Intégration du solveur Devito en tant que backend différences finies dans un solveur fluide

Contexte scientifique

Ce stage propose de travailler sur l'intégration d'un différences finies dans une bibliothèque résolvant des équations aux dérivées partielles par lutilisation de méthodes numériques appelées méthodes Vortex. Parmi les nombreuses méthodes numériques utilisées pour la simulation découlements incompressibles, les méthodes basées sur une approche particulaire (ou lagrangienne) occupent une place importante de part leur description intuitive et naturelle de lécoulement ainsi que par leur faible diffusion numérique et leur stabilité. Elles consistent à discrétiser les quantités physiques du problème sur des particules numériques qui trans- portent ces quantités dans le domaine physique en fonction de la dynamique de lécoulement.

Plus particulièrement, les méthodes Vortex dites semi-lagrangiennes ou avec remaillage ou particle-mesh, ont été exploitées avec succès pour la simulation de nombreux problèmes réels en mécanique des fluides, aux retombées industrielles. Parmi les domaines dapplications concernés on peut citer le contrôle aérodynamique découlement, la simulation découlements en milieux poreux, linteraction fluide-structure , la simulation découlements diphasiques, le transport de scalaire passif, loptimisation et lapprentissage par renforcement. Dans le cadre de ce stage, les simulations numériques seront donc basées sur une approche Vortex semi-lagrangienne et seront effectuées à partir dun code de calcul de recherche existant, nommé HySoP (Hybrid Simulations using Particles), dédié à la résolution des équations de Navier-Stokes en 2D et 3D à laide dune méthode Vortex avec remaillage ("particle-mesh"). Elle permet des simulations parallèles sur processeurs classiques (multi-CPU) ou sur architecture hybride CPU-GPU (exemple en Figure (a)).

Objectifs du stage

Devito est un paquet python qui permet de générer du code différences finies à partir d'expressions symboliques. Elle utilise numpy, sympy et numba avec support MPI pour les aspects parallèles.

L'intérêt premier d'intégrer devito dans HySoP est de proposer une implémentation sur le backend Python dans les cas où aucune plateforme OpenCL n'est installée. Cela permettrait également de comparer les performances HySoP/OpenCL à Devito/Numba/CUDA (numba propose un backend cuda en plus du backend C++ de base).

La librairie est écrite en python3, les filtres $F(\omega)$ utilisent numpy ou numba. Le support massivement parallèle est réalisé par la bibliothèque mpi4-fftw.

L'objectif de stage est d'explorer la possibilité d'ajouter un support CPU différences finies dans HySoP.

- Méthodes numériques : Calcul symbolique, différences finies.
- Bibliothèques utilisées : python3, numpy, numba, mpi4py, sympy

De nombreux exemples documentés sont présents sur le dépôt github : https ://github.com/devitocodes/devito

Plan de stage

1. Manipuler la bibliothèque Devito, implémenter des différences finies explicites FDC2 et FDC4 (sur un opérateur de diffusion par exemple).
2. Vérifier à l'aide de solutions analytiques que les solutions calculées et analytiques sont similaires (Méthode des solutions Manufacturées)
3. Composer des cas tests reproductibles.
4. Renouveler la démarche en utilisant les capacités parallèle de la bibliothèque Devito
5. Comprendre le fonctionnement de Devito :
   - A quel moment est allouée la mémoire pour les grilles ? Est-il possible de passer une grille pré allouée ?
   - Comment est gérée l'heuristique de découpe des grilles en MPI.
   - Peut-on fournir une topologie MPI mise en place en dehors de Devito ?
6. Essayer d'interagir avec Devito en allouant les grilles et un communicateur MPI indépendamment de Devito.
7. Analyser les performances de Devito en local et sur cluster.
8. ăÉcrire un convertisseur d'expressions symboliques HySoP vers expressions symboliques Devito (ie. recréer un arbre d'expressions devito à partir d'un arbre hysop).
9. Tenter une implémentation dans HySoP.

Compétences recherchées

Les candidatůeůs devront être issuůeůs dune formation mathématiques appliquées et/ou informatique. Des connaissances en mécanique des fluides (pour les applications) sont un plus. On attendra des candidatůeůs des compétences en programmation, notamment en calcul parallèle, ainsi quune connaissance des méthodes et schémas numériques pour la mécanique des fluides. Une expérience dans lutilisation de logiciels de gestion de versions (git, github, ...) est nécessaire.

Un bon niveau danglais (écrit et oral) sera requis.

Informations pratiques et contacts

Lieu : laboratoire LJK, Grenoble

Encadrement : Christophe Picard (Maître de Conférences)

Financement : Gratification de stage (environ 600/mois)

Durée : 5 mois à compter de mi-février

Contact : christophe.picard@univ-grenoble-alpes.fr

Candidature : CV et lettre de motivation au format PDF envoyés par courriel.