# Pointwise作成モデルから OpenFOAMでAMIを使うために

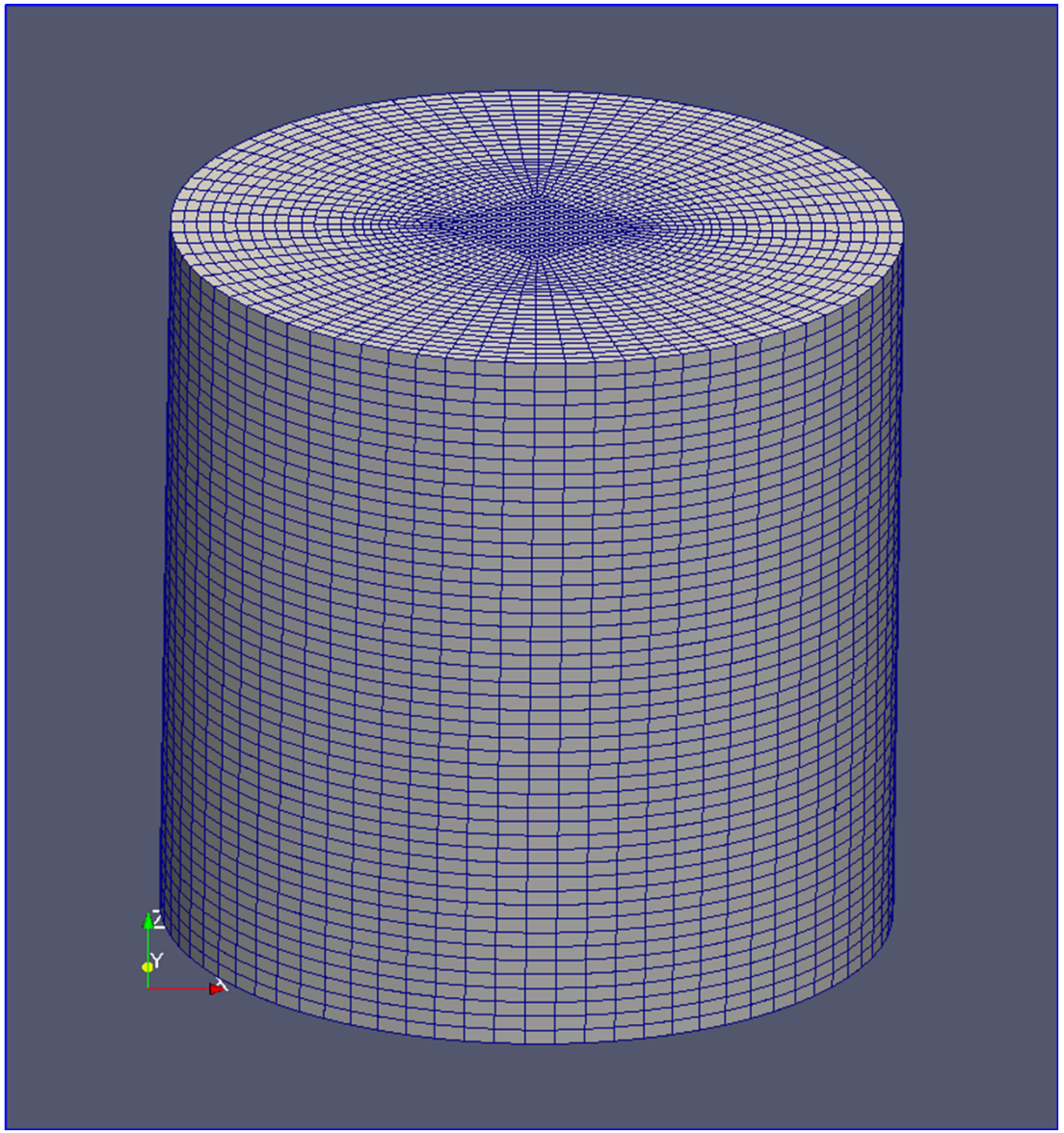
中川慎二

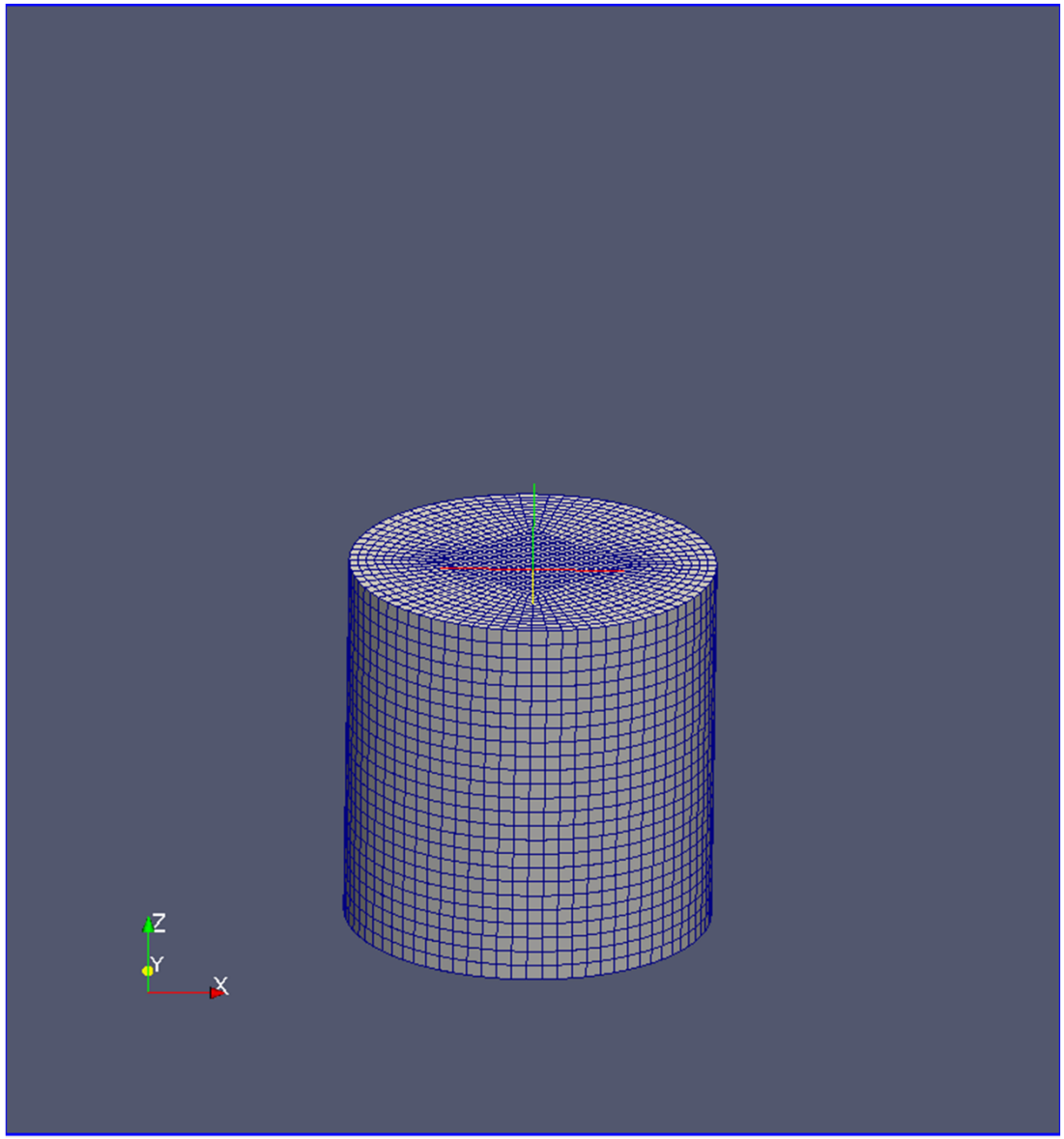
2012/09/29

# Pointwise作成モデル

- 全体：直径0.1m，高さ0.1mの円柱
  - 底面中心(0 0 0)，上面中心(0 0 0.1)
- 回転部：中心部で，直径0.05mの円柱，底面から高さ0.05m
- Pointwiseでは，回転部と非回転部を分けずに1つのモデルとする。境界条件も設定しない。
- 回転領域の設定，境界条件の設定などは，OpenFOAMのユーティリティーを使って実施する。

# 全体




Z
Y
X

# ケースディレクトリの準備

- pimpleDyFoamのpropellerチュートリアルを元に，ケースディレクトリを作成する。
- propellerケースディレクトリを複製して，amiPointwiseという名前に変更する。
- constant内のtriSurfaceフォルダを削除する。
- constant/polyMesh内のファイルを全て削除する。
- constant/polyMeshに，Pointwiseからエクスポートしたファイル（boundary, faces, neighbour, owner, points）を入れる。

# 操作の手順

- メッシュ操作：topoSetの実行。topoSetDictの中身は後述。

topoSet

- createPatchの実行

createPatch -overwrite

- // system/changeDictionaryDictの編集
- constant/polyMesh/facesファイルを開き，faceの数を確認する。この数を，changeDictionaryDictファイルのAMI1とAMI2のstartFaceに書き込む。

- changeDictionaryの実行

changeDictionary

createBaffles -internalFacesOnly -overwrite preAMI '(AMI1 AMI2)‘

mergeOrSplitBaffles -split -overwrite

- 以上で，モデルの準備が終了です。あとは，pimpleDyMFoamを実行するだけ。

# topoSetDictの内容（1/10）

```
// すべての外部境界面を格納
  {
    name    wholeFaces;   type    faceSet;   action  new;    source boundaryToFace;
    sourceInfo     { }
  }
  // top面(+Z)作成用にwholeFacesを複製
  {
    name    topFaces;   type    faceSet;   action  new;   source faceToFace;
    sourceInfo         {     set wholeFaces;    }
  }
  // top面(+Z)作成：+Z向きの面を取り出す
  {
    name    topFaces;   type    faceSet;   action  subset;   source  normalToFace;
    sourceInfo
    {
      normal  (0 0 1);   // Vector
      cos     0.3;       // Tolerance (max cos of angle)
    }
  }
```

# topoSetDictの内容（2/10）

```
// bottom面(-Z)作成用にwholeFacesを複製
   {
      name    bottomFaces;   type    faceSet;   action  new;   source faceToFace;
      sourceInfo
      {
         set wholeFaces;
      }
   }
   // bottom面(-Z) : -Z向きの面を取り出す
   {
      name    bottomFaces;   type    faceSet;   action  subset;   source
normalToFace;
      sourceInfo
      {
         normal  (0 0 -1);   // Vector
         cos     0.3;        // Tolerance (max cos of angle)
      }
   }
```

# topoSetDictの内容（3/10）

```
// side面(円筒)作成用にwholeFacesを複製
{
    name    sideFaces;  type    faceSet;  action  new;  source faceToFace;
    sourceInfo    {       set wholeFaces;       }
}
// side面(円筒):全体からtopとbottomを除外する
{
    name    sideFaces;  type    faceSet;  action  delete;  source  faceToFace;
    sourceInfo    {     set topFaces;       }
}
{
    name    sideFaces;  type    faceSet;  action  delete;  source  faceToFace;
    sourceInfo      {       set bottomFaces;       }
}
```

# topoSetDictの内容（4/10）

```
//全体長さ0.1m,半径0.05m
//下から0.05m,半径0.025mの円柱を回転ゾーンrotorCellsに
   {
      name    rotorCells;
      type    cellSet;
      action  new;
      source  cylinderToCell;
      sourceInfo
      {
         p1      (0 0 0);
         p2      (0 0 0.0500001);  //zを0.05で指定すると凸凹するため
         radius  0.025;
      }
   }
```

# topoSetDictの内容（5/10）

```
//rotor以外を非回転cellSetに
    //はじめにrotorCellsと同じcellSetをつくり、反転invertする
    {
        name    statorCells;        type    cellSet;        action  new;
        source  cellToCell;
        sourceInfo
        {
            set rotorCells;
        }
    }
    {
        name    statorCells;        type    cellSet;        action  invert;
    }
```

# topoSetDictの内容（6/10）

```
//回転部rotorCellsと非回転部statorCellsの間にfaceSetを作る
  //はじめにrotorCellsすべてのfaceを入れ、subsetとしてstatorCellsを指定する
  //両者の重なる部分だけが残る
  {
    name    preAMI;      type    faceSet;      action  new;      source  cellToFace;
    sourceInfo
    {
      set   rotorCells;
      option all;
    }
  }
  {
    name    preAMI;      type    faceSet;      action  subset;      source  cellToFace;
    sourceInfo
    {
      set   statorCells;
      option all;
    }
  }
```

# topoSetDictの内容（7/10）

```
//回転部のセルをcellZoneSetに入れておく
{
    name    rotorCellZoneSet;
    type    cellZoneSet;
    action  new;
    source  setToCellZone;
    sourceInfo
    {
        set     rotorCells;
    }
}
```

# topoSetDictの内容（8/10）

```
//回転部rotorCellsと非回転部statorCellsの間にfaceSetを作る
  //はじめにrotorCellsすべてのfaceを入れ、subsetとしてstatorCellsを指定する
  //両者の重なる部分だけが残る
  {
    name    rotorFace;      type    faceSet;      action  new;      source  cellToFace;
    sourceInfo
    {
      set    rotorCells;
      option  all;
    }
  }
  {
    name    rotorFace;      type    faceSet;      action  subset;      source  cellToFace;
    sourceInfo
    {
      set    statorCells;
      option  all;
    }
  }
```

# topoSetDictの内容（9/10）

//回転部のfaceSet rotorFaceとcellSet rotorCellZoneSetをまとめてpreAMIというfaceZoneSetに格納する

```
{
    name    preAMI;
    type    faceZoneSet;
    action  new;
    source  setsToFaceZone;
    sourceInfo
    {
        faceSet     rotorFace;
        cellSet     rotorCellZoneSet;
    }
}
```

# createPatchDictの内容

```
// Do a synchronisation of coupled points after creation of any patches.
// Note: this does not work with points that are on multiple coupled patches with transformations (i.e. cyclics).
pointSync false;

// Patches to create. topoSetで作ったsetをpatchに。
patches
(
   {
      name top;
      patchInfo      {         type         patch;      }
      constructFrom set;
      set topFaces;
   }
   {
      name bottom;
      patchInfo      {         type         patch;      }
      constructFrom set;
      set bottomFaces;
   }
   {
      name side;
      patchInfo      {         type         patch;      }
      constructFrom set;
      set sideFaces;
   }
);       //AMI patches will be created with createBaffles and mergeOrSplitBaffles
```

# changeDictionaryDict

```
dictionaryReplacement
{
    boundary
    {
        AMI1
        {
            type            cyclicAMI;
            nFaces          0;
            startFace       336490; //facesファイルで数を確認
            neighbourPatch  AMI2;
            transform       noOrdering;
            surface         {           }
        }
        AMI2
        {
            type            cyclicAMI;
            nFaces          0;
            startFace       336490; neighbourPatch  AMI1;
            transform       noOrdering;
            surface         {           }
        }
    }
}
```