オープンCAE勉強会@富山 第34回

～OpenFOAMの非ニュートンモデルカスタマイズ～

秋山善克

OpenFOAMに組み込まれている非ニュートンモデル

Src¥transportModels¥incompressible¥viscosityModels内

```
5
(
BirdCarreau
CrossPowerLaw
HerschelBulkley
Newtonian
powerLaw
)
```

BirdCarreau $$\eta = \eta_\infty + \left(\eta_0 - \eta_\infty\right)\left\{1.0 + \left(k\dot{\gamma}\right)^2\right\}^{\frac{n-1.0}{2.0}}$$

CrossPowerLaw $$\eta = \frac{\left(\eta_0 - \eta_\infty\right)}{1.0 + \left(m\dot{\gamma}\right)^n} + \eta_\infty$$

HerschelBulkley $$\eta = \min\left(\eta_0, \frac{\tau_0}{\dot{\gamma}} + k\dot{\gamma}^{n-1}\right)$$

Newtonian $$\eta = \eta_0$$

powerLaw $$\eta = \max\left(\eta_{\min}, \min\left(\eta_{\max}, k\dot{\gamma}^{n-1}\right)\right)$$

transportProperties

Newtonian $\eta = \eta_0$

```
nu              nu [ 0 2 -1 0 0 0 0 ] 1;
```

CrossPowerLaw $\eta = \frac{(\eta_0 - \eta_\infty)}{1.0 + (m\dot{\gamma})^n} + \eta_\infty$

```
CrossPowerLawCoeffs
{
    nu0         nu0 [ 0 2 -1 0 0 0 0 ] 10000;
    nuInf       nuInf [ 0 2 -1 0 0 0 0 ] 1e-6;
    m           m [ 0 0 1 0 0 0 0 ] 1.0;
    n           n [ 0 0 0 0 0 0 0 ] 0.6;
}
```

BirdCarreau $\eta = \eta_\infty + (\eta_0 - \eta_\infty)\left\{1.0 + (k\dot{\gamma})^2\right\}^{\frac{n-1.0}{2.0}}$

```
BirdCarreauCoeffs
{
    nu0         nu0 [ 0 2 -1 0 0 0 0 ] 10000;
    nuInf       nuInf [ 0 2 -1 0 0 0 0 ] 1e-06;
    k           k [ 0 0 1 0 0 0 0 ] 1;
    n           n [ 0 0 0 0 0 0 0 ] 0.4;
}
```

HerschelBulkley $\eta = \min\left(\eta_0, \frac{\tau_0}{\dot{\gamma}} + k\dot{\gamma}^{n-1}\right)$

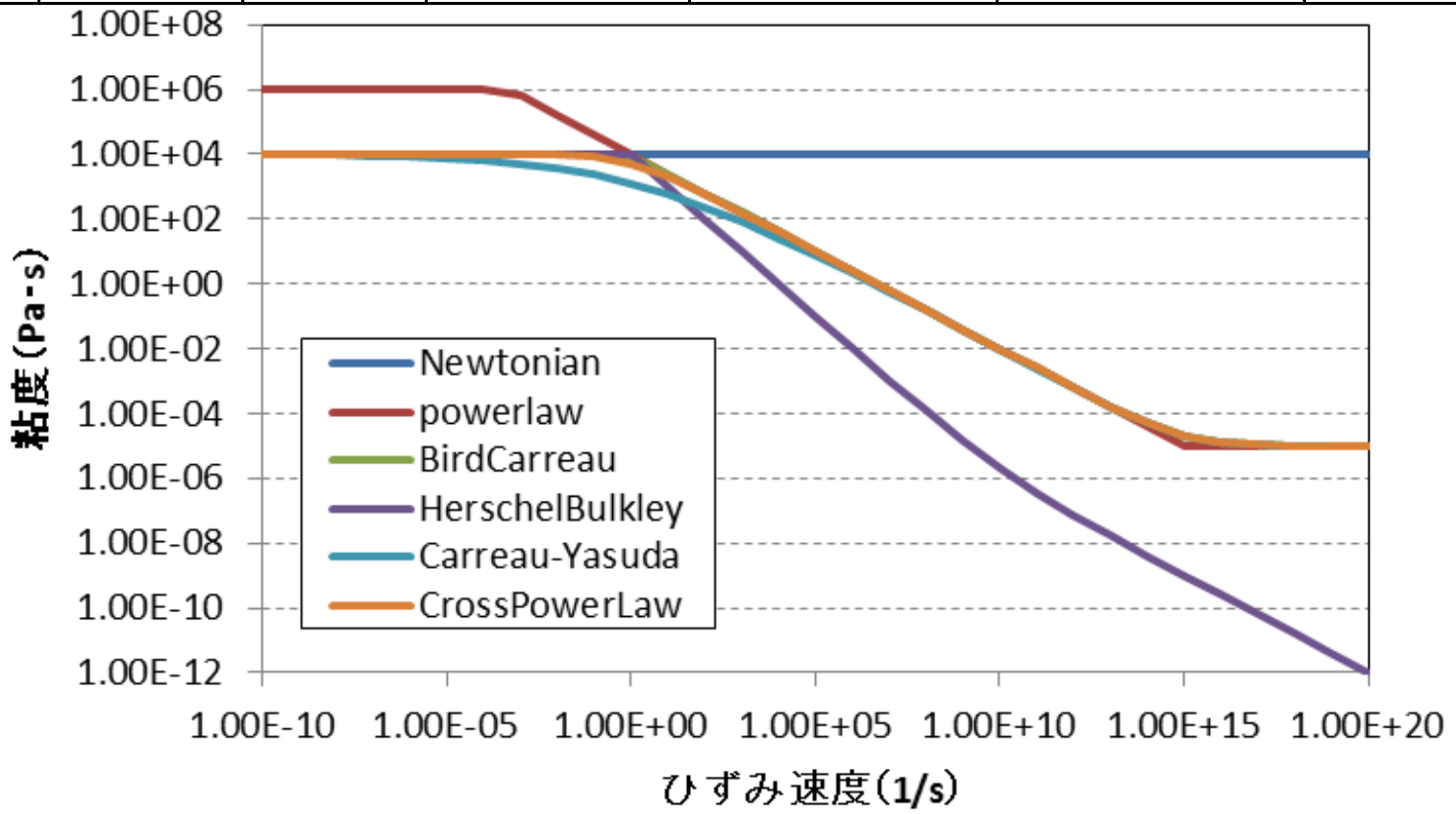
```
HerschelBulkleyCoeffs
{
nu0             nu0 [0 2 -1 0 0 0 0] 10000;
tau0            tau0 [0 2 -2 0 0 0 0] 10000;
k               k [0 2 -1 0 0 0 0] 1;
n               n [0 0 0 0 0 0 0] 0.4;
}
```

powerLaw $\eta = \max\left(\eta_{min}, \min\left(\eta_{max}, k\dot{\gamma}^{n-1}\right)\right)$

```
powerLawCoeffs
{
 k              k [ 0 2 -1 0 0 0 0 ] 10000;
 n              n [ 0 0 0 0 0 0 0 ] 0.4;
 nuMin          nuMin [ 0 2 -1 0 0 0 0 ] 1e-08;
 nuMax          nuMax [ 0 2 -1 0 0 0 0 ] 1e8;
}
```

非ニュートンモデルの比較

	Newtonian	powerlaw	BirdCarreau	HerschelBulkley	Carreau-Yasuda	CrossPowerLaw
η 0	1.00E+04	1.00E+06	1.00E+04	1.00E+04	1.00E+04	1.00E+04
η ∞		1.00E-05	1.00E-05		1.00E-05	1.00E-05
τ 0				1.00E+04		
k		10000	1	1	1	
m						1
n		0.4	0.4	0.4	0.4	0.6
a					0.2	



他の非ニュートンモデル

Ellis model

$$\eta = \frac{\eta_0}{1.0 + \left(\frac{\tau}{\tau_{1/2}} \right)^n}$$

Cross model

$$\eta = \frac{\eta_0}{1.0 + (m\dot{\gamma})^n}$$

Powell-Eyring model (PowellEyring)

$$\eta = \eta_0 + k\tau_0 \frac{\sinh^{-1} k\dot{\gamma}}{k\dot{\gamma}}$$

Sisko model (Sisko)

$$\eta = \eta_\infty + \eta_0 \left|\dot{\gamma}\right|^{n-1}$$

Casson model (Casson)

$$\sqrt{\eta} = \sqrt{\frac{\tau_y}{\dot{\gamma}}} + \sqrt{m}$$

Bingham model (Bingham)

$$\eta = \frac{\tau_y}{\dot{\gamma}} + \eta_0$$

Fluent HerschelBulkley (FluentHerschelBulkley)

$$\dot{\gamma} > \dot{\gamma}_c : \eta = \frac{\tau_y}{\dot{\gamma}} + k\left(\frac{\dot{\gamma}}{\dot{\gamma}_c} \right)^{n-1}$$

$$\dot{\gamma} < \dot{\gamma}_c : \eta = \tau_0 \frac{\left(2 - \frac{\dot{\gamma}}{\dot{\gamma}_c} \right)}{\dot{\gamma}_c} + k\left[(2-n) + (n-1)\frac{\dot{\gamma}}{\dot{\gamma}_c} \right]$$

CFX HerschelBulkley (CFXHerschelBulkley)

$$\eta = \frac{\tau_y}{\lambda\dot{\gamma}} + k(\lambda\dot{\gamma})^{n-1}$$

CFX Ostwald de Waele (CFXpowerLaw)

$$\eta = k(\lambda\dot{\gamma})^{n-1}$$

ひずみ速度の出力

applications¥solvers¥incompressible¥nonNewtonianIcoFoam内

createFields.H内に下記を追加

```
volScalarField strRatio
 (
    IOobject
    (
        "strRatio",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    mesh,
    dimensionedScalar("strRatio", dimensionSet(0,0,-1,0,0,0,0),
scalar(0.0))
 );
```

nonNewtonianIcoFoam.C内に下記を追加

```
strRatio = Foam::sqrt(2.0)*mag(symm(fvc::grad(U)));
runTime.write();
```

※runTime.write()の直前に追加する

同様に記述することで他のソルバー
simpleFoam
interFoam
buoyantBoussinesqSimpleFoam
等でも可能

商用ソフトではデフォルトで出力可能

非ニュートンモデル（CrossLaw）の作成

似たプログラムを流用

Src¥transportModels¥incompressible¥viscosityModels内のCrossPowerLawをコピー
名前をCrossPowerLawからCrossLawに変更（フォルダ、ファイル名）

プログラムのカスタマイズ

CrossLaw.Cのファイル内
粘性を計算しているところのnuInf_を削除（CrossLawの式に書き換える）

{　return (nu0_)/(scalar(1) + pow(m_*strainRate(), n_)) ;}　CrossLaw $\eta = \dfrac{(\eta_0)}{1.0 + (m\dot{\gamma})^n}$

CrossPowerLawをCrossLawに置き換え

CrossLaw.Hファイル内
CrossPowerLawをCrossLawに置き換え
dimensionedScalar　nuInf_を削除

コンパイルの作法

Src¥transportModels¥incompressible¥Make内の
filesファイル内
viscosityModels/CrossLaw/CrossLaw.Cを追加

Src¥transportModelsで./Allmake

正しくコンパイルできれば
使用可能

```
6
(
BirdCarreau
CrossLaw          ←CrossLawが追加される
CrossPowerLaw
HerschelBulkley
Newtonian
powerLaw
)
```

商用ソフトの場合

STAR-CCM+

```
$nu0/(1.0+pow($m*$StrainRate,$n))
```

CrossLaw $\eta = \dfrac{(\eta_0)}{1.0 + (m\dot{\gamma})^n}$

FLUENT

```
#include "udf.h"

DEFINE_PROPERTY(CrossLaw, cell, thread)
{
        real nu0,m,n;
        real nu,StrainRate;

        StrainRate=C_STRAIN_RATE_MAG(cell,thread);

        nu0=1.0e5;
        m=1.0;
        n=1.0;

        nu=nu0/(1.0+pow(m*StrainRate,n));

return nu;

}
```

CrossLawの作成

OpenFOAMのデフォルト　　CrossPowerLaw　$\eta = \dfrac{(\eta_0 - \eta_\infty)}{1.0 + (m\dot{\gamma})^n} + \eta_\infty$

η_∞　の差

今回の組込　　CrossLaw　$\eta = \dfrac{(\eta_0)}{1.0 + (m\dot{\gamma})^n}$

CrossPowerLawから η_∞ をソースから削除する

ソース

CrossPowerLaw　return (nu0_ - nuInf_)/(scalar(1) + pow(m_*strainRate(), n_)) + nuInf_;

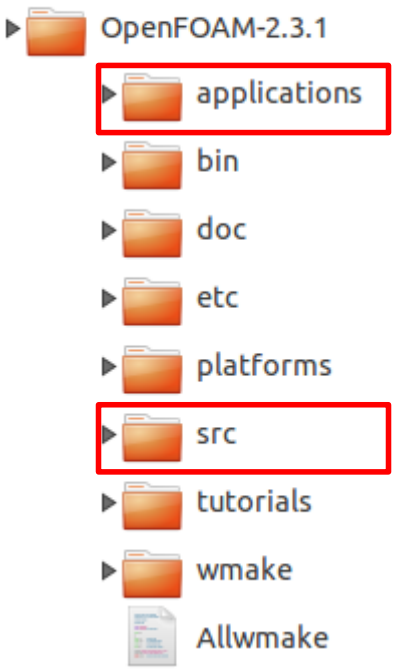
CrossLaw　return (nu0_)/(scalar(1) + pow(m_*strainRate(), n_)) ;

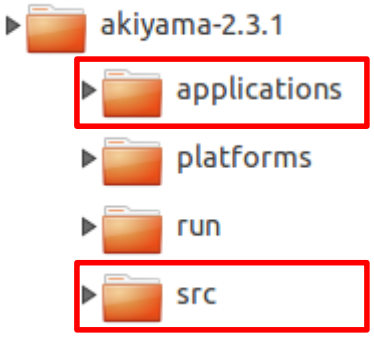
その他入出力の部分を削除する

カスタマイズ準備

OpenFOAMのユーザーディレクトリ以下にある実行ファイルが優先される

OpenFOAM-2.3.1内のapplicationsとsrcをuser-2.3.1にコピー

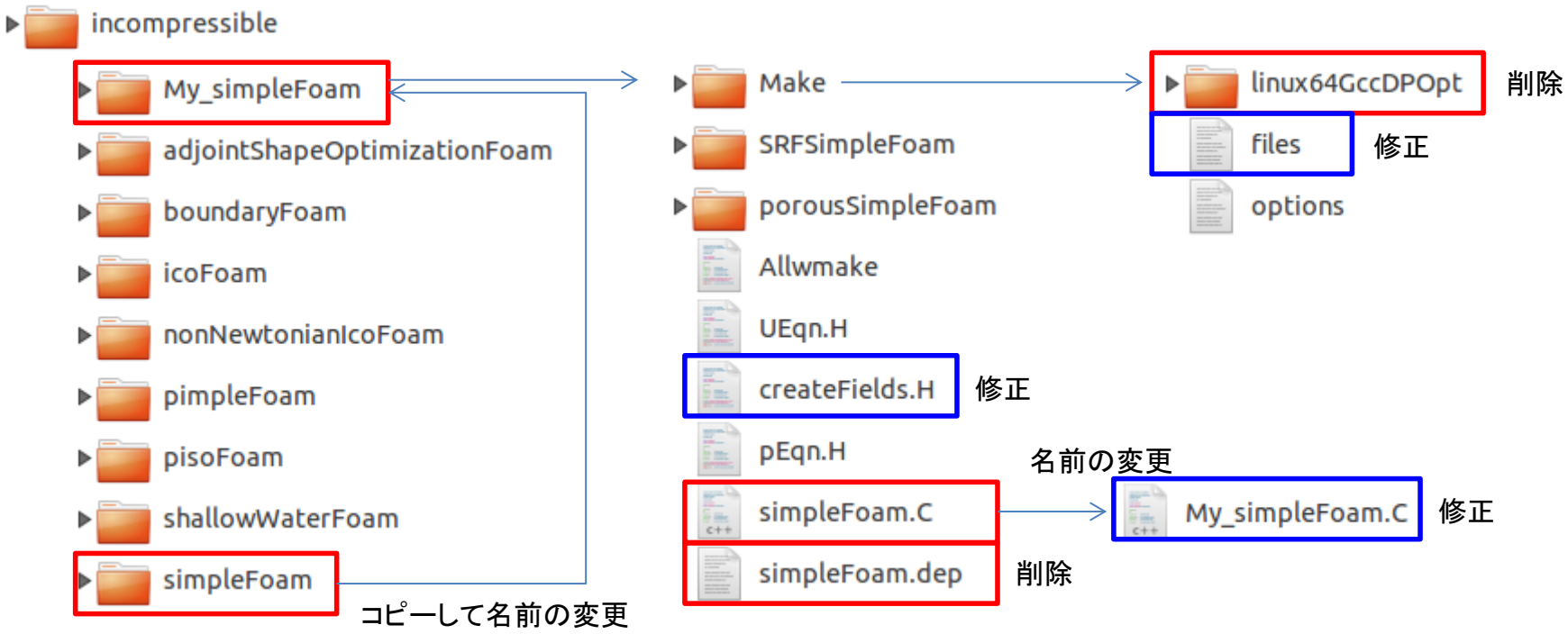




ユーザーディレクトリ

ひずみ速度の出力

~/OpenFOAM/akiyama-2.3.1/applications/solvers/incompressible/simpleFoam
をMy_simpleFoamとしてコピーする
同様にsimpleFoam.CをMy_simpleFoam.Cとファイル名を変更する



ファイル、フォルダーの削除または名前の変更

ファイル内容の修正

ひずみ速度の出力

createFields.H

ひずみ速度の出力変数を追加
（28行目に入力）

```
volScalarField strRatio
(
    IOobject
    (
        "strRatio",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    mesh,
    dimensionedScalar("strRatio", dimensionSet(0,0,-1,0,0,0,0),scalar(0.0))
);
```

```
15      Info<< "Reading field U\n" << endl;
16      volVectorField U
17      (
18          IOobject
19          (
20              "U",
21              runTime.timeName(),
22              mesh,
23              IOobject::MUST_READ,
24              IOobject::AUTO_WRITE
25          ),
26          mesh
27      );
28
29      #include "createPhi.H"
30
31
32      label pRefCell = 0;
33      scalar pRefValue = 0.0;
```

ひずみ速度の出力

files

simpleFoam.C

EXE = $(FOAM_APPBIN)/simpleFoam

My_simpleFoam.C

EXE = $(FOAM_USER_APPBIN)/My_simpleFoam

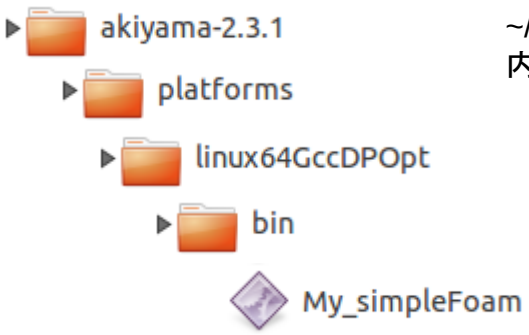
My_simpleFoam.C

66行目にひずみ速度の計算を追加
strRatio = Foam::sqrt(2.0)*mag(symm(fvc::grad(U)));

```
65          turbulence->correct();
66
67          runTime.write();
```

```
65          turbulence->correct();
66          strRatio = Foam::sqrt(2.0)*mag(symm(fvc::grad(U)));
67          runTime.write();
```

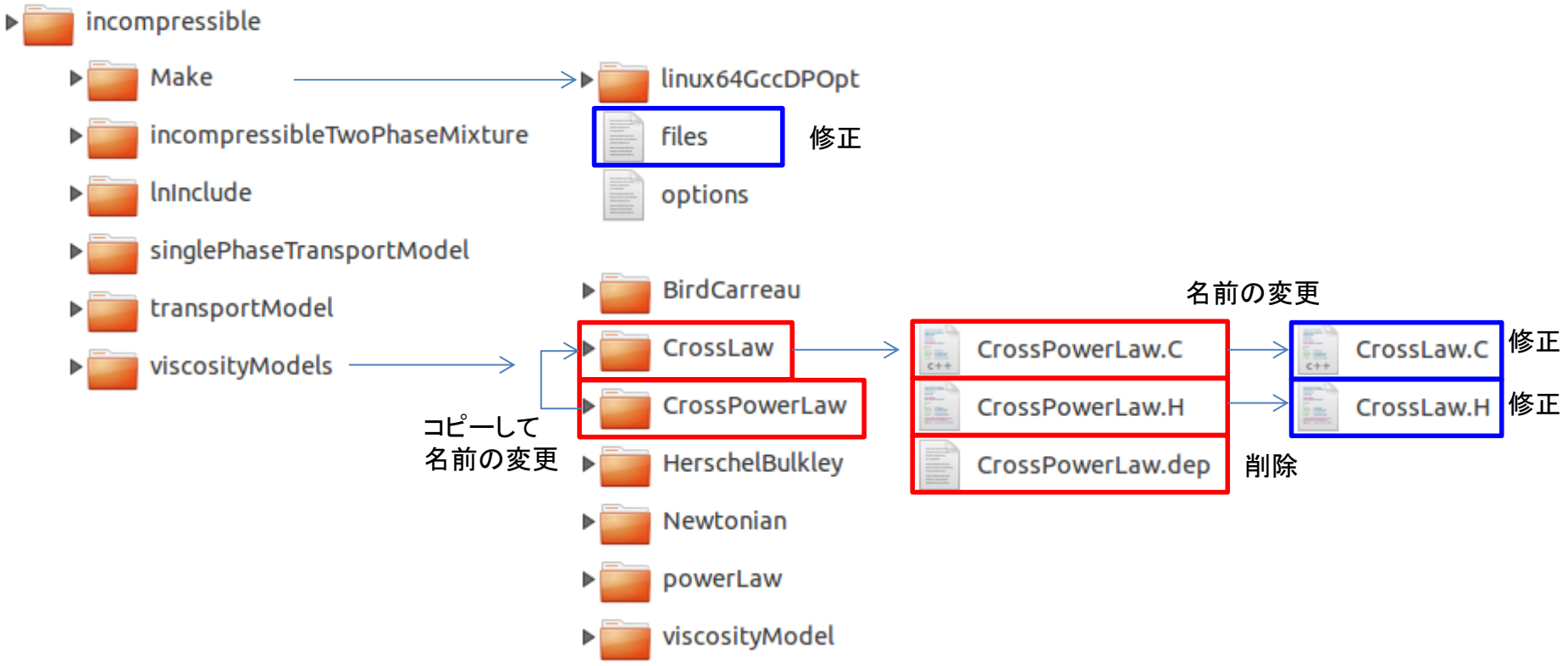
My_simpleFoamディレクトリで
$wmake
を実行



エラーがなければ
~/OpenFOAM/akiyama-2.3.1/platforms/linux64GccDPOpt/bin
内にMy_simpleFoamができる

CrossLawの作成

~/OpenFOAM/akiyama-2.3.1/src/transportModels/incompressible/viscosityModels/CrossPowerLaw
をCrossLawとしてコピーする
同様に内部のファイルもCrossLawとする



ファイル、フォルダーの削除または名前の変更

ファイル内容の修正

CrossLawの作成

files

5行目
viscosityModels/CrossPowerLaw/CrossPowerLaw.C
をコピーし6行目に貼り付け

viscosityModels/CrossPowerLaw/CrossPowerLaw.C

15行目
LIB = $(FOAM_LIBBIN)/libincompressibleTransportModels

6行目

viscosityModels/CrossLaw/CrossLaw.C

15行目
LIB = $(FOAM_USER_LIBBIN)/libincompressibleTransportModels

CrossLaw.C

CrossLaw.H

検索>置換でファイル内の
CrossPowerLawをCrossLawに置換

~/OpenFOAM/akiyama-2.3.1/src/transportModels/incompressible

incompressibleディレクトリで
$wmake
を実行

エラーがなければ
~/OpenFOAM/akiyama-2.3.1/platforms/linux64GccDPOpt/lib
内にlibincompressibleTransportModels.so

libincompressibleTransportModels.so

```
6
(
BirdCarreau
CrossLaw
CrossPowerLaw
HerschelBulkley
Newtonian
powerLaw
)
```

CrossLawが追加される

CrossLawの作成

CrossLaw.H

62行目
dimensionedScalar nuInf_;
削除またはコメントアウト

CrossLaw.C

53行目
return (nu0_ - nuInf_)/(scalar(1) + pow(m_*strainRate(), n_)) + nuInf_;
を目的の式に書き換える

70行目
nuInf_(CrossLawCoeffs_.lookup("nuInf")),
削除またはコメントアウト

100行目
CrossLawCoeffs_.lookup("nuInf") >> nuInf_;
削除またはコメントアウト

目的の式の書き換えと必要な変数の定義を行う

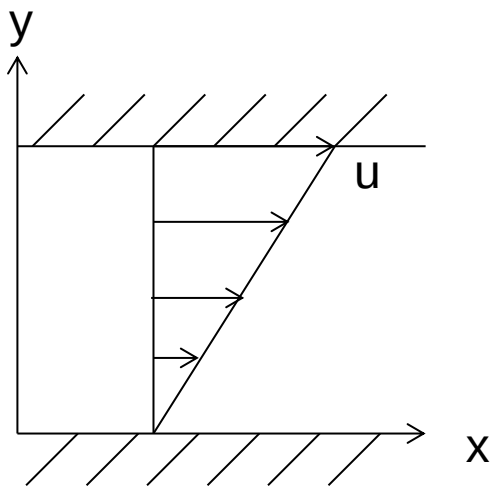
CrossLaw $\eta = \dfrac{(\eta_0)}{1.0 + (m\dot{\gamma})^n}$

return (nu0_)/(scalar(1) + pow(m_*strainRate(), n_));

必要な変数の定義

incompressibleディレクトリで
$wmake
を実行

検証問題（クエット流れ）



2枚の平行な板
下面は固定
上面は速度uでx方向に動いている

流体の速度は
X=0でU=0
X=YでU=u
Y方向に対してx方向流速は線形に増加する

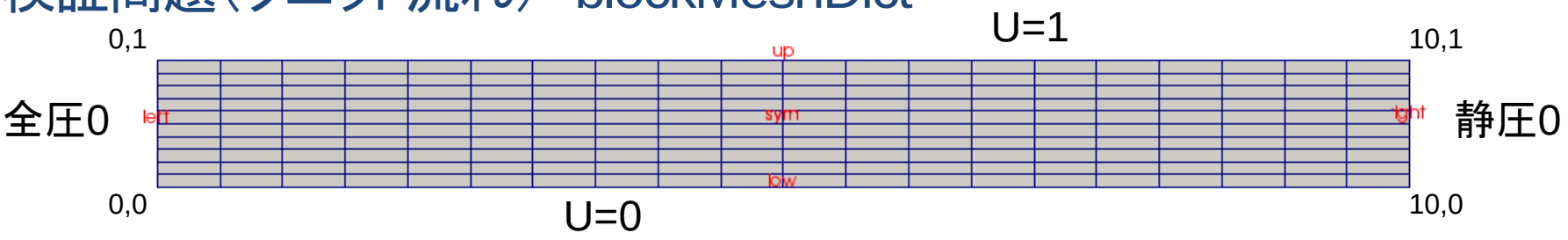
壁面せん断応力

$$\tau = \eta(\dot{\gamma})\dot{\gamma} = \eta(\dot{\gamma})\left(-\frac{du}{dy}\right)$$

ひずみ速度はy方向に対するx方向流速の勾配（変化率）
→壁面せん断応力が求まる

壁面流速uを変化させたときの壁面せん断応力を理論値と比較することで組込んだ非ニュートンモデルの妥当性を検証することができる

検証問題（クエット流れ） blockMeshDict




```
convertToMeters 1;

vertices
(
    (0 0 -0.1)
    (10 0 -0.1)
    (10 1 -0.1)
    (0 1 -0.1)
    (0 0 0.1)
    (10 0 0.1)
    (10 1 0.1)
    (0 1 0.1)
);

blocks
(
    hex (0 1 2 3 4 5 6 7) (20 10 1) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    up
    {
        type wall;
        faces
        (
            (3 7 6 2)
        );
    }
    left
    {
        type patch;
        faces
        (
            (0 4 7 3)
        );
    }
    low
    {
        type wall;
        faces
        (
            (1 5 4 0)
        );
    }
    right
    {
        type patch;
        faces
        (
            (2 6 5 1)
        );
    }
    sym
    {
        type empty;
        faces
        (
            (0 3 2 1)
            (4 5 6 7)
        );
    }
);

mergePatchPairs
(
);
```

検証問題（クエット流れ） 0/U 0/p

U

```
dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    up
    {
        type            fixedValue;
        value           uniform (1 0 0);
    }
    left
    {
        type            zeroGradient;
    }
    low
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }
    right
    {
        type            zeroGradient;
    }
    sym
    {
        type            empty;
    }
}
```

p

```
dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    up
    {
        type            zeroGradient;
    }

    left
    {
        type            totalPressure;
        p0              uniform 0;
        U               U;
        phi             phi;
        rho             none;
        psi             none;
        gamma           1;
        value           uniform 0;
    }

    low
    {
        type            zeroGradient;
    }

    right
    {
        type            fixedValue;
        value           uniform 0;
    }

    sym
    {
        type            empty;
    }
}
```

検証問題（クエット流れ） constant

RASProperties

```
RASModel        laminar;

turbulence      off;

printCoeffs     off;
```

transportProperties

```
transportModel  powerLaw;

powerLawCoeffs
{
 k                k [ 0 2 -1 0 0 0 0 ] 10000;
 n                n [ 0 0 0 0 0 0 0 ] 0.4;
 nuMin            nuMin [ 0 2 -1 0 0 0 0 ] 1e-08;
 nuMax            nuMax [ 0 2 -1 0 0 0 0 ] 1e8;
}
```

検証問題（クエット流れ） system

controlDict

```
application     simpleFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         10000;

deltaT          1;

writeControl    timeStep;

writeInterval   1000;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;
```

fvSchemes

```
ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         Gauss linear;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss upwind;
    div(phi,k)      bounded Gauss upwind;
    div(phi,epsilon) bounded Gauss upwind;
    div(phi,R)      bounded Gauss upwind;
    div(R)          Gauss linear;
    div(phi,nuTilda) bounded Gauss upwind;
    div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}
```

検証問題（クエット流れ） system

fvSolution

```
solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-06;
        relTol          0.1;
        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;
        cacheAgglomeration on;
        agglomerator    faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels     1;
    }

    "(U|k|epsilon|R|nuTilda)"
    {
        solver          smoothSolver;
        smoother        symGaussSeidel;
        tolerance       1e-05;
        relTol          0.1;
    }
}
```

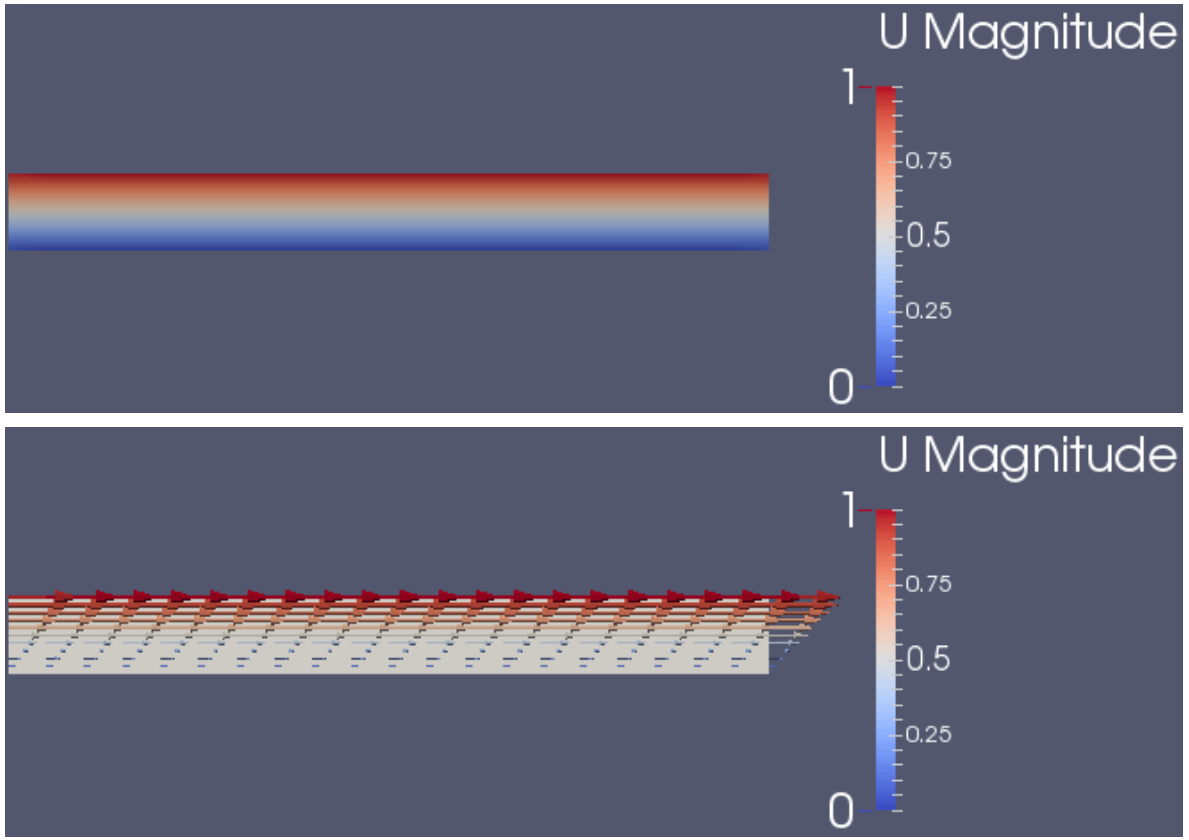
```
SIMPLE
{
    nNonOrthogonalCorrectors 0;

    residualControl
    {
        p               1e-6;
        U               1e-6;
        "(k|epsilon|omega)" 1e-6;
    }
}

relaxationFactors
{
    fields
    {
        p               0.3;
    }
    equations
    {
        U               0.7;
        k               0.7;
        epsilon         0.7;
        R               0.7;
        nuTilda         0.7;
    }
}
```

検証問題（クエット流れ）

ターミナル入力コマンド
blockMesh
simpleFoam
wallShearStress
paraFoam

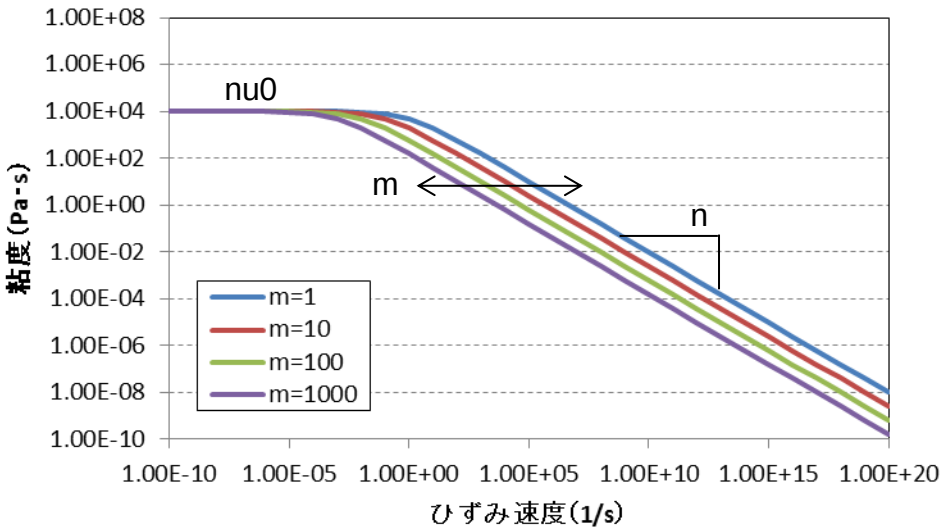


CrossLaw検証結果

CrossLaw $$\eta = \frac{(\eta_0)}{1.0 + (m\dot{\gamma})^n}$$

```
transportModel  CrossLaw;

CrossLawCoeffs
{
    nu0             nu0 [ 0 2 -1 0 0 0 0 ] 1e4;
    m               m [ 0 0 1 0 0 0 0 ] 1;
    n               n [ 0 0 0 0 0 0 0 ] 0.6;
}
```



各変数の意味

物性の変化位置とその前後でひずみ速度(壁面流速)を与える

ひずみ速度(流速)		粘度		壁面せん断応力	
入力	解析結果	解析結果	理論値	解析結果	理論値
1.00E-06	1.00E-06	9997.49	9997.489	0.009997	0.009997
1.00E-02	0.01	9406.49	9406.491	94.0649	94.06491
1	1	5000	5000	5000	5000
1.00E+02	100.3	592.506	592.5068	59428.4	59428.43
1.00E+06	1.94E+07	0.423717	0.423717	8227190	8227195

物性値は各モデルで必要な範囲を設定する
時間があれば物性値を変更して結果への影響を見る

検証方法

- 組込んだ非ニュートンモデルを用いてひずみ速度を与えて壁面せん断応力を解析から求める
  - 領域の高さを1とする
  - 粘度の変化点前後のひずみ速度を入力して壁面せん断応力を求める
  - ひずみ速度の入力は流速で変化させる

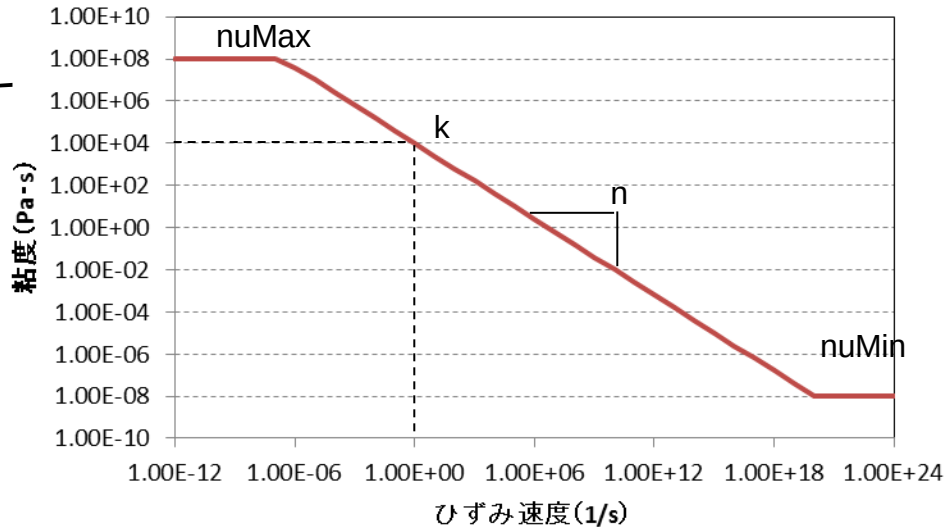
$$\dot{\gamma} = \frac{U}{Y}$$

  - 壁面せん断応力の誤差を小さくするには残差を小さくする（1e-6程度）
  - 余裕があればメッシュ感度も確認する
- 非ニュートンモデルにひずみ速度を入力して粘度、壁面せん断応力の理論値を算出し、解析結果（ひずみ速度、粘度、壁面せん断応力）と比較する

$$\tau = \eta(\dot{\gamma})\dot{\gamma} = \eta(\dot{\gamma})\left(-\frac{du}{dy}\right)$$

- グラフを作成して各パラメータの値を示す

powerLaw $\eta = \max\left(\eta_{\min}, \min\left(\eta_{\max}, k\dot{\gamma}^{n-1}\right)\right)$



- 結果のまとめと解説書を作成する

スケジュール

- 4月
  - 非ニュートンモデルの解説
- 5月
  - チュートリアル及び検証問題の実施
- 6月

- 7月
  - 非ニュートンモデルの組込（OpenFOAMカスタマイズ）解説
- 8月
  - 各自の進捗報告（非ニュートンモデルの組込）
- 9月
  - 各自の進捗報告（検証問題の報告）
- 10月
  - まとめ
- 11月
  - CAEシンポジウムにて発表の後公開

参考文献

- 流体力学　非圧縮性流体の流れ学,中山司著,森北出版
- 非ニュートン流体力学,中村喜代次著,コロナ出版
- エクセルによる樹脂流動解析,吉川秀雄著,東京図書出版
- Polymer Extrusion,Chris Rauwendaal, Hanser Gardner Pubns
- Principles of polymer processing, Zehev Tadmor Costas G. Gogos, Wiley-Interscience

- 参考資料
- 下記資料を参考にして組み込みの実施
  - http://opencae.gifu-nct.ac.jp/pukiwiki/index.php?%C2%E8%A3%B1%A3%B1%B2%F3%CA%D9%B6%AF%B2%F1%A1%A7H240204
  - http://www.tfd.chalmers.se/~hani/kurser/OS_CFD/Naser_Hamedi/Documents/Report.pdf#search='NonNewtonian+Models+in+OpenFoam+implementation+of+a+nonNewtonian+model‘