はじめてのOpenFOAM®

その１

富山県立大学　中川慎二

オープンCAE勉強会@富山 第80回 2020年5月23日

Acknowledgement: OPENFOAM® is a registered trade mark of OpenCFD Limited, producer and distributor of the OpenFOAM software via www.openfoam.com.

この講習会では，ユーザーガイドやチュートリアルガイドを参照しながら，作業を進めます。最新のユーザーガイドなどは下記サイトで参照可能です。（最近の User Guide は　CC-BY-NC-ND ライセンスです。ご注意ください。）

OpenFOAM®　User Guide（オリジナル）

https://www.openfoam.com/documentation/user-guide/

OpenFOAM®　Tutorial Guide

https://www.openfoam.com/documentation/tutorial-guide/

ソフトウエアマニュアル翻訳（OpenCAE学会）

https://gitlab.com/OpenCAE/OpenFOAM/tree/master/doc

重要なメモ(環境変数)

端末でOpenFOAMの環境を有効にし，環境変数の内容を表示する下記コマンドを実行し，結果をメモしましょう。

echo $WM_PROJECT_DIR

---

echo $FOAM_TUTORIALS

---

echo $FOAM_RUN

コース概要

目的：OpenFOAMを利用し，流動シミュレーションに必要な一連の作業を体験する。
OpenFOAMの基本的な使い方を学ぶ。

OpenFOAMマニュアル（ユーザーガイド）に掲載されている例題（チュートリアル）に，実際にコンピュータを使って取組む。

OpenFOAMで　できそうなこと，できないことなど，講習終了後の活用に向けた話題も取り上げる。

スケジュール

1. 使用システム説明
2. OpenFOAM概要
   - OpenFOAMとは?，使用例紹介，ディレクトリ構造
3. 例題：キャビティ流れ
   - 概要，格子生成，条件設定，流体解析，可視化，格子改造
4. 例題：ダムの崩壊
   - 概要，格子生成，条件設定，流体解析，可視化
5. さらにOpenFOAMを使うために
   - 情報元，主なソルバー説明，質疑応答

1.講習会で使用する計算機の環境

ハードウェア

ノートPC

ソフトウェア

OS：Windows＋VirtualBox（仮想マシン実行環境）

仮想マシン

OS: Xubuntu20.04 64bit

（Ubuntuベースの軽量ディストリビューション）

シミュレーションソフトウェア:

OpenFOAM v1912

http://www.openfoam.com/

OpenFOAM v7

http://www.openfoam.org/

動作環境の確認

- Windowsメニューから，Virtualboxを起動
- Virtualboxのメニューから，「ファイル」ー「仮想アプライアンスのインポート」を選択
- 取得した *.ovaファイルを選択する。
- インポート作業が終了するまで待つ。
- 仮想マシンを起動する

２．OpenFOAM概要
オープンソース・CFDソフトウェア
“OpenFOAM”について

数値シミュレーションについて

CFD (Computational Fluid Dynamics):数値流体力学，流体の数値シミュレーション

実現象→物理モデル→数学モデル→シミュレーションモデル　（様々な仮定）

作業の流れ：

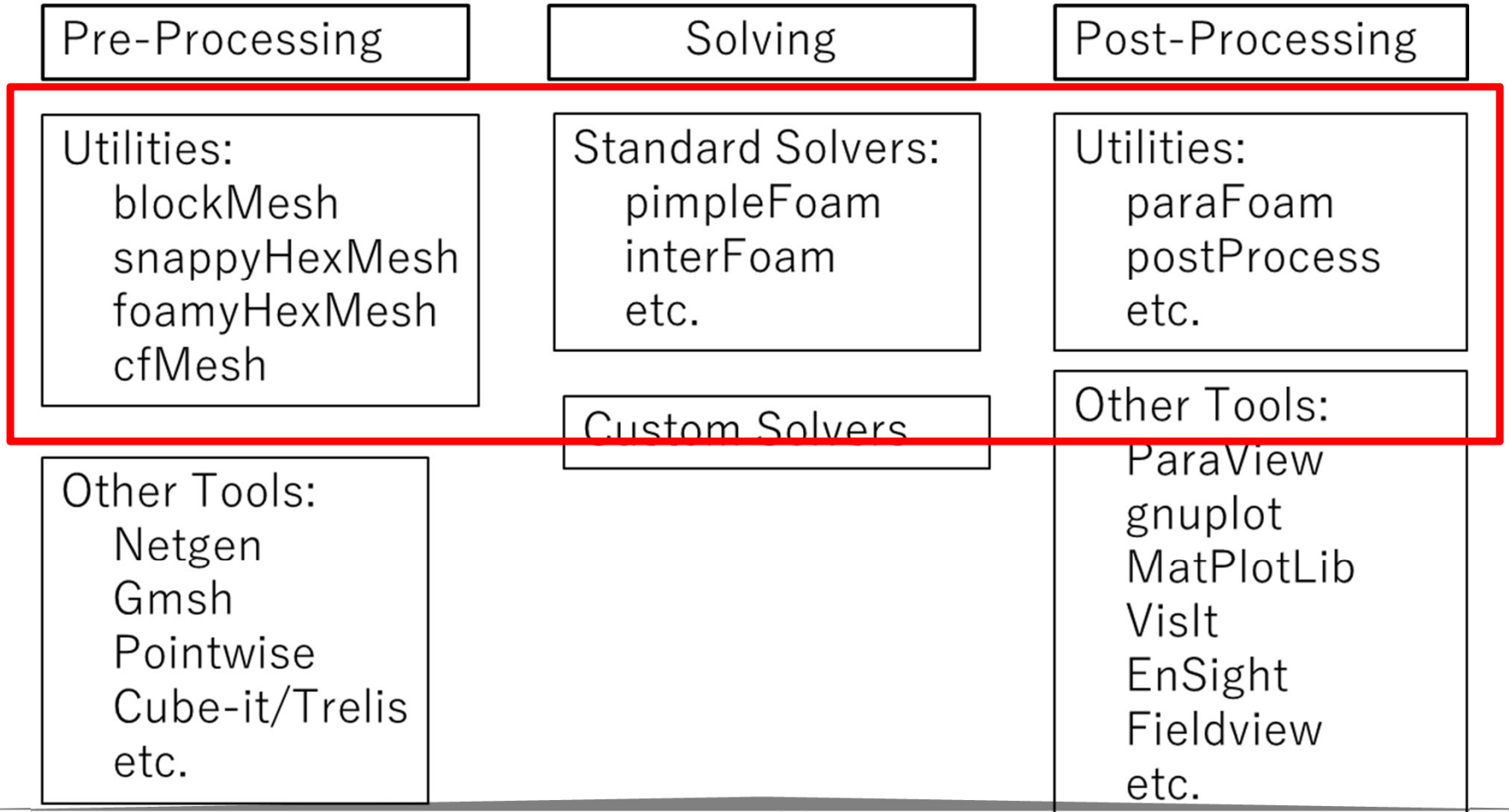
プレ処理　→　計算　→　ポスト処理

数値シミュレーションについて

作業の流れ：

User Guide　C.1

プリ処理　→　計算　→　ポスト処理



OpenFOAMで利用するソフト

1．OpenFOAM
OpenFOAM本体，c++で記述されたプログラムの集まり。
各種ユーティリティを含む

2．paraview
オープンソースの可視化ソフトウェア
http://www.paraview.org/
Windows版，Linux版，MacOS版有り

3．メッシュ生成ソフトウェア
効率的な運用には　必要

4．gccなど
コンパイラ（GNU Compiler Collection）
ソースコードをコンパイルする場合には 必要

ディレクトリ構造（OpenFOAM全体）

【インストール先】

$WM_PROJECT_DIR/　←インストール ディレクトリ

- ├ applications　←アプリケーションのソースファイル
- ├ bin　←シェルスクリプト
- ├ doc　←マニュアル
- ├ etc　←設定ファイル
- ├ platforms　←コンパイル済み実行ファイル（ソルバ, ライブラリ）
- ├ src　←各種部品のソースファイル
- ├ tutorials　←例題ファイル（オリジナル）
- └ wmake　←コンパイル関連（通常使用しません）

【作業フォルダ】

ユーザーのホームディレクトリ（$HOME）内に作成

$FOAM_RUN が標準的な場所，エイリアス run で移動

ソースコード

- ファイルブラウザで下記ディレクトリに移動
  $FOAM_SOLVERS/incompressible/icoFoam
- メインプログラム
  icoFoam.C

Docker の場合

- 見たいものをユーザディレクトリにコピーし，ファイルブラウザからアクセス。（端末でエディタが使えれば，コピーなしで直接参照。）

```
cp  -rp  $FOAM_SOLVERS/incompressible/icoFoam/  $WM_PROJECT_USER_DIR/solver/
```

ソースコード

- プログラム

```
fvVectorMatrix UEqn (
            fvm::ddt(U)
          + fvm::div(phi, U)
          - fvm::laplacian(nu, U)    );
solve(UEqn == -fvc::grad(p));
```

時間微分項

対流項

粘性項

圧力勾配項

- 基礎式

$$\frac{\partial \vec{U}}{\partial t} + \vec{U}\frac{\partial \vec{U}}{\partial \vec{X}} = -\frac{1}{\rho}\frac{\partial p}{\partial \vec{X}} + \nu\left(\nabla^2 \vec{U}\right)$$

ディレクトリ構造（例題）

インストールディレクトリ
(Docker内の/opt/OpenFOAM/OpenFOAM-v1912)

$WM_PROJECT_DIR ←インストールディレクトリ
　└ tutorials ←標準例題ディレクトリ（ $FOAM_TUTORIALS ）
　　└ incompressible ←非圧縮性流体ソルバ　ディレクトリ
　　　└ icoFoam ←icoFoamソルバー　ディレクトリ
　　　　└ cavity ←cavity例題の親ディレクトリ
　　　　　└ cavity ← cavityケース　ディレクトリ

この中に，各種計算条件を記載した
ファイルや，計算結果が収納される。
詳細は次のスライド。

標準的な作業ディレクトリ構造

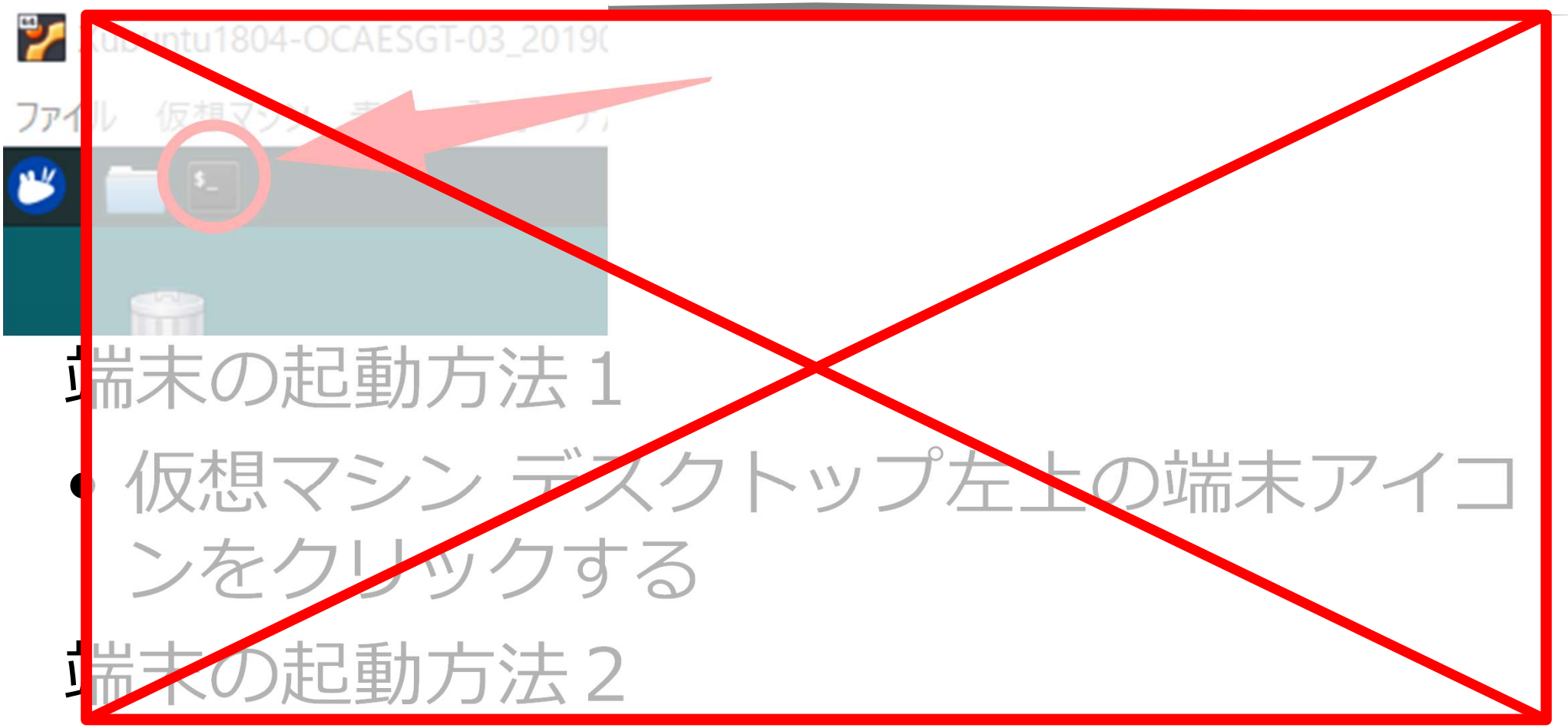
インストール先ではなく，各ユーザの作業用ディレクトリ（$FOAM_RUN）にコピーする。

ディレクトリ構造（ケース詳細）

cavity/　← cavityケース　ディレクトリ
　├○ system/　← 計算制御関連 ディレクトリ
　│　　├ blockMeshDict　←メッシュ生成情報記述ファイル
　│　　├ controlDict　←計算制御のテキストファイル
　│　　├ fvSchemes　←離散化関連設定ファイル
　│　　└ fvSolution　←解法関連設定ファイル
　├ ○ constant/　← 物性・メッシュ関連 ディレクトリ
　│　　├ transportProperties　←流体物性設定ファイル
　│　　└○ polyMesh/　←メッシュ関連ディレクトリ(メッシュ生成後)
　│　　　└ boundary　←境界（メッシュ生成後に作成される）
　│　　　・・・・　←メッシュ生成後にはファイルが増える
　└ ○*time directories*［数字］　← 結果を格納するディレクトリ
　　時刻がディレクトリ名となり，内部に複数のファイル(U，pなど)を格納する。初期条件・境界条件はここで設定

詳細な説明は，User Guide 2.1節

作業：端末の起動



端末の起動方法１

- 仮想マシン デスクトップ左上の端末アイコンをクリックする

端末の起動方法２

- 仮想マシン デスクトップ左上のメニュー（ネズミアイコン）をクリックし，Terminal Emulatorを起動する

作業：ファイルマネージャー起動

~~ファイルマネージャーの起動方法 1~~

- ~~仮想マシン デスクトップ左上のフォルダアイコンをクリックし，OpenFolderを選ぶ~~

~~ファイルマネージャーの起動方法 2~~

- 仮想マシン デスクトップ左上のメニュー（ネズミアイコン）をクリックし，File Managerを起動する

３．例題
キャビティー流れ

Tutorial Guide

Chapter 2 Incompressible flow　2.1 Lid-driven cavity flow

https://openfoam.com/documentation/tutorial-guide/tutorialse2.php#x6-60002.1

作業：例題のコピー

コピー元:
$FOAM_TUTORIALS/incompressible/icoFoam/cavity/cavity/

コピー先: $FOAM_RUN/cavity

[方法1] 端末から下記コマンドを実行

```
cd $FOAM_RUN
ls                          ←ディレクトリ・ファイルのリストを表示する
rm  -r  cavity      ← もし，すでにcavityが存在するときは削除
cp  -r  $FOAM_TUTORIALS/incompressible/icoFoam/cavity/cavity/  ./
```

[方法2] ファイルマネージャーでCopy, Paste

作業：ディレクトリ構造の確認

- 標準例題ディレクトリ

$FOAM_TUTORIALS

上記は変数，実際の場所は下記を指す。

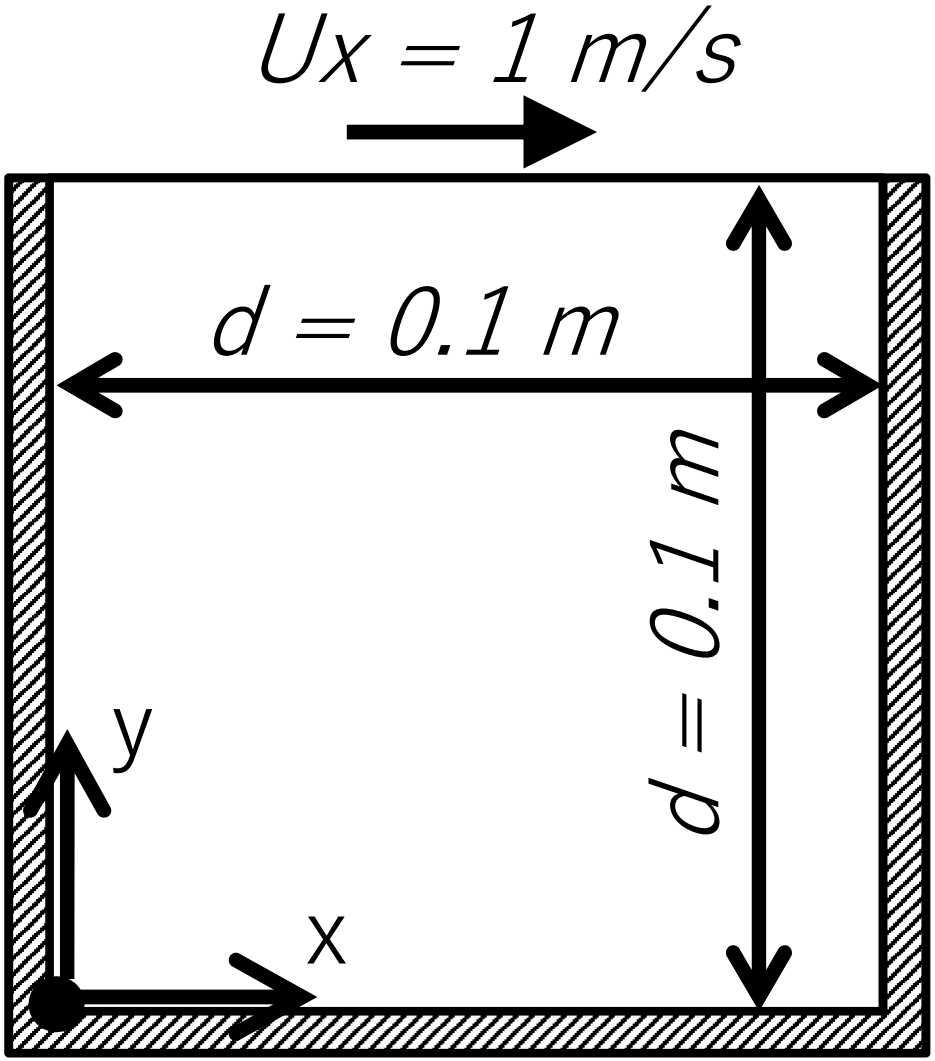
$WM_PROJECT_DIR/tutorials

- 問題の種類ごとにディレクトリが分かれている
- その中に，いろいろな場合に対する例題が納められている

https://develop.openfoam.com/Development/openfoam/-/tree/master/tutorials

例題１：キャビティ流れ

Tutorial Guide 2.1節



正方形容器

上蓋が移動

容器内に非圧縮性流体
（incompressible fluid ）

2次元流，層流，一定温度

速度と圧力を求める

Standard Application の
icoFoamを利用する

PIRT：重要度ランクテーブルの例

Phenomena Identification and Ranking Table

現象	内容	重要度	知見の程度	採否
A	非定常性	中	高	○
B	物体の移動	高	高	△*
C	圧縮性	低	高	ー
D	三次元性	低	高	ー**
E	乱流	低	中	ー
F	温度	低	中	ー
G	非ニュートン性	低	中	ー

＊ 上壁は移動するが，計算領域には含まれないため，境界条件として扱う。
＊＊ 全ソルバは3次元用である。セルと境界条件の設定で2次元計算とする。

icoFoamソルバは層流のみの対応のため，実際に使用することは少ない。学習用的なソルバと考えられる。

標準ソルバー：icoFoam

Standard Solvers

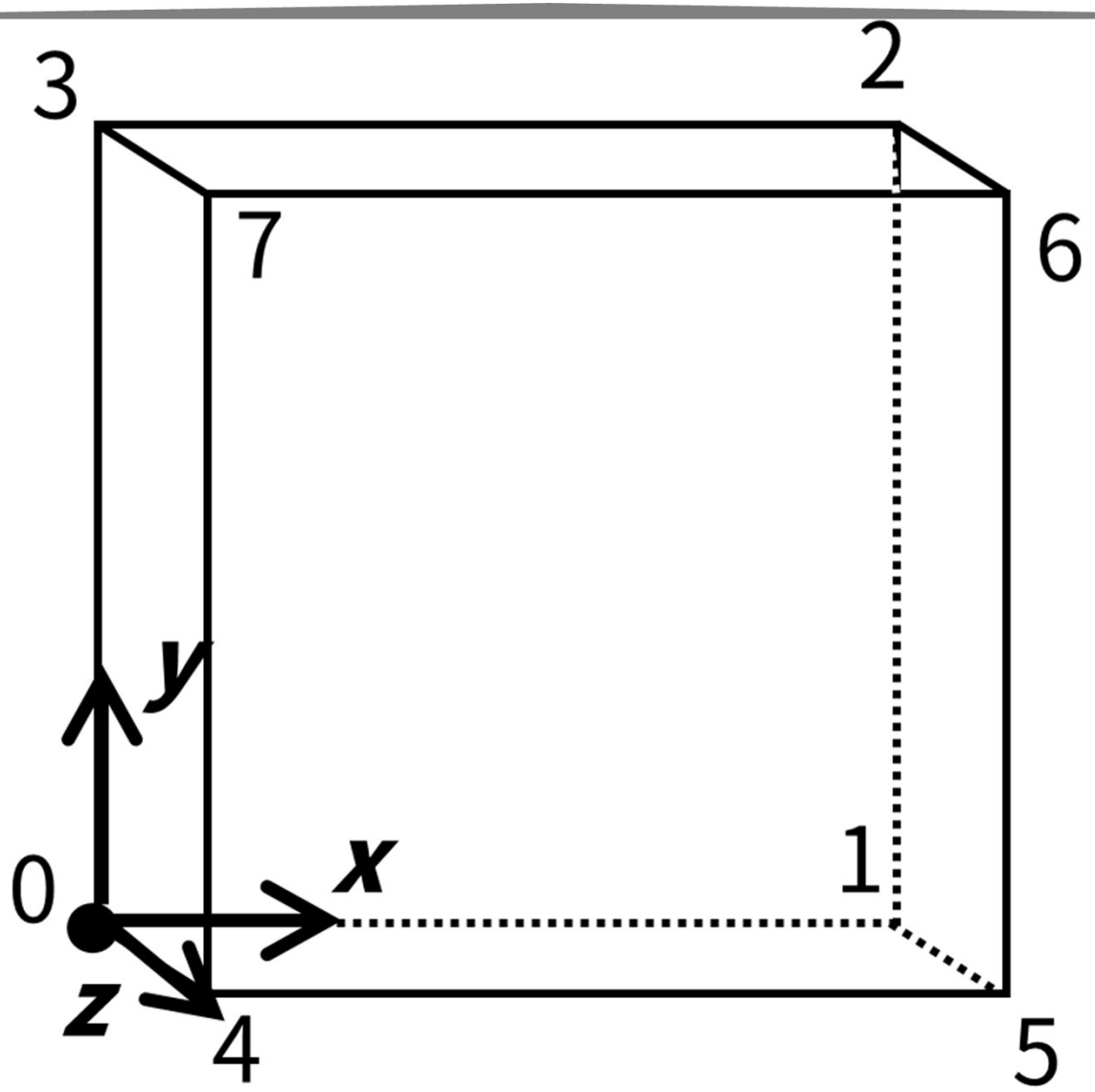
Incompressible flow

icoFoam

Transient solver for incompressible, laminar flow of Newtonian fluids

モデルの幾何形状

User Guide 2.1.1.1



節点とblock構成

プリ処理 Pre-processing

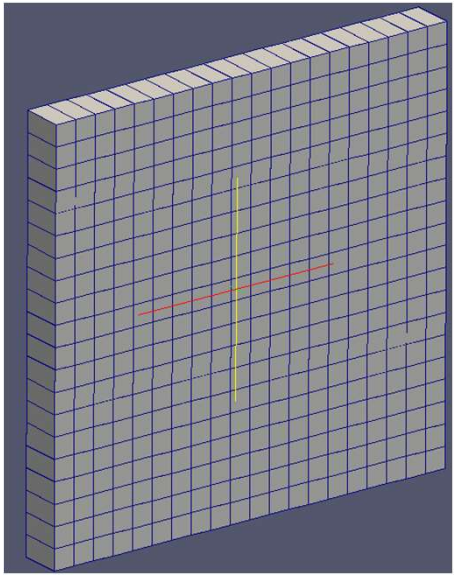
User Guide 2.1.1節

作業内容

- メッシュ生成（Mesh generation）
- 境界条件と初期条件設定（Boundary and initial conditions）
- 物性値設定（Physical properties）
- 計算制御設定（Control）
- 離散化と行列解法の設定（Discretisation and linear-solver settings）

メッシュ

- 計算領域を多くの小さな領域に分割する
- 小領域をセルという
- 分割線をメッシュという
- シミュレーションでは，各セルでの物理量を予測する



メッシュ生成

単純なメッシュ

- OpenFOAMで作成できる
- blockMeshDictというファイルにメッシュの生成方法を記述 → メッシュ生成コマンドblockMeshを実行

少し複雑なメッシュ

- OpenFOAMで作成できる
- 任意形状のSTLファイルに適合したメッシュの生成　→　snappyHexMesh ユーティリティやcfMeshを利用

【作業：ファイルマネージャ】

- ファイルマネージャーで，

$FOAM_RUN/cavity/system

まで移動し，blockMeshDictをダブルクリックして開く

blockMeshディクショナリの基本構造

User Guide 2.1.1.1節

blockMeshDictディクショナリの基本構造

scale　単位変換の係数　(旧名 convertToMeters)
　OpenFOAM内での基本単位はm
　たとえば，mm単位で記入するとき，この係数を0.001とする

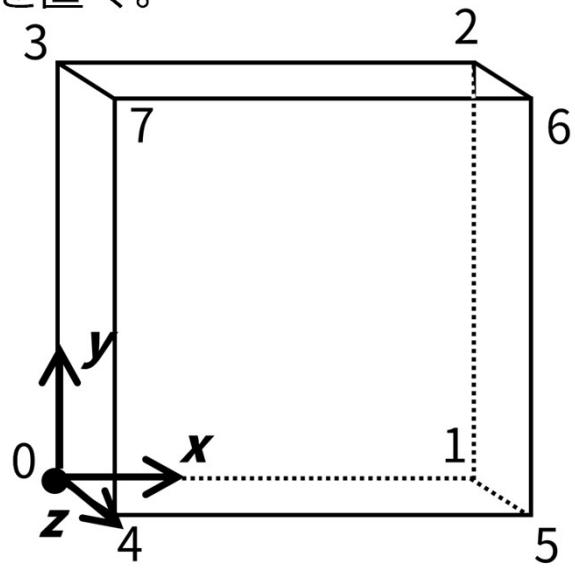
vertices　節点　　座標を与える
blocks　ブロック
patches　面に関する情報（境界条件）

メッシュ作成指令書：blockMeshDict

scale 0.1;　　←　これから書く数字を，0.1倍すると，単位がmになる

```
vertices　←　節点
(
    (0 0 0)　　←　0から7 までの各点の座標。この値を0.1倍するとm単位になる。
    (1 0 0)　　←　この場合，x座標が1×0.1=0.1mの位置に点を置く。
    (1 1 0)
    (0 1 0)
    (0 0 0.1)
    (1 0 0.1)
    (1 1 0.1)
    (0 1 0.1)
);

blocks　←　ブロック（直方体（hex），節点番号で指定する）
(
    hex (0 1 2 3 4 5 6 7) (20 20 1) simpleGrading (1 1 1)
        ←　節点0～7 で直方体を作る。　x，y，z 方向に20，20，1分割（等間隔）。
);
```

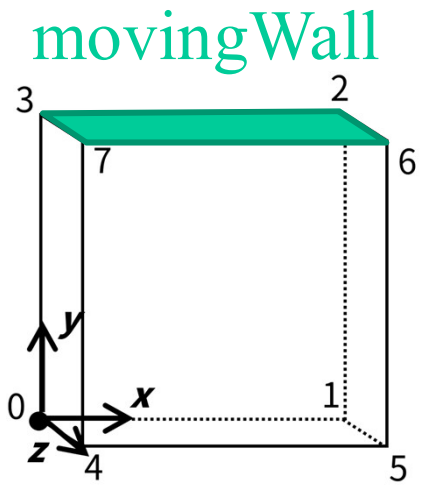


User Guide 2.1.1.1節

メッシュ作成指令書：blockMeshDict

User Guide 2.1.1.1節

```
boundary      ←　同じ境界条件をまとめる
(
    movingWall      ←　境界条件に名前をつける：一定の速度で動く壁面
    {
        type wall;      ←　タイプを壁面に設定する
        faces ( (3 7 6 2) );      ←　４つの節点で構成される面
    }
    fixedWalls      ←　境界条件に名前をつける： 固定された壁（速度は０）
    {
        type wall;      ←　タイプを壁面に設定する
        faces       ←　４つの節点で構成される面　が３個
            (  (0 4 7 3)     (2 6 5 1)  (1 5 4 0)  );
    }
    frontAndBack      ←　境界条件：対称面（物理量の勾配は０）
    {
        type empty;      ←　タイプをemptyに設定する（2次元計算のため）
        faces       ←　４つの節点で構成される面　が２個
        (  (0 3 2 1)    (4 5 6 7)      );
    }
);
```



【作業:端末】メッシュ生成：blockMeshの実行

User Guide 2.1.1.1節

- Terminal（端末）を起動し,OpenFOAMが使える状態にする。
- 対象ケースの場所（$FOAM_RUN/cavity/）に移動するため，下記コマンドを実行する。

```
run
cd  cavity
```

- 端末で下記コマンドを実行し，メッシュ生成ユーティリティblockMeshを実行する。

```
blockMesh
```

- 端末に，実行結果が表示される。エラーメッセージが表示されていないか，確認する。

境界条件および初期条件

- 時刻 0 のディレクトリに，初期条件が記述されたファイル（ p と U）を用意する。

【作業：ファイルマネージャ】

現在のケースディレクトリ($FOAM_RUN/cavity)の下にある 0 ディレクトリまで移動し，ファイル p をダブルクリックして開く。

/0/pファイルの読み方

```
                    Kg  m  s
dimensions      [0 2 -2 0 0 0 0];
```

この変数 p の単位は m^2/s^2
（圧力／密度となる）

```
internalField   uniform 0;
```

内部の値は 一様で 0

```
boundaryField
{
    movingWall
    {
        type            zeroGradient;
    }

    fixedWalls
    {
        type            zeroGradient;
    }

    frontAndBack
    {
        type            empty;
    }
}
```

境界条件

movingWall，fixedWallsという名の境界では，境界面に垂直な方向の圧力勾配は 0。

frontAndBack境界では，
empty →2次元流れの境界

/0/Uファイルの読み方

Kg m s

```
dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    movingWall
    {
        type            fixedValue;
        value           uniform (1 0 0);
    }

    fixedWalls
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }

    frontAndBack
    {
        type            empty;
    }
}
```

この変数 U の単位は m/s

ベクトルなので3成分をもつ

内部の値は 一様で(0,0,0)

境界条件

movingWall境界では，速度は一定値。x方向に1m/s

fixedWalls境界では，速度は一定値 0。静止。

frontAndBack境界では，empty →2次元流れの境界

物性値設定

User Guide 2.1.1.3節

- ～Propertiesという名のディクショナリに物性値を記録（ケース・ディレクトリ/constant/に保存）
- icoFoamでは，動粘度νをtransportPropertiesディクショナリで与える。
- 今回は　$\nu = 0.01\ m^2s^{-1}$ と設定
  - レイノルズ数　$Re = U\,d\,/\,\nu = 1 \times 0.1 / 0.01 = 10$

【作業：ファイルマネージャ】

$FOAM_RUN/cavity/constant の下にある transportPropertiesファイルをダブルクリックして開き，内容を確認。

計算制御設定

User Guide 2.1.1.4節

- 計算時間，データ読み込み，書き出し制御等に関する情報は　controlDict ディクショナリに記録
- controlDict ディクショナリは，ケースディレクトリ/systemディレクトリに保存する

【作業：ファイルマネージャ】

現在のケースディレクトリ

$FOAM_RUN/cavity

の下にある system ディレクトリまで移動し，ファイル controlDict をダブルクリックして開く。

controlDict の読み方 (1)

User Guide 2.1.1.4節

application	icoFoam;	ソルバは　icoFoam
startFrom startTime	startTime; 0;	計算をstartTime欄で指定して時刻から始める。(今回は0秒)
stopAt endTime	endTime; 0.5;	計算をendTime欄で指定した時刻で止める。(今回は0.5秒)
deltaT	0.005;	時間の刻み幅を指定する。
writeControl writeInterval	timeStep; 20;	結果ファイルの書出し時刻を指定する。writeIntervalで指定した回数ごとに書き出す。(今回は0.005秒×20回=0.1秒毎)

より詳細な説明(他の選択肢など)は、ユーザマニュアル4.3節 p.111

controlDict の読み方 (2)

User Guide 2.1.1.4節

`purgeWrite 0;`	書き出しファイル数を制限するか（今回は制限しない。）
`writeFormat ascii;`	書き出しファイルをASCII形式に
`writePrecision 6;`	書き出すデータの有効桁数
`writeCompression off;`	書き出しファイルの圧縮/非圧縮
`timeFormat general;` `timePrecision 6;`	書き出しディレクトリの名前の付け方と桁数
`runTimeModifiable true;`	各タイムステップの開始時に、各種ディクショナリを再読み込みするかどうか

より詳細な説明は，ユーザマニュアル4.3節 p.111

離散化と行列解法の設定

User Guide 2.1.1.5節

有限体積法での離散化方法

- ケース/system/fvSchemesディクショナリ

行列解法、トレランス、アルゴリズム設定など

- ケース/system/fvSolutionディクショナリ

メッシュの確認

User Guide 2.1.2節

ポスト処理ソフトParaViewを使って，メッシュを確認する

【作業：端末】

- ケース「cavity」ディレクトリにいることを確認するため、下記コマンドを実行する。

```
pwd
```

- /home/user/OpenFOAM/user-v1912/run/cavityと表示されればよい。違う場所にいるときは、下記コマンドを実行する。

```
cd  $FOAM_RUN/cavity
```

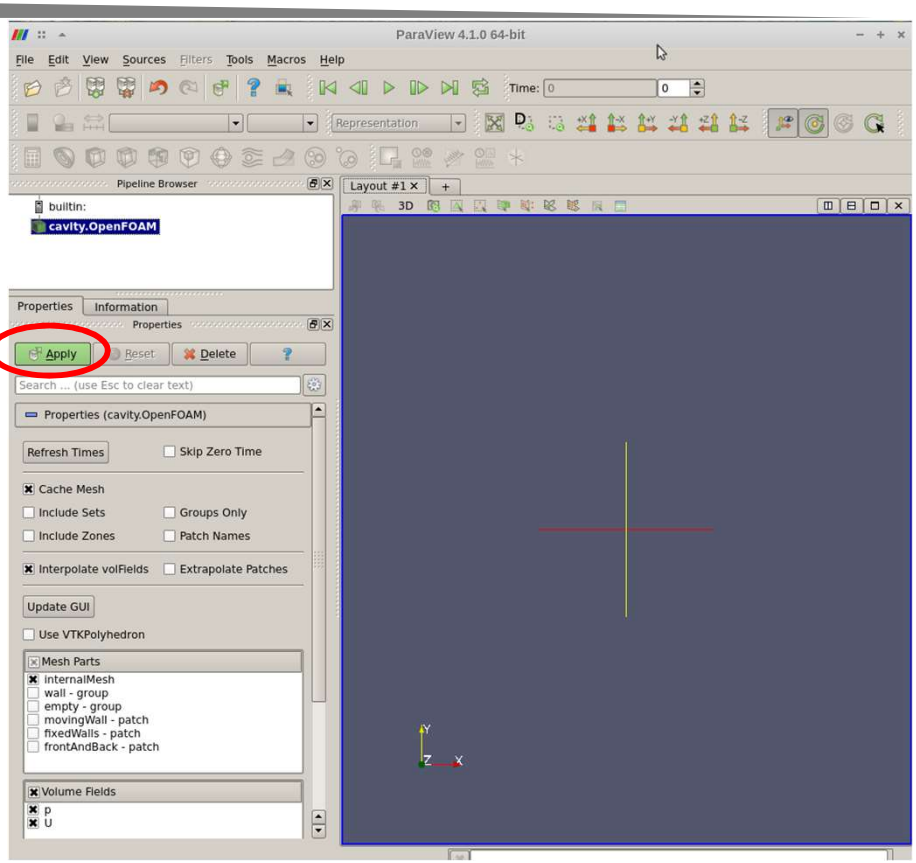
- 下記のコマンドを実行する

```
paraFoam
```

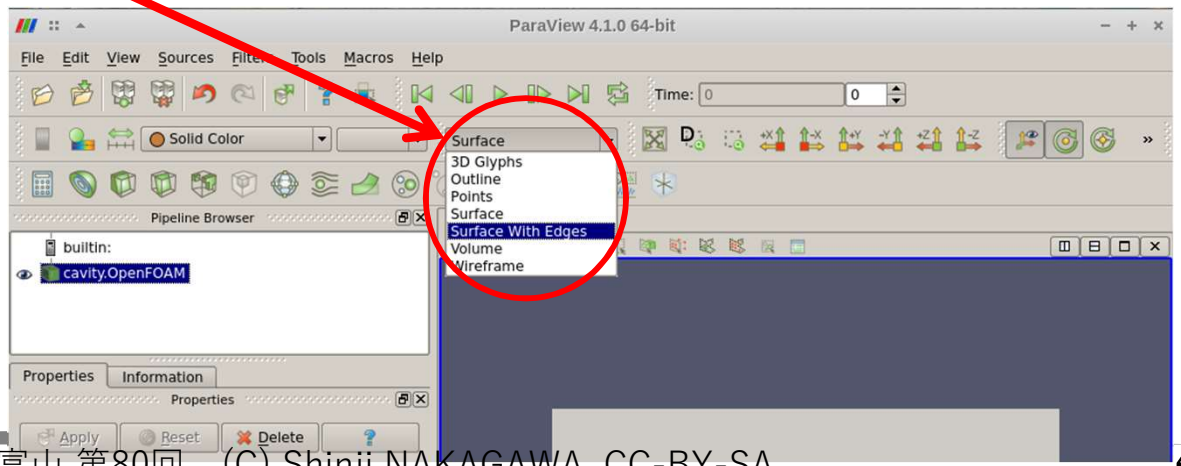
- ParaViewが起動する

メッシュの確認

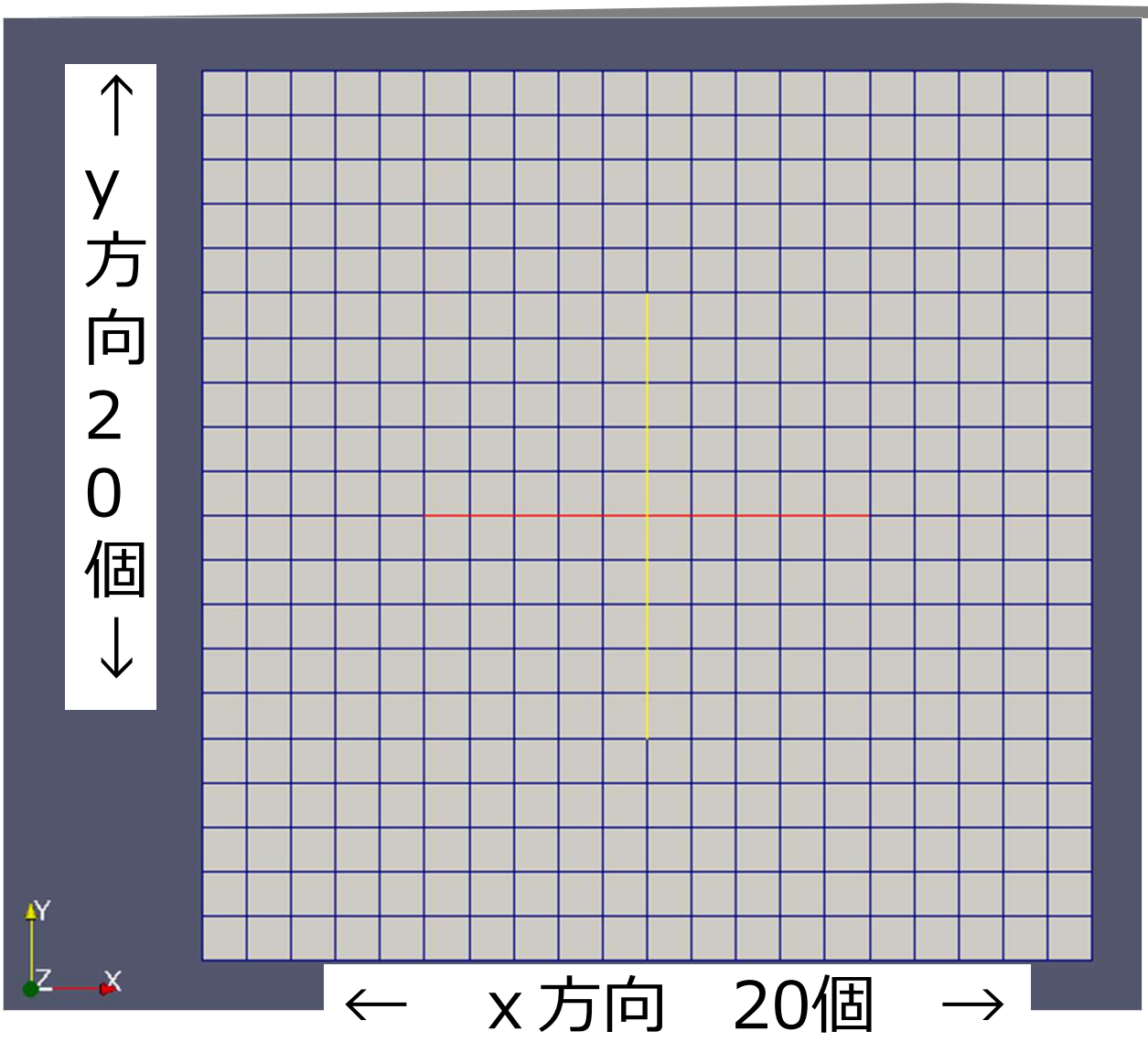
- ParaViewが起動する
- Applyボタン（緑）を押す



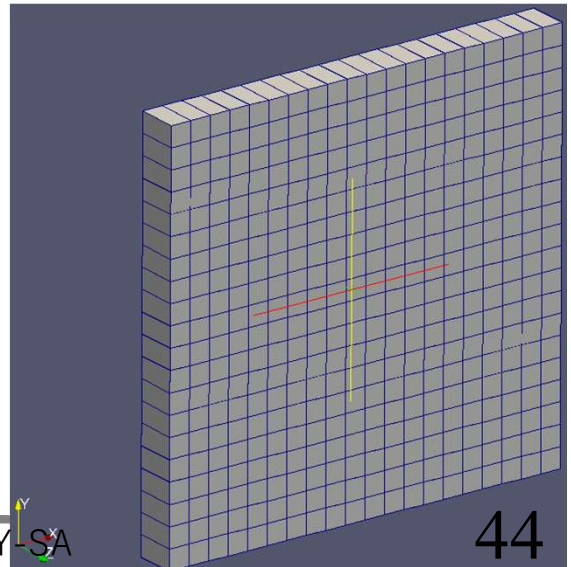
- 上部メニューで、
  「Surface」から
  「Surface Edges」or
  「Wireframe」に
  変更する。



メッシュの確認



- 左のような格子が表示されることを確認する。
- クリック＆ドラッグでモデルを動かし、z方向の格子も確認する。



確認後、paraviewを終了する。

計算　Solving

User Guide 2.1.3節

計算を実行する

【作業：端末】

- ケース「cavity」ディレクトリにいることを確認するため、下記コマンドを実行する。

```
pwd
```

- /home/user/OpenFOAM/user-v1912/cavityと表示されればよい。違う場所にいるときは、下記コマンドを実行する。

```
cd $FOAM_RUN/cavity
```

- 下記のコマンドを実行する

```
icoFoam
```

- 計算のレポートが端末に表示される。ケースディレクトリに結果が出力される。（0.1から0.5）

ポスト処理 Post-processing

User Guide 2.1.4節

ParaView（オープンソースソフトウェア）を使って結果を可視化

OpenFOAMの結果を可視化するコマンドは
paraFoam

このコマンドは，OpenFOAMのケースディレクトリに「ケース名.OpenFOAM」というファイルを作り、ParaViewを起動する。

結果の可視化

User Guide 2.1.4節

ポスト処理ソフトParaViewを使って，メッシュを確認する

【作業：端末】

- ケース「cavity」ディレクトリにいることを確認するため、下記コマンドを実行する。

```
pwd
```

- /home/user/OpenFOAM/user-v1812/run/cavityと表示されればよい。違う場所にいるときは、下記コマンドを実行する。

```
cd $FOAM_RUN/cavity
```

- 下記のコマンドを実行する

```
paraFoam
```

- ParaViewが起動する

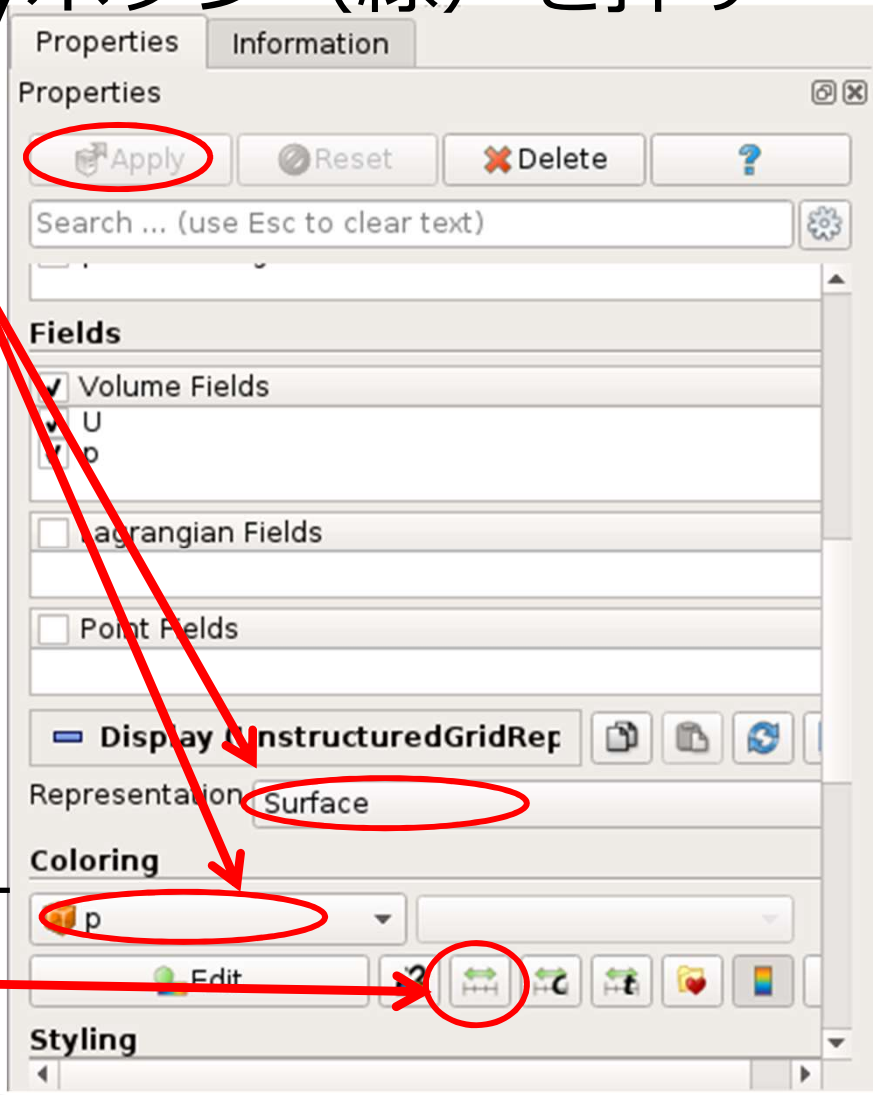
圧力の可視化

User Guide 2.1.4節

【作業：ParaView】

- ParaViewが起動したら、Applyボタン（緑）を押す
- 「Properties」ウィンドウで、
  Coloring を　■ *p*
  Style を Surface にする。
- 上部メニューの再生ボタンを押す

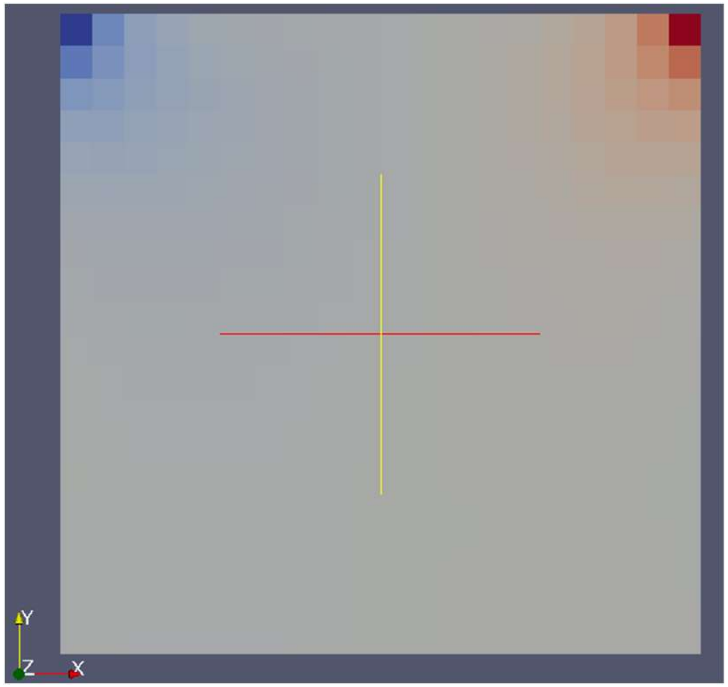
- 「 Properties 」ウィンドウで、
  Rescale to Data Rangeを押す



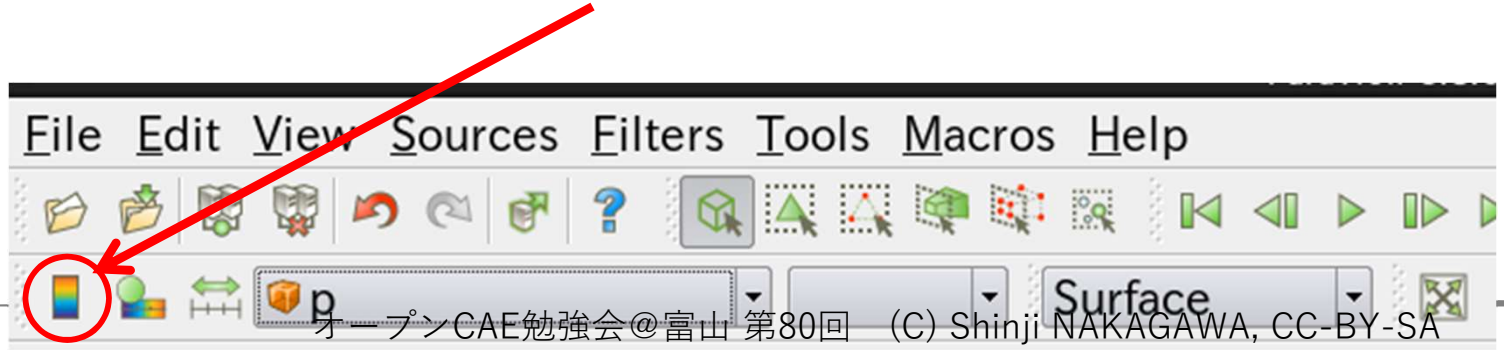
圧力の可視化

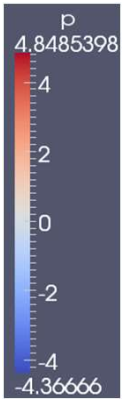
User Guide 2.1.4節

- 圧力分布が表示される。
- 「Edit Color Map」
  →「Choose Preset」
  →「上から2番目」で
  　　青から赤の虹色表示



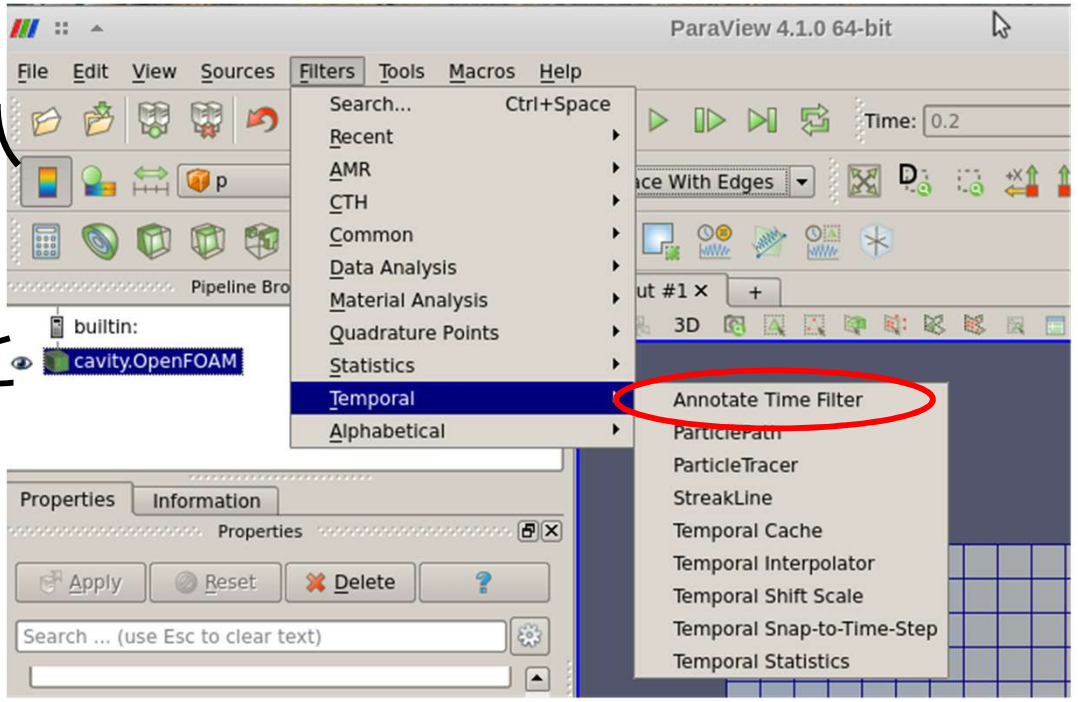
- 上部メニューの「Toggle Color Legend Visibility」ボタンを押すと、凡例（カラーバー）が表示される。





時刻の表示:Annotate Time フィルタ

- Pipeline Browserで cavity.OpenFOAMをハイライト
- Filters ー Temporal と進み、Annotate Time Filterをクリック
- Applyボタンを押す
- 表示している結果の時刻が書かれる。
- Text Positionの設定によっては，クリック&ドラッグ可能。

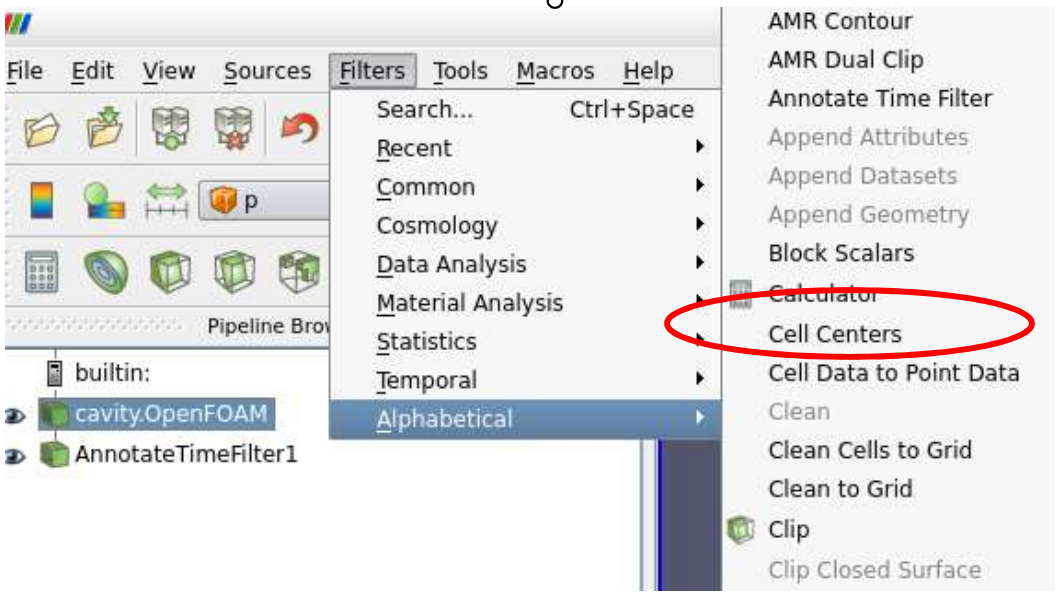


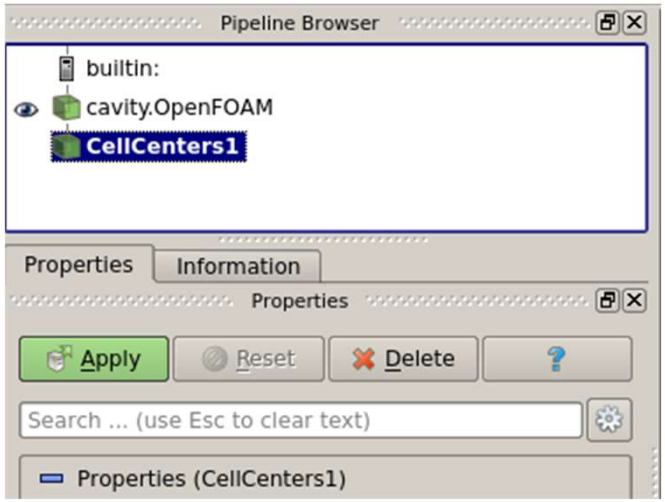
速度ベクトル図 Glyph （1）

セル中心での速度ベクトル

User Guide 2.1.4.2節

- Pipeline Browserで cavity.OpenFOAMが選ばれていることを確認
- Filtersメニューから、Alphabeticalと進み、CellCentersをクリックする。
- Propertiesタブを選び、Applyボタンを押す



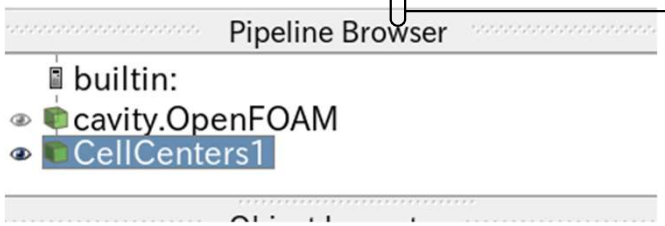


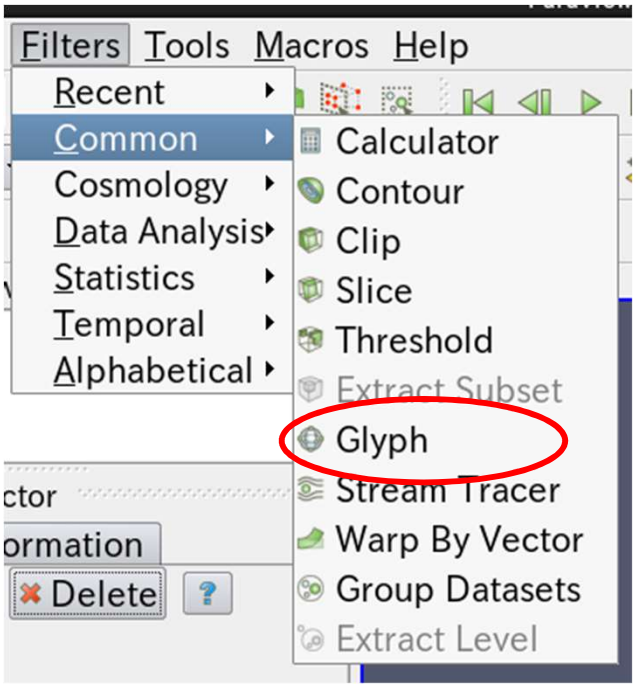
続く…

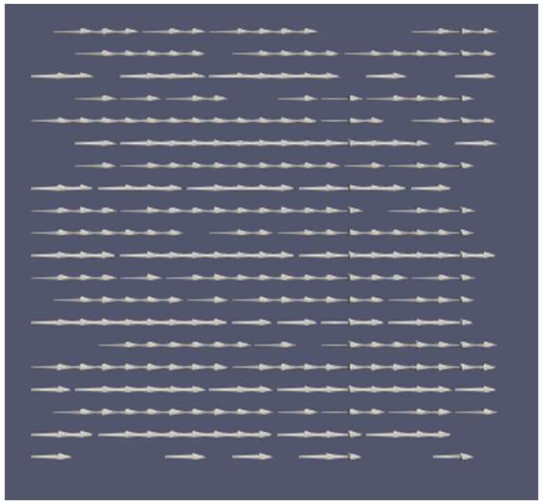
速度ベクトル図 Glyph (2)

- Pipeline Browserで CellCentersが選ばれていることを確認
- Filtersメニューから、Commonと進み、Glyphをクリックする。
- Applyボタンを押す

User Guide 2.1.4.2節







速度ベクトル図 Glyph (3)

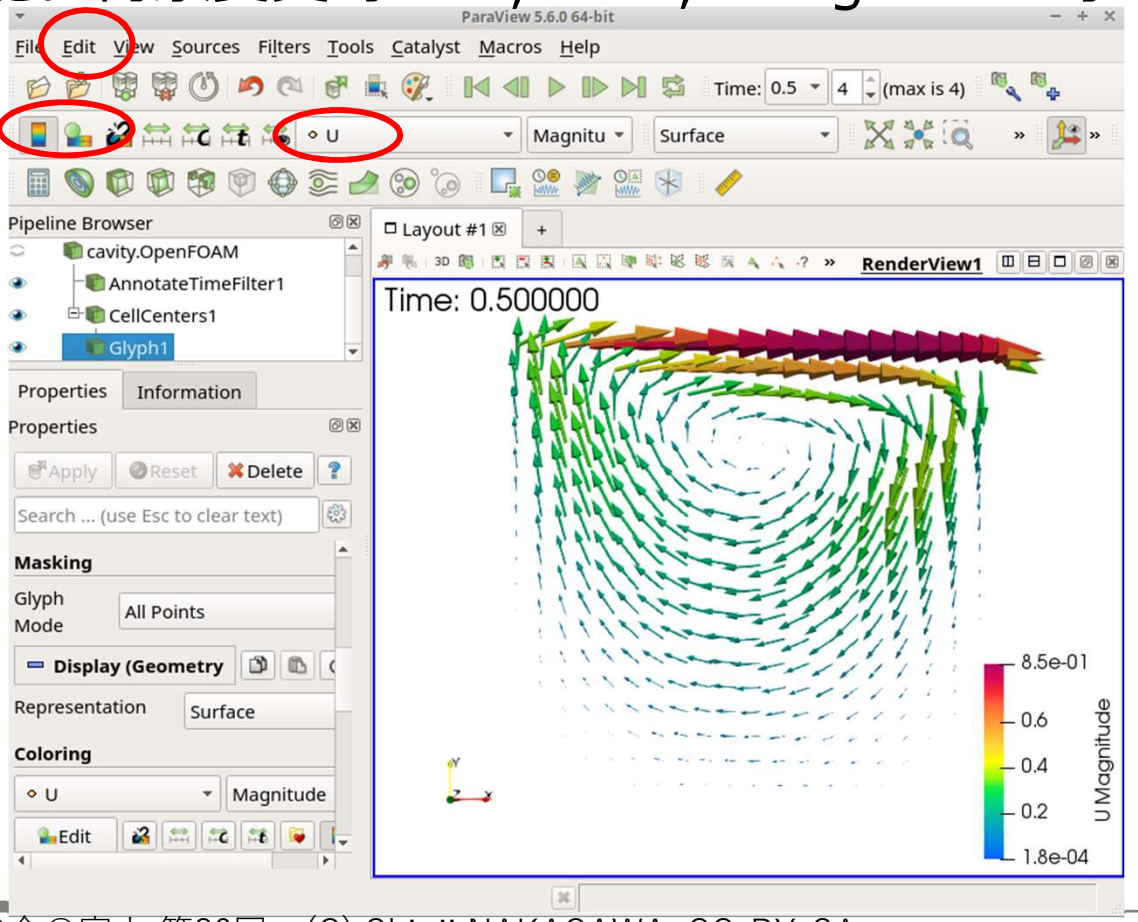
- ベクトルとして表示したいデータを選択する必要がある。
- PipelineBrowserでGlyph選択、Propertiesタブをクリックし、Orientation Array を U にする。
- ベクトルが描画される。長さは一定となっている。
- Scale Array を U に、ScaleFactor を0.04にしてApplyボタンをクリックする。
- 速度の大きさに合わせて、ベクトルの長さが決まる。

User Guide 2.1.4.2節

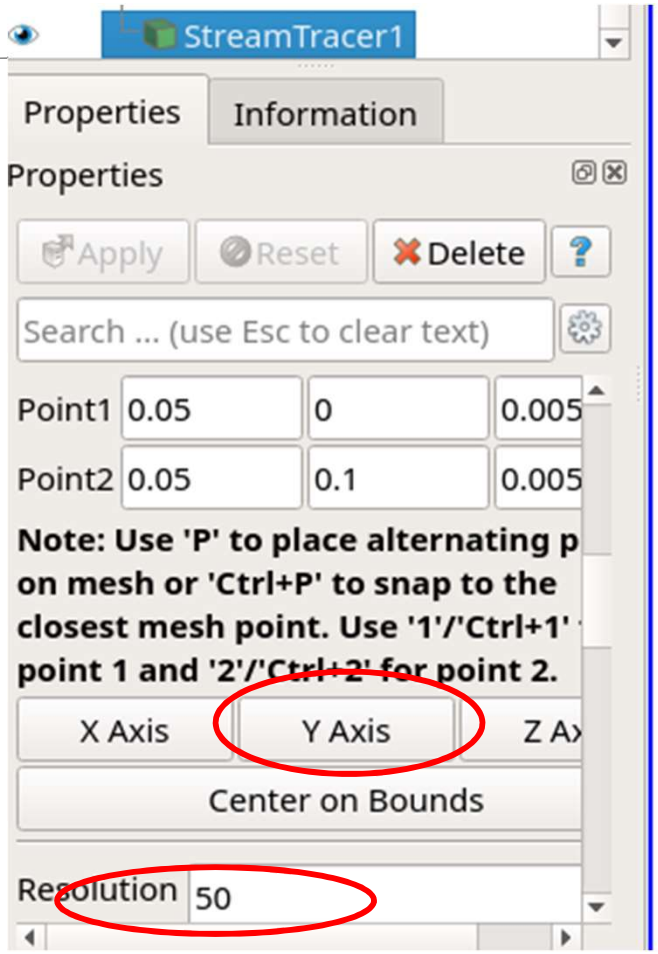
速度ベクトル図 Glyph (4)

User Guide 2.1.4.2節

- PipelineBrowserでGlyph選択、
  Colorling:U に、
  Colorling:Edit で BlueToRed Rainbow
- 背景色はメニューの Edit – Settings からウィンドを起動し，Color Paletteタブから変更可能。背景変更時には，text, foreground 等も適切に変更が必要である。



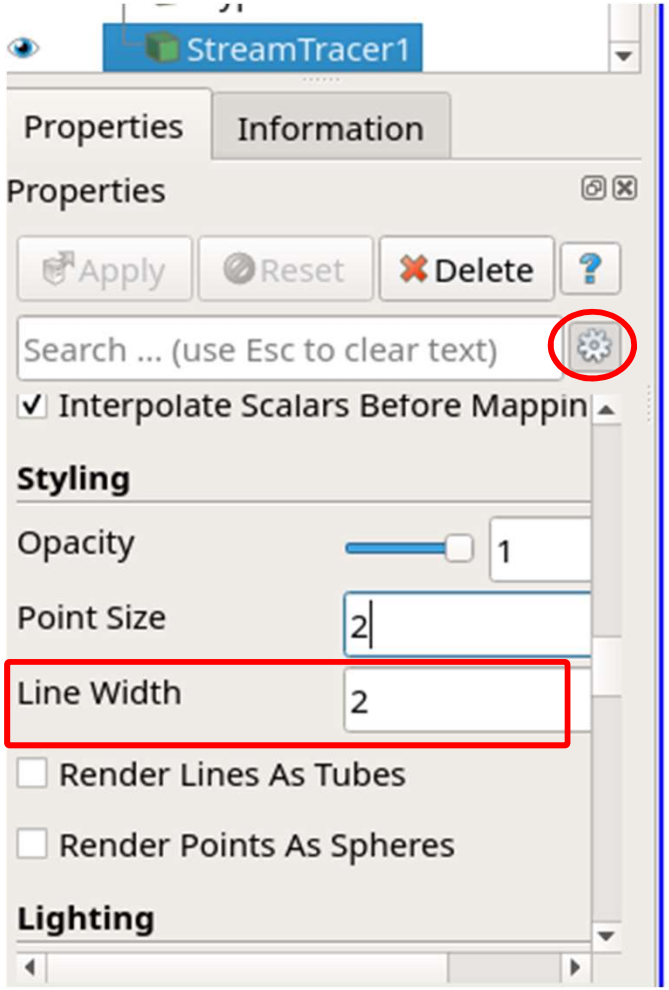
流線 Streamline

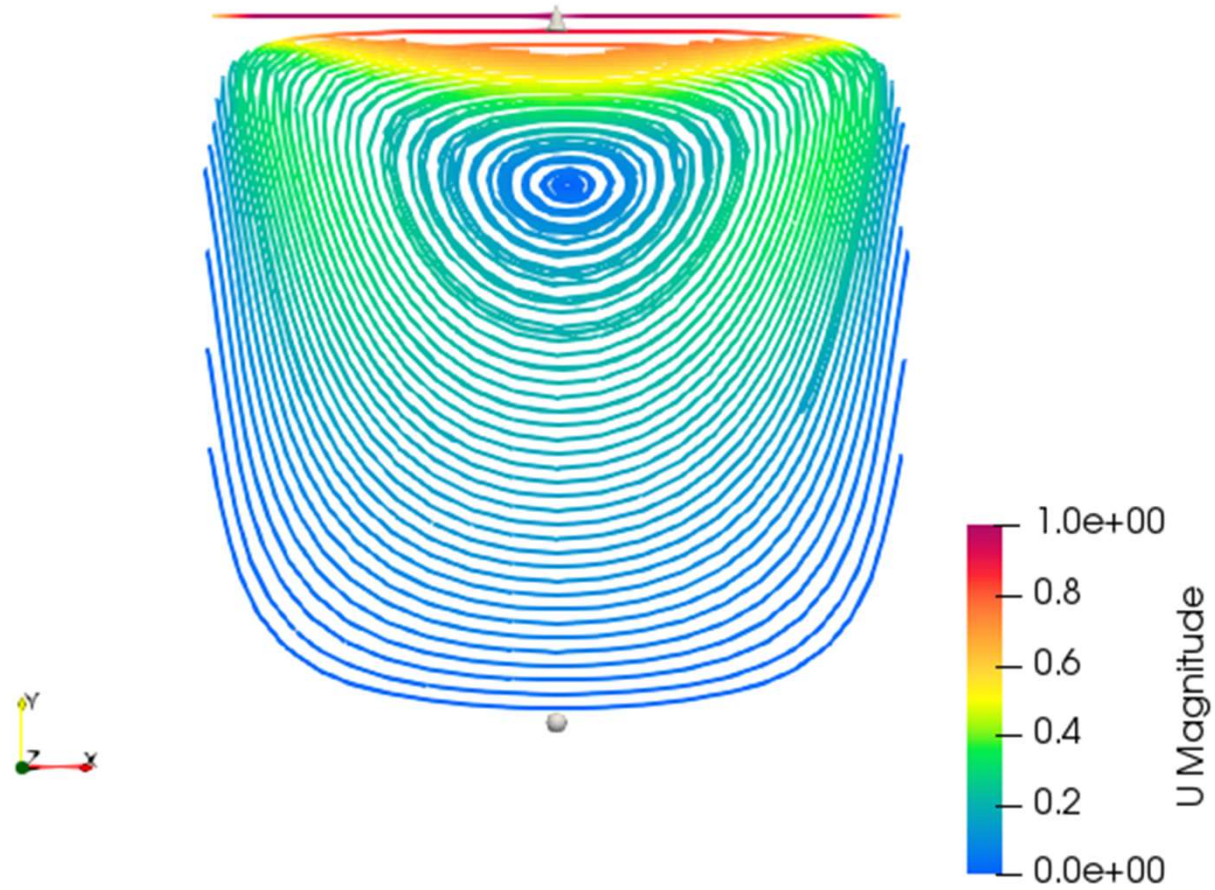


- Pipeline Browserでcavity.OpenFOAMが選ばれていることを確認
- すべてのfilterの表示をoffにするため、目玉をクリックして消す
- Filters - Searchと進み、StreamTracerを検索・クリックする。
- Line Parameters欄のY Axisをクリックし，Resolutionを50とする。Y軸に平行な場所から，50本の流線が描画される。

流線 Streamline

User Guide 2.1.4.3節





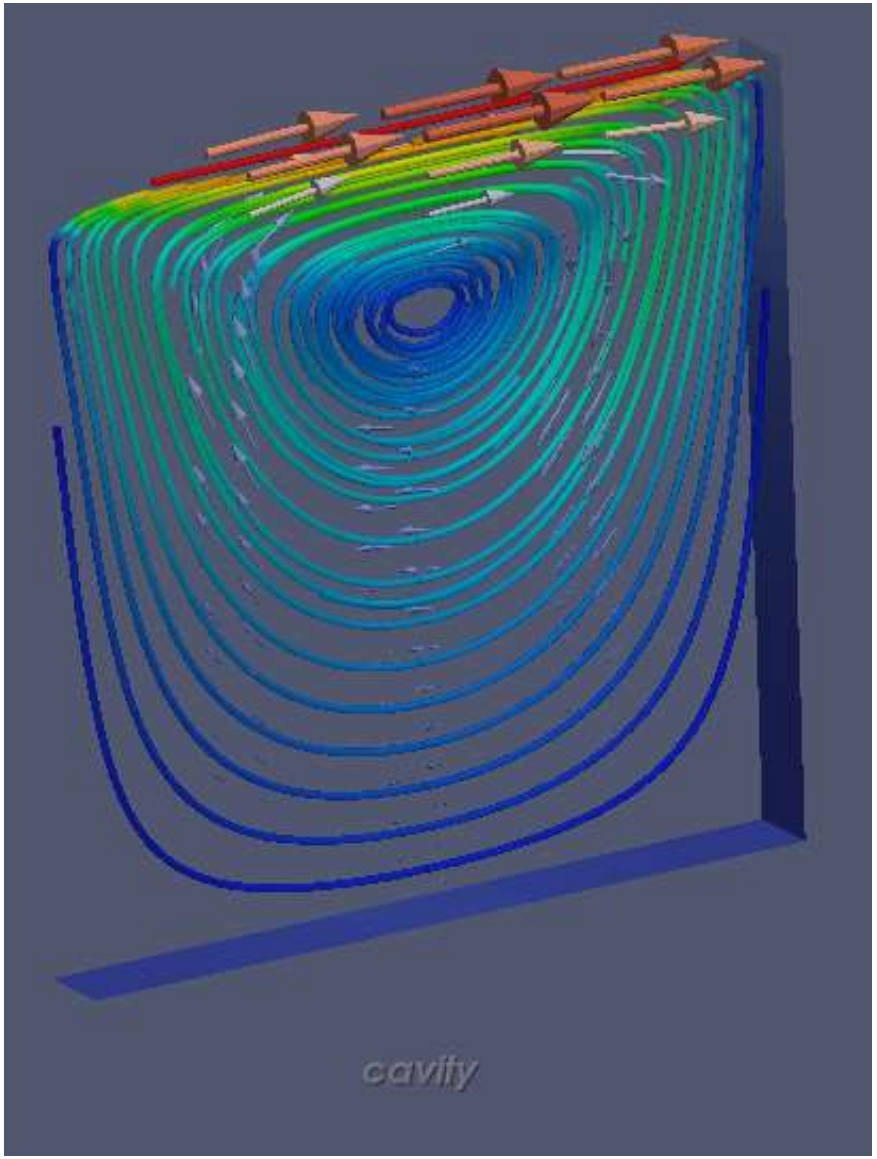
- Toggle advanced properties をクリックし，StylingのLine widthを大きくすると、線が太くなる。

可視化結果の保存

- 画像の保存
  - メニュー： File ー Save screenshot
- 動画の保存
  - メニュー： File ー Save Animation

- 作業状態の保存
  - メニュー： File ー Save State

おまけ：runTimePostProcessing



v1906
Improvements in runTimePostProcessing

The runTimePostProcessing function object provides a direct VTK insitu visualization alternative to the ParaView/Catalyst interface for well-defined, high throughput workflows.

v1912
New bi-directional streamlines more...

おまけ：runTimePostProcessing

- 計算実行時に，流線，カット面などの情報を取り出し，組み合わせた可視化画像を生成できる
- 計算終了とともに，可視化も完了

おまけ：runTimePostProcessing

system/visualization

```
streamLines
{
    type            streamLine;
    writeControl    writeTime;
    setFormat       vtk;
    U               U;
    direction       bidirectional;
    fields          (p U);
    lifeTime        150;
    nSubCycle       5;
    cloud           particleTracks;
    seedSampleSet
    {
        type        uniform;
        axis        x;
        start       (0.05 0.0 0.005 );
        end         (0.05 0.1 0.005 );
        nPoints     20;
    }
}
```

おまけ：runTimePostProcessing

```
postPro1
{
    #includeEtc "caseDicts/postProcessing/visualization/runTimePostPro.cfg“
    writeControl    writeTime;
    output
    {
        name        image;
        width       800;
        height      600;
    }
    camera
    {
        nFrameTotal     1;
        parallelProjection  yes;
        position        (-0.083 0.14 0.23);
        focalPoint      (0.05 0.05 0.005);
        up              (0.19 0.95 -0.26);
        zoom    13;
    }
```

確実な作業のために

- 実行手順，やり直し手順を，スクリプトとして残しましょう。
- 標準例題には，実行手順 Allrun，やり直し手順 Allclean といったスクリプトが用意されているケースがあります。
- 今回の例題は，親ディレクトリに，複数条件で連続実行するためのスクリプトが存在します。

$FOAM_TUTORIALS/incompressible/icoFoam/cavity/Allrun など

作業：スクリプト作成(1)

- ファイルマネージャーで，ケースディレクトリ内に新規ファイルを作成する。名前をAllclean とする。
- ファイルをダブルクリックし，テキストエディタ(ソフト名：Mousepad)で開く。
- 下記の内容を記入し，保存する。

標準例題ディレクトリに存在するAllcleanを参考にするとよい。

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

cleanCase
```

作業：スクリプト作成(2)

- 保存した Allclean ファイルを右クリックし，Propetiesを表示する。
- Permissionsタブで，「プログラムとして実行可能にする」にチェックを入れて有効にする。
- 端末から，下記を入力・実行して，Allcleanコマンドによって結果が削除されることを確認する。

```
./Allclean
```

作業：スクリプト作成(3)

- ファイルマネージャーで，ケースディレクトリ内に新規ファイルを作成する。名前をAllrun とする。
- ファイルをダブルクリックし，テキストエディタ(ソフト名：Mousepad)で開く。
- 下記の内容を記入し，保存する。

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

runApplication blockMesh
runApplication $(getApplication)
```

作業：スクリプト作成(4)

- 保存した Allrun ファイルを右クリックし，Propetiesを表示する。
- Permissionsタブで，「プログラムとして実行可能にする」にチェックを入れて有効にする。
- 端末から下記を入力・実行して，Allrunコマンドによってメッシュの生成と計算が実行されること，画面出力されていたlogがファイルに書き出されていることを確認する。

```
./Allrun
```

発展：ケースのコピー

【作業：ファイルマネージャ】

User Guide 2.1.5.1節

現在のケースディレクトリ($FOAM_RUN/cavity)をコピーして、貼り付ける。

名前を cavityFine とする。

/cavityFine/constant/polyMesh/blockMeshDictをダブルクリックし、ファイルを開く。

メッシュの細分化

User Guide 2.1.5.2節

- /cavityFine/system/blockMeshDict をダブルクリックし、ファイルを開く。
- blocks部分を下記のように変更して保存。

```
 blocks
 (
   hex (0 1 2 3 4 5 6 7) (40 40 1) simpleGrading
   (1 1 1)
);
```

計算パラメータ修正

User Guide 2.1.5.4節

- /cavityFine/system/controlDictをダブルクリックし、ファイルを開く。
- 時間刻みを 0.005 から 0.0025にする。
  - セルが半分の大きさになったため、クーラン数を1以下にするには、時間も半分にする必要がある。
- writeInterval を 20 から 40 に変更する。
  - 先ほどと同じ間隔（0.1秒毎）でデータを書き出すため。（0.0025×40=0.1）
  - あるいは、writeControl を runTime とし、writeInterval を 0.1 としてもよい。（書き出す時刻を直接指定する。）

計算（メッシュ細分化）

計算を実行する

User Guide 2.1.5.5節

【作業】

- ケース「cavityFine」ディレクトリを右クリック，端末で開く。
- 下記のコマンドを実行する（メッシュの生成，バックグラウンド実行とログのファイルへの書き出し）

```
blockMesh
icoFoam > log &
```

- 計算のレポートが端末に表示されるかわりに，ケースディレクトリにlogという名のファイルが生成される。
- ケースディレクトリに結果が出力される。（0.1から0.5）