buoyantPimpleFoam解析

李东岳

1. 引言

buoyantPimpleFoam是OpenFOAM的传热求解器之一，其用于求解瞬态的、由于温度变化导致的密度变化、浮力驱动流动。相对于pimpleFoam，buoyantPimpleFoam的求解变量增加了能量e（或h）和密度ρ。求解思想和pimpleFoam大体相同。另外一个容易混淆的求解器为boussinesqBuoyantPimpleFoam，二者的区别在于后者采用Boussinesq进行假定，且仅仅求解温度方程。在实际应用中，boussinesqBuoyantPimpleFoam使用的范围存在限制。有关Boussinesq假定可以参考CFD中的能量方程。

OpenFOAM中并没有附加重力的恒密度单相流求解器。例如在恒密度的icoFoam和simpleFoam中，是压力差引起的流体流动。但在某些情况下，用户想要在不可压缩流体内添加重力场。考虑一个管道内的静止的液体，在进出口没有压差的情况下，附加重力或不附加重力预测的速度是相同的：液体均不会产生流动。但附加重力会引起压力自上而下的一个均匀增强的渐进变化。因此在附加重力的不可压缩求解器中，可以预测这种压力自顶而下越来越大的情况。buoyantPimpleFoam可以通过将beta（热膨胀系数）设置为常量，达到类似植入重力的pimpleFoam求解器。在这个情况下，重力引起均一的压力梯度变化，且无对流速度。

2. 控制方程与算法

首先有连续性方程：

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{U}) = 0 \tag{1}$$

动量方程：

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) + \nabla \cdot \tau = -\nabla p + \rho \mathbf{g} \tag{2}$$

其中ρ为流体的密度，$\mathbf{U}$为速度矢量，t表示时间，p表示压力，$\mathbf{g}$表示重力加速度。参考icoFoam解析中的方程(18)，对方程(2)半离散化之后有：

$$A_{\mathrm{P}} \mathbf{U}_{\mathrm{P}}^{r} + \sum A_{\mathrm{N}} \mathbf{U}_{\mathrm{N}}^{r} - S_{\mathrm{P}}^{n} = -\nabla p_{\mathrm{P}}^{n} + \rho_{\mathrm{P}}^{n} \mathbf{g} \tag{3}$$

$$\mathbf{U}_{\mathrm{P}}^{r} = \frac{1}{A_{\mathrm{P}}}\left(-\sum A_{\mathrm{N}} \mathbf{U}_{\mathrm{N}}^{r} + S_{\mathrm{P}}^{n}\right) - \frac{1}{A_{\mathrm{P}}}(\nabla p_{\mathrm{P}}^{n} - \rho_{\mathrm{P}}^{n} \mathbf{g}) \tag{4}$$

其中A_{P}，A_{N}为离散后的矩阵系数。令

$$\mathbf{HbyA}^{n} = \frac{1}{A_{\mathrm{P}}}\left(-\sum A_{\mathrm{N}} \mathbf{U}_{\mathrm{N}}^{r} + S_{\mathrm{P}}^{n}\right) \tag{5}$$

$$\mathbf{HbyA}_{\mathrm{P}}^{n} = \frac{1}{A_{\mathrm{P}}}\left(-\sum A_{\mathrm{N}}\mathbf{U}_{\mathrm{N}}^{r} + S_{\mathrm{P}}^{n}\right) \tag{5}$$

把方程(5)中代入到方程(4)中，有

$$\mathbf{U}_{\mathrm{P}}^{r} = \mathbf{HbyA}_{\mathrm{P}}^{n} - \frac{1}{A_{\mathrm{P}}}\left(\nabla p_{\mathrm{P}}^{n} - \rho_{\mathrm{P}}^{n}\mathbf{g}\right) \tag{6}$$

参考interFoam解析中的推导：$\nabla p_{\mathrm{P}}^{n} - \rho_{\mathrm{P}}^{n}\mathbf{g} = \nabla p_{\mathrm{rgh,P}}^{n} + \mathbf{gh}\nabla\rho_{\mathrm{P}}^{n}$，其中$\mathbf{h}$为位置矢量，有

$$\mathbf{U}_{\mathrm{P}}^{r} = \mathbf{HbyA}_{\mathrm{P}}^{n} - \frac{1}{A_{\mathrm{P}}}\left(\nabla p_{\mathrm{rgh,P}}^{n} + \mathbf{gh}\nabla\rho_{\mathrm{P}}^{n}\right) \tag{7}$$

在收敛的情况下有

$$\mathbf{U}_{\mathrm{P}}^{n+1} = \mathbf{HbyA}_{\mathrm{P}}^{n+1} - \frac{1}{A_{\mathrm{P}}}\left(\nabla p_{\mathrm{rgh,P}}^{n+1} + \mathbf{gh}\nabla\rho_{\mathrm{P}}^{n+1}\right) \tag{8}$$

把方程(8)代入到(1)，对任意网格单元（略去下标$_{\mathrm{P}}$）有

$$\frac{\partial\rho}{\partial t} + \nabla\cdot\left(\rho^{n+1}\left(\mathbf{HbyA}^{n+1} - \frac{1}{A}\left(\nabla p_{\mathrm{rgh}}^{n+1} + \mathbf{gh}\nabla\rho^{n+1}\right)\right)\right) = 0 \tag{9}$$

$$\frac{\partial\rho}{\partial t} + \nabla\cdot\left(\rho^{n+1}\mathbf{HbyA}^{n+1} - \frac{\rho^{n+1}}{A}\mathbf{gh}\nabla\rho^{n+1}\right) = \nabla\cdot\left(\frac{\rho^{n+1}}{A}\nabla p_{\mathrm{rgh}}^{n+1}\right) \tag{10}$$

方程(10)即压力泊松方程。现参考icoFoam解析中引入略去临点影响的假定有

$$\mathbf{U}_{\mathrm{P}}^{*} = \mathbf{HbyA}_{\mathrm{P}}^{n} - \frac{1}{A_{\mathrm{P}}}\left(\nabla p_{\mathrm{rgh,P}}^{*} + \mathbf{gh}\nabla\rho_{\mathrm{P}}^{n}\right) \tag{11}$$

将方程(11)代入到(1)，对任意网格单元（略去下标$_{\mathrm{P}}$）有

$$\frac{\partial\rho}{\partial t} + \nabla\cdot\left(\rho^{n}\mathbf{HbyA}^{n} - \frac{\rho^{n}}{A}\mathbf{gh}\nabla\rho^{n}\right) = \nabla\cdot\left(\frac{\rho^{n}}{A}\nabla p_{\mathrm{rgh}}^{*}\right) \tag{12}$$

方程(12)中除了第一项均对密度进行了欧拉显性离散，即使用的ρ^n否则会出现另外一个未知量ρ^{n+1}。同时，第一项中的密度时间变化量也只能进行显性离散。为了更好的收敛性，对于可压缩流体有

$$\rho^{*} = \rho^{n} + \mathrm{psi}(p^{*} - p^{n}) = \rho^{n} + \mathrm{psi}\left(p_{\mathrm{rgh}}^{*} - p_{\mathrm{rgh}}^{n}\right) \tag{13}$$

其中psi表示可压缩性。其对时间t的微分为

$$\frac{\partial\rho^{*}}{\partial t} = \frac{\partial\rho^{n}}{\partial t} + \mathrm{psi}\frac{\partial\left(p_{\mathrm{rgh}}^{*} - p_{\mathrm{rgh}}^{n}\right)}{\partial t} \tag{14}$$

把方程(14)代入到(10)有

$$\frac{\partial\rho^{n}}{\partial t} + \mathrm{psi}\frac{\partial\left(p_{\mathrm{rgh}}^{*} - p_{\mathrm{rgh}}^{n}\right)}{\partial t} + \nabla\cdot\left(\rho^{n}\mathbf{HbyA}^{n} - \frac{\rho^{n}}{A}\mathbf{gh}\nabla\rho^{n}\right) = \nabla\cdot\left(\frac{\rho^{n}}{A}\nabla p_{\mathrm{rgh}}^{n+1}\right) \tag{15}$$

下面分析能量方程，主要的推导过程请参考CFD中的能量方程：

$$\frac{\partial \rho h}{\partial t}+\nabla\cdot(\rho\mathbf{U}h)+\frac{\partial \rho K}{\partial t}+\nabla\cdot(\rho\mathbf{U}K)-\frac{\partial p}{\partial t}=\rho r-\nabla\cdot q+\rho\mathbf{g}\cdot\mathbf{U}+\nabla\cdot(\tau\cdot\mathbf{U}) \tag{16}$$

在buoyantPimpleFoam中，忽略了$\nabla\cdot(\tau\cdot\mathbf{U})$和$\rho r$。并且认为热通量$q$可以指定为$\alpha_{\text{eff}}h$。因此，能量方程变为

$$\frac{\partial \rho h}{\partial t}+\nabla\cdot(\rho\mathbf{U}h)+\frac{\partial \rho K}{\partial t}+\nabla\cdot(\rho\mathbf{U}K)-\frac{\partial p}{\partial t}=-\nabla\cdot(\alpha_{\text{eff}}\nabla h)+\rho\mathbf{g}\cdot\mathbf{U} \tag{17}$$

方程(17)即最终的能量方程。

3. 代码分析

下面分析主要的代码。首先进入createFields.H

```
// Force p_rgh to be consistent with p
p_rgh = p - rho*gh; // 声明p_rgh场
```

然后求解密度方程rhoEqn.H，其对应方程(1)：

```
    fvScalarMatrix rhoEqn
    (
        fvm::ddt(rho)
      + fvc::div(phi)
      ==
        fvOptions(rho)
    );
```

然后然后求解UEqn.H：

```
fvVectorMatrix UEqn
(
    fvm::ddt(rho, U) + fvm::div(phi, U)
  + MRF.DDt(rho, U)
  + turbulence->divDevRhoReff(U)
 ==
    fvOptions(rho, U)
);

UEqn.relax();

fvOptions.constrain(UEqn);

if (pimple.momentumPredictor())
{
    solve
    (
        UEqn
     ==
        fvc::reconstruct
        (
            (
              - ghf*fvc::snGrad(rho)
              - fvc::snGrad(p_rgh)
            )*mesh.magSf()//方程(7)右端的两项
        )
    );

    fvOptions.correct(U);
    K = 0.5*magSqr(U);//K的定义参考"CFD中的能量方程"
}
```

能量方程EEqn.H

```
{
    volScalarField& he = thermo.he();

    fvScalarMatrix EEqn
    (
        fvm::ddt(rho, he) + fvm::div(phi, he)//方程(17)第1,2项
      + fvc::ddt(rho, K) + fvc::div(phi, K)//方程(17)第3,4项
      + (
            he.name() == "e"
          ? fvc::div
            (
                fvc::absolute(phi/fvc::interpolate(rho), U),
                p,
                "div(phiv,p)"
            )//当求解内能的时候，参考"CFD中的能量方程"中的方程(17)中的第8项
          : -dpdt//当求解内能的时候，方程(14)第5项
        )
      - fvm::laplacian(turbulence->alphaEff(), he)//方程(14)第6项
     ==
        rho*(U&g)//方程(14)第7项
      + radiation->Sh(thermo)//其他辐射源项
      + fvOptions(rho, he)//其他源项
    );
}
```

接下来求解压力方程。压力方程的组建思想和其他求解器相同。即通过连续性方程组建压力泊松方程，在此不再赘述。仅摘取压力方程如下：

```
    fvScalarMatrix p_rghDDtEqn
    (
        fvc::ddt(rho)  //方程(15)第1项
      + psi*correction(fvm::ddt(p_rgh))//方程(15)第2项
      + fvc::div(phiHbyA)//方程(15)第3项
      ==
        fvOptions(psi, p_rgh, rho.name())
    );

...

        fvScalarMatrix p_rghEqn
        (
            p_rghDDtEqn
          - fvm::laplacian(rhorAUf, p_rgh)//方程(15)第4项
        );
```

求解后需要更新压力：$p = p_{\mathrm{rgh}} + \rho\mathbf{g}\cdot\mathbf{h}$。然后再次进去密度方程rhoEqn.H。 另外在代码中，psi*correction(fvm::ddt(p_rgh))主要对应方程(15)的第2项（离散的对角阵）。correction()实际返回的为A - (A & A.psi())，其中A为我们离散后的矩阵系数。A.psi()为$p_{\mathrm{rgh}0}$。首先，通过fvm::ddt(p_rgh)离散获取p_rgh的离散矩阵系数，然后通过correction函数，将已知的A & A.psi()减去（需要注意的是这一步重组后进入方程组的右方）。另外需要提及的是，correction函数在最终收敛的时候，失去作用。

2019.03.26 大修