          = dico_rystation,
            station_scalaire   = liste_station_scalaire,
            station_coord_fich = station_coord_fic,
           )
```

Description du script Python *Infrason_relief_meteo_2D.py*

```
cycle_temps(date_max = tps_f,
            dt_stop  = 1.e-14)

raffinement(fact_raff      = raff,
            niv_max        = nmax,
            extra          = 3,
            extra_frt      = {"air/relief": 5},
            propag_sev     = 3,
            seuil_raffine  = 0.1,
            seuil_regroupe = 0.05,
            freq_raffine   = 9999999,
            freq_regroupe  = 2,
            niv_min_frt    = {"air/relief": nmax},
           )

materiau["air"]   .raffinement(niv_min         = nmax,
                               niv_max         = nmax,
                               niv_min_initial = nmax,
                               extra           = 3)

materiau["relief"].raffinement(niv_min = 0,
                               niv_max = 0,
                               extra   = 3)

maillage(nombre_mailles = (nx, ny, ),
         min_coord      = (xmin, ymin, ),
         max_coord      = (xmax, ymax, ),
         recouvrement   = 3)

interface()

##############################################################################

modele_utilisateur(gamma, frequence, xmin, xmax, XC, YC)

simu(nom            = nom_simu,          # nom du cas
     liste_schemas  = ["Hydro"],         # solveur a utiliser
     liste_materiaux= ("air", "relief"), # liste des materiaux
     extensions      = "CSharpUserModel" # modèle utilisateur
     )


##############################################################################################
##############################################################################################
##############################################################################################
```

Description du script Python *Infrason_relief_meteo_2D.py*

```
##########################################################################################
##########################################################################################
##########################################################################################

TPS_F = 100      # en secondes
#PAS   = 10       # en metres
PAS   = 20       # en metres
FREQ  = 1.0      # en Hz
XC    = 19382.0; # Position x de la source
YC    = 4290.0;  # Position y de la source

radical = "InfraSonMeteo0_2D_" ; # radical += str(XC) + "_" + str(YC) + "_pas_"
NOM     = radical + "pas_" + str(PAS) + "m"
CAS_Infrason(NOM, PAS, FREQ, TPS_F, XC, YC)

# Lancement du cas
launch_run("--print_stats --hercule_prot -tp", cpu = ParallelParam(npart = 8, nomp = 4))
```

Description du modèle C#
source_et_meteo.cs

```
//*****************************************************************************
// Copyright ou (C) ou Copr. Commissariat à Energie Atomique et aux Energies   *
// Alternatives, 2020.                                                         *
//                                                                             *
// Ce logiciel est la propriété intellectuelle entière du CEA.                 *
// Vous pouvez utiliser ce programme uniquement sur des machines du CEA à la   *
// condition que les responsables du programme vous en ont donné               *
// l'autorisation et l'accès.                                                  *
// Il est interdit de copier ce programme et ses sources sur des machines      *
// non administrées par le CEA.                                                *
//                                                                             *
// Le fait que vous puissiez accéder à cet en-tête signifie que vous en avez   *
// pris connaissance et que vous en acceptez les termes.                       *
//*****************************************************************************
using System;
using System.IO;
using System.Collections.Generic;
using Real = System.Double;
using SystemMath = System.Math;
using Random = System.Random;

#if HERA
using HeraAPI;
#elif TROLL
using Arcane;
#endif

namespace HeraTrollAPI {

    public class MyModel : HeraTrollModel {
    public Real time;
    public Real gamma;
    public Real nu;
    public Real xmin;
    public Real xmax;
    public Real XC;
    public Real YC;
    public Real pi;

    public MyModel() {

        gamma = gamma_parse;
        nu    = nu_parse;
        xmin  = xmin_parse;
        xmax  = xmax_parse;
        XC    = XC_parse;
        YC    = YC_parse;

        pi    = SystemMath.PI;

    }
```

Description du modèle C#
source_et_meteo.cs

```
public override void CreateInternalVar() {

    int ordre     = 2;
    bool cl       = true;
    bool adv_pure = true;

    CreateVar("rho_ini" , OUTPUT | INTENSIVE | PROJECTABLE_VOLUME, ordre, cl, adv_pure);

}


public override void InitEOS(CellGroup group) {

    Var l_rho   = GetVar("Density");
    Var l_pmat  = GetVar("Pressure");
    Var l_eint  = GetVar("InternalEnergy");
    Var l_cmat2 = GetVar("SoundSpeed");
    Var l_rho0  = GetVar("rho_ini");
    Var c_x     = GetVar("c_coord[0]");
    Var c_y     = GetVar("c_coord[1]");

    Real Lx = xmax - xmin;
    Real fact;

    foreach(Cell cell in group) {

        Real x = c_x[cell];
        Real l = (x - xmin) / Lx;

        /* ******************************************************** */
        // Meteo
        if (l < 0.5) fact = 0.5 + 3.0 * l;
        else         fact = 3.5 - 3.0 * l;

        Real h = c_y[cell] - YC;
        Real a0 = (h*h*h - 2550.0 * h*h + 900000.0 * h + 343.0) / 1e8 * fact;
        /* ******************************************************** */

        if (a0 < -100.0) a0 = -100.0;
        if (a0 >  100.0) a0 =  100.0;

        Real c    = 343.0 + a0;
        Real c2   = c * c; // CHAMP IMPOSE DE VITESSE DU SON : PRISE EN COMPTE DU VENT

        l_rho[cell]   = gamma * l_pmat[cell] / c2;
        l_eint[cell]  = l_pmat[cell] / ((gamma-1.0) * l_rho[cell]);
        l_cmat2[cell] = c2;
        l_rho0[cell]  = l_rho[cell]; //  SAUVEGARDE DE LA DENSITE INITIALE

    }
}
```

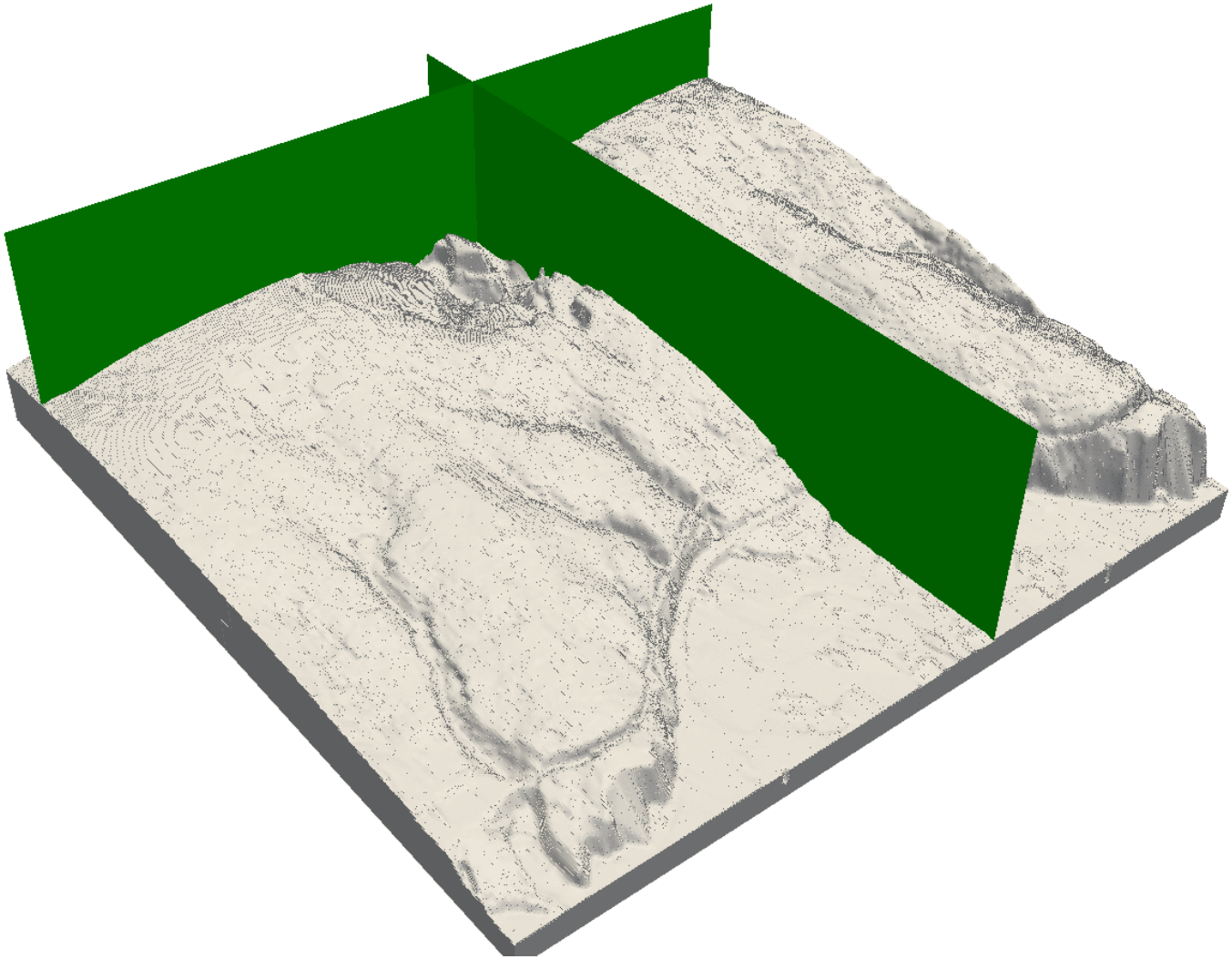
Description du modèle C#
source_et_meteo.cs

```
    public override void ApplyEOS(CellGroup group) {

        Var l_rho   = GetVar("Density");
        Var l_eint  = GetVar("InternalEnergy");
        Var l_pmat  = GetVar("Pressure");
        Var l_cmat2 = GetVar("SoundSpeed");
        Var l_rho0  = GetVar("rho_ini");

        Var c_x      = GetVar("c_coord[0]");
        Var c_y      = GetVar("c_coord[1]");

        foreach(Cell cell in group) {

            Real x = c_x[cell] - XC;
            Real y = c_y[cell] - YC;
            Real r = SystemMath.Sqrt( x*x + y*y );

            if (r < 100.0) {
                l_pmat[cell] = 1.0e5 * 0.01 * SystemMath.Sin( 2.0 * SystemMath.PI * nu * time ) + 1.0e5;
                l_rho[cell]  = l_rho0[cell] * SystemMath.Pow( l_pmat[cell] / 1.0e5, 1.0/gamma );
                if (l_rho0[cell] < 1.0e-4) {
                    l_rho[cell]  = 1.19 * SystemMath.Pow( l_pmat[cell] / 1.0e5, 1.0/gamma );
                }
            } else {
                l_pmat[cell] = (gamma -1.0) * l_rho[cell] * l_eint[cell];
            }

            l_cmat2[cell] =  gamma * l_pmat[cell] / l_rho[cell];

        }
    }

    public override void Compute(CellGroup group) {
        time = CurrentTime;
    }

    #if HERA
    public static void Main(string[] args) { }
    #endif
    }
}
```

- Exemple de simulations 2D pour trois cartes de vent données

- Exemple de simulations 2D pour trois cartes de vent données

▶ Exemple de simulation 3D



Principales étapes de l'étude

- Complément d'information sur le sujet
  - Modèles physiques et méthodes numériques associées (présentation du code d'hydrodynamique compressible)
  - Identification des paramètres du code et de leurs impacts sur sa durée d'exécution parallèle
  - Prise en main de ressources dans un centre de calcul distant
- Formalisation du problème et des grandeurs à optimiser
  - Choix d'une méthode d'optimisation adaptée au problème
- Conception et implantation d'une boucle d'optimisation appelant le code de calcul intensif //
  - En Python
  - La boucle d'optimisation pourra éventuellement être elle-même parallèle si son algorithme le permet
- Tests et mise au point du code complet d'optimisation sur des cas 2D
  - Sur des problèmes de taille « modeste » (cartes réduites, physique simplifiée, …)
- Passage à l'échelle en terme de physique ou de terrain couvert, en 2D
  - Expérimentations sur de plus gros problèmes et un plus grand nombre de cœurs de calcul
  - Analyse des performances et de la qualité de la solution trouvée
  - Optimisations
- Estimation de la taille maximale de problème traitable en fonction
  - Des ressources disponibles
  - Du temps imparti à l'étude
- Analyse de la faisabilité de l'extension au cas 3D dans le temps imparti pour le projet

→ Remise d'un rapport et soutenance orale à l'école CentraleSupelec

ARMEN

Partie 4

Connexion à Inti et mise en place de l'environnement

Connexion distante

Notes pour Windows

- Lancer un serveur X (Xming)
- Ouvrir putty et activer le Forwarding X11

Se connecter au serveur de Metz

chome.metz.supelec.fr → instructions S. Vialle / J. Cagnol

Se connecter au calculateur du CEA

```
$ ssh -X inti-amd.ocre.cea.fr -l nomlogin
$ mot de passe (12 caractères minimum)
```

Vos espaces de travail

```
$ cd ; pwd                    : votre "home" (espace disque très limité)
$ cd $CCCWORKDIR ; pwd        : votre "workdir" (bcp plus de place*)
$ cd $CCCSCRATCHDIR ; pwd     : votre "scratchdir" (bcp plus de place*)
```

→ **Les simulations sont à lancer sur un de ces deux derniers espaces**

* Attention, pas de sauvegardes, purgés régulièrement.

Récupérer les données pour votre projet

▶ Créer un répertoire dédié à votre projet dans le « work » directory

```
$ cd `ccc_home -W` ou cd $CCCWORKDIR
$ cp -r /ccc/work/cont003/ocre/jaouens03/CentraleSupelec/PROJET1 .
ou
$ cp -r /ccc/work/cont003/ocre/jaouens03/CentraleSupelec/PROJET2 .
```

▶ « Sourcer » la version 1.0.0 d'ARMEN

```
$ module purge
$ module load /ccc/work/cont003/armen/armen/opt/armen-1.1.0/share/armen/1.1.0
```

▶ Vérification

```
$ which hrun                ← commande de lancement d'ARMEN
> /ccc/work/cont003/armen/armen/opt/armen-1.0.0/bin/hrun
```

▶ Lancement d'un cas

```
$ cd PROJET1
$ python3 SouffleBati2D.py
ou
$ cd PROJET2
$ python3 Infrason_relief_meteo_2D.py
```

Quelques commandes utiles pour la gestion des jobs //

- Quelques commandes utiles
  - `$ ccc_mpinfo` : visualisation des partitions // disponibles sur Inti
  - `$ ccc_mpstat` : visualisation des jobs // lancés (-h : help)
  - `$ ccc_mpstat -u` : idem, mais uniquement les vôtres (-h : help)
  - `$ ccc_mdel -k #jobid` : arrêt du job #jobid

- Partition recommandée avec l'exécutable fourni (python3 nom_cas.py –host=nom_partition)

  1 – rome (par défaut)

- Editeurs disponibles sur Inti
  - `vi, gvim`
  - `Emacs`

- Il est permis de rapatrier les résultats de vos calculs (fichiers STATION_ST* par exemple), sur vos laptops. MAIS en aucun cas le code ARMEN !!

→ Pour toute question ou difficulté : stephane.jaouen@cea.fr

Questions ?