Simulation du Jet D'Eau de A à Y: Algorithme numérique pour résoudre les équations de Navier-Stokes incompressibles

Pablo Strasser

University of Geneva

22 février 2013

Résumé

Résumé

Résumé

Résumé

Résumé

Résumé

Résumé

Opérateur différentiel

Gradient :

$$\boldsymbol{\nabla} p = \sum_i \partial_i p$$

Divergence :

$$\boldsymbol{\nabla} \cdot \boldsymbol{v} = \sum_i \partial_i \boldsymbol{v}_i$$

Rotationnel

$$\boldsymbol{\nabla} \times \boldsymbol{v} = \begin{pmatrix} \partial_2 \boldsymbol{v}_3 - \partial_3 \boldsymbol{v}_2 \\ \partial_3 \boldsymbol{v}_1 - \partial_1 \boldsymbol{v}_3 \\ \partial_1 \boldsymbol{v}_2 - \partial_2 \boldsymbol{v}_1 \end{pmatrix}$$

Convection

$$(\boldsymbol{v} \cdot \boldsymbol{\nabla})\, \boldsymbol{v} = \sum_i \boldsymbol{v}_i \partial_i \boldsymbol{v}$$

Propriété des opérateurs différentiels

Propriété (Divergence d'un rotationnel)

$$\nabla \cdot (\nabla \times \boldsymbol{v}) = 0$$

Propriété (Rotationnel d'un gradient)

$$\nabla \times (\nabla p) = 0$$

Propriété des opérateurs différentiels

Propriété (Divergence d'un rotationnel)

$$\nabla \cdot (\nabla \times \boldsymbol{v}) = 0$$

Propriété (Rotationnel d'un gradient)

$$\nabla \times (\nabla \boldsymbol{p}) = 0$$

Projection

Propriété

Pour chaque vecteur $\boldsymbol{v}$ *on peut* projeter *dans un espace à* divergence *nulle sans changer le* rotationnel *en résolvant :*

$$\boldsymbol{v}_{new} = \boldsymbol{v} - \boldsymbol{\nabla} p$$

$$\Delta p = \boldsymbol{\nabla} \cdot \boldsymbol{v}$$

Preuve

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}_{new} = \boldsymbol{\nabla} \cdot \boldsymbol{v} - \Delta p = \boldsymbol{\nabla} \cdot \boldsymbol{v} - \boldsymbol{\nabla} \cdot \boldsymbol{v} = 0$$

$$\boldsymbol{\nabla} \times \boldsymbol{v}_{new} = \boldsymbol{\nabla} \times \boldsymbol{v} - \boldsymbol{\nabla} \times \boldsymbol{\nabla} p = \boldsymbol{\nabla} \times \boldsymbol{v}$$

Définition (Projecteur)

$$P\boldsymbol{v} = \boldsymbol{\nabla} \Delta^{-1} \boldsymbol{\nabla} \cdot \boldsymbol{v}$$

Projection

Propriété

Pour chaque vecteur $\boldsymbol{v}$ *on peut* projeter *dans un espace à* divergence *nulle sans changer le* rotationnel *en résolvant :*

$$\boldsymbol{v}_{new} = \boldsymbol{v} - \boldsymbol{\nabla} p$$

$$\Delta p = \boldsymbol{\nabla} \cdot \boldsymbol{v}$$

Preuve

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}_{new} = \boldsymbol{\nabla} \cdot \boldsymbol{v} - \Delta p = \boldsymbol{\nabla} \cdot \boldsymbol{v} - \boldsymbol{\nabla} \cdot \boldsymbol{v} = 0$$

$$\boldsymbol{\nabla} \times \boldsymbol{v}_{new} = \boldsymbol{\nabla} \times \boldsymbol{v} - \boldsymbol{\nabla} \times \boldsymbol{\nabla} p = \boldsymbol{\nabla} \times \boldsymbol{v}$$

Définition (Projecteur)

$$P\boldsymbol{v} = \boldsymbol{\nabla} \Delta^{-1} \boldsymbol{\nabla} \cdot \boldsymbol{v}$$

Projection

Propriété

Pour chaque vecteur $\boldsymbol{v}$ *on peut* projeter *dans un espace à* divergence *nulle sans changer le* rotationnel *en résolvant :*

$$\boldsymbol{v}_{new} = \boldsymbol{v} - \boldsymbol{\nabla} p$$

$$\Delta p = \boldsymbol{\nabla} \cdot \boldsymbol{v}$$

Preuve

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}_{new} = \boldsymbol{\nabla} \cdot \boldsymbol{v} - \Delta p = \boldsymbol{\nabla} \cdot \boldsymbol{v} - \boldsymbol{\nabla} \cdot \boldsymbol{v} = 0$$

$$\boldsymbol{\nabla} \times \boldsymbol{v}_{new} = \boldsymbol{\nabla} \times \boldsymbol{v} - \boldsymbol{\nabla} \times \boldsymbol{\nabla} p = \boldsymbol{\nabla} \times \boldsymbol{v}$$

Définition (Projecteur)

$$P\boldsymbol{v} = \boldsymbol{\nabla} \Delta^{-1} \boldsymbol{\nabla} \cdot \boldsymbol{v}$$

Reformulation avec projection

Navier-Stokes

$$\nabla \cdot \boldsymbol{v}(\boldsymbol{x}, t) = 0$$

$$\partial_t \boldsymbol{v}(\boldsymbol{x}, t) = \underbrace{f(\boldsymbol{v}(\boldsymbol{x}, t)) - \nabla p}_{\text{Accélération}}$$

Où

$$f(\boldsymbol{v}(\boldsymbol{x}, t)) = -(\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + \frac{\boldsymbol{F}}{\rho} + \nu \Delta \boldsymbol{v}$$

Reformulation

$$\partial_t (\nabla \cdot \boldsymbol{v}) = 0 = \nabla \cdot f(\boldsymbol{v}) - \Delta p$$

$$\Delta p = \nabla \cdot f(\boldsymbol{v})$$

$$\partial_t \boldsymbol{v} = (1 - P) f(\boldsymbol{v})$$

Reformulation avec projection

Navier-Stokes

$$\begin{aligned} \boldsymbol{\nabla} \cdot \boldsymbol{v}(\boldsymbol{x}, t) &= 0 \\ \partial_t \boldsymbol{v}(\boldsymbol{x}, t) &= \underbrace{f(\boldsymbol{v}(\boldsymbol{x}, t)) - \boldsymbol{\nabla} p}_{\text{Accélération}} \end{aligned}$$

Où

$$f(\boldsymbol{v}(\boldsymbol{x}, t)) = -(\boldsymbol{v} \cdot \boldsymbol{\nabla})\boldsymbol{v} + \frac{\boldsymbol{F}}{\rho} + \nu \Delta \boldsymbol{v}$$

Reformulation

$$\begin{aligned} \partial_t (\boldsymbol{\nabla} \cdot \boldsymbol{v}) = 0 &= \boldsymbol{\nabla} \cdot f(\boldsymbol{v}) - \Delta p \\ \Delta p &= \boldsymbol{\nabla} \cdot f(\boldsymbol{v}) \\ \partial_t \boldsymbol{v} &= (1 - P) f(\boldsymbol{v}) \end{aligned}$$

Définition lagrangienne

Caracteristique

$$\partial_t \boldsymbol{\xi}_\lambda(t) = \boldsymbol{v}(\boldsymbol{\xi}_\lambda(t), t)$$
$$\boldsymbol{\xi}_\lambda(t_0) = \boldsymbol{\xi}_\lambda^0$$

Vitesse lagrangienne

$$\boldsymbol{u}_\lambda(t) = \boldsymbol{v}(\boldsymbol{\xi}_\lambda(t), t)$$

Dérivée matérielle

$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = \frac{d\boldsymbol{v}(\xi_\lambda, t)}{dt} = \partial_t \boldsymbol{v} + \left(\frac{\partial \boldsymbol{\xi}_\lambda(t)}{\partial t} \cdot \nabla\right) \boldsymbol{v}$$
$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = \partial_t \boldsymbol{v} + (\boldsymbol{v} \cdot \nabla)\, \boldsymbol{v}$$

Définition lagrangienne

Caracteristique

$$\partial_t \boldsymbol{\xi}_\lambda(t) = \boldsymbol{v}(\boldsymbol{\xi}_\lambda(t), t)$$
$$\boldsymbol{\xi}_\lambda(t_0) = \boldsymbol{\xi}_\lambda^0$$

Vitesse lagrangienne

$$\boldsymbol{u}_\lambda(t) = \boldsymbol{v}(\boldsymbol{\xi}_\lambda(t), t)$$

Dérivée matérielle

$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = \frac{d\boldsymbol{v}(\xi_\lambda, t)}{dt} = \partial_t \boldsymbol{v} + \left(\frac{\partial \boldsymbol{\xi}_\lambda(t)}{\partial t} \cdot \nabla\right) \boldsymbol{v}$$
$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = \partial_t \boldsymbol{v} + (\boldsymbol{v} \cdot \nabla)\, \boldsymbol{v}$$

Définition lagrangienne

Caracteristique

$$\partial_t \boldsymbol{\xi}_\lambda(t) = \boldsymbol{v}(\boldsymbol{\xi}_\lambda(t), t)$$
$$\boldsymbol{\xi}_\lambda(t_0) = \boldsymbol{\xi}_\lambda^0$$

Vitesse lagrangienne

$$\boldsymbol{u}_\lambda(t) = \boldsymbol{v}(\boldsymbol{\xi}_\lambda(t), t)$$

Dérivée matérielle

$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = \frac{d\boldsymbol{v}(\xi_\lambda, t)}{dt} = \partial_t \boldsymbol{v} + \left(\frac{\partial \boldsymbol{\xi}_\lambda(t)}{\partial t} \cdot \boldsymbol{\nabla}\right) \boldsymbol{v}$$
$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = \partial_t \boldsymbol{v} + (\boldsymbol{v} \cdot \boldsymbol{\nabla})\, \boldsymbol{v}$$

Navier-Stokes lagrangien

Navier-Stokes lagrangien

$$\boldsymbol{\nabla} \cdot \boldsymbol{u}_\lambda(t) = 0$$

$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = -\boldsymbol{\nabla} p(\boldsymbol{\xi}_\lambda(t), t) + \frac{\boldsymbol{F}(\boldsymbol{\xi}_\lambda(t), t)}{\rho(\boldsymbol{x}, t)} + \nu\Delta\boldsymbol{u}_\lambda(t)$$

Différentes versions des équations de Navier-Stokes

Eulerien

$$\begin{aligned} \boldsymbol{\nabla} \cdot \boldsymbol{v}(\boldsymbol{x}, t) &= 0 \\ \partial_t \boldsymbol{v}(\boldsymbol{x}, t) &= f(\boldsymbol{v}(\boldsymbol{x}, t)) - \boldsymbol{\nabla} p \end{aligned}$$

Projection

$$\partial_t \boldsymbol{v} = (1 - P) f(\boldsymbol{v})$$

Lagrangien

$$\begin{aligned} \boldsymbol{\nabla} \cdot \boldsymbol{u}_\lambda(t) &= 0 \\ \frac{d\boldsymbol{u}_\lambda(t)}{dt} &= -\boldsymbol{\nabla} p(\boldsymbol{\xi}_\lambda(t), t) + \frac{\boldsymbol{F}(\boldsymbol{\xi}_\lambda(t), t)}{\rho(\boldsymbol{x}, t)} + \nu \Delta \boldsymbol{u}_\lambda(t) \end{aligned}$$

Différentes versions des équations de Navier-Stokes

Eulerien

$$\nabla \cdot \boldsymbol{v}(\boldsymbol{x}, t) = 0$$
$$\partial_t \boldsymbol{v}(\boldsymbol{x}, t) = f(\boldsymbol{v}(\boldsymbol{x}, t)) - \nabla p$$

Projection

$$\partial_t \boldsymbol{v} = (1 - P) f(\boldsymbol{v})$$

Lagrangien

$$\nabla \cdot \boldsymbol{u}_\lambda(t) = 0$$
$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = -\nabla p(\boldsymbol{\xi}_\lambda(t), t) + \frac{\boldsymbol{F}(\boldsymbol{\xi}_\lambda(t), t)}{\rho(\boldsymbol{x}, t)} + \nu \Delta \boldsymbol{u}_\lambda(t)$$

Différentes versions des équations de Navier-Stokes

Eulerien

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}(\boldsymbol{x}, t) = 0$$
$$\partial_t \boldsymbol{v}(\boldsymbol{x}, t) = f(\boldsymbol{v}(\boldsymbol{x}, t)) - \boldsymbol{\nabla} p$$

Projection

$$\partial_t \boldsymbol{v} = (1 - P) f(\boldsymbol{v})$$

Lagrangien

$$\boldsymbol{\nabla} \cdot \boldsymbol{u}_\lambda(t) = 0$$
$$\frac{d\boldsymbol{u}_\lambda(t)}{dt} = -\boldsymbol{\nabla} p(\boldsymbol{\xi}_\lambda(t), t) + \frac{\boldsymbol{F}(\boldsymbol{\xi}_\lambda(t), t)}{\rho(\boldsymbol{x}, t)} + \nu \Delta \boldsymbol{u}_\lambda(t)$$

Idée

- Vitesse et pression sont discrétisées sur une grille.
- Les équations de Navier-Stokes sont résolues sur la grille.
- Mais la topologie est determinée par la position des particules.
- Les particules se déplacent par la vitesse donnée par interpolation de la grille.
- Les conditions aux bords sont imposées en créant des points fantômes en vitesse et pression

Idée

- Vitesse et pression sont discrétisées sur une grille.
- Les équations de Navier-Stokes sont résolues sur la grille.
- Mais la topologie est determinée par la position des particules.
- Les particules se déplacent par la vitesse donnée par interpolation de la grille.
- Les conditions aux bords sont imposées en créant des points fantômes en vitesse et pression

Idée

- Vitesse et pression sont discrétisées sur une grille.
- Les équations de Navier-Stokes sont résolues sur la grille.
- Mais la topologie est determinée par la position des particules.
- Les particules se déplacent par la vitesse donnée par interpolation de la grille.
- Les conditions aux bords sont imposées en créant des points fantômes en vitesse et pression

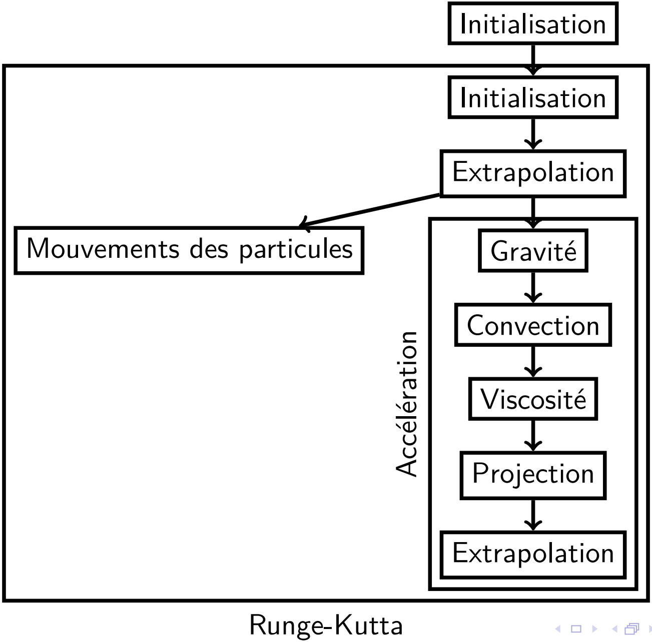
Idée

- Vitesse et pression sont discrétisées sur une grille.
- Les équations de Navier-Stokes sont résolues sur la grille.
- Mais la topologie est determinée par la position des particules.
- Les particules se déplacent par la vitesse donnée par interpolation de la grille.
- Les conditions aux bords sont imposées en créant des points fantômes en vitesse et pression

Idée

- Vitesse et pression sont discrétisées sur une grille.
- Les équations de Navier-Stokes sont résolues sur la grille.
- Mais la topologie est determinée par la position des particules.
- Les particules se déplacent par la vitesse donnée par interpolation de la grille.
- Les conditions aux bords sont imposées en créant des points fantômes en vitesse et pression

Schéma de l'algorithme



Projection

Projection en accélération

$$\partial_t \boldsymbol{v} = (1 - P) f(\boldsymbol{v})$$

Projection en vitesse

$$\begin{aligned} \partial_t \tilde{\boldsymbol{v}} &= f(\boldsymbol{v}) \\ \boldsymbol{v} &= (1 - P)\tilde{\boldsymbol{v}} \end{aligned}$$

Projection

Projection en accélération

$$\partial_t \boldsymbol{v} = (1 - P) f(\boldsymbol{v})$$

Projection en vitesse

$$\begin{aligned} \partial_t \tilde{\boldsymbol{v}} &= f(\boldsymbol{v}) \\ \boldsymbol{v} &= (1 - P)\tilde{\boldsymbol{v}} \end{aligned}$$

Intégration

Runge-Kutta sur l'accélération

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \sum_{i=1}^{s} b_i \boldsymbol{k}_i$$

$$\boldsymbol{k}_i = \Delta t(1 - P) f\left(\boldsymbol{v}_n + \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j\right)$$

Runge-Kutta sur la vitesse

$$\tilde{\boldsymbol{v}}_{n+1} = (1 - P)\left(\tilde{\boldsymbol{v}}_n + \sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i\right)$$

$$\tilde{\boldsymbol{k}}_i = \Delta t f\left((1 - P)\left(\tilde{\boldsymbol{v}}_n + \sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j\right)\right)$$

Intégration

Runge-Kutta sur l'accélération

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \sum_{i=1}^{s} b_i \boldsymbol{k}_i$$

$$\boldsymbol{k}_i = \Delta t(1 - P) f \left(\boldsymbol{v}_n + \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j \right)$$

Runge-Kutta sur la vitesse

$$\tilde{\boldsymbol{v}}_{n+1} = (1 - P) \left(\tilde{\boldsymbol{v}}_n + \sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i \right)$$

$$\tilde{\boldsymbol{k}}_i = \Delta t f \left((1 - P) \left(\tilde{\boldsymbol{v}}_n + \sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j \right) \right)$$

Équivalence

Théorème

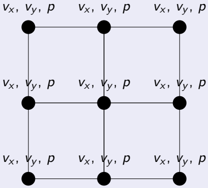
Si $\mathbf{v}_n = \tilde{\mathbf{v}}_n$ *et* $\mathbf{v}_n$ *sont à divergence nulle alors* $\mathbf{v}_{n+1} = \tilde{\mathbf{v}}_{n+1}$.

Corollaire

En résolvant l'équation de Navier-Stokes par une méthode de Runge-Kutta, on est éxactement à divergence nulle.

Discrétisation "standard"

Position des variables



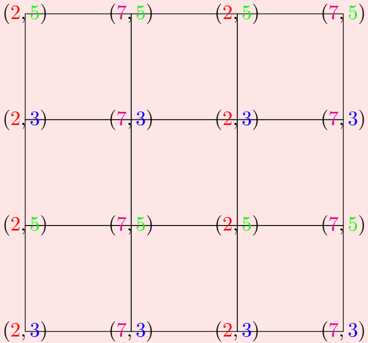
Dérivée

$$\partial_x a_i = \frac{a_{i+1} - a_{i-1}}{\Delta x}$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}_{i,j} = \frac{v^x_{i+1,j} - v^x_{i-1,j}}{\Delta x} + \frac{v^y_{i+1,j} - v^y_{i-1,j}}{\Delta y}$$

Problème damier

Divergence nulle ! ! ! !

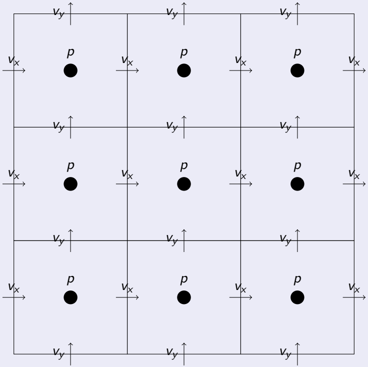


Solution :

Changer la grille.

Discrétisation sur grille décalée

Position des variables



Dérivée sur grill décalée

Gradient

$$\boldsymbol{\nabla} p_{ij} = \begin{pmatrix} \frac{p_{i,j}-p_{i-1,j}}{\Delta x} \\ \frac{p_{i,j}-p_{i,j-1}}{\Delta y} \end{pmatrix}$$

Divergence

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}_{i,j} = \frac{v_{x\,i+1,j} - v_{x\,i,j}}{\Delta x} + \frac{v_{y\,i,j+1} - v_{y\,i,j}}{\Delta y}$$

Laplacien

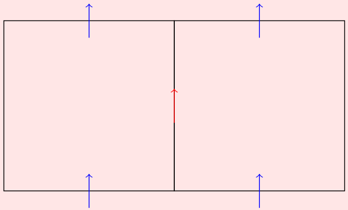
$$\Delta p_{i,j} = \frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{(\Delta x)^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{(\Delta y)^2}$$

Terme convectif sur grille décalée

Sorte de terme

$$v^j \partial_j v^i$$

Problème :



Terme convectif sur grille décalée (suite)

Dérivée centrée

$$\left(v^j \partial_j v^i\right)_{\boldsymbol{k}}^i = v^{ji}{}_{\boldsymbol{k}} \frac{v^i{}_{\boldsymbol{k}+\boldsymbol{e}_j} - v^i{}_{\boldsymbol{k}-\boldsymbol{e}_j}}{2\Delta x_j}$$

Dérivée upwind

si $v^{ji}{}_{\boldsymbol{k}} < 0$

$$\left(v^j \partial_j v^i\right)_{\boldsymbol{k}}^i = v^{ji}{}_{\boldsymbol{k}} \frac{v^i{}_{\boldsymbol{k}+\boldsymbol{e}_j} - v^i{}_{\boldsymbol{k}}}{\Delta x_j}$$

si $v^{ji}{}_{\boldsymbol{k}} > 0$

$$\left(v^j \partial_j v^i\right)_{\boldsymbol{k}}^i = v^{ji}{}_{\boldsymbol{k}} \frac{v^i{}_{\boldsymbol{k}} - v^i{}_{\boldsymbol{k}-\boldsymbol{e}_j}}{\Delta x_j}$$

si $v^{ji}{}_{\boldsymbol{k}} = 0$

$$\left(v^j \partial_j v^i\right)_{\boldsymbol{k}}^i = 0$$

Algorithme en topologie fixe

Principe

- Utiliser une méthode de Runge-Kutta de notre choix.
- Evaluer l'accéleration sur une grille décalée.
- Projeter chaque accélération ou chaque vitesse.

Runge-Kutta sur l'accélération

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \sum_{i=1}^{s} b_i \boldsymbol{k}_i$$
$$\boldsymbol{k}_i = \Delta t (1-P) f(\boldsymbol{v}_n + \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j)$$

Runge-Kutta sur la vitesse

$$\tilde{\boldsymbol{v}}_{n+1} = (1-P)(\tilde{\boldsymbol{v}}_n + \sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i)$$
$$\tilde{\boldsymbol{k}}_i = \Delta t f((1-P)(\tilde{\boldsymbol{v}}_n + \sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j))$$

Algorithme en topologie fixe

Principe

- Utiliser une méthode de Runge-Kutta de notre choix.
- Evaluer l'accéleration sur une grille décalée.
- Projeter chaque accélération ou chaque vitesse.

Runge-Kutta sur l'accélération

$$\begin{aligned} \boldsymbol{v}_{n+1} &= \boldsymbol{v}_n + \sum_{i=1}^{s} b_i \boldsymbol{k}_i \\ \boldsymbol{k}_i &= \Delta t (1-P) f(\boldsymbol{v}_n + \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j) \end{aligned}$$

Runge-Kutta sur la vitesse

$$\begin{aligned} \tilde{\boldsymbol{v}}_{n+1} &= (1-P)(\tilde{\boldsymbol{v}}_n + \sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i) \\ \tilde{\boldsymbol{k}}_i &= \Delta t f((1-P)(\tilde{\boldsymbol{v}}_n + \sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j)) \end{aligned}$$

Algorithme en topologie fixe

Principe

- Utiliser une méthode de Runge-Kutta de notre choix.
- Evaluer l'accéleration sur une grille décalée.
- Projeter chaque accélération ou chaque vitesse.

Runge-Kutta sur l'accélération

$$\begin{aligned} \boldsymbol{v}_{n+1} &= \boldsymbol{v}_n + \textstyle\sum_{i=1}^{s} b_i \boldsymbol{k}_i \\ \boldsymbol{k}_i &= \Delta t (1-P) f(\boldsymbol{v}_n + \textstyle\sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j) \end{aligned}$$

Runge-Kutta sur la vitesse

$$\begin{aligned} \tilde{\boldsymbol{v}}_{n+1} &= (1-P)(\tilde{\boldsymbol{v}}_n + \textstyle\sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i) \\ \tilde{\boldsymbol{k}}_i &= \Delta t f((1-P)(\tilde{\boldsymbol{v}}_n + \textstyle\sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j)) \end{aligned}$$

Algorithme en topologie fixe

Principe

- Utiliser une méthode de Runge-Kutta de notre choix.
- Evaluer l'accéleration sur une grille décalée.
- Projeter chaque accélération ou chaque vitesse.

Runge-Kutta sur l'accélération

$$\begin{aligned} \boldsymbol{v}_{n+1} &= \boldsymbol{v}_n + \textstyle\sum_{i=1}^{s} b_i \boldsymbol{k}_i \\ \boldsymbol{k}_i &= \Delta t (1-P) f(\boldsymbol{v}_n + \textstyle\sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j) \end{aligned}$$

Runge-Kutta sur la vitesse

$$\begin{aligned} \tilde{\boldsymbol{v}}_{n+1} &= (1-P)(\tilde{\boldsymbol{v}}_n + \textstyle\sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i) \\ \tilde{\boldsymbol{k}}_i &= \Delta t f((1-P)(\tilde{\boldsymbol{v}}_n + \textstyle\sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j)) \end{aligned}$$

Algorithme en topologie fixe

Principe

- Utiliser une méthode de Runge-Kutta de notre choix.
- Evaluer l'accéleration sur une grille décalée.
- Projeter chaque accélération ou chaque vitesse.

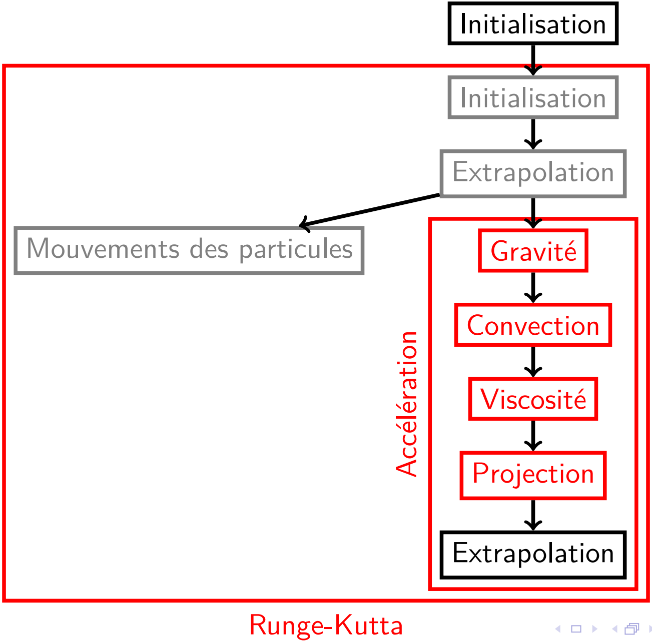
Runge-Kutta sur l'accélération

$$\begin{aligned} \boldsymbol{v}_{n+1} &= \boldsymbol{v}_n + \textstyle\sum_{i=1}^{s} b_i \boldsymbol{k}_i \\ \boldsymbol{k}_i &= \Delta t (1-P) f\left(\boldsymbol{v}_n + \textstyle\sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j\right) \end{aligned}$$

Runge-Kutta sur la vitesse

$$\begin{aligned} \tilde{\boldsymbol{v}}_{n+1} &= (1-P)\left(\tilde{\boldsymbol{v}}_n + \textstyle\sum_{i=1}^{s} b_i \tilde{\boldsymbol{k}}_i\right) \\ \tilde{\boldsymbol{k}}_i &= \Delta t f\left((1-P)\left(\tilde{\boldsymbol{v}}_n + \textstyle\sum_{j=1}^{s} a_{ij} \tilde{\boldsymbol{k}}_j\right)\right) \end{aligned}$$

Schéma de l'algorithme



Détermination analytique de la topologie

Déplacement de la position

$$\partial_t \boldsymbol{x} = \boldsymbol{v}$$

Surface libre

$$\sigma_{ij} = -p\delta_{ij} + \nu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right)$$

$\sum_{i,j} \sigma_{ij} n_i n_j = 0$

$\sum_{i,j} \sigma_{ij} t_i^1 n_j = 0$

$\sum_{i,j} \sigma_{ij} t_i^2 n_j = 0$

$$p_{surf} = A\boldsymbol{v}_{int}$$

$$\boldsymbol{v}_{surf} = B\boldsymbol{v}_{int}$$

Détermination analytique de la topologie

Déplacement de la position

$$\partial_t \boldsymbol{x} = \boldsymbol{v}$$

Surface libre

$$\sigma_{ij} = -p\delta_{ij} + \nu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right)$$

$$\sum_{i,j} \sigma_{ij} n_i n_j = 0$$

$$\sum_{i,j} \sigma_{ij} t_i^1 n_j = 0$$

$$\sum_{i,j} \sigma_{ij} t_i^2 n_j = 0$$

$$p_{surf} = A\boldsymbol{v}_{int}$$

$$\boldsymbol{v}_{surf} = B\boldsymbol{v}_{int}$$

Détermination analytique de la topologie

Déplacement de la position

$$\partial_t \boldsymbol{x} = \boldsymbol{v}$$

Surface libre

$$\sigma_{ij} = -p\delta_{ij} + \nu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right)$$

$$\sum_{i,j} \sigma_{ij} n_i n_j = 0$$
$$\sum_{i,j} \sigma_{ij} t_i^1 n_j = 0$$
$$\sum_{i,j} \sigma_{ij} t_i^2 n_j = 0$$

$$p_{surf} = A\boldsymbol{v}_{int}$$
$$\boldsymbol{v}_{surf} = B\boldsymbol{v}_{int}$$

Discrétisation

Chose nécessaire

- Résoudre :

$$\partial_t \boldsymbol{x} = \boldsymbol{v}$$

- Imposer les conditions aux bords variables :

$$p_{surf} = A\boldsymbol{v}_{int}$$

$$\boldsymbol{v}_{surf} = B\boldsymbol{v}_{int}$$

Discrétisation

Chose nécessaire

- Résoudre :

$$\partial_t \boldsymbol{x} = \boldsymbol{v}$$

- Imposer les conditions aux bords variables :

$$p_{surf} = A\boldsymbol{v}_{int}$$
$$\boldsymbol{v}_{surf} = B\boldsymbol{v}_{int}$$

Discrétisation

Particule

On discrétise l'équation avec des particules. Et on interpole la vitesse de la grille sur les particules pour résoudre :

$$\partial_t \boldsymbol{x} = \boldsymbol{v}_I(\boldsymbol{x})$$

Imposition des conditions aux bords

On calcul des points fantômes hors du domaine pour imposer les conditions aux bords. Ces points fantômes sont obtenus par des méthodes d'extrapolations.

$$p_{ext} = A\boldsymbol{v}_{int}$$

$$\boldsymbol{v}_{ext} = B\boldsymbol{v}_{int}$$

Discrétisation

Particule

On discrétise l'équation avec des particules. Et on interpole la vitesse de la grille sur les particules pour résoudre :

$$\partial_t \boldsymbol{x} = \boldsymbol{v}_I(\boldsymbol{x})$$

Imposition des conditions aux bords

On calcul des points fantômes hors du domaine pour imposer les conditions aux bords. Ces points fantômes sont obtenus par des méthodes d'extrapolations.

$$p_{ext} = A\boldsymbol{v}_{int}$$

$$\boldsymbol{v}_{ext} = B\boldsymbol{v}_{int}$$

Intégration

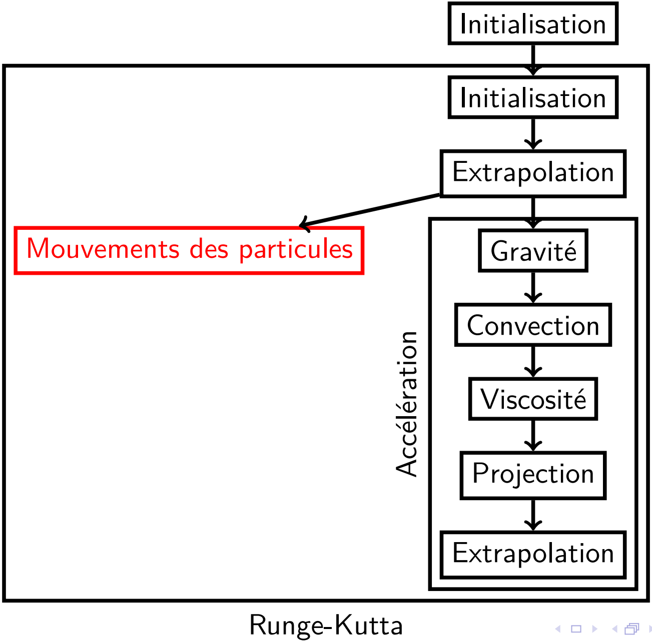
Équations

$$\begin{aligned}
\partial_t \boldsymbol{v}_{int} &= (1+P) f(\boldsymbol{v}) \\
\partial_t \boldsymbol{x} &= \boldsymbol{v}_I(\boldsymbol{x}) \\
p_{ext} &= A \boldsymbol{v}_{int} \\
\boldsymbol{v}_{ext} &= B \boldsymbol{v}_{int}
\end{aligned}$$

Une équation différentielle

$$\begin{aligned}
\boldsymbol{v}_{ext} &= B \boldsymbol{v}_{int} \\
\partial_t \boldsymbol{v}_{ext} &\approx B \partial_t \boldsymbol{v}_{int} = B(1+P) f(\boldsymbol{v})
\end{aligned}$$

Schéma de l'algorithme



Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

Extrapolation

Méthode

- On extrapole par les conditions aux bords.
- On extrapole par une méthode générique pour le reste.

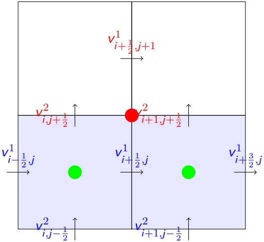
Conditions aux bords en 2d

- Plan vertical ou horizontal.
- Plan diagonal.

Conditions aux bords en 3d

- Plan vertical ou horizontal.
- Plan diagonal sur 2 axes.
- Plan diagonal sur 3 axes.

En 2d plan vertical ou horizontal



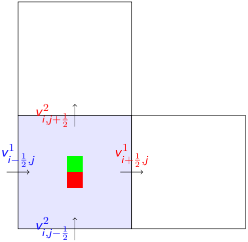
$$\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} = 0$$

$$0 = \frac{v^1_{i+\frac{1}{2},j+1} - v^1_{i+\frac{1}{2},j}}{\Delta x_2} + \frac{v^2_{i+1,j+\frac{1}{2}} - v^2_{i,j+\frac{1}{2}}}{\Delta x_1}$$

$$0 = \frac{v^1_{i+\frac{1}{2},j} - v^1_{i-\frac{1}{2},j}}{\Delta x_1} + \frac{v^2_{i,j+\frac{1}{2}} - v^2_{i,j-\frac{1}{2}}}{\Delta x_2}$$

$$0 = \frac{v^1_{i+\frac{3}{2},j} - v^1_{i+\frac{1}{2},j}}{\Delta x_1} + \frac{v^2_{i+1,j+\frac{1}{2}} - v^2_{i,j-\frac{1}{2}}}{\Delta x_2}$$

En 2d plan diagonal



$$\frac{\partial v_1}{\partial x_1} - \frac{\partial v_2}{\partial x_2} = 0$$

Extrapolations

Extrapolation générique pour la surface

1. On extrapole les vitesses inconnues de la surface par moyenne des vitesses intérieures voisines.
2. On calcul la divergence et on la partage avec toutes les vitesses inconnues.

Extrapolation générique pour plus loin

1. On calcul une variable nommée layer qui indique la distance à la surface.
2. On extrapole par vitesse moyenne plus proche de la surface (layer plus petit).

Extrapolations

Extrapolation générique pour la surface

1. On extrapole les vitesses inconnues de la surface par moyenne des vitesses intérieures voisines.
2. On calcul la divergence et on la partage avec toutes les vitesses inconnues.

Extrapolation générique pour plus loin

1. On calcul une variable nommée layer qui indique la distance à la surface.
2. On extrapole par vitesse moyenne plus proche de la surface (layer plus petit).

Extrapolations

Extrapolation générique pour la surface

1. On extrapole les vitesses inconnues de la surface par moyenne des vitesses intérieures voisines.
2. On calcul la divergence et on la partage avec toutes les vitesses inconnues.

Extrapolation générique pour plus loin

1. On calcul une variable nommée layer qui indique la distance à la surface.
2. On extrapole par vitesse moyenne plus proche de la surface (layer plus petit).

Extrapolations

Extrapolation générique pour la surface

1. On extrapole les vitesses inconnues de la surface par moyenne des vitesses intérieures voisines.
2. On calcul la divergence et on la partage avec toutes les vitesses inconnues.

Extrapolation générique pour plus loin

1. On calcul une variable nommée layer qui indique la distance à la surface.
2. On extrapole par vitesse moyenne plus proche de la surface (layer plus petit).

Conditions aux bords pour la pression

Méthode

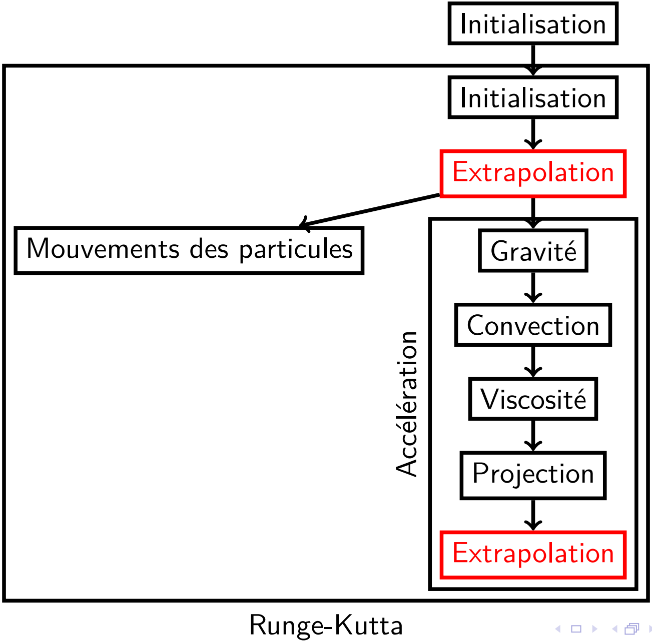
On impose les conditions aux bords de façon centrée.

$$\boldsymbol{n} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$p = 2\nu \frac{\partial v_1}{\partial x_1}$$

S'il y a plusieurs plans comme cela on somme.

Schéma de l'algorithme



Interpolation

Raison

Nécessaire pour connaître la vitesse à la position des particules.

n-linéaire

1d :

$$f(x,y) = c_2 \frac{x - x_1}{x_2 - x_1} + c_1 \frac{x - x_2}{x_1 - x_2}$$

2d :

$$f(x,y) = c_{11} \frac{x - x_2}{x_1 - x_2} \frac{y - y_2}{y_1 - y_2} + c_{12} \frac{x - x_2}{x_1 - x_2} \frac{y - y_1}{y_2 - y_1} + c_{21} \frac{x - x_1}{x_2 - x_1} \frac{y - y_2}{y_1 - y_2} + c_{22} \frac{x - x_2}{x_1 - x_2} \frac{y - y_2}{y_1 - y_2}$$

Interpolation

Raison

Nécessaire pour connaître la vitesse à la position des particules.

n-linéaire

1d :

$$f(x,y) = c_2 \frac{x - x_1}{x_2 - x_1} + c_1 \frac{x - x_2}{x_1 - x_2}$$

2d :

$$f(x,y) = c_{11} \frac{x - x_2}{x_1 - x_2} \frac{y - y_2}{y_1 - y_2} + c_{12} \frac{x - x_2}{x_1 - x_2} \frac{y - y_1}{y_2 - y_1} + c_{21} \frac{x - x_1}{x_2 - x_1} \frac{y - y_2}{y_1 - y_2} + c_{22} \frac{x - x_2}{x_1 - x_2} \frac{y - y_2}{y_1 - y_2}$$

Interpolation

Raison

Nécessaire pour connaître la vitesse à la position des particules.

n-linéaire

1d :

$$f(x,y) = c_2 \frac{x - x_1}{x_2 - x_1} + c_1 \frac{x - x_2}{x_1 - x_2}$$

2d :

$$\begin{aligned} f(x,y) = c_{11} \frac{x - x_2}{x_1 - x_2} \frac{y - y_2}{y_1 - y_2} + c_{12} \frac{x - x_2}{x_1 - x_2} \frac{y - y_1}{y_2 - y_1} \\ + c_{21} \frac{x - x_1}{x_2 - x_1} \frac{y - y_2}{y_1 - y_2} + c_{22} \frac{x - x_2}{x_1 - x_2} \frac{y - y_2}{y_1 - y_2} \end{aligned}$$

Initialisation

Utilité

Obtenir la nouvelle topologie de la position des particules. On utilise une variable nommée layer pour indiquer la distance de la surface.

Pas de temps idéal (condition CFL)

On calcul le pas de temps idéal tel qu'une particule ne peut se déplacer que d'une fraction de la distance d'une cellule.

Initialisation

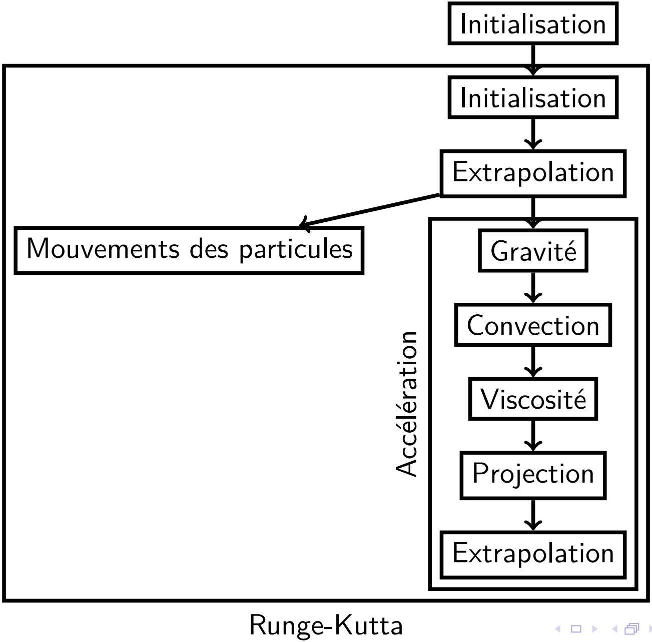
Utilité

Obtenir la nouvelle topologie de la position des particules. On utilise une variable nommée layer pour indiquer la distance de la surface.

Pas de temps idéal (condition CFL)

On calcul le pas de temps idéal tel qu'une particule ne peut se déplacer que d'une fraction de la distance d'une cellule.

Schéma de l'algorithme



Jet latéral

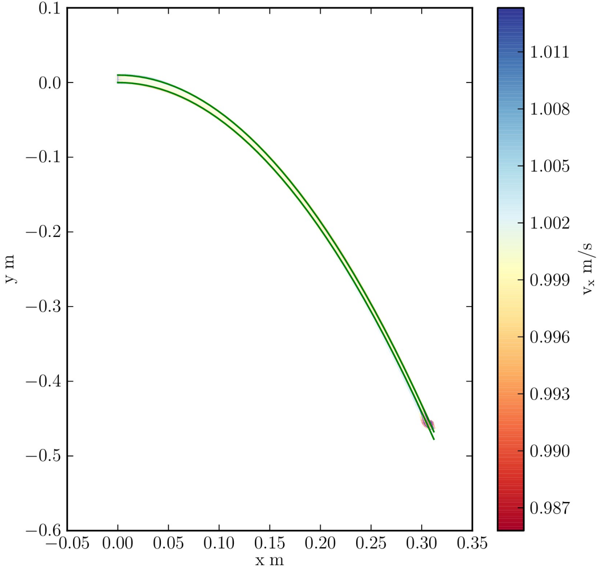
Particulaire

$$
\begin{aligned}
v_x &= v_0 \\
v_y &= -\frac{g}{2}(t - t_0)^2 \\
x &= v_0(t - t_0) \\
y &= y_0 - \frac{g}{2}(t - t_0)^2
\end{aligned}
$$

Cartésienne

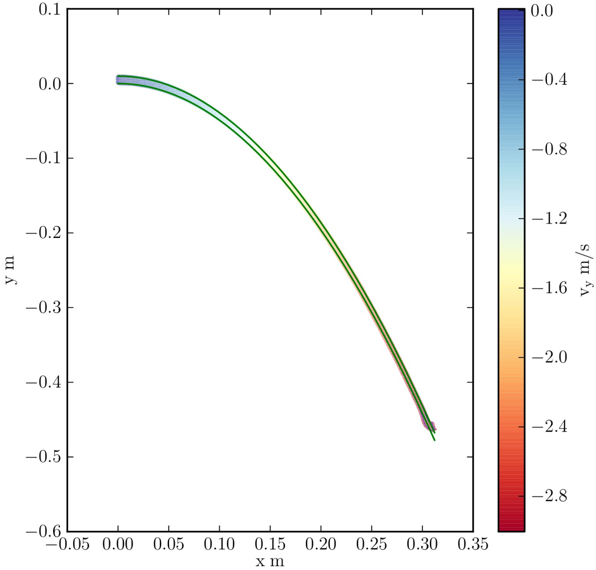
$$
\begin{aligned}
v_x &= v_0 \\
v_y &= -\frac{g}{2}\frac{x^2}{v_0}
\end{aligned}
$$

Jet latéral



0.1
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
y m
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
x m
1.011
1.008
1.005
1.002
0.999
0.996
0.993
0.990
0.987
v_x m/s

Jet latéral



0.1
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
y m
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
x m
0.0
−0.4
−0.8
−1.2
−1.6
−2.0
−2.4
−2.8
v_y m/s

Jet latéral et Jet d'eau

Avantage et défaut

- ⊕ Méthode de discrétisation bien connu.
- ⊕ Système linéaire facil à résoudre.
- ⊖ Terme convectif plus dur.
- ⊕ Moins de voisin à traiter.
- ⊖ Dépendant du choix des axes.

Avantage et défaut

- $\oplus$ Méthode de discrétisation bien connu.
- $\oplus$ Système linéaire facil à résoudre.
- $\ominus$ Terme convectif plus dur.
- $\oplus$ Moins de voisin à traiter.
- $\ominus$ Dépendant du choix des axes.

Avantage et défaut

- $\oplus$ Méthode de discrétisation bien connu.
- $\oplus$ Système linéaire facil à résoudre.
- $\ominus$ Terme convectif plus dur.
- $\oplus$ Moins de voisin à traiter.
- $\ominus$ Dépendant du choix des axes.

Avantage et défaut

- $\oplus$ Méthode de discrétisation bien connu.
- $\oplus$ Système linéaire facil à résoudre.
- $\ominus$ Terme convectif plus dur.
- $\oplus$ Moins de voisin à traiter.
- $\ominus$ Dépendant du choix des axes.

Avantage et défaut

- $\oplus$ Méthode de discrétisation bien connu.
- $\oplus$ Système linéaire facil à résoudre.
- $\ominus$ Terme convectif plus dur.
- $\oplus$ Moins de voisin à traiter.
- $\ominus$ Dépendant du choix des axes.

Amélioration possible

MAC

- Écrire une version parallèle du code.
- Solveur specialisé pour changement de topologie sans tout recalculer.
- Éviter l'effet escalier aux bords.
- Utiliser une autre forme de topologie que les particules.
- Ordre supérieur de discrétisation.
- Éviter le problème de saut dans les conditions aux bords.

Amélioration possible

MAC

- Écrire une version parallèle du code.
- Solveur specialisé pour changement de topologie sans tout recalculer.
- Éviter l'effet escalier aux bords.
- Utiliser une autre forme de topologie que les particules.
- Ordre supérieur de discrétisation.
- Éviter le problème de saut dans les conditions aux bords.

Amélioration possible

MAC

- Écrire une version parallèle du code.
- Solveur specialisé pour changement de topologie sans tout recalculer.
- Éviter l'effet escalier aux bords.
- Utiliser une autre forme de topologie que les particules.
- Ordre supérieur de discrétisation.
- Éviter le problème de saut dans les conditions aux bords.

Amélioration possible

MAC

- Écrire une version parallèle du code.
- Solveur specialisé pour changement de topologie sans tout recalculer.
- Éviter l'effet escalier aux bords.
- Utiliser une autre forme de topologie que les particules.
- Ordre supérieur de discrétisation.
- Éviter le problème de saut dans les conditions aux bords.

Amélioration possible

MAC

- Écrire une version parallèle du code.
- Solveur specialisé pour changement de topologie sans tout recalculer.
- Éviter l'effet escalier aux bords.
- Utiliser une autre forme de topologie que les particules.
- Ordre supérieur de discrétisation.
- Éviter le problème de saut dans les conditions aux bords.

Amélioration possible

MAC

- Écrire une version parallèle du code.
- Solveur specialisé pour changement de topologie sans tout recalculer.
- Éviter l'effet escalier aux bords.
- Utiliser une autre forme de topologie que les particules.
- Ordre supérieur de discrétisation.
- Éviter le problème de saut dans les conditions aux bords.

Remerciements

Personne

Prof. Martin Gander : Pour m'avoir suivi dans ce travail ainsi que pour son cours d'analyse numérique.

Dr. Felix Kwok : Pour m'avoir suivi dans ce travail et pour les tps et exercices.

Prof. Peter Wittwer : Pour les cours de mathématiques et physique.

Reto, Norma, Roland, Tamara, Bruno, François : Pour me soutenir et les bons moments passés ensemble.

Remerciements

Logiciel

LuaLaTeX : Pour l'écriture du rapport.

Gcc : Pour compiler mon code c++11.

Valgrind : Pour la détection des fuites memoires et autre érreurs de memoire.

Vtk : Pour l'exportation des données.

Paraview : Pour la visualisation des données.

Umfpacks : Comme solveur linéaire directe.

Pyamg : Comme solveur linéaire itératif.

Boost : Pour contenir des librairies utiles comme Boost-Python utilisé comme binding avec Pyamg.