Implementing the pimpleFoam to oscillating flow solver porousOsciPimpleFoam using volume-averaged kOmega turbulence model

Yanyan Zhai

Applied Mechanics/Fluid Dynamics,
Technical University of Denmark,
Copenhagen, Denmark

2019-11-28

Outline

- Theory Background
- porosityZones Library
- solvers
- Turbulence model library
- Running the case

General flow equations

Continuity equation

$$\frac{\partial u_i}{\partial x_i} = 0$$

Momentum equation

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\nu(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}) + S_i$$

Flow equations in porous media

$$\frac{\partial \langle \overline{u}_i \rangle}{\partial x_i} = 0$$

$$(1 + C_m)\frac{\partial}{\partial t}\frac{\langle \overline{u}_i \rangle}{n} + \frac{1}{n}\frac{\partial}{\partial x_j}\frac{\langle \overline{u}_i \rangle \langle \overline{u}_i \rangle}{n} = -\frac{1}{\rho}\frac{\partial \langle p \rangle^f}{\partial x_i} + g_j + \frac{1}{n}\frac{\partial}{\partial x_j}\nu\left(\frac{\partial \langle \overline{u}_i \rangle}{\partial x_j} + \frac{\partial \langle \overline{u}_j \rangle}{\partial x_i}\right) + \frac{F_i + B_i}{\rho}$$

$$C_m = \gamma_p \frac{1-n}{n}$$

$$F_i = a\rho \langle \overline{u}_i \rangle + b\rho\sqrt{\langle \overline{u}_j \rangle \langle \overline{u}_j \rangle}\,\langle \overline{u}_i \rangle$$

Standard kOmega model

Turbulence Kinetic Energy

$$\rho\frac{\partial k}{\partial t}+\rho U_j\frac{\partial k}{\partial x_j}=\tau_{ij}\frac{\partial U_i}{\partial x_j}-\beta^*\rho k\omega+\frac{\partial}{\partial x_j}[(\mu+\sigma^*\mu_T)\frac{\partial k}{\partial x_j}]$$

Specific Dissipation Rate

$$\rho\frac{\partial \omega}{\partial t}+\rho U_j\frac{\partial \omega}{\partial x_j}=\alpha\frac{\omega}{k}\tau_{ij}\frac{\partial U_i}{\partial x_j}-\beta\rho\omega+\frac{\partial}{\partial x_j}[(\mu+\sigma\mu_T)\frac{\partial \omega}{\partial x_j}]$$

Closure Coefficients

$$\alpha=5/9 \quad \beta=3/40 \quad \beta^*=9/100 \quad \sigma=1/2 \quad \sigma^*=1/2$$

Auxiliary Relation

$$\epsilon=\beta^*\omega k \quad l=k^{\frac{1}{2}}/\omega$$

Eddy Viscosity

$$\mu_T=\rho k/\omega$$

Volume-averaged kOmega model

Turbulence Kinetic Energy

$$\frac{\partial \langle k \rangle}{\partial t} + \frac{1}{n}\frac{\partial \langle k \rangle \langle \overline{u_j} \rangle}{\partial x_j} = \frac{\partial}{\partial x_j}[(\nu + \langle \nu_t \rangle \sigma^*)\frac{\partial \langle k \rangle}{\partial x_j}] - \frac{\left\langle \overline{u_i' u_j'} \right\rangle}{n}\frac{\partial \langle \overline{u_i} \rangle}{\partial x_j} - \beta^* \langle \omega \rangle \langle k \rangle + n\beta^* \omega_\infty k_\infty$$

Specific Dissipation Rate

$$\frac{\partial \langle \omega \rangle}{\partial t} + \frac{1}{n}\frac{\partial \langle \omega \rangle \langle \overline{u_j} \rangle}{\partial x_j} = \frac{\partial}{\partial x_j}[(\nu + \langle \nu_t \rangle \sigma)\frac{\partial \langle \omega \rangle}{\partial x_j}] - \frac{\gamma \langle \omega \rangle}{n \langle k \rangle}\left\langle \overline{u_i' u_j'} \right\rangle \frac{\partial \langle \overline{u_i} \rangle}{\partial x_j} - \beta \langle \omega \rangle^2 + \beta \omega_\infty^2$$

$$k_\infty = 3.7\frac{1-n}{\sqrt{n}}(\langle \overline{u_1} \rangle^2 + \langle \overline{u_2} \rangle^2)$$

$$\omega_\infty = \frac{39.0}{3.7 D_{50}}\frac{(1-n)^{1.5}}{n^2}(\langle \overline{u_1} \rangle^2 + \langle \overline{u_2} \rangle^2)^{1/2}$$

porosityZones Library

The library is referred via a dictionary file named porosityZones in the tutorial as shown below.

```
porousmedias
    {
        coordinateSystem
        { e1  (1 0 0);
          e2  (0 1 0);
        }
resistanceFormulation vanGent1995;
        porosity 0.4;
        gammaAddedMass 0.34;
d50 d50 [0 1 0 0 0 0 0] 0.036;
alpha alpha [0 0 0 0 0 0 0] 500;
beta beta [0 0 0 0 0 0 0] 2.0;
    }
```

porosityZones Library

Add the viscous and inertial resistance force contribution to the momentum equation

```
void Foam::porosityZones::addResistance(fvVectorMatrix& UEqn)
{
    forAll(*this, i)
    {
        operator[](i).addResistance(UEqn);
    }
}
```

porosityZones Library

Porosity field will be written into 0 folder with the code in createPorosityFields.H

```
porosityZones pZones(mesh);
// Obtain the porosity field
tmp<volScalarField> tporosity(pZones.porosity());
volScalarField& porosity = tporosity();
// Writes the porosity field to starting time folder
porosity.write();
// Create a field with the porosity on the faces
// Interpolated from the centers.
surfaceScalarField porosityFace
(fvc::interpolate(pZones.porosity()()));
```

UEqn.H pimpleFoam

Constructing the original momentum eqaution.

```
tmp<fvVectorMatrix> UEqn
(
    fvm::ddt(U)
  + fvm::div(phi, U)
  + turbulence->divDevReff(U)
 ==
    fvOptions(U)
);
```

UEqn.H porousOsciPimpleFoam

```
tmp<fvVectorMatrix> UEqn
(
    pZones.ddt(U) //
  + 1.0/pZones.porosity()*fvm::div(phi/(porosityFace), U)
  + turbulence->divDevReff(U) // Turbulence added
 ==
fvOptions(U)  // Source term (not used) use via fvOptionsDict
  + bodyForcing->correct() // Source term due to body force
);
```

The following code is added between 'UEqn().relax();' and 'fvOptions.constrain(UEqn());'

```
pZones.addResistance(UEqn());
```

createFields.H porousOsciPimpleFoam

The following code is added after U field

```
volVectorField F
(   IOobject
    (
        "F",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE ),mesh,
dimensionedVector("F", dimensionSet(0,1,-2,0,0,0,0),
Foam::vector(0,0,0))
);
```

createFields.H porousOsciPimpleFoam

Creating new dictionary in constant related to body force source term

```
IOdictionary bodyForcingProperties
(
IOobject
(
"bodyForcingProperties",
runTime.constant(),
mesh,
IOobject::MUST_READ,
IOobject::NO_WRITE
)
);
```

createFields.H porousOsciPimpleFoam

Creating new sub-dictionary for bodyForcingProperties dictionary

```
autoPtr<Foam::bodyForcing> bodyForcing
(
Foam::bodyForcing::New
(
mesh,
mesh.lookupObject<IOdictionary>
("bodyForcingProperties").subDict("bodyForcing")
)
);
```

bodyForcingProperties porousOsciPimpleFoam

The dictionay in the case gives the basic information of oscillating flow

```
bodyForcing
{
    forceModel sinusoidalBodyForce;
    period 12;
    Um     2;
    constantSlope 0.000;
    forceDirection (1 0 0);
}
```

head file .C porousOsciPimpleFoam

Two new files are added to use the class porosityZones and bodyforcing.

```
#include "fvCFD.H"
#include "singlePhaseTransportModel.H"
#include "turbulenceModel.H"
#include "pimpleControl.H"
#include "fvIOoptionList.H"
#include "IOporosityModelList.H"
#include "IOMRFZoneList.H"
#include "porosityZones.H"
#include "bodyForcing.H"        //BF
```

main() in .C porousOsciPimpleFoam

In this part, the following code is added after #include "createMesh.H" to create the porosity field.

```
#include "createPorosityFields.H" // porosity field
```

The resistance induced by porous media is considered before "runTime.write();".

```
F = pZones.porousBodyForce(U);
// Calculating resistance from porosity
```

files in Make folder porousOsciPimpleFoam

files in porousOsciPimpleFoam

```
porousOsciPimpleFoam.C
EXE = $(FOAM_USER_APPBIN)/porousOsciPimpleFoam
```

files in pimpleFoam

```
pimpleFoam.C
EXE = $(FOAM_APPBIN)/pimpleFoam
```

options in Make folder porousOsciPimpleFoam

In EXE_INC, the following code is added in porousOsciPimpleFoam

```
-I./../bodyForce/lnInclude \
-DOFVERSION=222 \
-DEXTBRANCH=0 \
-DXVERSION=$(WAVES_XVERSION) \
-I$(WM_PROJECT_USER_DIR)/utilities/waves2Foam/src
/waves2FoamPorosity/lnInclude
```

In EXE_LIBS, the following lines are added.

```
-L$(FOAM_USER_LIBBIN) \
-lwaves2FoamPorosity \
-lbodyForce
```

H file porouskOmega

declare new fields in protected part

```
// Fields
volScalarField d50_;
volScalarField Ci_;
volScalarField porosity_;
surfaceScalarField porosityFace;
```

The 'kOmega' in this file is replced by 'porouskOmega'

C file porouskOmega

```
d50_
    (   IOobject
        (
            "d50",
            runTime_.timeName(),
            mesh_,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
       mesh_,
       dimensionedScalar("d50", dimLength, 0.036)
    ),
```

The 'kOmega' in this file is replced by 'porouskOmega'

C file porouskOmega

```
Ci_
    (   IOobject
        (
            "Ci",
            runTime_.timeName(),
            mesh_,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
       mesh_,
       dimensionedScalar("Ci", dimless, 1)
    ),
```

C file porouskOmega

```
  porosity_
    (   IOobject
        (   "porosity",
            runTime_.timeName(),
            mesh_,
            IOobject::MUST_READ,
            IOobject::AUTO_WRITE
        ),
       mesh_
    ),
porosityFace(fvc::interpolate(porosity_)),
```

C file porouskOmega

The additional terms induced by volume-averaging in dissipation rate and kinetic energy equations are declared and defined.

```
volScalarField omegai_(39/3.7*pow(Ci_-porosity_, 1.5)
/pow(porosity_, 2)/d50_*sqrt(U_ & U_));

volScalarField ki_(3.7*(Ci_-porosity_)
/pow(porosity_, 0.5)*(U_ & U_));
```

C file porouskOmega

```
tmp<fvScalarMatrix> omegaEqn
    (
        fvm::ddt(omega_)
      + 1.0/porosity_*fvm::div(phi_/porosityFace, omega_)
      - fvm::laplacian(DomegaEff(), omega_)
     ==
        1.0/porosity_*alpha_*G*omega_/k_
      - fvm::Sp(beta_*omega_, omega_)
      + beta_*omegai_*omegai_
    );
```

C file porouskOmega

```
tmp<fvScalarMatrix> kEqn
  (  fvm::ddt(k_)
    + 1.0/porosity_*fvm::div(phi_/porosityFace, k_)
    - fvm::laplacian(DkEff(), k_)
   ==
      1.0/porosity_*G //
    - fvm::Sp(Cmu_*omega_, k_)
    + porosity_*Cmu_*omegai_*ki_
  );
```

compared porousOsciPimpleFoam with porouskOmega

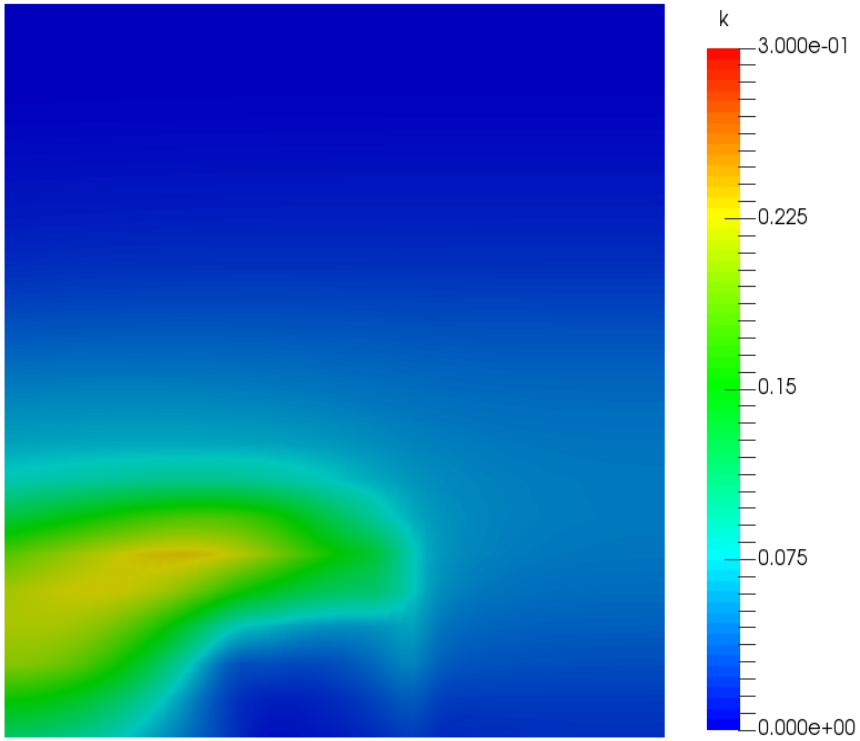

Figure: kOmega

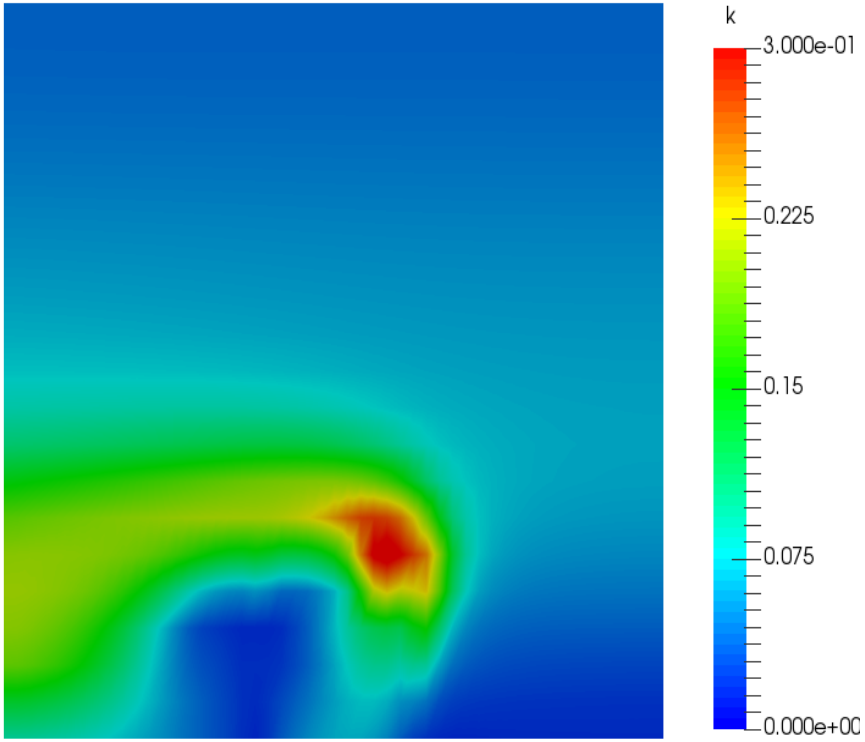

Figure: porouskOmega

porousOsciPimpleFoam with porouskOmega

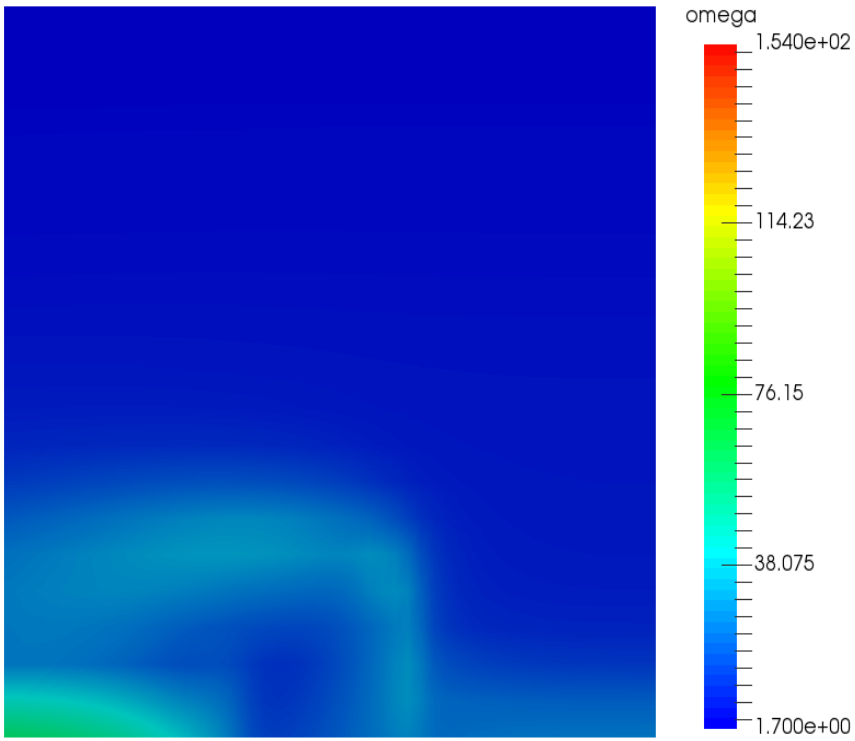

Figure: kOmega

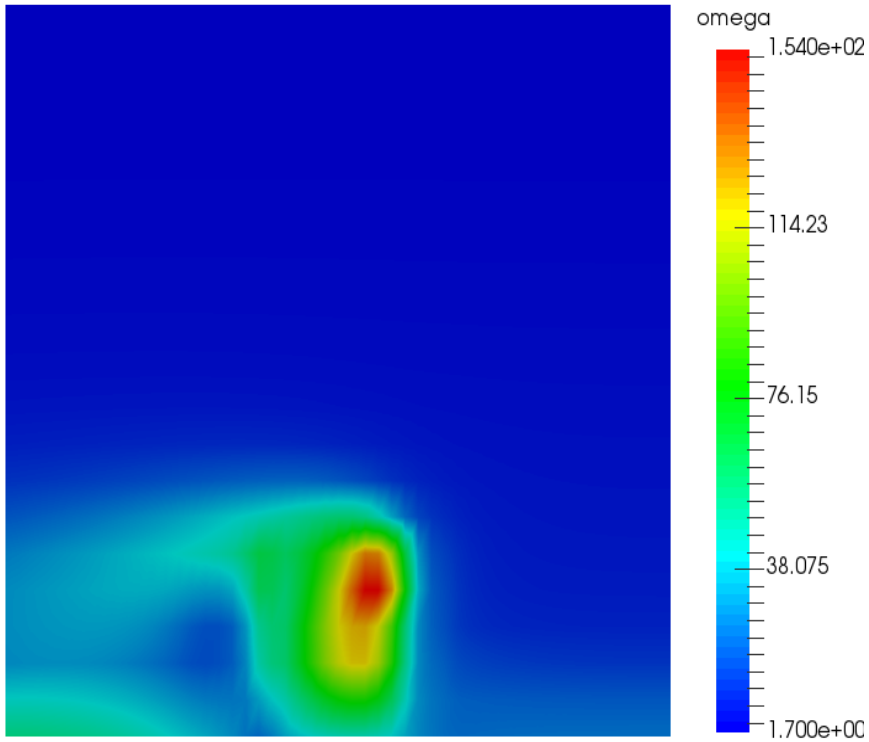

Figure: porouskOmega

porousOsciPimpleFoam with porouskOmega

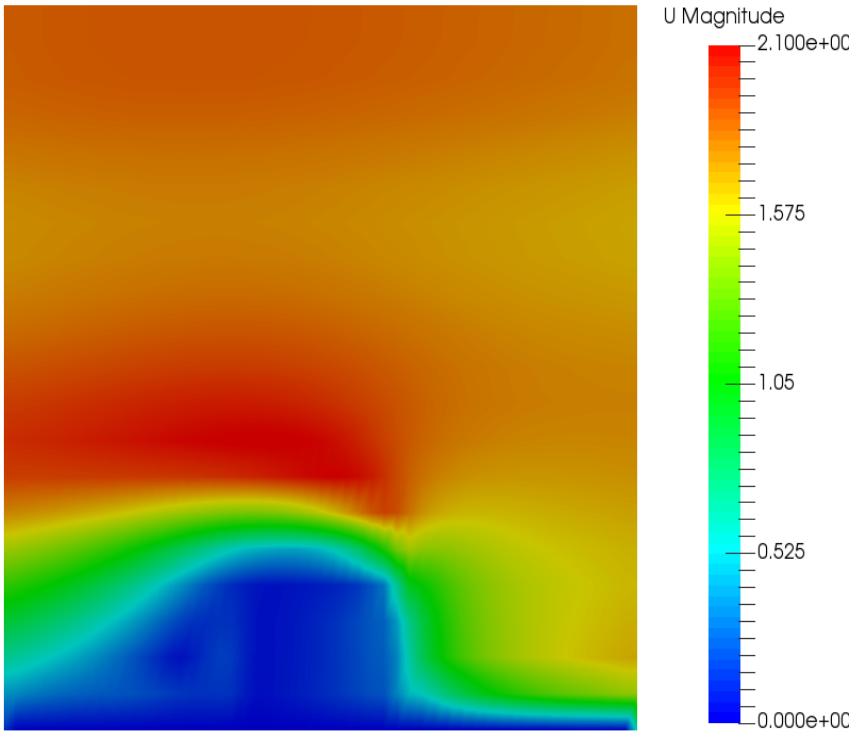

Figure: kOmega

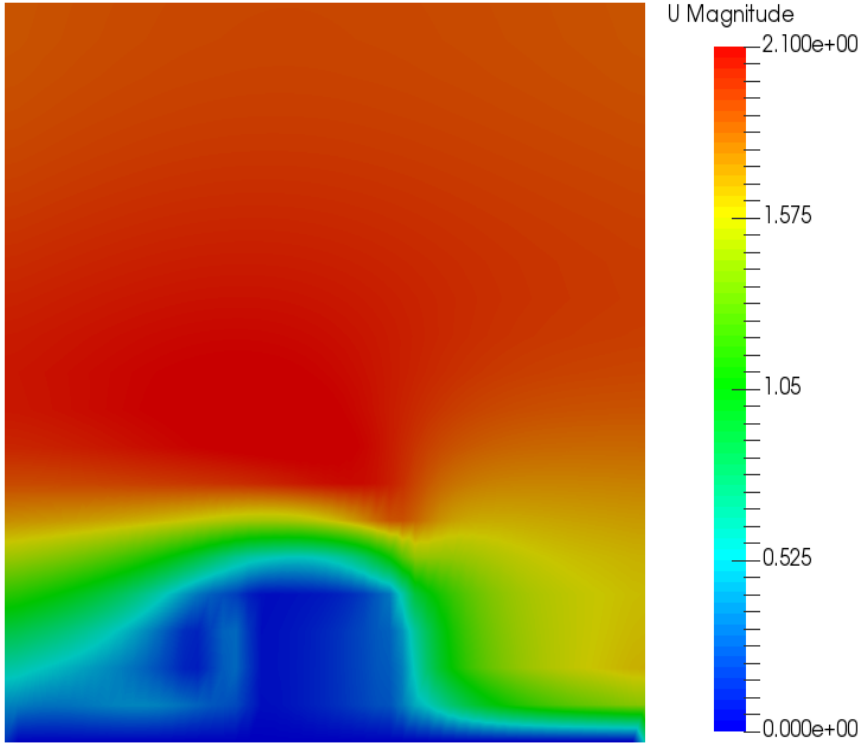

Figure: porouskOmega

Questions?

Good luck!