

## Raffinement statique de maillage cartésien dans Phénix

**Rencontres Arcane** | 17/04/2023

Maxime Stauffert

CEA/DAM

[maxime.stauffert@cea.fr](mailto:maxime.stauffert@cea.fr)

## Sommaire

1 Contexte

2 Mise en œuvre

3 Exemples d'utilisation

4 Conclusion





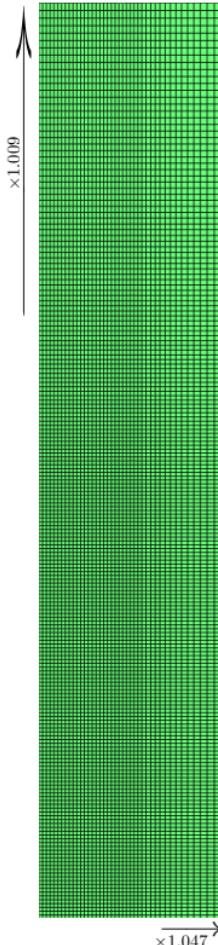
# 1. CONTEXTE



# Contexte dans Phénix

*Besoin d'un raffinement statique de maillage cartésien*

- Domaine de taille  $1\text{cm} \times 5\text{cm}$
- 8080 mailles en tout
- 2880 mailles de  $0.021\text{cm} \times 0.021\text{cm}$  en bas à gauche
- Rapport d'aspect 2
- Maillage conforme

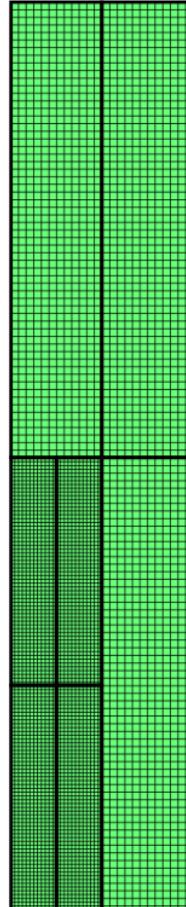




# Contexte dans Phénix

*Besoin d'un raffinement statique de maillage cartésien*

- Domaine de taille  $1\text{cm} \times 5\text{cm}$
- 8080 5040 mailles en tout
- 2880 mailles de  $0.021\text{cm} \times 0.021\text{cm}$  en bas à gauche
- Rapport d'aspect  $\geq 1$
- Maillage non-conforme

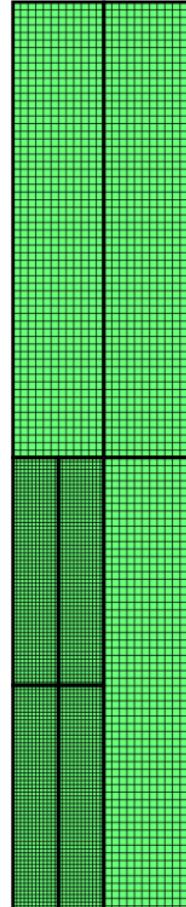


# Contexte dans Phénix

*Besoin d'un raffinement statique de maillage cartésien*



- Maillage fin dans une zone turbulente
- Limitation du nombre de mailles (notamment pour le 3D)
- Bons rapports d'aspect pour la robustesse du schéma
- Pas de nécessité de raffinement adaptatif





# Contexte dans Phénix

Schéma Lagrange-projection existant en conforme

- Une phase Lagrange pure

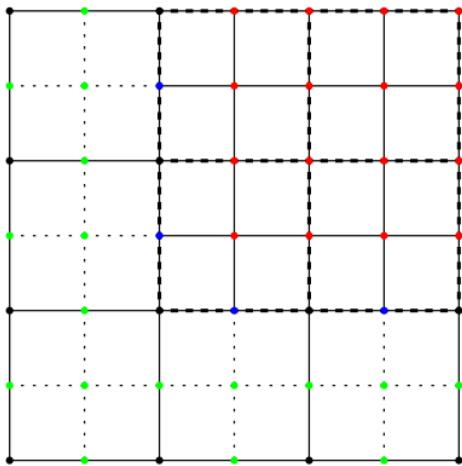
- ▶ On calcule des masses / forces à l'aide de quantités globalisées aux mailles
  - ▶ On en déduit des vitesses aux nœuds

→ En non-conforme : on veut imposer une unique vitesse aux nœuds à l'interface des patchs

- Une phase de projection directionnelle

- ▶ On calcule des volumes déplacés par direction à chaque face
  - ▶ On en déduit des quantités par matériau sur le maillage cartésien

→ En non-conforme : on veut appliquer les boucles inchangées à des groupes bien choisis





# Contexte dans Arcane

## Outils complémentaires

- Maillage cartésien
  - ▶ Permet de naviguer entre les entités dans chaque direction
  - ▶ Existe seulement pour le maillage conforme de départ
- AMR tree-based
  - ▶ Possibilité de raffiner chaque maille en créant 4 enfants
  - ▶ Pas de notion de raffinement sur un patch avec un niveau commun à toutes les mailles
  - ▶ Pas de maillage cartésien associé à un patch pour la projection
- Drapeaux
  - ▶ Permet de repérer de manière efficace certaines mailles
  - ▶ Aide à la création des groupes pour les algorithmes et les mises à jour entre patchs de différents niveaux
  - ▶ Nombre limité de drapeaux dans Arcane → mécanique copiée dans Phénix



# 2. MISE EN ŒUVRE



# Mise en œuvre dans Arcane

## Raffinement par patch

- API de raffinement par patch :

```
cartesian_mesh->refinePatch2D(origin, length);
cartesian_mesh->refinePatch3D(origin, length);
```

- Maillages cartésiens pour chaque patch :

```
for (auto patch_i = Integer{0}; patch_i < cartesian_mesh->nbPatch(); ++patch_i) {
    auto cell_dmx = cartesian_mesh->patch(patch_i)->cellDirection(MD_DirX);
    ENUMERATE_CELL (cell_i, cell_group) {
        auto cell_next = cell_dmx[*cell_i].next();
        // algorithm not modified
    }
}
```



# Mise en œuvre dans Phénix

## *Initialisation non-conforme*

- Initialisation conforme inchangée lors de la lecture des données
- Raffinement selon les patchs dans le jeu de données
- Localisation des valeurs dans les patchs raffinés
- Mise à zéro / NaN des valeurs dans les mailles inactives
- Fin de l'initialisation en non-conforme



# Mise en œuvre dans Phénix

*Création de patches avec mailles de recouvrement*

- L'utilisateur renseigne une zone de raffinement

```
<eul-mesh-refinement>
  <patch>
    <origin>0.2 0.02 0.0</origin>
    <length>0.4 0.04 0.0</length>
  </patch>
</eul-mesh-refinement>
```

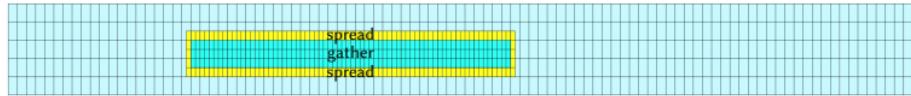
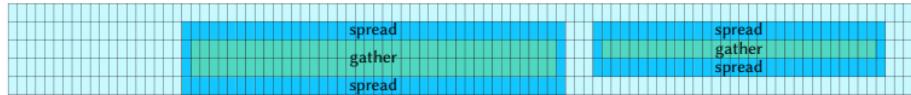
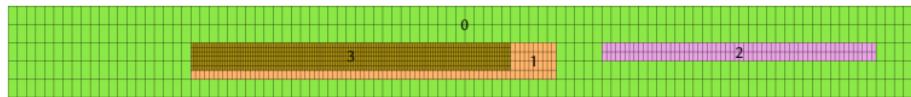
- On augmente la zone de raffinement que l'on va demander à Arcane  
→ Recouvrement comme en parallèle pour ne pas modifier les algorithmes conformes
- On interdit l'intersection des zones de raffinement pour la cohérence des niveaux de mailles



# Mise en œuvre dans Phénix

## *Création des groupes*

- Boucle de remontée puis de descente sur les patchs pour affecter des drapeaux aux mailles
  - Les mailles peuvent êtres actives, de recouvrement ou inactives
  - Les mailles enfants de recouvrement doivent êtres actives ou elles-mêmes de recouvrement
- Création des groupes de mailles à l'aide des drapeaux
  - Le groupe des mailles actives pave le domaine
  - Le groupe des mailles de recouvrement permet de mettre à jour les valeurs entre les patchs
  - Le groupe des mailles de calcul contient les mailles actives et leurs voisines





## Mise en œuvre dans Phénix

*Moyennes / localisations des quantités entre les patchs*

- Moyenne des valeurs des mailles enfants pour remonter l'information :

$$V_p = \sum_{i=1}^{n_d} V_i, \quad \rho_p = \frac{1}{V_p} \sum_{i=1}^{n_d} \rho_i V_i, \quad n_d = \begin{cases} 4 & \text{en 2D,} \\ 8 & \text{en 3D.} \end{cases}$$

- Localisation de la valeur de la maille parent pour descendre l'information :

$$\forall i \in \{1, \dots, n_d\}, \quad V_i = \frac{V_p}{n_d}, \quad \rho_i = \rho_p.$$



# Mise en œuvre dans Phénix

## *Modifications minimales des algorithmes*

- Boucles sur les groupes d'entités :

```
const auto& node_group =
    getGroupMngStrategy()->meshGroup(Group::Active, ItemKind::Node, IfConformReturn::AllNodes);
ENUMERATE_NODE (node_i, node_group) {
    // algorithm not modified
}
```

- Boucles sur les patchs :

```
ENUMERATE_PATCH (patch_i, cartesian_mesh) {
    auto face_dm = patch_i->faceDirection(dir);
    const auto& face_group =
        getGroupMngStrategy()->patchFaces(*patch_i, Group::InnerCompute, dir, IfConformReturn::InnerFaces);
    ENUMERATE_FACE (face_i, face_group) {
        // algorithm not modified
    }
}
```

- Boucles sur les environnements :

```
ENUMERATE_ENV (env_i, mesh_material_mng) {
    const auto& envcells_view =
        *getGroupMngStrategy()->ptrEnvCellVectorView(*env_i, Group::OwnActive, IfConformReturn::OwnEnvCellsView);
    ENUMERATE_ENVCELL (envcell_i, envcells_view) {
        // algorithm not modified
    }
}
```



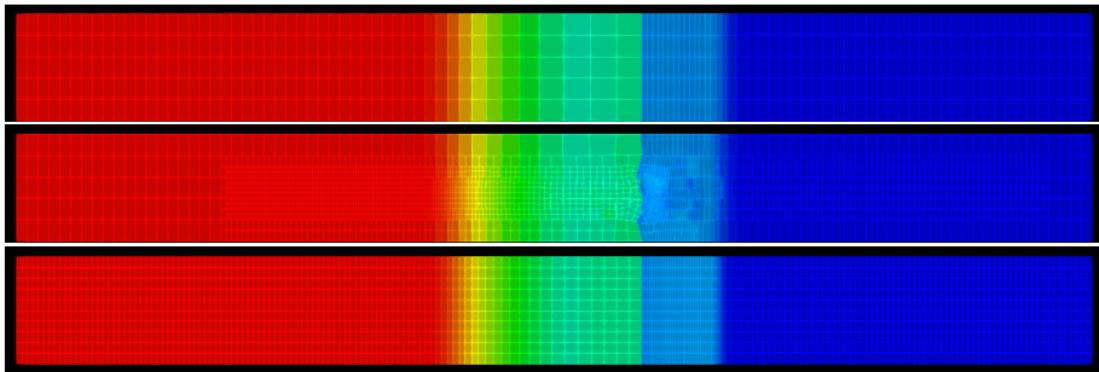
# 3.

## EXEMPLES D'UTILISATION

# Exemples d'utilisation

Tube à choc en Lagrange pur

100 × 5 mailles



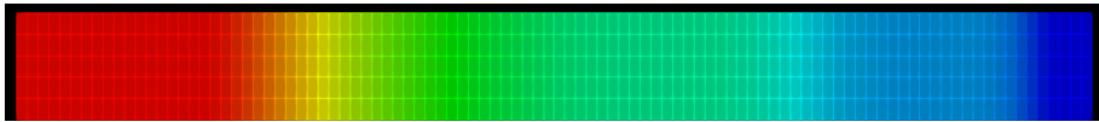
$\rho$  à  $t = 0.0876 \text{ s}$



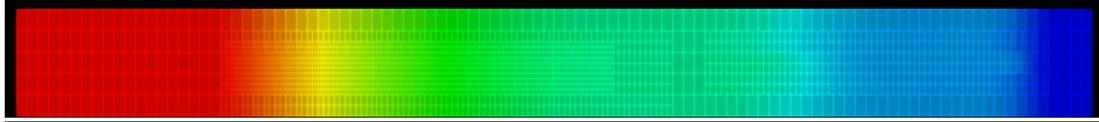
# Exemples d'utilisation

Tube à choc Lagrange-projection

100 × 5 mailles



1460 mailles



200 × 10 mailles



$\rho$  à  $t = 0.25$  s



# 4. CONCLUSION

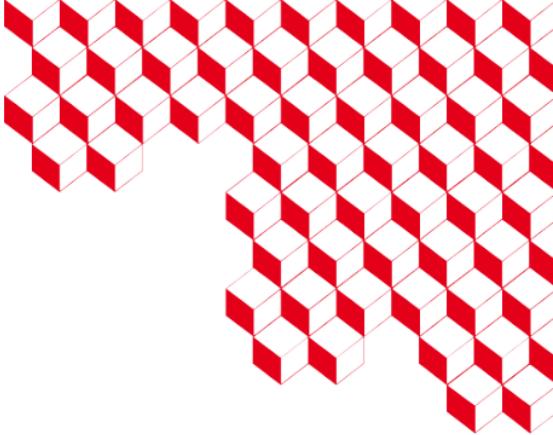


- Réalisations :

- ▶ Possibilité de raffiner de manière statique avec un schéma de type Lagrange-projection
- ▶ Utilisation de fonctionnalités Arcane
- ▶ Impact minimal dans les algorithmes de Phénix

- Perspectives :

- ▶ Déboggage du 3D (en cours : N. Peton)
- ▶ Sorties non-conforme en structuré (en cours : M. Stauffert)
- ▶ Déboggage du parallèle (à venir : N. Peton)
- ▶ Passage à l'ordre 2 en espace (à venir : M. Stauffert)
- ▶ Développements côté Arcane et Phénix pour le déraffinement (G. Gospellier et M. Stauffert)
- ▶ Optimisation du non-conforme



**Merci de votre attention !**

Maxime Stauffert  
**CEA DAM Île-de-France**  
Bruyères-le-Châtel  
91297 Arpajon cedex  
[maxime.stauffert@cea.fr](mailto:maxime.stauffert@cea.fr)