

```

/*-----*/
=====
\\      /   F i e l d       |   OpenFOAM: The Open Source CFD Toolbox
\\     /    O p e r a t i o n |   Copyright (C) 2011-2013 OpenFOAM Foundation
\\    /     A n d             |
\\   /      M a n i p u l a t i o n |
-----

```

License

This file is part of OpenFOAM.

OpenFOAM is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

OpenFOAM is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with OpenFOAM. If not, see <http://www.gnu.org/licenses/>.

Application
icoFoam

my_icoFoam に書き換える

Description

Transient solver for incompressible, laminar flow of Newtonian fluids.

```

/*-----*/

```

```

#include "fvCFD.H"

```

```

// * * * * *

```

```

int main(int argc, char *argv[])
{

```

```

    #include "setRootCase.H"

    #include "createTime.H"
    #include "createMesh.H"
    #include "createFields.H"
    #include "initContinuityErrs.H"

```

```

// * * * * *

```

```

Info<< "\nStarting time loop\n" << endl;

```

```

while (runTime.loop())
{
    Info<< "Time = " << runTime.timeName() << nl << endl;

```

```

    #include "readPISOControls.H"
    #include "CourantNo.H"

```

```

    fvVectorMatrix UEqn
    (
        fvm::ddt(U)
        + fvm::div(phi, U)
        - fvm::laplacian(nu, U)
    );

    solve(UEqn == -fvc::grad(p));

```

U の式

```

// --- PISO loop
for (int corr=0; corr<nCorr; corr++)
{
    volScalarField rAU(1.0/UEqn.A());

    volVectorField HbyA("HbyA", U);
    HbyA = rAU*UEqn.H();
    surfaceScalarField phiHbyA
    (
        "phiHbyA",
        (fvc::interpolate(HbyA) & mesh.Sf())
        + fvc::interpolate(rAU)*fvc::ddtCorr(U, phi)
    );

    adjustPhi(phiHbyA, U, p);

    for (int nonOrth=0; nonOrth<=nNonOrthCorr; nonOrth++)
    {
        fvScalarMatrix pEqn
        (
            fvm::laplacian(rAU, p) == fvc::div(phiHbyA)
        );

        pEqn.setReference(pRefCell, pRefValue);
        pEqn.solve();

        if (nonOrth == nNonOrthCorr)
        {
            phi = phiHbyA - pEqn.flux();
        }

        #include "continuityErrs.H"

        U = HbyA - rAU*fvc::grad(p);
        U.correctBoundaryConditions();
    }

    fvScalarMatrix TEqn
    (
        fvm::ddt(T)
        + fvm::div(phi, T)
        - fvm::laplacian(DT, T)
    );

    TEqn.solve();

    runTime.write();

    Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
        << " ClockTime = " << runTime.elapsedClockTime() << " s"
        << nl << endl;
}

Info<< "End\n" << endl;

return 0;
}

// *****

```

PISO ループ
補正

T の式を追加

```

fvScalarMatrix TEqn
(
    fvm::ddt(T)
    + fvm::div(phi, T)
    - fvm::laplacian(DT, T)
);

TEqn.solve();

```

```
Info<< "Reading transportProperties\n" << endl;
```

```
IOdictionary transportProperties
(
    IOobject
    (
        "transportProperties",
        runTime.constant(),
        mesh,
        IOobject::MUST_READ_IF_MODIFIED,
        IOobject::NO_WRITE
    )
);
```

DT の定義を追加

```
dimensionedScalar nu
(
    transportProperties.lookup("nu")
);
```

```
dimensionedScalar DT
(
    transportProperties.lookup("DT")
);
```

```
Info<< "Reading field p\n" << endl;
volScalarField p
(
    IOobject
    (
        "p",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

T の定義を追加

```
Info<< "Reading field T\n" << endl;
volScalarField T
(
    IOobject
    (
        "T",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

```
Info<< "Reading field U\n" << endl;
volVectorField U
(
    IOobject
    (
        "U",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

```
# include "createPhi.H"
```

```
label pRefCell = 0;
scalar pRefValue = 0.0;
setRefCell(p, mesh.solutionDict().subDict("PISO"), pRefCell, pRefValue);
```

```
/*-----*- C++ -*-----*\
|=====|
| \\      / F i e l d      |   OpenFOAM: The Open Source CFD Toolbox
| \\      / O p e r a t i o n |   Version: 2.3.0
| \\      / A n d           |   Web:      www.OpenFOAM.org
| \\      / M a n i p u l a t i o n |
|-----*\
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "constant";
    object       transportProperties;
}
// ***** //

nu          nu [ 0 2 -1 0 0 0 ] 0.01;

DT          DT [ 0 2 -1 0 0 0 ] 0.002;

// ***** //
```

Tの拡散係数 (熱拡散率) を追加

```
/*-----*- C++ -*-----*/
|=====|
| \\      / F i e l d      | OpenFOAM: The Open Source CFD Toolbox
|  \\    / O p e r a t i o n | Version: 2.3.0
|   \\  / A n d              | Web:      www.OpenFOAM.org
|    \\ / M a n i p u l a t i o n |
|-----|
/*-----*/
FoamFile
{
    version      2.0;
    format       ascii;
    class        volScalarField;
    object       T;
}
// *****

dimensions      [0 0 0 1 0 0 0];

internalField   uniform 300;

boundaryField
{
    movingWall
    {
        type      fixedValue;
        value      uniform 350;
    }

    fixedWalls
    {
        type      fixedValue;
        value      uniform 300;
    }

    frontAndBack
    {
        type      empty;
    }
}
// *****
```

```
/*-----*- C++ -*-----*/
|=====|
| \\ \\ / | F i e l d | OpenFOAM: The Open Source CFD Toolbox
| \\ \\ / | O p e r a t i o n | Version: 2.3.0
| \\ \\ / | A n d | Web: www.OpenFOAM.org
| \\ \\ / | M a n i p u l a t i o n |
|-----|
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "system";
    object       fvSchemes;
}
// ***** //

ddtSchemes
{
    default      Euler;
}

gradSchemes
{
    default      Gauss linear;
    grad(p)      Gauss linear;
}

divSchemes
{
    default      none;
    div(phi,U)   Gauss linear;
    div(phi,T)   Gauss upwind; //注意：コンマと変数の間には空白なし
}

laplacianSchemes
{
    default      Gauss linear orthogonal;
}

interpolationSchemes
{
    default      linear;
}

snGradSchemes
{
    default      orthogonal;
}

fluxRequired
{
    default      no;
    p            ;
}

// ***** //
```

```
/*----- C++ -----*/
|=====|
| \ \ / / | F i e l d | OpenFOAM: The Open Source CFD Toolbox
| \ \ / / | O p e r a t i o n | Version: 2.3.0
| \ \ / / | A n d | Web: www.OpenFOAM.org
| \ \ / / | M a n i p u l a t i o n |
|-----|
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    location     "system";
    object       fvSolution;
}
// ***** //

solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }

    U
    {
        solver          smoothSolver;
        smoother        symGaussSeidel;
        tolerance       1e-05;
        relTol          0;
    }
}

PISO
{
    nCorrectors      2;
    nNonOrthogonalCorrectors 0;
    pRefCell         0;
    pRefValue        0;
}

// ***** //
```

T の解法設定を追加

```
T
{
    solver          BICCG;
    preconditioner  DILU;
    tolerance       1e-7;
    relTol          0;
};
```

