Using PyFoam as library

第25回オープンCAE勉強会@ 関西

2013. 9.21

片山　達也

目次

- 背景/目的
- 準備
- Pythonの基礎
- PyFoamについて
- cavity tutorial
- まとめ

背景/目的

- 背景
  - 最適化ツールOpenMDAOを活用するため OpenFOAMをpythonから実行したい
  - 様々な機能のあるPyFoamをpythonのライブラリとして使えば便利なはず・・・

- 目的
  - 解析の自動化
  - プリポストの効率化
  - pythonの勉強

準備

- 必要な環境
  - python　　(当方の環境： 2.7.3)
  - OpenFOAM　(当方の環境： 2.2.x)
  - PyFoam　　(当方の環境： 0.6.1)
  - numpyモジュール
- あると便利
  - pythonパッケージ(モジュール)
    - ipython,matplotlib,scipy …etc
  - MPI
  - eclipse (+PyDev)

Pythonの基礎(1)

- はじめに
  - 端末にてpython（or ipython）と入力。以降shellコマンドは水色の枠に表示
  - pythonのコマンドは褐色の枠内に表示
    - 早速入力してみる

shell command

```
> python （or ipython）
```

python code

```
>>>print “Hello World!”
Hello World!
```

Pythonの基礎(2)

- 変数の型を宣言しなくてもよい

python code

```
1  >>> a=1
2  >>> type(a)
   int
3  >>> b=1.0
4  >>> type(b)
   float
5  >>> type(a+b)
   float
6  >>> a+”a”
   Traceback (most recent call last):
     File "<stdin>", line 1, in <module>
   TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Pythonの基礎(3)

- リスト
  - キー付の辞書、変更できないタプルもある

python code

```
1  >>> l=[1,2,"san"]
2  >>> l[2]
   'san'
3  >>> l[2]=3
4  >>> l[2]
   3
5  >>> d={"west":"Mac","east":u"Makudo"}
6  >>> d["east"]
   'Makudo'
7  >>> t=(l,a)
8  >>> t[1]=1
   Traceback   …TypeError: 'tuple' …
```

Pythonの基礎(4)

- インデントには意味がある
  - if 文

python code

```
1  >>>  a=1
2  >>>  if a==1:
3  . . .  <tab>  print “a is ”,a
4  . . .   else:
5  . . .  <tab>  print “a is not ”,a
6  . . .
   a is 1
7  >>> if a==1:
8  . . . print “a is ”,a
   IndentationError: expected an indented block
```

Pythonの基礎(5)

- インデントには意味がある
  - for 文

python code

```
1  >>>  for i in l:
2  . . .  <tab>  print 2**i
3  . . .
   2
   4
   8
4  >>> for i in xrange(0,1):
5  . . .  <tab>  print i
6  . . .
   0
   1
```

Pythonの基礎(6)

- インデントには意味がある
  - 関数

python code

```
1  >>>  def myfunc(val):
2  . . .  <tab>  return val+10
3  . . .
4  >>> myfunc(1)
   11
```

Pythonの基礎(7)

- モジュールを使う

python code

```
1  >>>  import numpy
2  >>>  matA=numpy.array([[1,2],[2,3]])
3  >>>  matA
   array([[ 1, 2],
          [ 2, 3]])
4  >>> import numpy as np
5  >>> x=np.array([1,2])
6  >>> from numpy import dot
7  >>> y=dot(matA,x)
8  >>> y
   array([5,8])

9  >>> exit()
```

Pythonの基礎(8)

- shellからpython

shell command

```
> echo ‘print “Hello ” ’ > test.py
> python test.py
Hello
```

PyFoamについて

- PyFoamについて
  - 残差グラフ、境界条件の設定など便利な機能がある
  - tar file版(subversionでない)にAPIもある
- PYTHONPATH
  - インストールの際prefixオプションで場所をしていている場合、環境変数 PYTHONPATHにパスを通す

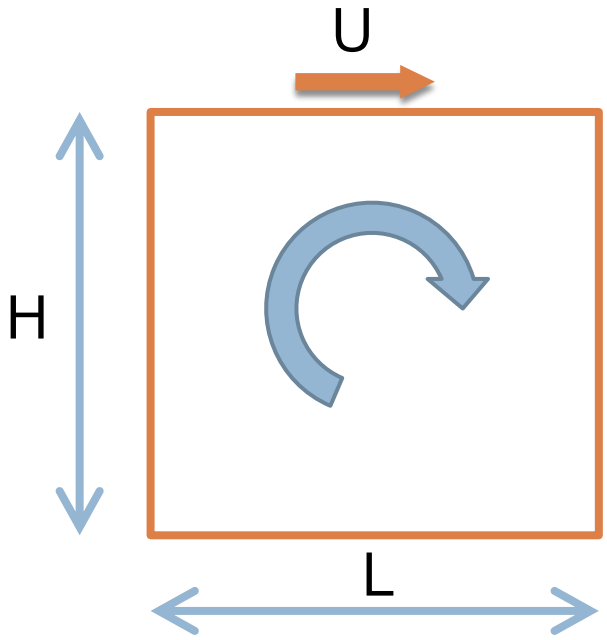
shell command

```
> cd  PyFoam
> python setup.py install --prefix=~/PyFoam
> export PYTHONPATH=~/PyFoam/lib/python2.7/site-packages:$PYTHONPATH
```

Cavity tutorial

- cavityの自動化
  - 形状・境界条件を変更し最大圧力を取得する
  - L，H，U各２水準を組み合わせ2^3回実施

	L[m]	H[m]	U[m/s]
水準1	0.1	0.1	0.5
水準2	0.2	0.2	1.0



  - 次ページ以降、前半で各クラスの使い方、後半が自動化スクリプトの作成を説明する

Cavity tutorial(1)

- SolutionDirectory (Caseディレクトリ)

python code

```
1 >>> import PyFoam.FoaInformation
2 >>> cavitydir= PyFoam.FoamInformation.foamTutorials() +
  "/incompressible/icoFoam/cavity"
3 >>> from PyFoam.RunDictionary.SolutionDirectory import
  SolutionDirectory
4 >>> cavityTut=SolutionDirectory(PyFoam.FoamInformation.
  foamTutorials() +"/incompressible/icoFoam/cavity")
5 >>> cavityCase=cavityTut.cloneCase("./cavity")

6 >>> cavityCase.name
```

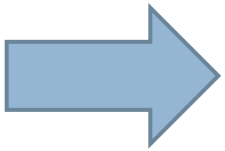
Cavity tutorial(2)

- templateファイル

shell command

> cd cavity
> cp constant/polyMesh/blockMeshDict constant/polyMesh/blockMeshDict.template
> gedit constant/polyMesh/blockMeshDict.template

```
19 vertices
20 (
21     (0 0 0)
22     (1 0 0)
23     (1 1 0)
24     (0 1 0)
25     (0 0 0.1)
26     (1 0 0.1)
27     (1 1 0.1)
28     (0 1 0.1)
29 );
```



blockMeshDict.templateを編集

```
19 vertices
20 (
21     (0 0 0)
22     ($L$ 0 0)
23     ($L$ $H$ 0)
24     (0 $H$ 0)
25     (0 0 0.1)
26     ($L$ 0 0.1)
27     ($L$ $H$ 0.1)
28     (0 $H$ 0.1)
29 );
```

Cavity tutorial(3)

- TemplateFile

python code

```
1 >>> from PyFoam.Basics.TemplateFile import TemplateFile
2 >>> bMDictTemp=TemplateFile(cavityCase.blockMesh() +
  ".template",expressionDelimiter="$")
3 >>> bMDictTemp.writeToFile(cavityCase.blockMesh(),
  {"L":1,"H":2})
```

- blockMeshDictを確認
  L,Hに値が代入されている
- 他のDictファイルも同様に
  TemplateFileが使える

```
19 vertices
20 (
21     (0 0 0)
22     (1 0 0)
23     (1 2 0)
24     (0 2 0)
25     (0 0 0.1)
26     (1 0 0.1)
27     (1 2 0.1)
28     (0 2 0.1)
29 );
```

Cavity tutorial(4)

- ParsedParameterFile
  - 直接ファイルを読み込み、変更、書き戻す

python code

```
1  >>> from PyFoam.RunDictionary.ParsedParameterFile import
   ParsedParameterFile
2  >>> U=ParsedParameterFile(cavityCase.initialDir() + "/U")
3  >>> U['boundaryField']['movingWall']['value']
   'uniform (1 0 0)'
4  >>> from PyFoam.Basics.DataStructures import Vector
5  >>> U['boundaryField']['movingWall']['value'].setUniform(
   Vector(0.5,0,0))
   >>> U.writeFile()
```

Cavity tutorial(5)

- BasicRunner
  - OpenFOAMのコマンドを実行する

python code

```
1  >>> from PyFoam.Execution.BasicRunner import BasicRunner
2  >>> import shlex
3  >>> args=shlex.split("blockMesh -case " + cavityCase.name)
4  >>> block=BasicRunner(args,silent=True)
5  >>> summary=block.start()
6  >>> summary
   {'OK': True,
    'casefullname': …
7  >>>block.runOK()
   True
```

Cavity tutorial(6)

- LogAnalyzerとLineAnalyzer
  - LineAnalyzerがlog一文からデータを拾う
  - LogAnalyzerにLineAnalyzerを一つ以上セットしlog全体の分析を行う

python code

```
1  >>> from PyFoam.LogAnalysis.FoamLogAnalyzer import
   FoamLogAnalyzer
2  >>> from PyFoam.LogAnalysis.ContinuityLineAnalyzer import
   GeneralContinuityLineAnalyzer
3  >>> analyzer=FoamLogAnalyzer(progress=False)
4  >>> analyzer.addAnalyzer("Continuity",
   GeneralContinuityLineAnalyzer(doTimelines=True, doFiles=False,
   singleFile=False, startTime=None, endTime=None))
```

Cavity tutorial(7)

- AnalyzedRunner
  - LogAnalyzerをセットし解析を実行する

python code

```
1  >>> from PyFoam.Execution.AnalyzedRunner import
   AnalyzedRunner
2  >>> args=shlex.split("icoFoam -case " + cavityCase.name)
3  >>> ico=AnalyzedRunner(analyzer,args,silent=True)
4  >>> summary=ico.start()
5  >>> summary
   {'OK': True,
    'analyzed': {'Continuity': …
6  >>>ico.runOK()
   True
```

Cavity tutorial(8)

- AnalyzedRunner
  - AnalyzedRunnerの代わりにPyFoam.LogAnalysis.LogAnalyzerApplicationを使えば、解析実行後のlogファイルを分析できる
  - FoamLogAnalyzerは基本クラスで、既に様々なLineAnalyzerがセット済みのクラスもある
  - LineAnalyzerの結果はfileへの出力、メモリへの保存などが可能で、LineAnalyzerのクラスフィールドで設定
  - LineAnalyzerの特性(どんな正規表現を拾ってくるのか)はAPIやソースコードにて確認する必要がある

Cavity tutorial(9)

- SolutionDirectory.clear()
  - cavityCaseを初期化

python code

```
1 >>> cavityCase.clear()
```

Cavity tutorial(10)

- 自動化の練習
  - Meshの横、縦、movingWallの速度を変化させる
  - 0.5 [s] 後の圧力pの最大値を取得する
  - cavityと同じディレクトリにautoRun.pyを作成する

autoRun.py

```
#!/usr/bin/env python
import shlex,sys,json
import numpy as np

                ⋮
```

Cavity tutorial(11)

- 自動化の練習(つづき)

autoRun.py　(つづき)

```
from PyFoam.RunDictionary.SolutionDirectory import
SolutionDirectory
from PyFoam.Basics.TemplateFile import TemplateFile
from PyFoam.Basics.DataStructures import Vector
from PyFoam.RunDictionary.ParsedParameterFile import
ParsedParameterFile
from PyFoam.Execution.BasicRunner import BasicRunner
from PyFoam.Error import error
```

⋮

Cavity tutorial(12)

- 自動化の練習(つづき)

autoRun.py　(つづき)

```
def main():
        #cavity case set
        cavityCase=SolutionDirectory("./cavity")

        #result list
        results=[]
        #blockMeshTemplate
        bMDictTemp=TemplateFile(cavityCase.blockMesh() +
".template",expressionDelimiter="$")
        #0/U
        U=ParsedParameterFile(cavityCase.initialDir() + "/U")

                        ⋮
```

Cavity tutorial(13)

- 自動化の練習(つづき)

autoRun.py　(つづき)

```
        for blockL in [1,2]:
            for blockH in [1,2]:
                for Uwall in [0.5, 1.0]:
                    #clear case
                    cavityCase.clear()
```

(インデント４つ)

```
    #edit files
    bMDictTemp.writeToFile(cavityCase.blockMesh(),{"L":blockL,"H"
    :blockH})
    U['boundaryField']['movingWall']['value'].setUniform(Vector(Uw
    all,0,0))
    U.writeFile()
```

Cavity tutorial(14)

- 自動化の練習(つづき)

autoRun.py　(つづき) (４つインデントを入れて)

```
    #blockMesh
    args=shlex.split("blockMesh -case " + cavityCase.name)
    block=BasicRunner(args,silent=True)
    block.start()
    if not block.runOK():
            error("There was a problem with blockMesh")
            sys.exit(1)
                    ⋮
```

Cavity tutorial(15)

- 自動化の練習(つづき)

autoRun.py　(つづき) (４つインデントを入れて)

```
#icoFoam
args=shlex.split("icoFoam -case " + cavityCase.name)
ico=BasicRunner(args,silent=True)
ico.start()
if not ico.runOK():
        error("There was a problem with icoFoam")
        sys.exit(1)
                ⋮
```

Cavity tutorial(16)

- 自動化の練習(つづき)

autoRun.py　(つづき) (４つインデントを入れて)

```
    #post
    p_latest=ParsedParameterFile(cavityCase.latestDir() + '/p')
    pmax=np.array(p_latest['internalField'].val).max()

    #append results
    results.append({"L":blockL,"H":blockH,"Uwall":Uwall,"pmax":p
    max})
                                ⋮
```

Cavity tutorial(17)

- 自動化の練習(つづき)

autoRun.py　(つづき) (インデントはなし。一行目は1つ)

```
        #print results
        print json.dumps(results,indent=4)


if __name__ == '__main__':
    main()
```

Cavity tutorial(18)

- autoRun.pyの実行

shell command

```
>python autoRun.py
[
    {
        "H": 1,
        "Uwall": 0.5,
        "L": 1,
        "pmax": 2.3693599999999999
    },
    {
        "H": 1,
        "Uwall": 1.0,
…
```

まとめ/参考資料

- まとめ
  - OpenFOAMの自動化にPyFoamは欠かせない
  - 特にプリポスト処理はshellでsedコマンドを使うことを考えるとすばらしい
  - 今後OpenMDAOに活用する

- 文献
  - より洗練されているオリジナル資料
    http://en.sourceforge.jp/projects/sfnet_openfoam-extend/downloads/OpenFOAM_Workshops/OFW5_2010_Gothenburg/Advanced_Training/pyFoamAdvanced.pdf