OpenFOAMにおける混相流計算

2013/1/19

大阪大学大学院基礎工学研究科
岡野研　M1 山本　卓也

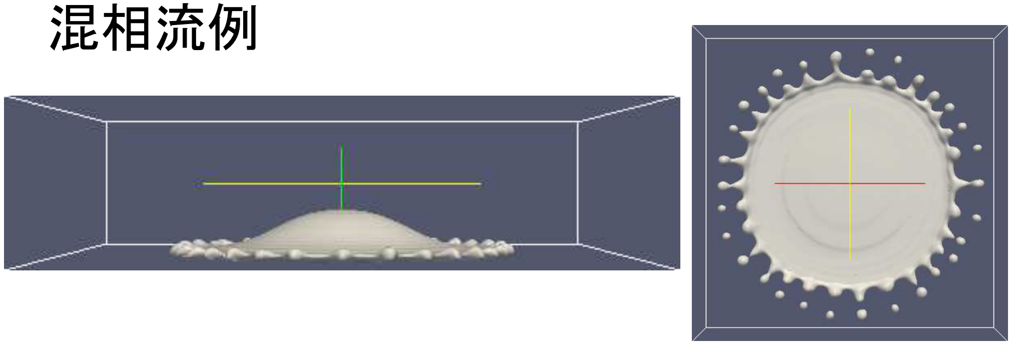
混相流とは

混相流･･･複数の相が混ざり合う流れ

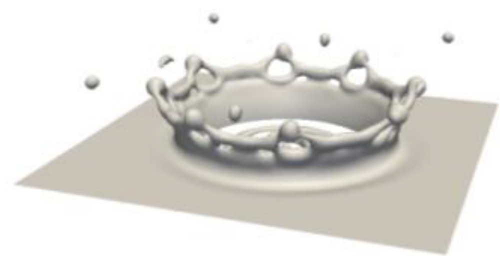
例)気液二相流(空気-水)　キャビテーション、気泡塔
　液液二相流(水-油)　有機溶媒と水の混合溶液
　固液二相流(粒子-水)　懸濁液

<u>工業的に重要であることが多い</u>

混相流例



横井,数値流体シンポジウム,2012,C03-4



富原ら,数値流体シンポジウム,2011,C04-3

様々な数値計算法が存在する

混相流の数値計算法

混相流のシミュレーションを分類すると以下の通りになる

- メッシュフリー法
- 界面捕獲法 (Interface Tracking)
- 界面追跡法 (Interface Capturing)
- 平均化(二流体)モデル

- **メッシュフリー法**

粒子法(MPS, SPH)

- **界面捕獲法**

VOF法

Level-Set法

Phase Field法

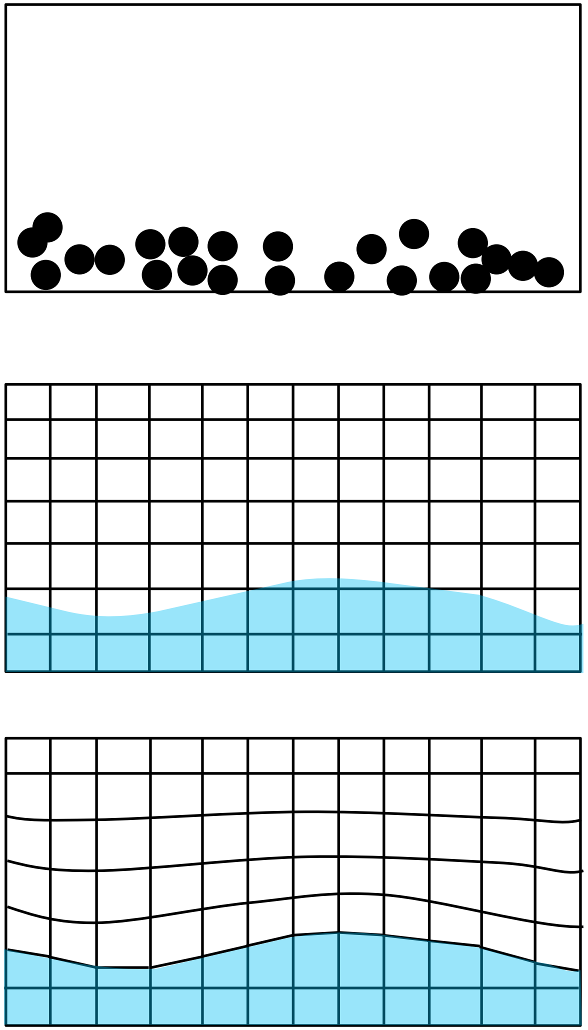
- **界面追跡法**

BFC(界面適合座標)

ALE(Arbitrary Lagrangian-Eulerian)

それぞれの手法の特徴

概念図



- **メッシュフリー法**

微小の粒子の運動で表現する
メッシュ分割が不必要
衝撃波等の不連続場の扱いが容易
大変形、歪みに対して精度保持
精度が悪い
計算時間多大

- **界面捕獲法**

計算格子を移動せずに計算する
手法によって異なるが界面がなまる

- **界面追跡法**

計算格子を時々刻々と移動する
精度がかなり高い
計算が破綻しやすい
砕波現象等の大変形をするものに不向き

OpenFOAMにおける実装

- **メッシュフリー法**

？粒子法(MPS, SPH)

- **界面捕獲法**

〇VOF法

×Level-Set法

×Phase Field法

×(Front tracking法)

- **界面追跡法**

△BFC(界面適合座標)

？ALE(Arbitrary Lagrangian-Eulerian)

混相流のコードが少ない
ほとんどVOF法を少し変えたもの（interFoam系）

そこで皆さん

一緒にコード開発しませんか？？

VOF(Volume of Fluid)法

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} = -\nabla P + \nu\nabla^2\boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

k: 界面の曲率

連続式

$$\nabla\cdot\boldsymbol{v} = 0$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla\cdot(\alpha\boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

- VOF法の欠点

界面の形状が明確に定義されない

- VOF法の長所

境界面の複雑な変形を伴う現象をシミュレート可能

アルゴリズムが単純

現在の研究ではVOF法を解くのみの研究は少ない

⇩

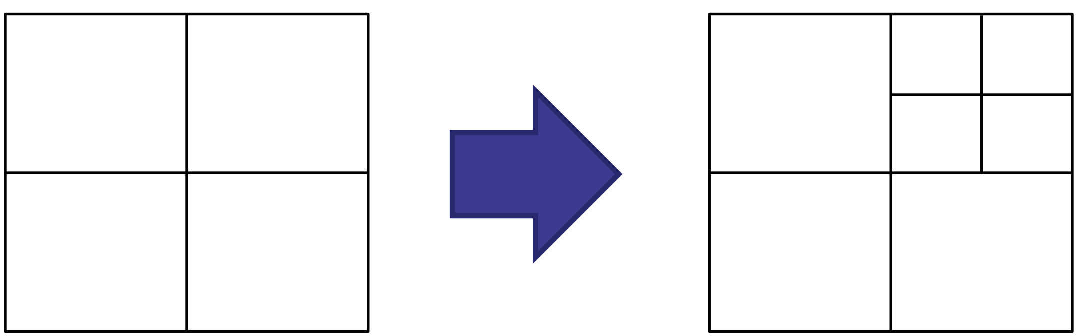
VOF法と様々なものを組み合わせてシミュレーション

OpenFOAMにおけるVOF法の実装

InterFoam	VOF法のみ
InterDymFoam	VOF法＋AMR
InterMixingFoam	VOF法(3つの流体の混合)
…	

OpenFOAMではAMRを用いることによりVOF法の誤差を低減

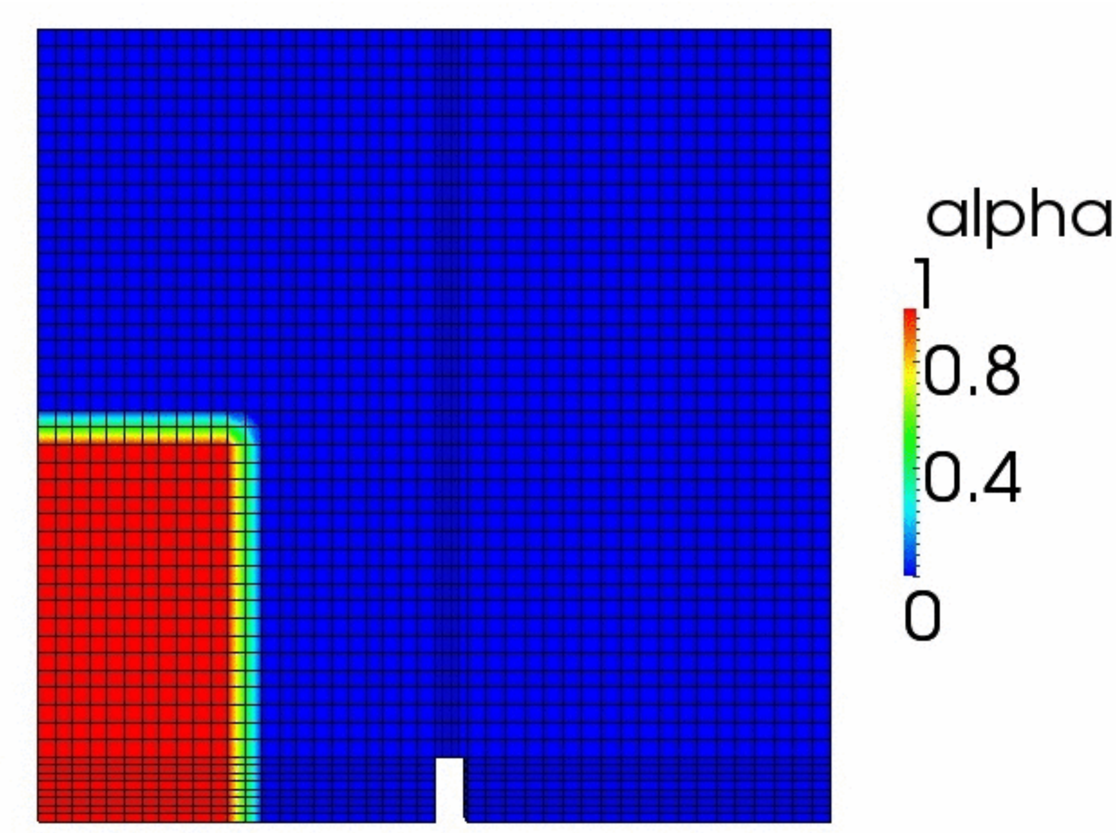
AMR(Adaptive Mesh Refinement)
局所格子分割するライブラリ

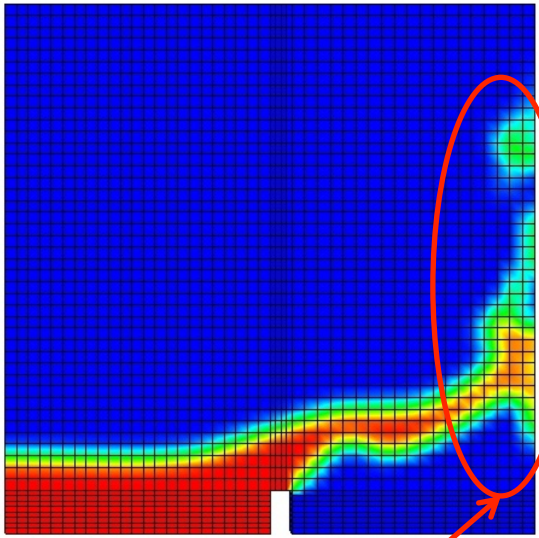


界面近傍で
局所格子分割

VOF法(InterFoam)

Dam Break (Tutorial)

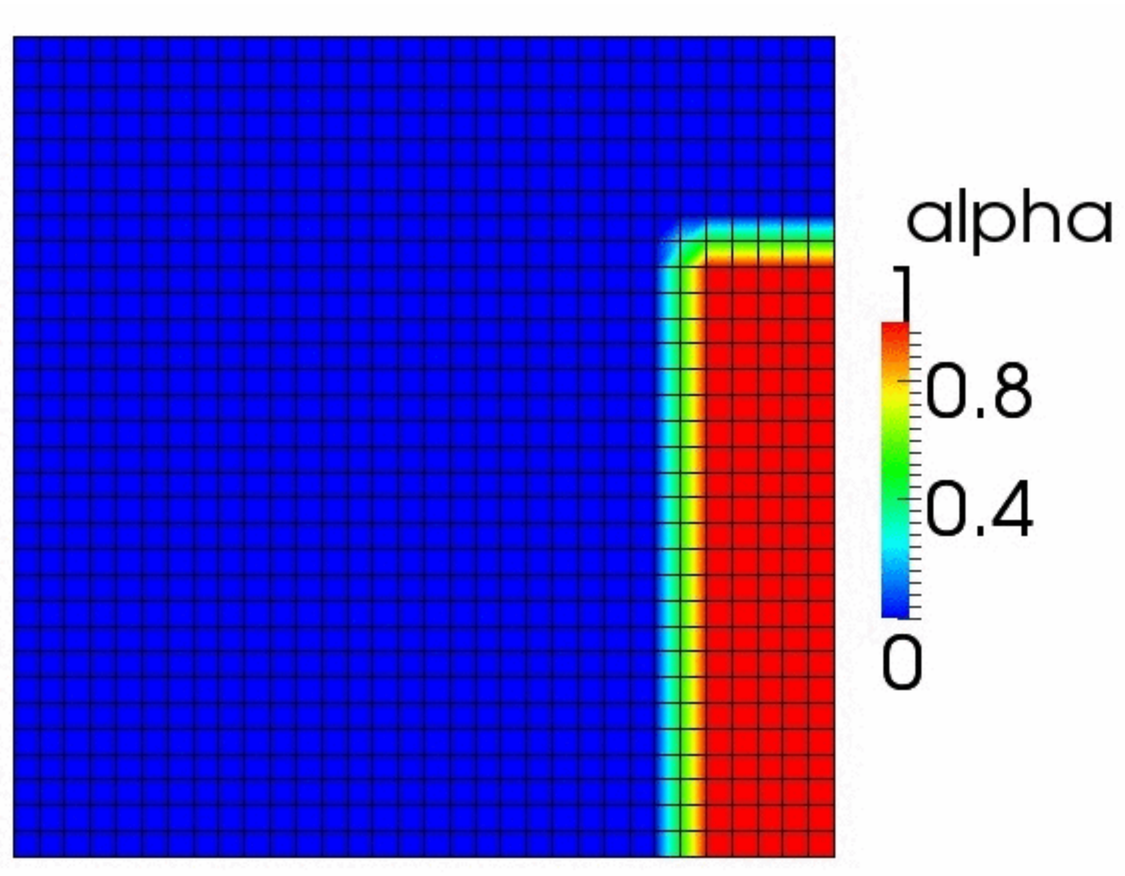


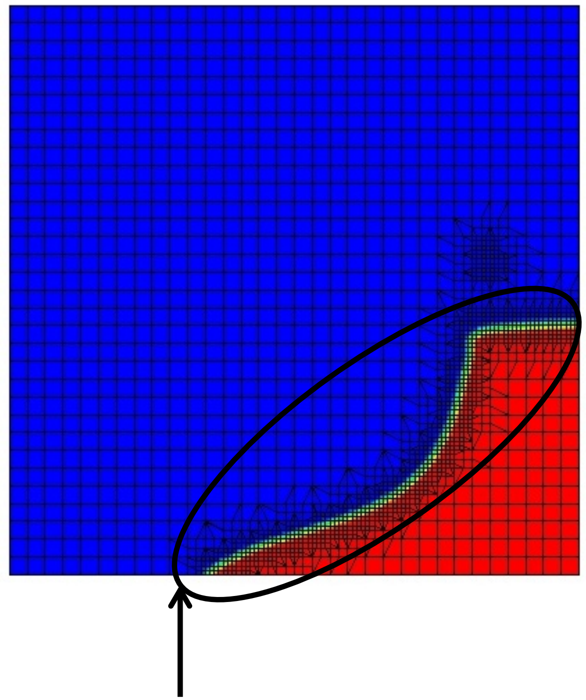


界面の拡散（誤差）
VOF法のみでは誤差が大きい

VOF法＋AMR(InterDymFoam)

Dam Break (Tutorial)





格子局所分割を行っている
格子分割のおかげで界面の拡散(誤差)が低下

VOF法の精度改善方法

<u>VOF法</u>

VOF法＋AMR

CLSVOF法

VOF/PLIC法

VOF(THINC/WLIC)法

数値スキーム

CIP法

WENO法

界面再構築のアルゴリズム

PLIC (Piesewise Linear Interface Calculation)

SLIC (Simple Line Interface Calculation)

WLIC (Weighted Line Interface Calculation)

様々なものが存在

CLSVOF法のOpenFOAMに対する実装を目指す

VOF法のコード解読から始める

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} = -\nabla P + \nu\nabla^2\boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n}\delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla\cdot(\alpha\boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\underline{\rho = \alpha\rho_g + (1-\alpha)\rho_l}$$

$$\mu = \alpha\mu_g + (1-\alpha)\mu_l$$

$$\underline{\rho = \alpha\rho_g + (1-\alpha)\rho_l}$$

createFields.Hの中

```
// Need to store rho for ddt(rho, U)
   volScalarField rho
   (
       IOobject
       (
           "rho",
           runTime.timeName(),
           mesh,
           IOobject::READ_IF_PRESENT
       ),
       alpha1*rho1 + (scalar(1) - alpha1)*rho2,
       alpha1.boundaryField().types()
   );
```

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla P + \nu \nabla^2 \boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\underline{\rho = \alpha \rho_g + (1-\alpha)\rho_l}$$

$$\underline{\mu = \alpha \mu_g + (1-\alpha)\mu_l}$$

$$\underline{\mu = \alpha \mu_g + (1-\alpha)\mu_l}$$

/src/transportModels/incompressible/
incompressibleTwoPhaseMixture/
twoPhaseMixture.C
118行目

```
tmp<volScalarField> twoPhaseMixture::mu() const
{
    volScalarField limitedAlpha1 = min(max(alpha1_, scalar(0)), scalar(1));

    return tmp<volScalarField>
    (
        new volScalarField
        (
            "mu",
            limitedAlpha1*rho1_*nuModel1_->nu()
          + (scalar(1) - limitedAlpha1)*rho2_*nuModel2_->nu()
        )
    );
}
```

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla P + \nu \nabla^2 \boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\rho = \alpha \rho_g + (1 - \alpha) \rho_l$$

$$\mu = \alpha \mu_g + (1 - \alpha) \mu_l$$

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) = 0$$

液相領域 $\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}_l) = 0$

気相領域 $\frac{\partial \alpha}{\partial t} + \nabla \cdot ((1 - \alpha) \boldsymbol{v}_g) = 0$

小文字l, gはそれぞれ液相、気相を表す。

再定義

$$\boldsymbol{v} = \alpha \boldsymbol{v}_l + (1 - \alpha) \boldsymbol{v}_g$$

$$\boldsymbol{v}_r = \boldsymbol{v}_l - \boldsymbol{v}_g$$

$\boldsymbol{v}_r$: 相関速度

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla P + \nu \nabla^2 \boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\rho = \alpha \rho_g + (1-\alpha)\rho_l$$

$$\mu = \alpha \mu_g + (1-\alpha)\mu_l$$

最終形

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) + \nabla \cdot ((1-\alpha)\alpha \boldsymbol{v}_r) = 0$$

alphaEqn.H中で設定

α式の設定については後に説明

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} = -\nabla P + \nu\nabla^2\boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n}\delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla\cdot(\alpha\boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\rho = \alpha\rho_g + (1-\alpha)\rho_l$$

$$\mu = \alpha\mu_g + (1-\alpha)\mu_l$$

表面張力モデルCSFモデル

(Brackbill (1992))

(Continuum Surface Force)

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n}\delta_s$$

σ: 表面張力

k: 曲率

$\boldsymbol{n}$: 法線ベクトル

δ_s: δ関数

表面張力

面積力
（面にかかる力） ⇒ 体積力
体積にかかる力

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

$$F_\sigma = \sigma k \boldsymbol{n} \delta_s$$

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla P + \nu \nabla^2 \boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\rho = \alpha \rho_g + (1 - \alpha) \rho_l$$

$$\mu = \alpha \mu_g + (1 - \alpha) \mu_l$$

Ueqn.H中
19行目

```
if (momentumPredictor)
  {
      solve
      (
          UEqn
       ==
          fvc::reconstruct
          (
              fvc::interpolate(rho)*(g & mesh.Sf())
            + (
                  fvc::interpolate(interface.sigmaK())*fvc::snGrad(alpha1)
                - fvc::snGrad(p)
              ) * mesh.magSf()
          )
      );
  }
```

sigmaK()とは？？

snGrad(alpha1)とは？？

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} = -\nabla P + \nu\nabla^2\boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla\cdot(\alpha\boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\rho = \alpha\rho_g + (1-\alpha)\rho_l$$

$$\mu = \alpha\mu_g + (1-\alpha)\mu_l$$

sigmaK()とは？？
interfaceProperties.H　中
140行目

```
tmp<volScalarField> sigmaK() const
    {
        return sigma_*K_;
    }
```

sigma_ :surface tension
K_ :curvature

sigmaK()

$$F_\sigma = \underline{\sigma k}\, \boldsymbol{n}\, \delta_s$$

InterFoamのソースコード解読

Ver. 1.6.x

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} = -\nabla P + \nu\nabla^2\boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma\kappa\boldsymbol{n}\delta_s$$

流体率αの移流方程式

$$\frac{\partial \alpha}{\partial t} + \nabla\cdot(\alpha\boldsymbol{v}) = 0$$

$\alpha = 1$ :: liquid phase

$0 < \alpha < 1$ :: interface

$\alpha = 0$ :: gas phase

$$\rho = \alpha\rho_g + (1-\alpha)\rho_l$$

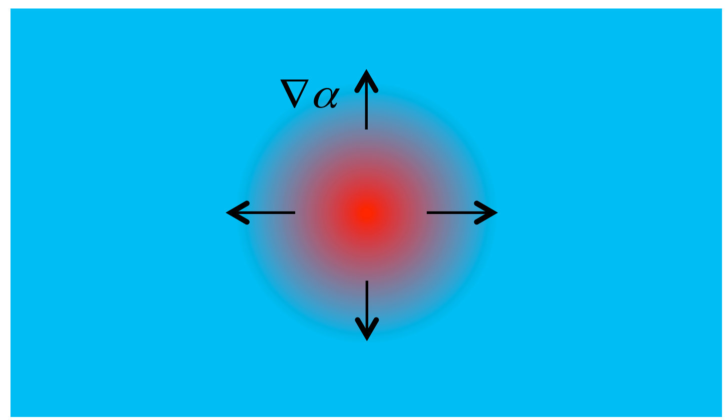
$$\mu = \alpha\mu_g + (1-\alpha)\mu_l$$

snGrad(alpha1)とは？？

プログラマズガイドより

面に垂直な勾配の単位ベクトルを表す。

α場（赤:流体,青:気体）



$$\boldsymbol{n} = \frac{\nabla\alpha}{|\nabla\alpha|}$$

InterFoamのソースコード解読

Ver. 1.6.x

α式の設定

最終形　(微分形)

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{v}) + \nabla \cdot ((1-\alpha)\alpha \boldsymbol{v}_r) = 0$$

有限体積法なので
積分系に変換

ガウスの発散定理

$$\int_s \boldsymbol{n} \cdot \boldsymbol{a} ds = \int_V \nabla \cdot \boldsymbol{a} dV$$

積分系

$$\frac{d}{dt}\int_{\Delta V} \alpha dV + \int_{\Delta V} \nabla \cdot (\alpha \boldsymbol{v}) dV + \int_{\Delta V} \nabla \cdot ((1-\alpha)\alpha \boldsymbol{v}_r) dV = 0$$

$$\frac{d}{dt}\int_{\Delta V} \alpha dV + \int_S \alpha \boldsymbol{v} \cdot \boldsymbol{n} dS + \int_S (1-\alpha)\alpha \boldsymbol{v}_r \cdot \boldsymbol{n} dS$$

InterFoamのソースコード解読

Ver. 1.6.x

非定常項はとりあえず無視して

α式の設定

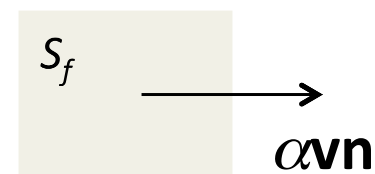
$$\frac{d}{dt}\int_{\Delta V}\alpha dV + \boxed{\int_S \alpha \boldsymbol{v}\cdot \boldsymbol{n} dS} + \boxed{\int_S (1-\alpha)\alpha \boldsymbol{v}_r \cdot \boldsymbol{n} dS}$$

離散化

中点公式により近似

$$(\alpha \boldsymbol{v}\cdot \boldsymbol{n})_f \cdot S_f \qquad ((1-\alpha)\alpha \boldsymbol{v}_r \cdot \boldsymbol{n})_f \cdot S_f$$

イメージ



ここで、fはセル界面上を表す。
S_fは表面積

これがOpenFOAMにどう組み込まれているか？？

InterFoamのソースコード解読

Ver. 1.6.x

alphaEqn.H 中
5~8行目

```
surfaceScalarField phic = mag(phi/mesh.magSf());
phic = min(interface.cAlpha()*phic, max(phic));
surfaceScalarField phir = phic*interface.nHatf();
```

プログラム上では

$$\phi_c = \frac{\phi}{S_f}$$

それぞれ
代入

$$\phi_c = \min(C_\alpha \times \phi_c, \max(\phi_c))$$

$$\phi_r = \phi_c \times n_f$$

$$\phi_r = n_f \min\left[C_\alpha \frac{\phi}{s_f}, \max\left(\frac{\phi}{s_f}\right)\right]$$

文字rから判断すると??

$$((1-\alpha)\alpha \boldsymbol{v}_r \cdot \boldsymbol{n})_f \cdot S_f$$

InterFoamのソースコード解読

Ver. 1.6.x

alphaEqn.H 中
5~8行目

```
surfaceScalarField phic = mag(phi/mesh.magSf());
phic = min(interface.cAlpha()*phic, max(phic));
surfaceScalarField phir = phic*interface.nHatf();
```

find, grepでソース(src)内検索

src/transportModels/interfaceProperties/interfaceProperties.C 中

cAlpha, nHatf_, K_等の設定

InterFoamのソースコード解読

Ver. 1.6.x

src/transportModels/interfaceProperties/interfaceProperties.C 中

117行目

```
// Face unit interface normal flux
   nHatf_ = nHatfv & Sf;
```

$$n_f = n_{fv} \cdot S_f$$

131行目

```
// Simple expression for curvature
   K_ = -fvc::div(nHatf_);
```

$$k = \nabla \cdot n_f$$

146行目

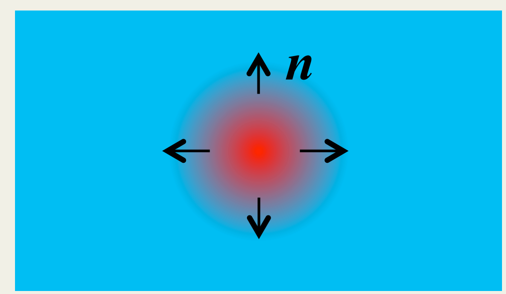
```
transportPropertiesDict_(dict),
   cAlpha_
   (
       readScalar
       (
alpha1.mesh().solutionDict().subDict("PISO").lookup("cAlpha")
       )
   ),
```

C_αの読み込み

表面張力モデルCSFモデル
(Brackbill (1992))
(Continuum Surface Force)

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n}$$

$$k = \nabla \cdot n$$



InterFoamのソースコード解読

Ver. 1.6.x

src/transportModels/interfaceProperties/interfaceProperties.C　中

113行目

```
// Face unit interface normal
    surfaceVectorField nHatfv = gradAlphaf/
(mag(gradAlphaf) + deltaN_);
```

$$n_{fv} = \frac{(\nabla \cdot \alpha)_f}{\left| (\nabla \cdot \alpha)_f + \delta_N \right|}$$

156行目

```
deltaN_
    (
        "deltaN",
        1e-8/pow(average(alpha1.mesh().V()), 1.0/3.0)
    ),
```

$$\delta_N = \frac{1.0e^{-8}}{(\sum_N V_i / N)^{1/3}}$$

個人的に物理的意味はまだ分かっていない。

InterFoamのソースコード解読

Ver. 1.6.x

<u>alphaEqn.H 中</u>
5~8行目

```
surfaceScalarField phic = mag(phi/mesh.magSf());
phic = min(interface.cAlpha()*phic, max(phic));
surfaceScalarField phir = phic*interface.nHatf();
```

<u>最初の推測</u>

$$\phi_r = n_f \min\left[C_\alpha \frac{\phi}{s_f}, \max\left(\frac{\phi}{s_f}\right)\right]$$

文字から判断すると??

$$\left(\left(1-\alpha\right)\alpha \boldsymbol{v}_{\boldsymbol{r}} \cdot \boldsymbol{n}\right)_f \cdot S_f$$

なんとなく
それっぽいという所までの理解

InterFoamのソースコード解読

Ver. 1.6.x

<u>alphaEqn.H 中</u>
9行目~

```
for (int aCorr=0; aCorr<nAlphaCorr; aCorr++)
    {
        surfaceScalarField phiAlpha =
            fvc::flux
            (
                phi,
                alpha1,
                alphaScheme
            )
          + fvc::flux
            (
                -fvc::flux(-phir, scalar(1) - alpha1,
alpharScheme),
                alpha1,
                alpharScheme
            );

        MULES::explicitSolve(alpha1, phi, phiAlpha, 1, 0);

        rhoPhi = phiAlpha*(rho1 - rho2) + phi*rho2;
    }
```

$$\phi\alpha$$

$$\phi_r(1-\alpha)\alpha$$

<u>Fvc::flux 流束を返す。</u>

/src/finiteVolume/finiteVolume/fvc/fvcFlux.c中

プログラム上では

$$\phi\alpha = \phi\alpha + \phi_r(1-\alpha)\alpha$$

MULES(Multidimensional Universal Limiter for Explicit Solution)??

InterFoamのソースコード解読

Ver. 1.6.x

$$\phi\alpha = \phi\alpha + \phi_r (1-\alpha)\alpha$$

$$\phi_r = n_f \min\left[C_\alpha \frac{\phi}{s_f}, \max\left(\frac{\phi}{s_f} \right) \right]$$

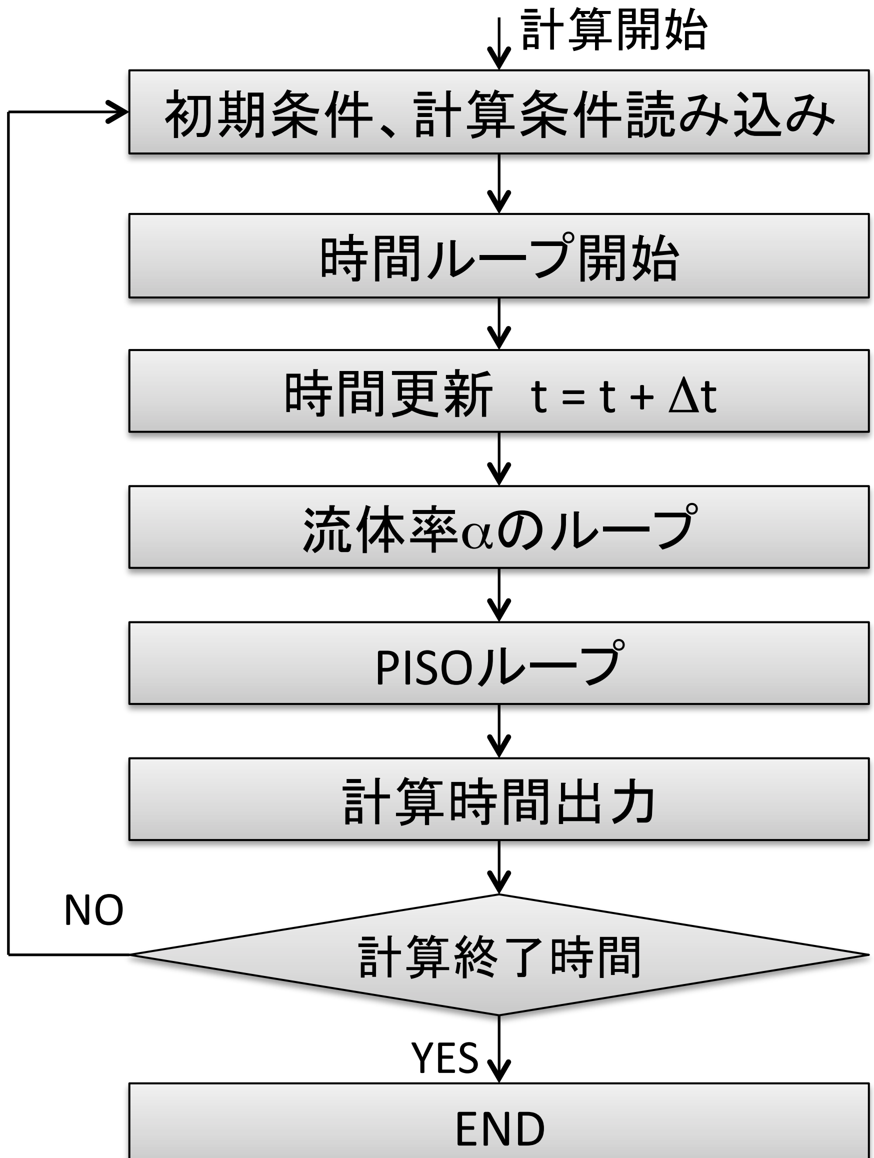
$$\phi = \boldsymbol{v} \cdot S_f$$

ではないかと推測できる

α式の設定

$$\frac{d}{dt}\int_{\Delta V} \alpha dV + \boxed{\int_S \alpha \boldsymbol{v} \cdot \boldsymbol{n} dS} + \boxed{\int_S (1-\alpha)\alpha \boldsymbol{v}_r \cdot \boldsymbol{n} dS}$$

InterFoamのアルゴリズム



InterFoam.C より

CLSVOF法

CLSVOF(Conjugate Level-Set and Volume Of Fluid)

- 支配方程式

Navier-Stokes 式

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} = -\nabla P + \nu \nabla^2 \boldsymbol{v} + \boldsymbol{F}_\sigma + \rho g$$

$$\boldsymbol{F}_\sigma = \sigma k \boldsymbol{n} \delta_s$$

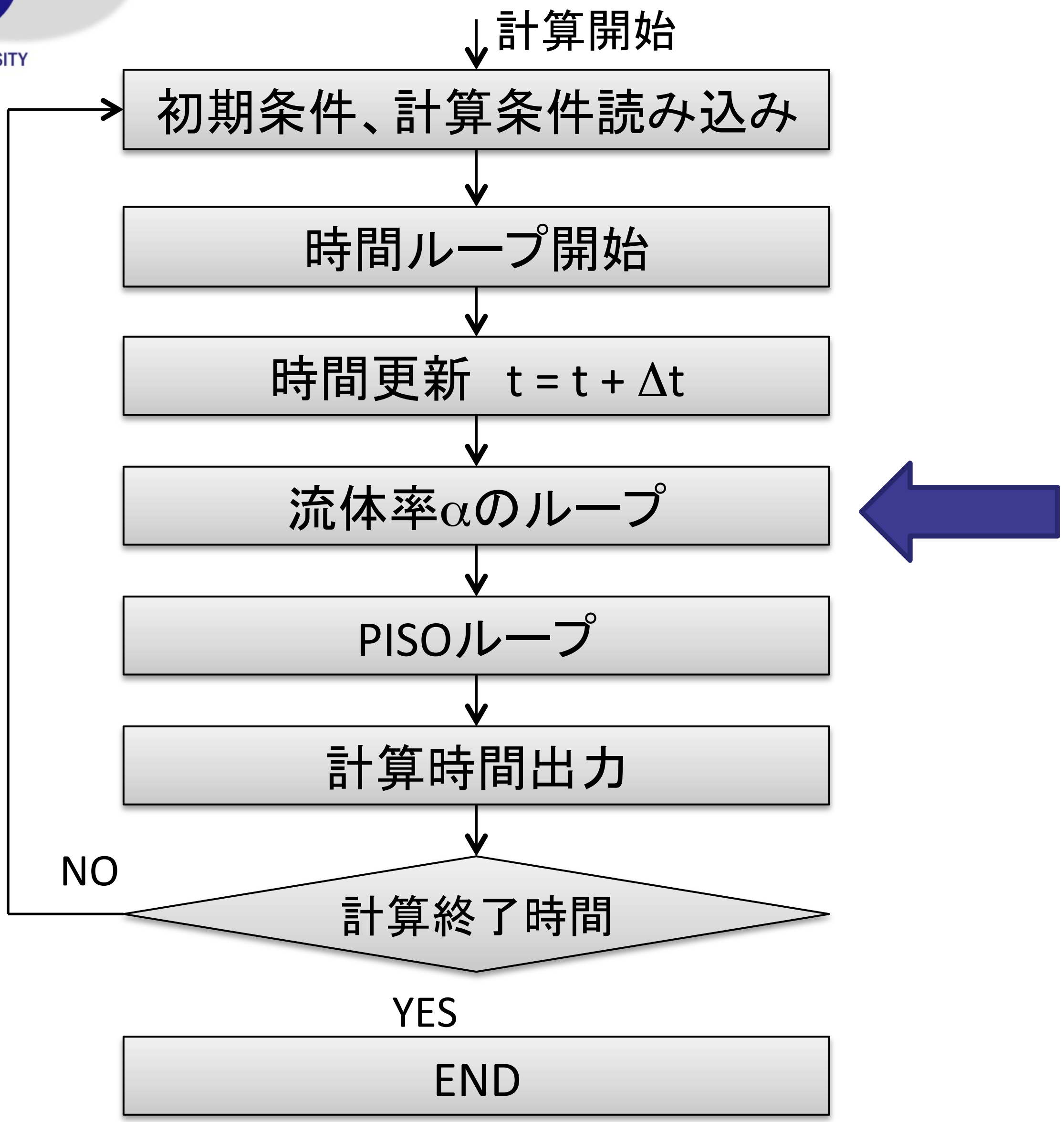
k; 曲率

$$k = \nabla \cdot \boldsymbol{n}$$

曲率等の計算を流体率αを用いて計算する

Level-Set関数を用いて曲率等の計算

CLSVOF法のために変えるところ



αの計算
外力項表面張力
CSFモデル

最後に

もう少し進捗する予定でしたがあまり進みませんでした。
すみませんでした。
現在はOpenFOAMの開発は趣味でやっています。

もう一度言いますが、

どなたか一緒に

OpenFOAMで様々なコード開発に挑戦しませんか？？

特に混相流領域