

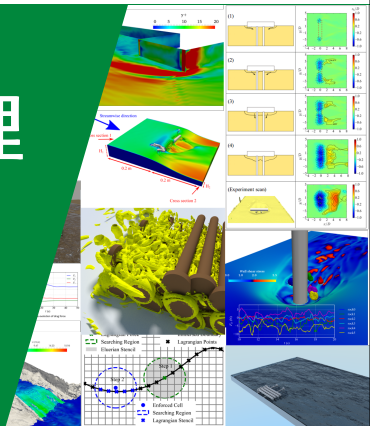


2025年春季《计算流体力学编程实践》

第六章 计算结果后处理

徐云成

✉ycxu@cau.edu.cn



2025 年 4 月 25 日

- ▶ ParaView — GUI形式，易于操作
 - 绘制云图、矢量图、流线图
 - 二次加工数据
 - 输出图片、动画、视频
- ▶ 命令行后处理
 - 计算结束后的处理，例如 `simpleFoam -postProcess -func Lambda2`
 - 计算进行中(run-time)的处理
- ▶ 第三方软件进行后处理
 - 使用Tecplot直接读取算例数据
 - 使用matlab python等对后处理得到的数据进行再处理



```
Time = 287

smoothSolver: Solving for Ux, Initial residual = 0.000119475, Final residual = 1.19332e-05, No Iterations 5
smoothSolver: Solving for Uy, Initial residual = 0.000988822, Final residual = 6.43953e-05, No Iterations 6
GAMG: Solving for p, Initial residual = 0.00147862, Final residual = 0.00014116, No Iterations 3
time step continuity errors : sum local = 0.00595802, global = -0.000643205, cumulative = 1.18609
smoothSolver: Solving for epsilon, Initial residual = 0.000135797, Final residual = 8.98303e-06, No Iterations 3
smoothSolver: Solving for k, Initial residual = 0.000217753, Final residual = 1.3218e-05, No Iterations 4
ExecutionTime = 4.72 s  ClockTime = 6 s
```

图: simpleFoam 二维计算屏幕输出信息

- ▶ 对于系数矩阵 $Ax = b$, 残差为 $r = b - Ax$
- ▶ 定义归一化因子 $n = \sum (|Ax - A\bar{x}| + |b - A\bar{x}|)$
- ▶ 归一化的残差值为 $r = \frac{1}{n} \sum |b - Ax|$
- ▶ Initial residual: 使用上一个时间步的 x 和本时间步的 A 、 b 进行计算
- ▶ Final residual: 使用本时间步的 x 、 A 、 b 进行计算
- ▶ 一般使用 Initial residual 曲线来判断稳态计算是否收敛
- ▶ No Iteration 指系数矩阵计算迭代次数



```
{
  volScalarField contErr(fvc::div(phi));

  scalar sumLocalContErr = runTime.deltaTValue()*
    mag(contErr()).weightedAverage(mesh.V()).value();

  scalar globalContErr = runTime.deltaTValue()*
    contErr.weightedAverage(mesh.V()).value();
  cumulativeContErr += globalContErr;

  Info<< "time step continuity errors : sum local = " << sumLocalContErr
    << ", global = " << globalContErr
    << ", cumulative = " << cumulativeContErr
    << endl;
}
```

- ▶ Local continuity error（绝对值）：

$$\epsilon_{\text{local}} = \Delta t \langle |\nabla \phi| \rangle$$

- ▶ Global continuity error:

$$\epsilon_{\text{global}} = \Delta t \langle \nabla \phi \rangle$$

- ▶ $\langle x \rangle$ 指网格单元体积平均

src/finiteVolume/cfdTools/incompressible/continuityErrs.H



方法一使用foamLog

- ▶ 将计算屏幕输出保存至文件，例如 `simpleFoam>log.simpleFoam`
- ▶ 运行 `foamLog log.simpleFoam`，可以得到logs/，该文件夹内包括

```
clockTime_0      epsAvg_0      epsMax_0      k_0      pFinalRes_0  Ux_0      UyFinalRes_0
contCumulative_0  epsilon_0    epsMin_0      kFinalRes_0  pItrs_0      UxFinalRes_0  UyItrs_0
contGlobal_0     epsilonFinalRes_0  executionTime_0  kItrs_0      Separator_0  UxItrs_0
contLocal_0      epsilonItrs_0  foamLog.awk    p_0      Time_0      Uy_0
```

其中 U_{x_0} 是 U_x 的（初始）残差



方法二使用run-time function object

- ▶ 在system/controlDict中加入

```
functions
{
#includeFunc  residuals
...  other function objects here ...
}
```

- ▶ 计算会实时输出结果至postProcessing/residuals/0/residuals.dat
- ▶ 使用以下命令可以实时看到残差变化
foamMonitor -l postProcessing/residuals/0/residuals.dat



- ▶ 到底有多少这样的后处理函数 function object?

`postProcess -list`

- ▶ 上述命令列出的函数可以在以下位置找到

`$FOAM_ETC/caseDicts/postProcessing`

- ▶ 后处理函数种类包括:

combustion	fields	forces	lagrangian	numerical	probes	surfaceFieldValue
control	flowRate	graphs	minMax	pressure	solvers	visualization



CourantNo

库郎数，要求 $\phi(\text{phi})$ 进行计算
可用以下三种方式进行计算

- ▶ `simpleFoam -postProcess -func CourantNo`
- ▶ `postProcess -func CourantNo`
- ▶ 在`system/controlDict`中加入

```
functions
{
#includeFunc CourantNo
... other function objects here ...
}
```



场计算 field

9/24

Lambda2

计算速度梯度张量对称和非对称部分平方和的第二特征值，用于表征流场结构

MachNo

从u场计算马赫数

PecletNo

从phi场计算佩克莱数

Q

计算速度梯度张量的第二不变量，用于表征流场结构

R

计算雷诺应力张量（6个变量）

add

两个或多个场相加

- ▶ `simpleFoam -postProcess -func "add(U,U)"`
- ▶ `postProcess -func "add(U,U)"`
- ▶ `#includeFunc add(U,U)`

输出变量名`add(U,U)`，此外`subtract`是两个场相减

age

计算流体颗粒从进口到该位置时间，也被称为“空气龄”

components

输出分量场，例如`U`可分为`Ux`、`Uy`、`Uz`



ddt

计算某个变量的时间梯度 Eulerian time derivative

div

计算某个变量的散度 divergence

grad

计算某个变量的梯度 gradient

enstrophy

通过速度场计算涡度拟能 $\xi = 0.5|\nabla \times \mathbf{u}|^2$



fieldAverage

计算时间平均，尤其适用于LES中的run-time process，例如
#includeFunc fieldAverage(U, p, prime2Mean = yes)
运行 foamInfo fieldAverage 选择第2个可得

```
fieldAverage1
{
    type            fieldAverage;
    libs            ("libfieldFunctionObjects.so");

    writeControl    writeTime;

    restartOnRestart    false;
    restartOnOutput     false;
    periodicRestart     false;
    restartPeriod       0.002;

    base              time;
    window             10.0;
    windowName        w1;

    mean               yes;
    prime2Mean         yes;

    fields              (U p);
}
```

Usage

\table

Property	Description	Required	Default
type	type name: fieldAverage		yes
restartOnRestart	Restart the averaging on restart	no	no
restartOnOutput	Restart the averaging on output	no	no
periodicRestart	Periodically restart the averaging	no	no
restartPeriod	Periodic restart period		conditional
fields	list of fields and averaging options		yes

\endtable

flowType

输出速度场的流动状态：-1是旋转流动，0是剪切流，1是平面流

$$\lambda = \frac{|\mathbf{D}| - |\mathbf{\Omega}|}{|\mathbf{D}| + |\mathbf{\Omega}|}$$

λ 是流动状态指标， \mathbf{D} 是速度梯度张量的对称部分 $\text{symm}(\text{gradU})$ ， $\mathbf{\Omega}$ 是速度梯度张量的偏对称部分 $\text{skew}(\text{gradU})$

可在src/OpenFOAM/primitives/Tensor/TensorI.H中找到

$$\text{symm}(\mathbf{a}) = \begin{bmatrix} a_{11} & 0.5(a_{12} + a_{21}) & 0.5(a_{13} + a_{31}) \\ & a_{22} & 0.5(a_{23} + a_{32}) \\ & & a_{33} \end{bmatrix}$$

$$\text{skew}(\mathbf{a}) = \begin{bmatrix} 0.0 & 0.5(a_{12} - a_{21}) & 0.5(a_{13} - a_{31}) \\ 0.5(a_{21} - a_{12}) & 0.0 & 0.5(a_{23} - a_{32}) \\ 0.5(a_{31} - a_{13}) & 0.5(a_{32} - a_{23}) & 0.0 \end{bmatrix}$$

flowType

输出速度场的流动状态：-1是旋转流动，0是剪切流，1是平面流

$$\lambda = \frac{|\mathbf{D}| - |\mathbf{\Omega}|}{|\mathbf{D}| + |\mathbf{\Omega}|}$$

λ 是流动状态指标， \mathbf{D} 是速度梯度张量的对称部分 $\text{symm}(\text{gradU})$ ， $\mathbf{\Omega}$ 是速度梯度张量的偏对称部分 $\text{skew}(\text{gradU})$

可在src/OpenFOAM/primitives/Tensor/TensorI.H中找到

$$\text{symm}(\mathbf{a}) = \begin{bmatrix} a_{11} & 0.5(a_{12} + a_{21}) & 0.5(a_{13} + a_{31}) \\ & a_{22} & 0.5(a_{23} + a_{32}) \\ & & a_{33} \end{bmatrix}$$

$$\text{skew}(\mathbf{a}) = \begin{bmatrix} 0.0 & 0.5(a_{12} - a_{21}) & 0.5(a_{13} - a_{31}) \\ 0.5(a_{21} - a_{12}) & 0.0 & 0.5(a_{23} - a_{32}) \\ 0.5(a_{31} - a_{13}) & 0.5(a_{32} - a_{23}) & 0.0 \end{bmatrix}$$

场计算 field

4/24

mag

计算某个变量的大小 magnitude

magSqr

计算某个变量的平方 magnitude-squared

randomise

给某一个变量加扰动

```
postProcess -func "randomise(p,magPerturbation=0.1)"
```

scale

使某个变量增大一定倍数，只能是标量(volScalarField)

```
postProcess -func "scale(U,result=U_scaled,scale=1.2)"
```

场计算 field

15/24

shearStress

计算剪切应力场，输出张量（6个分量）

shearStress

计算剪切应力场，输出张量（6个分量）

turbulenceFields

计算湍流相关变量，R是雷诺应力

```
simpleFoam -postProcess -func "turbulenceFields(fields=(R devSigma))"
```

等同于 `simpleFoam -postProcess -func R`

turbulenceIntensity

计算湍流强度，由于涉及湍流模型，必须要用<solver>



场计算 field

6/24

vorticity

计算涡强度，必须要用<solver>

wallShearStress

计算壁面剪切力，必须要用<solver>

writeCellCentres

输出网格中心坐标以及坐标三个分量，有助于进一步后处理

writeVTK

输出VTK格式的数据 `postProcess -func "writeVTK(objects=(U p nut))"`

yPlus

计算壁面第一层网格的无量纲尺度，必须要用<solver>

flowRatePatch

利用phi计算边界上流量 `postProcess -func "flowRatePatch(name=inlet)"`



力的计算

8/24

forcesIncompressible

计算不可压流体壁面所受力，包括三个方向的压力和黏滞力

```
simpleFoam -postProcess -func "forcesIncompressible(patches=(lowerWall),rhoInf=1000)
```

forceCoeffsIncompressible

计算相关升力、阻力系数，还可以设置面积Aref，速度magUInf，升力方向liftDir，阻力方向dragDir

forcesCompressible

计算可压流体壁面所受力

forceCoeffsCompressible

计算相关升力、阻力系数

最大值最小值

9/24

cellMax

输出网格中心的最大值 `postProcess -func "cellMax(fields=(U p))"`

cellMin

输出网格中心的最小值

faceMax

输面的最大值

`postProcess -func "faceMax(fields=(U p),regionType=patch,name=inlet)"`

faceMin

输面的最小值



minMaxComponents

输出最大最小值以及所在的网格序号和空间位置

```
postProcess -func "minMaxComponents(fields=(U p))"
```

minMaxMagnitude

输出最大最小值（模）以及所在的网格序号和空间位置

```
postProcess -func "minMaxMagnitude(fields=(U p))"
```



pressureDifferencePatch

计算两个边界上的平均压力以及他们的差

```
simpleFoam -postProcess -func "pressureDifferencePatch(patch1=inlet,patch2=outlet)"
```

其实是subtract+areaAverage

```
$FOAM_ETC/caseDicts/postProcessing/pressure/pressureDifference.cfg
```



pressureDifferencePatch

计算两个边界上的平均压力以及他们的差

```
simpleFoam -postProcess -func "pressureDifferencePatch(patch1=inlet,patch2=outlet)"
```

其实是subtract+areaAverage

```
$FOAM_ETC/caseDicts/postProcessing/pressure/pressureDifference.cfg
```



probes

输出离定义点最近的网格中心值

```
postProcess -func "probes(fields=(U p),probeLocations=((1 0 0)(2 0 0)))"
```

```
probes
{
    // Where to load it from
    libs      ("libsampling.so");

    type      probes;

    // Name of the directory for probe data
    name      probes;

    // Write at same frequency as fields
    writeControl    timeStep;
    writeInterval    1;

    // Fields to be probed
    fields          (p U);

    probeLocations
    (
        ( -0.0206 0 0.0 )      //
        (0 -0. 0.0) //
        (0.1 0. 0.0) //
        (0.29 0 0.0) //
    );
}
```


internalProbes

输出离定义点最近的网格中心值

```
postProcess -func "probes(fields=(U p),probeLocations=((1 0 0)(2 0 0)))"
```

```
probes
{
    // Where to load it from
    libs      ("libsampling.so");

    type      probes;

    // Name of the directory for probe data
    name      probes;

    // Write at same frequency as fields
    writeControl    timeStep;
    writeInterval    1;

    // Fields to be probed
    fields          (p U);

    probeLocations
    (
        ( -0.0206 0 0.0 )      //
        (0 -0. 0.0) //
        (0.1 0. 0.0) //
        (0.29 0 0.0) //
    );
}
```

Thank you.

欢迎私下交流，请勿私自上传网络，谢谢！

