

【OpenFOAM球体周りの抗力係数(4)】

simpleFoamで球体周りの定常流れ

2022年2月20日



FreeCAD 0.19
Paraview 5.9.0
OpenFOAM v2006
Python 3.8.10(Jupyter lab)

WSL2

Python

プリ処理

バスケットボールのモデル作成

- FreeCAD

メッシュ作成

- blockMesh
- snappyHexMesh

解析設定

- OpenFOAM

ソルバ

計算実行

- OpenFOAM

ポスト処理

結果処理

- Paraview
- PyFoam

今回のモデルは「20220216_sphere_coff_blog」というフォルダの中に作成します。

フォルダ構成

20220216_sphere_coff_blog

model

orgCase ←今回はこちらに球体周りのメッシュ作成

resultDir

□フォルダ構成

\$tree

(0)Terminal上で
「tree」と打ってフォルダ構成を確認します。

※treeコマンドがインストールされていない場合はTerminalで「sudo apt install tree」と打ってインストールしてください。

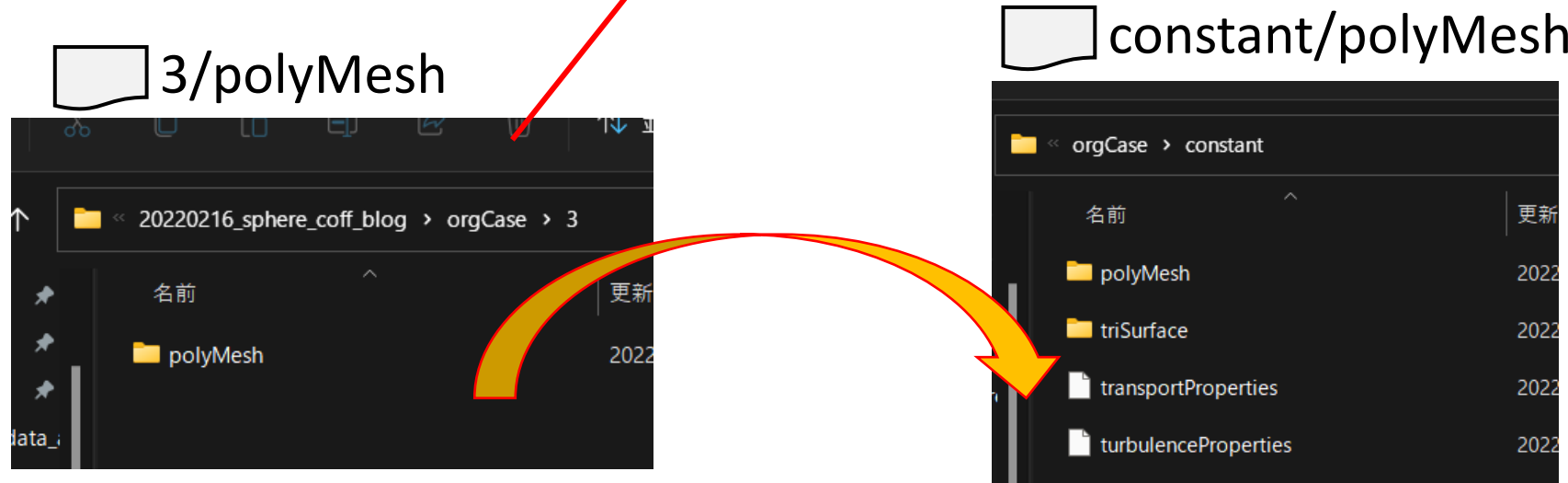
0		
U	速度ベクトル	
epsilon	乱流消失率	
k	乱流エネルギー	
nut	渦粘性係数	
old		
nuTilda	使わない	
omega		
p	圧力場	
constant		
transportProperties	物性値の設定	
turbulenceProperties	乱流モデルの指定	
system		
blockMeshDict	ベースメッシュの設定	
controlDict	実行制御	
decomposeParDict	並列計算の設定	
fvSchemes	離散化スキームの設定	
fvSolution	時間解法やマトリックスソルバの設定	
meshQualityDict	メッシュ品質設定	
snappyHexMeshDict	メッシュ設定	

```
[kamakiri@ws1]orgCase$tree
.
├── U
├── epsilon
├── k
├── nut
├── nuTilda
├── omega
├── p
├── boundary
├── cellZones
├── faceZones
├── faces
├── neighbour
├── owner
├── pointZones
├── points
├── codeStreamTemplate.C.dep
├── codeStreamTemplate.o
├── options
├── sourceFiles
└── variables
```

※blockMesh時に生成されたdynamicCodeなどは載せていません

□メッシュファイルの移動

(1)snappHexMeshで生成したメッシュ「3/polyMesh」を「constant/polyMesh」に移します。

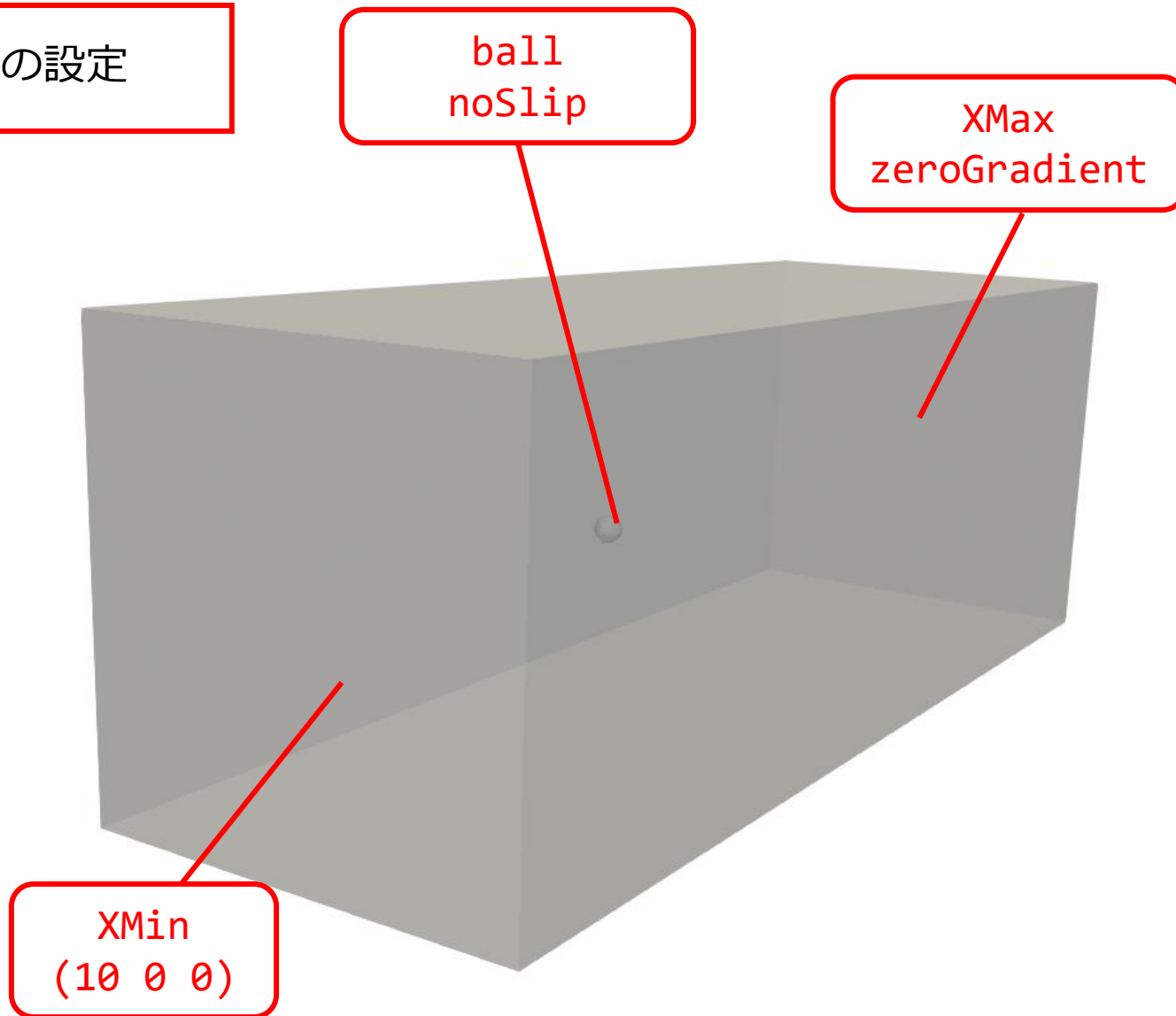


(2)「3」フォルダは使わないので削除。
「0.org」をコピーして「0」という名前にする



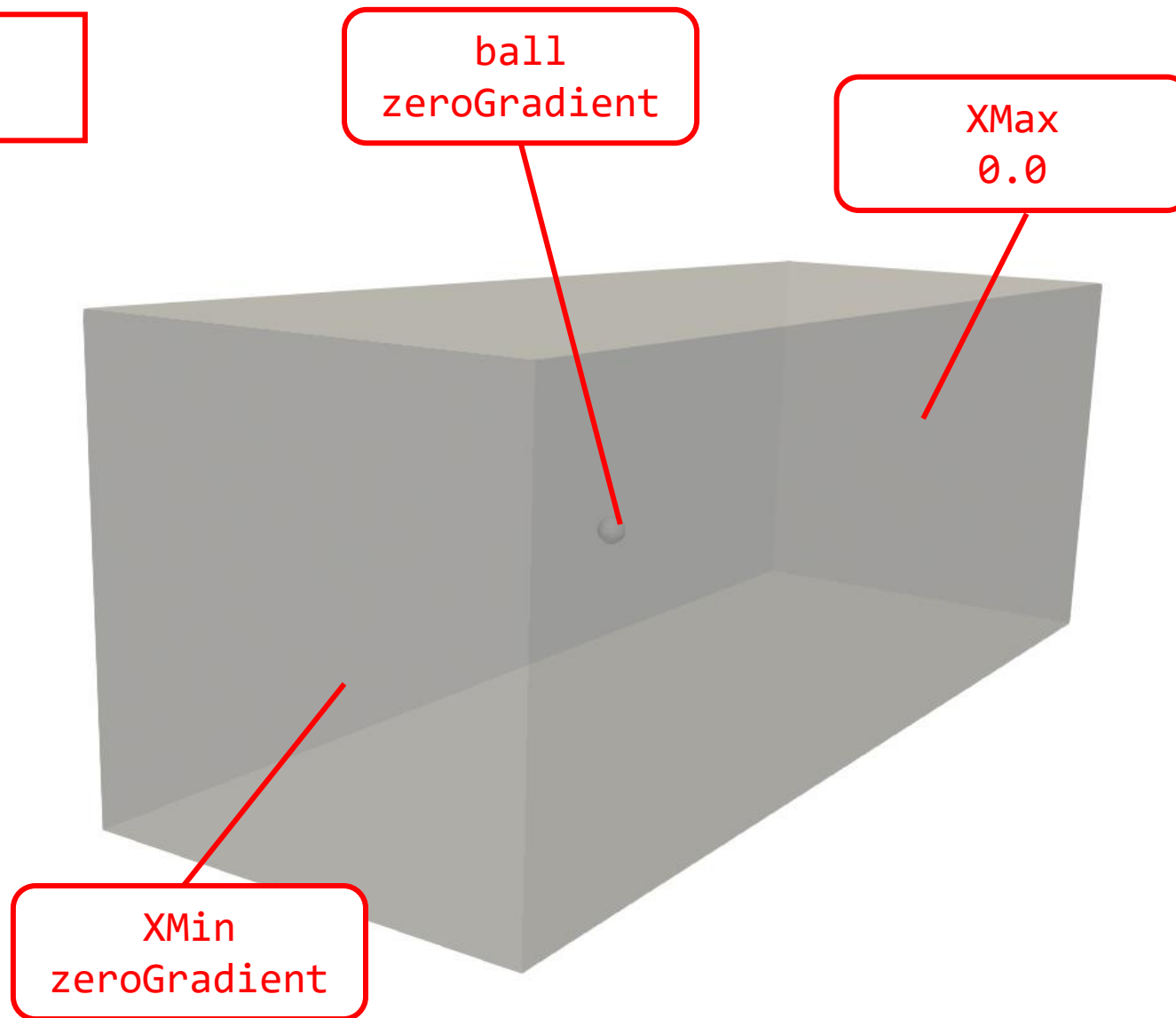
(3) 「U」 ファイルの設定

```
internalField uniform (1 0 0);
boundaryField
{
    XMin
    {
        type fixedValue;
        value uniform (1 0 0);
    }
    XMax
    {
        type zeroGradient;
    }
    YMin
    {
        type slip;
    }
    YMax
    {
        type slip;
    }
    ZMin
    {
        type slip;
    }
    ZMax
    {
        type slip;
    }
    ball
    {
        type noSlip;
    }
}
```



(4) 「p」 ファイルの設定

```
boundaryField
{
    XMin
    {
        type zeroGradient;
    }
    XMax
    {
        type fixedValue;
        value uniform 0.0;
    }
    YMin
    {
        type zeroGradient;
    }
    YMax
    {
        type zeroGradient;
    }
    ZMin
    {
        type zeroGradient;
    }
    ZMax
    {
        type zeroGradient;
    }
    ball
    {
        type zeroGradient;
    }
}
```



(5) 「k」 ファイルの設定

```
internalField uniform 0.00015; //U = 1.0 (1%);
boundaryField
{
    XMin
    {
        type            turbulentIntensityKineticEnergyInlet;
        type fixedValue;
        value uniform 0.00015;
    }
    XMax
    {
        type zeroGradient;
    }
    YMin
    {
        type zeroGradient;
    }
    YMax
    {
        type zeroGradient;
    }
    ZMin
    {
        type zeroGradient;
    }
    ZMax
    {
        type zeroGradient;
    }
    ball
    {
        type kqRWallFunction;
        value uniform 0.0;
    }
}
```

乱流エネルギー十分に発達した乱流 : $U_x' = U_y' = U_z'$

$$k = \frac{1}{2} \overline{U' \cdot U'} = \frac{1}{2} (U_x'^2 + U_y'^2 + U_z'^2) = \frac{3}{2} U'^2$$

乱流強度1%と仮定すると

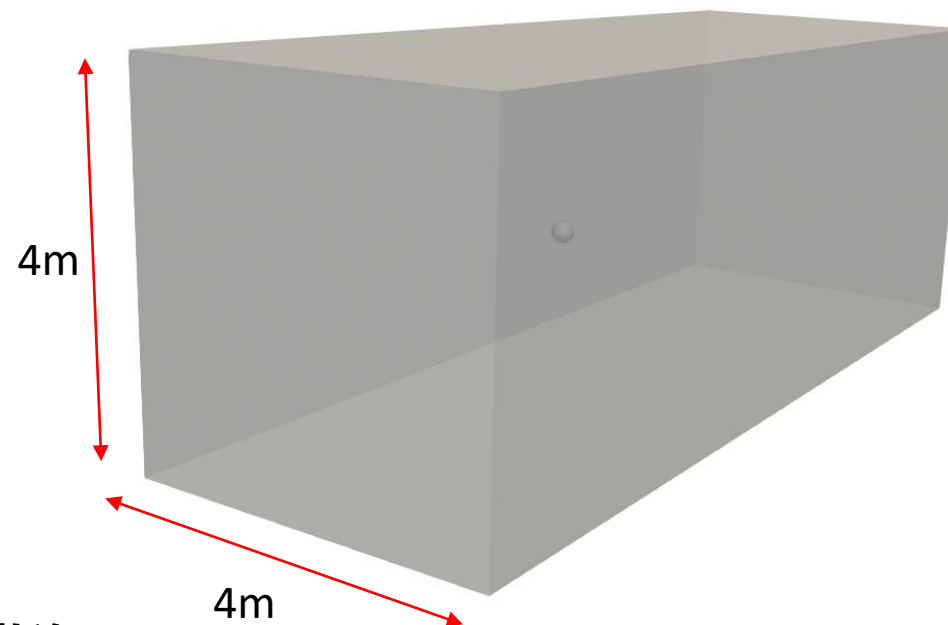
$$k = \frac{3}{2} U'^2 = \frac{3}{2} (IU)^2 = 1.5 \times (0.01 \times 1)^2 = \mathbf{0.00015}$$

※乱流強度が大きすぎると抗力係数が大きくずれる

0/epsilon

(6) 「epsilon」 ファイルの設定

```
internalField uniform 7.54673E-07; //L=4m (10%),
boundaryField
{
    XMin
    {
        type fixedValue;
        value $internalField;
    }
    XMax
    {
        type zeroGradient;
    }
    YMin
    {
        type zeroGradient;
    }
    YMax
    {
        type zeroGradient;
    }
    ZMin
    {
        type zeroGradient;
    }
    ZMax
    {
        type zeroGradient;
    }
    ball
    {
        type epsilonWallFunction;
        value uniform 0.0;
    }
}
```



乱流散逸

$$C_\mu = 0.09$$

$$k = 0.00015$$

混合長 l をボックス幅の10%として

$$\varepsilon = \frac{C_\mu^{3/4} k^{3/2}}{l} = \frac{0.09^{3/4} \times 0.00015^{3/2}}{0.1 \times 4} = 7.54673\text{E-}07$$

0/nut

(7) 「nut」 ファイルの設定

$v_T = C_\mu \frac{k^2}{\varepsilon}$ より計算するようにしている。

```
internalField uniform 0.001;
boundaryField
{
    XMin
    {
        type calculated;
        value uniform 0.001;
    }
    XMax
    {
        type calculated;
        value uniform 0.001;
    }
    YMin
    {
        type zeroGradient;
    }
    YMax
    {
        type zeroGradient;
    }
    ZMin
    {
        type zeroGradient;
    }
    ZMax
    {
        type zeroGradient;
    }
    ball
    {
        type nutkWallFunction;
        value uniform 0.0;
    }
}
```

 constant/transportProperties

(8)物性値の設定

```
transportModel  Newtonian;
```

```
nu              1e-05;
```

 constant/turbulenceProperties


(9)乱流モデルの設定

```
simulationType RAS;

RAS
{
    // Tested with kEpsilon, realizableKE, kOmega, kOmegaSST,
    // ShihQuadraticKE, LienCubicKE.
    RASModel      kEpsilon;

    turbulence     on;

    printCoeffs    on;
}
```

 system/controlDict

(10)ソルバ、ステップ数の指定

```
application      simpleFoam;
startFrom         startTime;
startTime         0;
stopAt           endTime;
endTime          500;
deltaT           1;
writeControl      timeStep;
writeInterval     100;
purgeWrite        0;
writeFormat       ascii;
writePrecision    6;
writeCompression off;
timeFormat        general;
timePrecision     6;
runTimeModifiable true;
```

(11)抗力係数、揚力係数の出力設定

```
functions
{
  forceCoeffs
  {
    type forceCoeffs;
    libs
    (
      "libforces.so"
    );
    writeControl timeStep;
    timeInterval 1;
    log no;
    patches
    (
      ball
    );
    rho rhoInf;
    rhoInf 1;
    liftDir (0 1 0);
    dragDir (1 0 0);
    CofR (-1.4503e-05 2.28624e-06 -0.000131355);
    pitchAxis (0 0 1);
    magUInf 1.0;
    lRef 0.25;
    Aref 0.049093946015625;
  }
}
```

(12)残差の出力設定

```
residuals
{
  type residuals;
  libs
  (
    "libutilityFunctionObjects.so"
  );
  fields
  (
    U
    p
    k
    epsilon
  );
} // #includeFunc residuals(U,p,k,epsilon);
#includeFunc yPlus
#includeFunc CourantNo
}
```

system/decomposeParDict

(13)並列数の指定

```
numberOfSubdomains 4;

method            scotch;
//method          hierarchical;
// method         ptscotch;

simpleCoeffs
{
    n              (4 1 1);
    delta          0.001;
}

hierarchicalCoeffs
{
    n              (3 2 1);
    delta          0.001;
    order          xyz;
}

manualCoeffs
{
    dataFile       "cellDecomposition";
}
```

methodにscotchをしているのでこちらの設定は使われていない

□ 計算実行

\$mkdir Allrun

(14) Allrunファイルを作成

□ Allrun

```
#!/bin/sh  
cd ${0%/*} || exit 1  
. $WM_PROJECT_DIR/bin/tools/RunFunctions
```

```
decomposePar  
mpirun -np 4 simpleFoam -parallel  
reconstructPar
```

```
rm -rf ./processor*
```

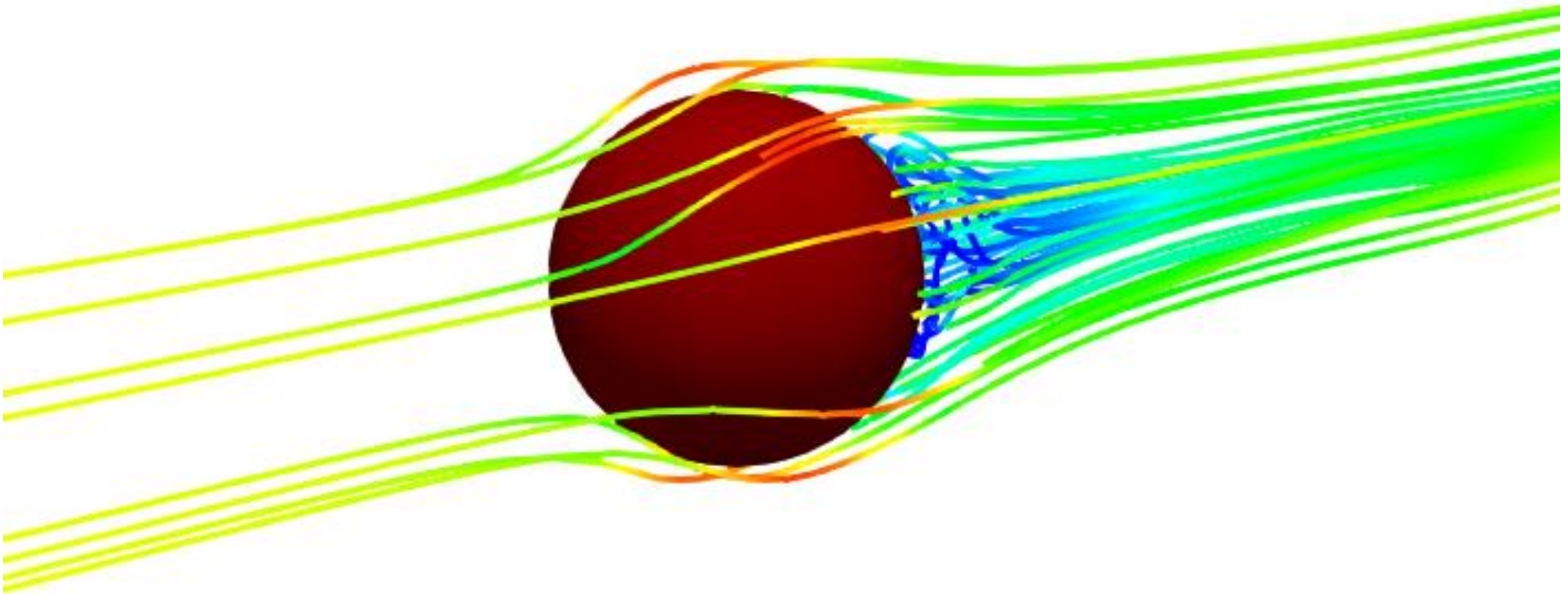
(15) Allrunファイルの中身を記述

\$./Allrun

(16) 計算実行

□結果の可視化

(15)Paraviewを起動して流線を表示



□結果の確認

抗力係数を確認

postprocessing/forceCoeffs/0/forceCoeffs.dat

```
1 # Force coefficients
2 # liftDir : (0.000000e+00 1.000000e+00 0.000000e+00)
3 # dragDir : (1.000000e+00 0.000000e+00 0.000000e+00)
4 # pitchAxis : (0.000000e+00 0.000000e+00 1.000000e+00)
5 # magUInf : 1.000000e+00
6 # lRef : 2.500000e-01
7 # Aref : 4.909395e-02
8 # CofR : (-1.450300e-05 2.286240e-06 -1.313550e-04)
9 # Time Cm Cd Cl Cl(f) Cl(r)
10 1 3.310849e-05 3.654918e+00 7.873303e-06 3.704514e-05 -2.917184e-05
11 2 5.350090e-05 5.864870e+00 -2.409411e-04 -6.696966e-05 -1.739715e-04
12 3 6.965077e-05 7.394349e+00 1.349978e-03 7.446398e-04 6.053383e-04
13 4 8.636128e-05 8.251301e+00 3.085065e-03 1.628894e-03 1.456171e-03
14 5 9.585839e-05 8.276388e+00 3.088211e-03 1.639964e-03 1.448247e-03
15 6 9.907515e-05 7.423688e+00 2.284616e-03 1.241383e-03 1.043233e-03
16 7 7.244855e-05 5.832100e+00 1.397785e-03 7.713412e-04 6.264441e-04
487 -4.707368e-06 4.530000e-01 5.957551e-04 2.931702e-04 3.025849e-04
488 -3.737924e-06 4.531787e-01 4.353955e-04 2.139598e-04 2.214357e-04
489 -2.320701e-06 4.532995e-01 4.929657e-04 2.441621e-04 2.488035e-04
490 -3.138723e-06 4.534252e-01 4.058868e-04 1.998047e-04 2.060821e-04
491 -3.051209e-06 4.534262e-01 4.811010e-04 2.374993e-04 2.436017e-04
492 -4.938661e-06 4.534091e-01 4.016936e-04 1.959081e-04 2.057855e-04
493 -5.900625e-06 4.532716e-01 4.939207e-04 2.410597e-04 2.528610e-04
494 -8.043428e-06 4.531299e-01 4.020854e-04 1.929993e-04 2.090861e-04
495 -9.377357e-06 4.529099e-01 5.193429e-04 2.502941e-04 2.690488e-04
496 -1.128543e-05 4.527669e-01 4.489855e-04 2.132073e-04 2.357782e-04
497 -1.164440e-05 4.526245e-01 5.451828e-04 2.609470e-04 2.842358e-04
498 -1.271793e-05 4.526163e-01 4.688658e-04 2.217149e-04 2.471508e-04
499 -1.273690e-05 4.526484e-01 5.870944e-04 2.808103e-04 3.062841e-04
500 -1.318806e-05 4.527945e-01 5.309290e-04 2.522764e-04 2.786526e-04
```

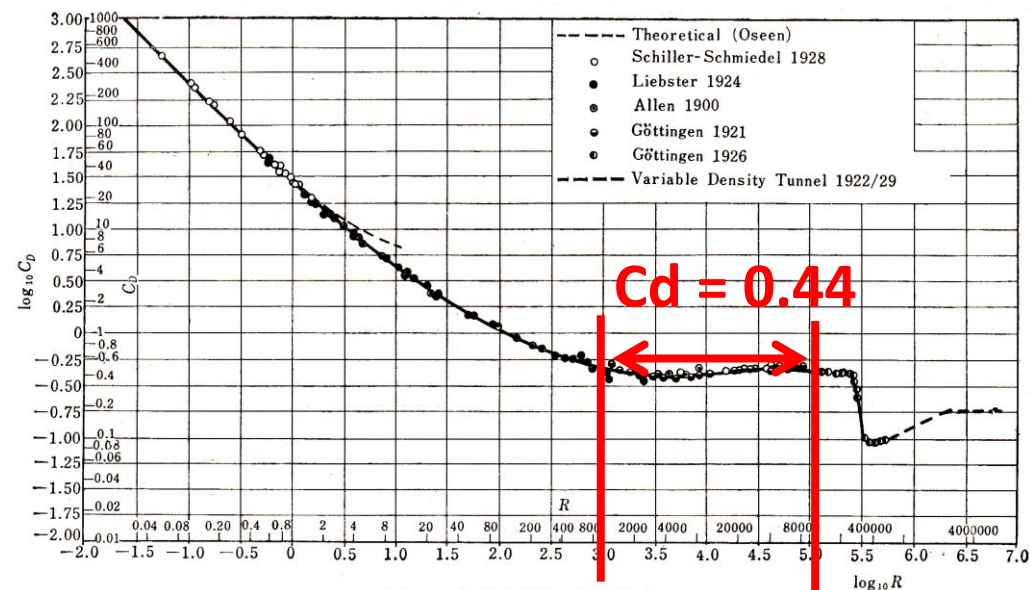


図62-1 球の抗力係数の実験データ

抗力係数 c_d が0.44くらいに収束

Appendix

□わかったこと

抗力係数が文献値4.2から外れるのは以下のモデル化の見直しが必要

1. 球体モデルのメッシュの細かさ
2. 乱流強度が強すぎる or 乱流散逸率が小さすぎる
3. 緩和係数などの見直し

□解くべき方程式

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial (\bar{u}_i \bar{u}_j)}{\partial x_j} = -\frac{1}{\rho} \frac{\partial \bar{P}}{\partial x_i} + \frac{\partial}{\partial x_j} \left\{ \nu_e \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) \right\}$$

$$\nu_e = \nu + \nu_t, \quad \bar{P} = \bar{p} + \frac{2}{3} \rho k$$

ν_t : 渦動粘性係数

ν_t を乱流エネルギー k と乱流散逸率 ε を用いてモデル化

$$\nu_t = C_\mu \frac{k^2}{\varepsilon}$$

$$\frac{\partial k}{\partial t} + \bar{u}_j \frac{\partial k}{\partial x_j} = P_k - \varepsilon + \frac{\partial}{\partial x_j} \left\{ \left(\frac{\nu_t}{\sigma_k} + \nu \right) \frac{\partial k}{\partial x_j} \right\}$$

$$\frac{\partial \varepsilon}{\partial t} + \bar{u}_j \frac{\partial \varepsilon}{\partial x_j} = (C_{\varepsilon_1} P_k - C_{\varepsilon_2} \varepsilon) \frac{\varepsilon}{k} + \frac{\partial}{\partial x_j} \left\{ \left(\frac{\nu_t}{\sigma_\varepsilon} + \nu \right) \frac{\partial \varepsilon}{\partial x_j} \right\}$$

$$C_\mu = 0.09, \quad \sigma_k = 1.0, \quad \sigma_\varepsilon = 1.3, \quad C_{\varepsilon_1} = 1.44, \quad C_{\varepsilon_2} = 1.92$$

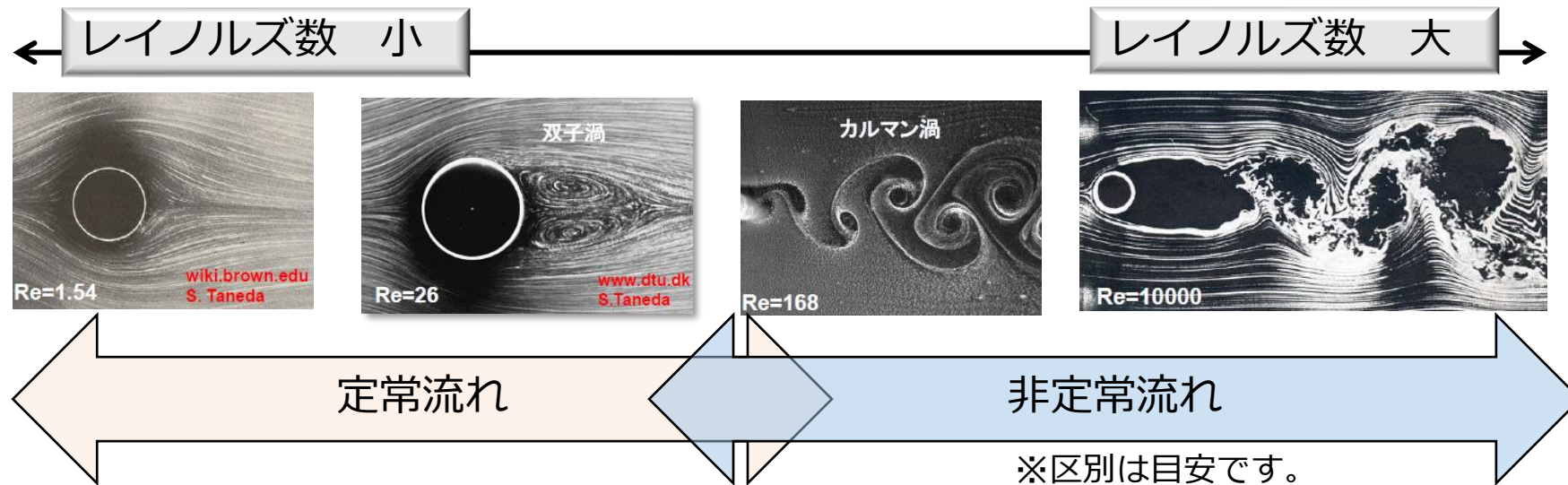
□定常・非定常流れ

定常流れ：時間が経っても流れは変化せず一定の状態

非定常流れ：時間の経過とともに流れが変化する状態

レイノルズ数による定常流れ・非定常流れの区別

※写真は円柱周りの流れ

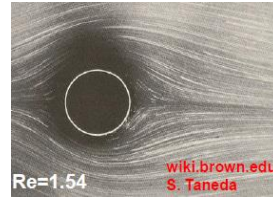


simpleFoam は SIMPLE 法を用いた非圧縮性流体の定常乱流解析ソルバーです。

- 運動方程式と圧力方程式の 2 つの方程式の形で解く
- 反復計算により収束したら (方程式の残差が小さくなったら) 計算を終了します。

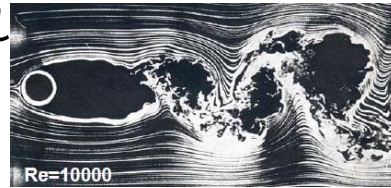
□層流・乱流

層流：整然とした流れ



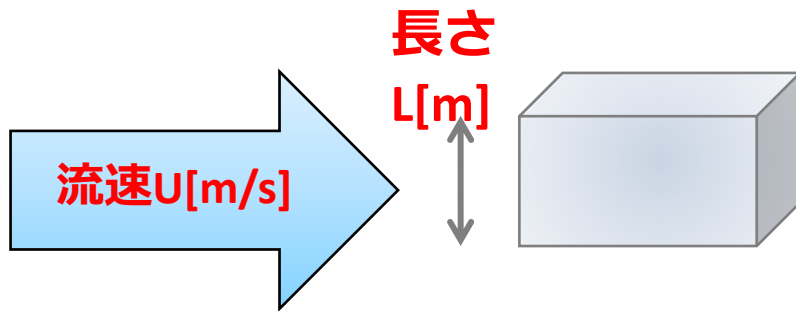
例：流速が遅い場合

乱流：時々刻々と変動する乱れた流れ



例：流速が速い場合

層流・乱流を見極めるひとつの目安



粘性係数： μ [Pa · s] **動粘性係数： $\nu = \mu / \rho$**
密度： ρ [kg/sm³]

レイノルズ数 Re

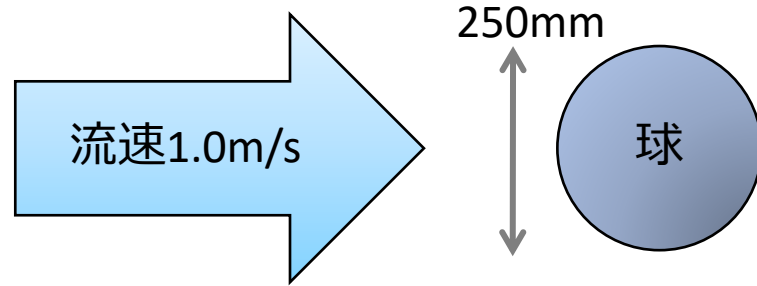
Re = (代表速度) × (代表長さ) / (動粘性係数)

$$Re = UL/\nu$$

□レイノルズ数による流れの分類

レイノルズ数：Re=50000の流れ

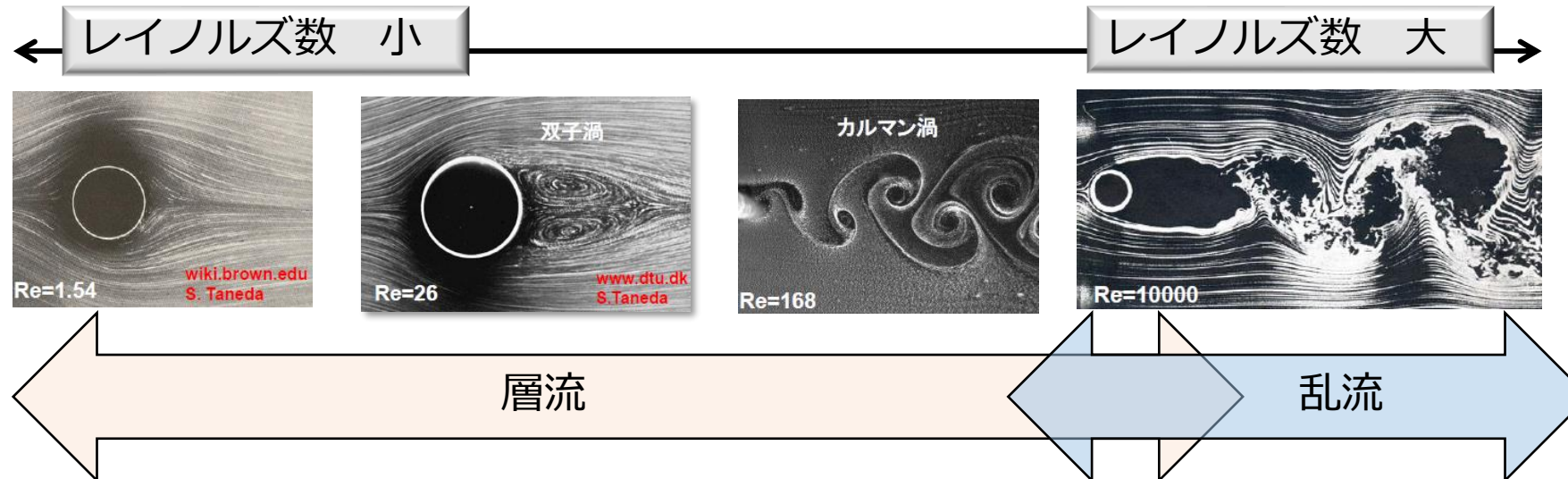
本資料の解析対象



レイノルズ数
 $Re = 0.25 * 1.0 / 10^{-5} = \mathbf{50000}$

レイノルズ数による層流・乱流の区別

※写真は円柱周りの流れ



※区別は目安です。

乱流モデル

乱流現象を再現する難しさ

乱流とは、

「**大小様々な渦**が複雑に絡み合った時々刻々と変動する流れ」である。

大小様々な渦を再現できるだけの解像度（メッシュ分割）が必要である。

⇒とてつもないメッシュ数 **現実的ではない！！**



渦流れ

幸い十分発達した乱流や限定的な場合において、少ないメッシュ数で計算出来る理論的なモデルが開発されている⇒**乱流モデル**

乱流モデル

RANS

十分発達した乱流であるとして乱流流れを平均化

レイノルズ平均

LES

メッシュで再現できない渦だけモデル化

LESでの格子, $\bar{u} = u - u'$

乱流モデルなし

DNS

最小渦まで再現できるだけメッシュを細かくする

DNSでの格子, u

計算コスト 小

計算コスト 大

精度 ×

精度 ○

□初期状態

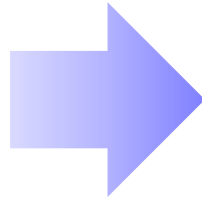
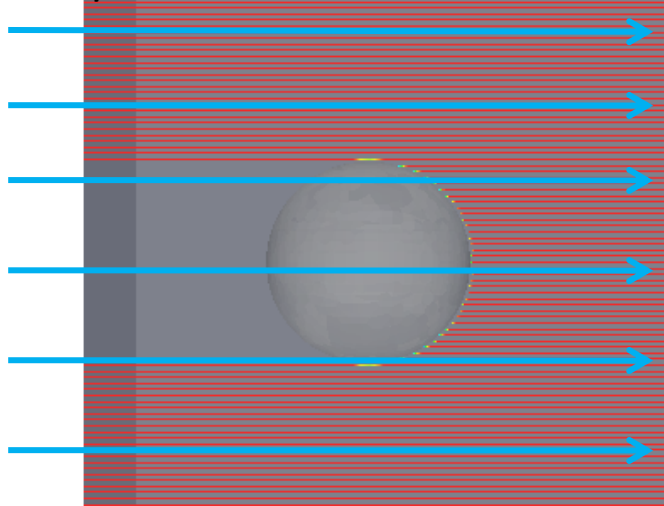
初期条件

解析の開始時点における状態を設定する条件

全体に初期の流れ場

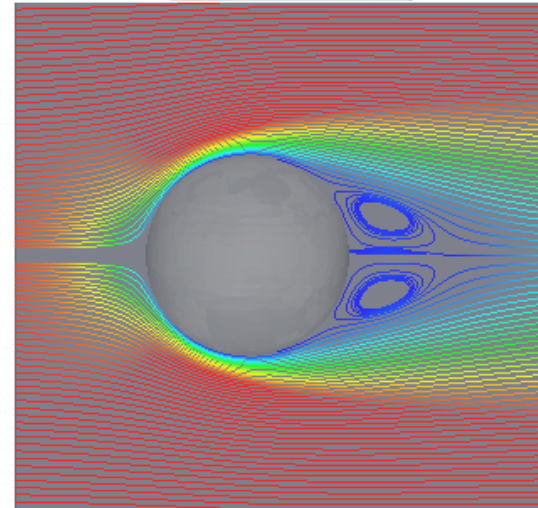
初期状態

0.0003m/s



時間が経てば、球体周りの流れは定常状態に落ち着く

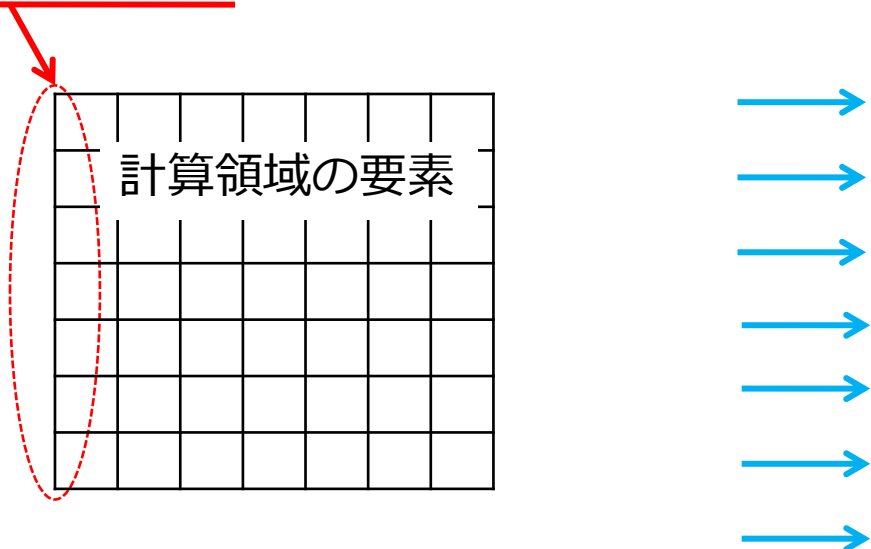
定常状態



定常解析：0サイクル目の流れを初期状態に与える
非定常解析：0秒の流れを与える

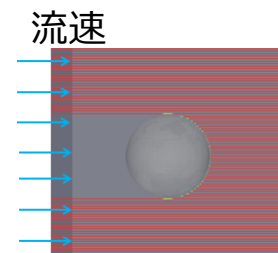
境界条件

計算領域の端は、情報が無いために境界条件を与える必要がある。



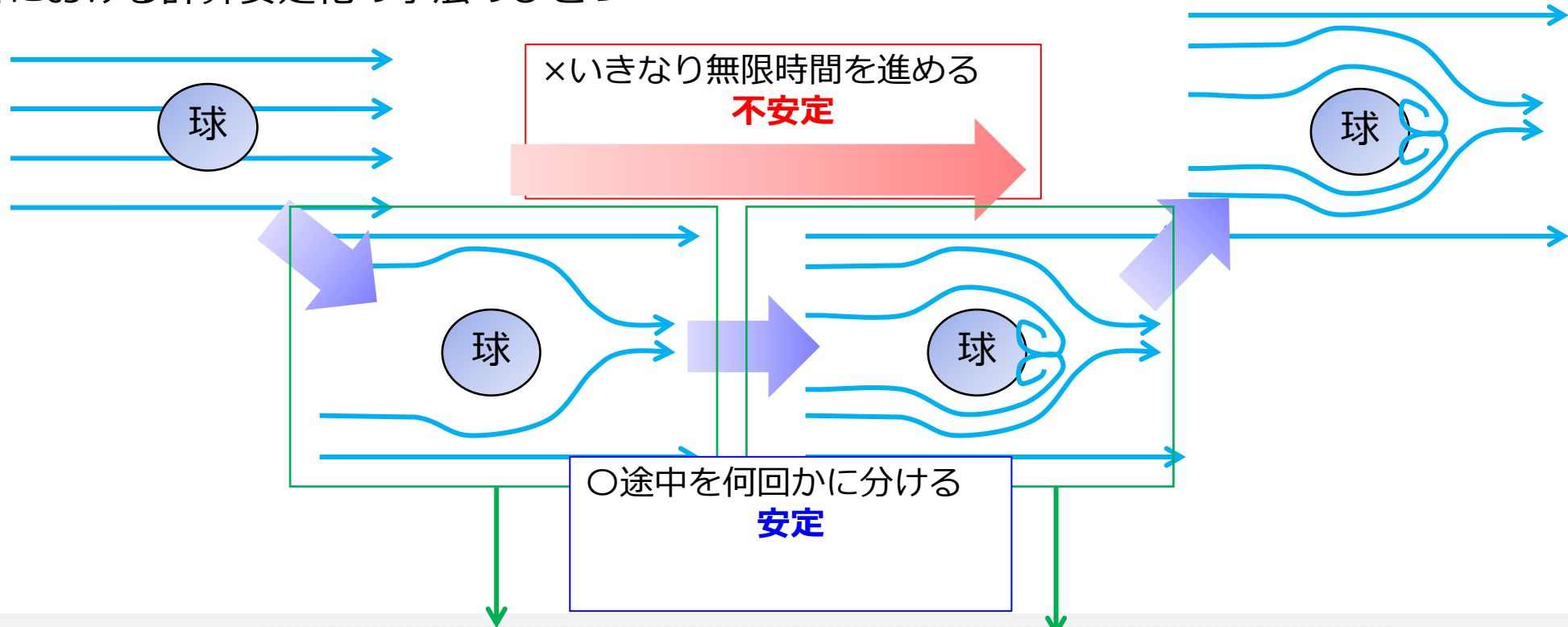
境界条件の種類

- **ディリクレ境界条件**：境界面の値を直接指定する条件
流速規定、質量流規定、圧力規定、静圧規定
- **ノイマン境界条件**：変数の勾配を与える条件 $\frac{\partial f}{\partial x} = 0$
フリースリップ条件、断熱条件
- **周期境界条件**：2つの面の値が等しくなるという条件



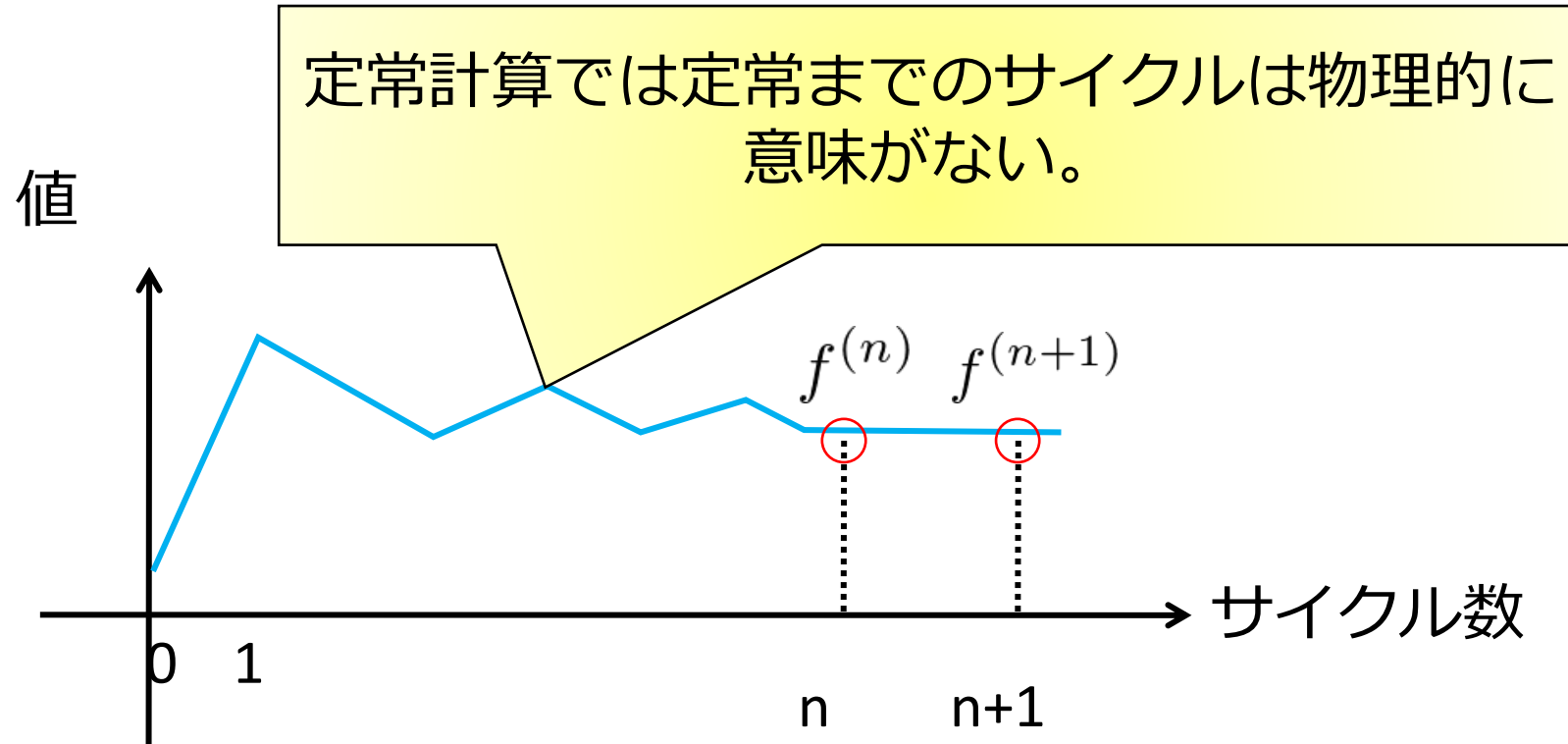
□緩和係数

定常解析における計算安定化の手法のひとつ



$$\begin{array}{ccc} \text{前のサイクルの解} & + & \text{今から求めたい解} \\ 30\% & & 70\% \\ \hline & & \text{緩和係数}0.7 \\ \hline & & \text{緩和係数を考慮した解} \end{array}$$

□ 定常収束判定

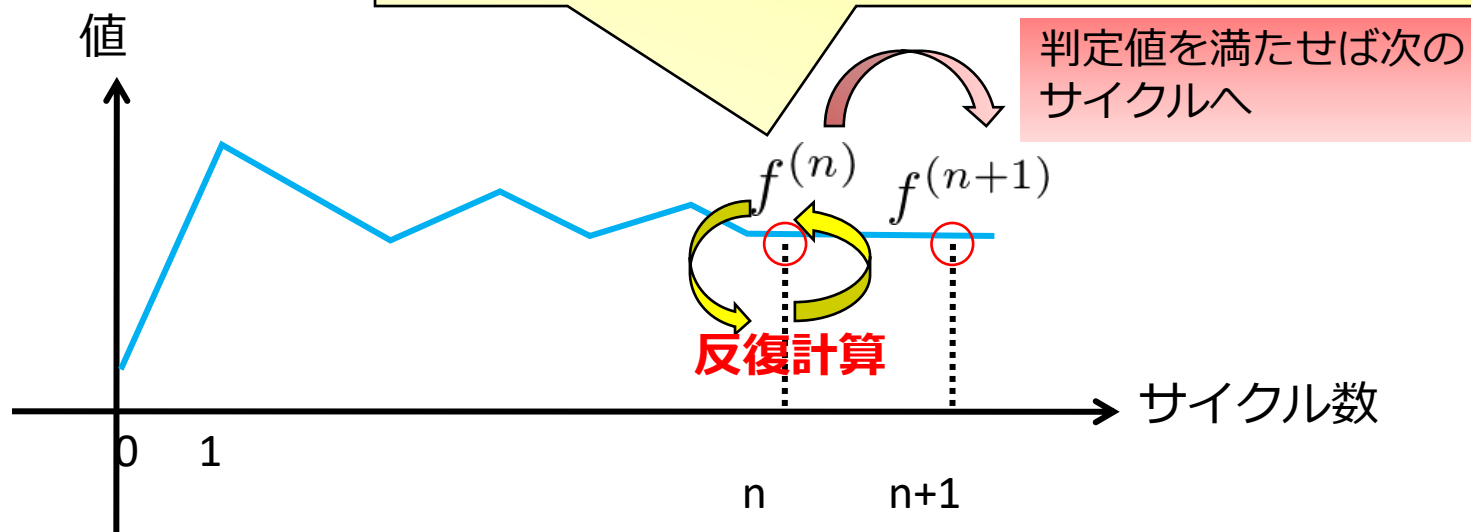


$|f^{(n+1)} - f^{(n)}| < \text{定常収束判定値}$
のとき定常状態になったとみなして計算終了。

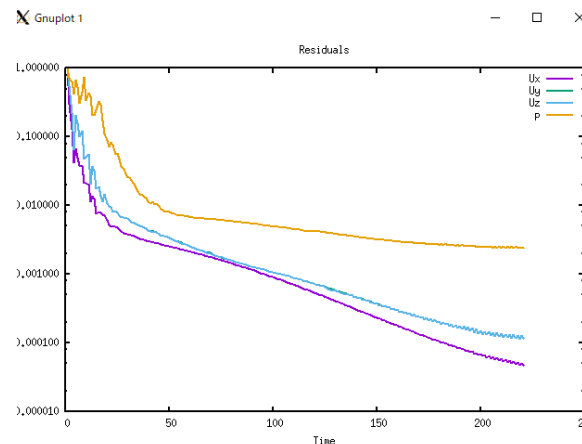
□ 残差

$Ax = b$ を解くために**反復計算**が必要。

真の解にどれだけ近いかを判定：**残差** $R = \left| \frac{b - Ax}{b} \right|$



毎サイクルの初期の残差をgnuplotで表示



system/fvSchemes

```
ddtSchemes{
    default steadyState;
}
gradSchemes{
    default cellLimited Gauss linear 1;
}
divSchemes{
    default none;
    div(phi,U) bounded Gauss limitedLinearV 1;
    div((nuEff*dev2(T(grad(U))))) Gauss linear;
    div(phi,k) bounded Gauss limitedLinear 1;
    div(phi,epsilon) bounded Gauss limitedLinear 1;
}
laplacianSchemes{
    default Gauss linear corrected;
}
interpolationSchemes{
    default linear;
}
snGradSchemes{
    default corrected;
}
fluxRequired{
    default no;
    p;
}
```


system/fvSolution

```
solvers{
  p{
    solver PCG;
    preconditioner DIC;
    tolerance 1e-06;
    relTol 0.01;
  }
  U{
    solver PBiCG;
    preconditioner DILU;
    tolerance 1e-05;
    relTol 0.1;
  }
  k{
    solver PBiCG;
    preconditioner DILU;
    tolerance 1e-05;
    relTol 0.1;
  }
  epsilon{
    solver PBiCG;
    preconditioner DILU;
    tolerance 1e-05;
    relTol 0.1;
  }
}

SIMPLE{
  nNonOrthogonalCorrectors 0;
  residualControl{
    p 0.01;
    U 0.001;
    k 0.001;
    epsilon 0.001;
  }
  relaxationFactors{
    fields{
      p 0.1;
    }
    equations{
      U 0.3;
      k 0.3;
      epsilon 0.3;
    }
  }
}
```

おわり