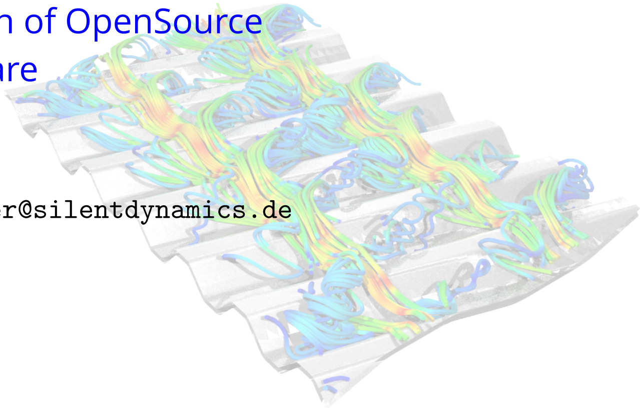
silentdynamics

Automation of OpenSource CAE Software



Hannes Kröger
hannes.kroeger@silentdynamics.de

GOFUN2019

Contents

Present solutions for analysis workflow automation

- with focus on numerical simulations $\Rightarrow$CFD, FEM
- using open-source tools $\Rightarrow$OpenFOAM

Automated workflow,

- to avoid errors
- to increase productivity, speed
- make complicated analyses available to unexperienced users

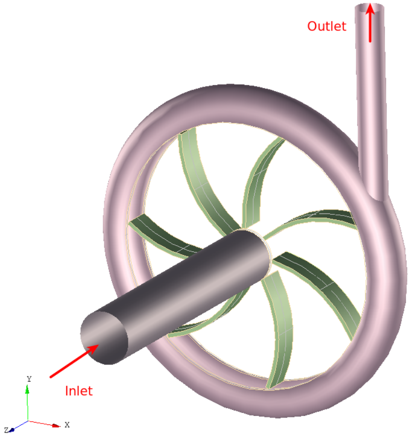
Tools for case setup and manipulation

- pyFoam
  - contains several command line tools for case manipulation
  - in python
- InsightCAE
  - also contains library of tools for case manipulation
  - in C++ with python wrappers
  - workflow GUI, report creation
  - CAD handling

We want to go through different steps of escalation

- create case manually, use supporting tools
- create a python script for the same task
- then turn into fully automated workflow with GUI

Simplified centrifugal pump with geometrical parameter



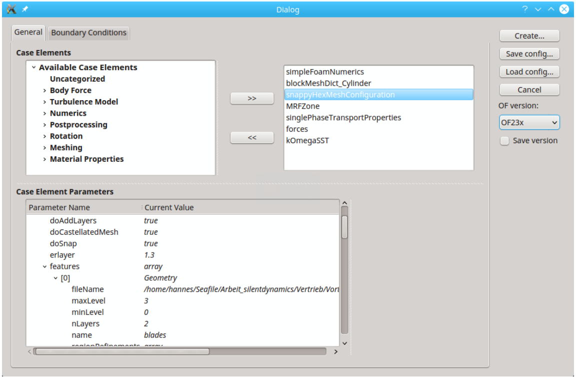
```
D=100;
d=20;
dd=15;
h=10;
t=1.5;
nb=7;
Lin=100;

blade_skel: SplineCurve(
 0.5*d*EX,
 rot(0.25*(d+D)*EX by -15*deg around EZ),
 rot(0.5*D*EX by -30*deg around EZ)
);

blade1=Thicken(Extrusion(blade_skel, h*EZ), -t);
blades=
 CircularPattern(blade1, 0, (360*deg/nb)*EZ, nb);

inlettube: asModel(
 Cylinder(0+(h+t)*EZ, (Lin+h)*EZ, d)?faces('isCylinder') );
```

```
rotordomain=Cylinder(O, h*EZ, D);

rotor:
((Cylinder(O -t*EZ, (h+t)*EZ, D) - rotordomain) | blades)
- Cylinder(h*EZ, (h+t)*EZ, d);

volute1=
(( Cylinder(O, ax D*EY, 0.7*dd) << (0.5*D*EX + 0.5*h*EZ) )
 | Torus( O +0.5*h*EZ, EZ*(D+0.6*dd), dd ))
- Cylinder(O, 10*h*EZ, D, centered);

volute_inlet=volute1?faces('isPlane');

volute_outlet=volute1
 ?faces('isCylinder␣&&␣isCoincident(%0)',
  rotordomain?allfaces);

volute: StitchedShell(
 volute1?faces('!in(%0)␣&&␣!in(%1)',
  volute_inlet, volute_outlet));
```

```
@post

saveAs("rotor.stlb") << rotor;
saveAs("volute.stlb") << volute;
saveAs("inlettube.stlb") << inlettube;
saveAs("rotordomain.stlb") << rotordomain;
```

Create the OpenFOAM by combining case elements with case builder

```
$ isofCaseBuilder
```



Add:

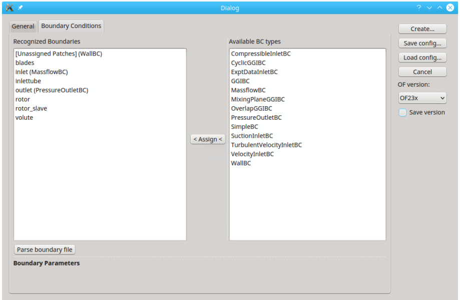
1. simpleFoamNumerics
2. blockMeshDict_Cylinder
3. snappyHexMeshConfiguration
4. MRFZone
5. singlePhaseTransportProperties
6. forces
7. kOmegaSST

Setup Parameters:

1. blockMeshDict_Cylinder: D 0.18, L 0.15, p0 $(0\ 0\ -0.05)^T$, defaultPatchName “outlet”, topPatchName “inlet”
2. snappyHexMeshParameters: PiM, add features:
   - 2.1 Geometry: rotor.stlb, name “blades”, scale 10^{-3}
   - 2.2 Geometry: inlettube.stlb, name “inlettube”, scale 10^{-3}
   - 2.3 Geometry: volute.stlb, name “volute”, scale 10^{-3}
   - 2.4 Geometry: rotordomain.stlb, name “rotor”, zoneName “rotor”, scale 10^{-3}
3. MRFZone: name “rotor”, rpm 1000
4. forces: name “forces”, patches: add “blades”, rhoInf 998.0

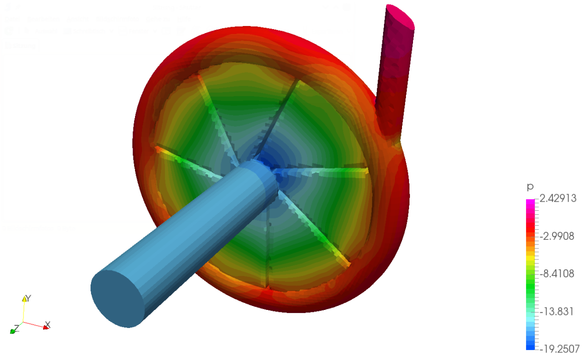
1. Save config, “case.iscb”
2. Create...
3. run `blockMesh`
4. run `snappyHexMesh -overwrite`

1. Restart Case Builder: `isofCaseBuilder case.iscb`
2. change to tab “Boundary Conditions”



3. Assign BCs:
   - inlet: “MassflowBC”, massflow 0.0001
   - outlet: “PressureOutletBC”
   - [Unassigned Patches]: “WallBC”
4. Save config, Create...

- run `simpleFoam`
- then evaluate manually, e.g. `isPV.py`:



Scripted Solution

Repeat after geometry change, goal: self-contained script

- copy input files into script:
  `cat model.iscad case.iscb > run.py`
- edit script

complete the script:

```
#!/usr/bin/env python

from Insight.toolkit import *
import subprocess, pprint, numpy, math

cadscript="""
D=100;
d=20;
dd=15;
...
saveAs("rotordomain.stlb") << rotordomain;
"""

openfoamcase="""
<?xml version="1.0" encoding="utf-8"?>
<root>
...
</root>
"""
```

```
## Generate geometry
subprocess.Popen(['iscad', '-b', '-'],
 stdin=subprocess.PIPE).communicate(input=cadscript)

## Mesh
case=OpenFOAMCase(OFEs_get("OFesi1806"))
case.setFromXML(openfoamcase, workdir, True, True) #skip BCs
case.createOnDisk ( workdir );
case.modifyCaseOnDisk ( workdir )
case.executeCommand( workdir, "blockMesh" )
case.executeCommand( workdir, "snappyHexMesh",
  ["-overwrite"] )

## Run
case=OpenFOAMCase(OFEs_get("OFesi1806"))
case.setFromXML(openfoamcase, workdir, True) # incl. BCs
case.createOnDisk ( workdir );
case.modifyCaseOnDisk ( workdir )
case.executeCommand( workdir, "simpleFoam" )
```

```
## Evaluate
f_vs_t=numpy.array(
 forces_readForces(case, workdir, "forces"))
Qfinal=f_vs_t[-1,8]+f_vs_t[-1,12]

patch_in = patchIntegrate(case, workdir, "p", "inlet")
patch_out = patchIntegrate(case, workdir, "p", "outlet")

p_in=patch_in.integral_values_[-1]/patch_in.A_[-1]
p_out=patch_out.integral_values_[-1]/patch_out.A_[-1]
delta_p=(p_out-p_in)*rho
H=delta_p/rho/9.81
```

Add a GUI

Complete automation by adding GUI and Report Creation

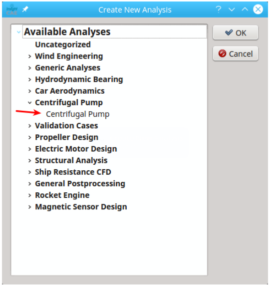
- Insight supports Python analysis modules
  go into `$HOME/.insight/share/python_modules`
- copy script to `$HOME/.insight/share/python_modules/Centrifugal\ Pump.py`
- Do following modifications

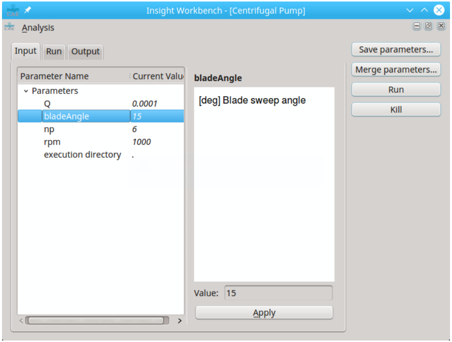
```python
#!/usr/bin/env python

from Insight.toolkit import *
import subprocess, pprint, numpy, math
import matplotlib.pyplot as plt

def category():
    return "Centrifugal␣Pump"

def defaultParameters():
    p=ParameterSet([
        ("rpm", DoubleParameter(1000.0, "[rpm]␣Rotation␣speed")),
        ("Q", DoubleParameter(0.0001, "[m^3/s]␣Volume␣flux")),
        ("bladeAngle", DoubleParameter(15., "[deg]␣Blade␣sweep␣angle"))
        ])
    return p
```

```
def executeAnalysis(ps, workdir):

    rpm=ps.getDouble("rpm")
    Q=ps.getDouble("Q")
    bladeAngle=ps.getDouble("bladeAngle")

    cadscript="""
D=100;
...
blade_skel:
SplineCurve(
 0.5*d*EX,
 rot(0.25*(d+D)*EX by -%g*deg around EZ),
 rot(0.5*D*EX by -30*deg around EZ)
);
...
saveAs("rotordomain.stlb") << rotordomain;
"""%bladeAngle
```

```
    openfoamcase="""
<?xml version="1.0" encoding="utf-8"?>
<root>
...
              <vector name="rotationCentre" value="0 0 0"/>
              <double name="rpm" value="%g"/>
       </OpenFOAMCaseElement>
...
              <Patch patchName="inlet" BCtype="MassflowBC">
                     <double name="T" value="300"/>
                     <string name="UName" value="U"/>
                     <double name="gamma" value="1"/>
                     <double name="massflow" value="%g"/>
...
</root>
"""%(rpm,Q)
```

```
    ## Generate geometry
    subprocess.Popen([’iscad’, ’-b’, ’-’],
    ...
    H=delta_p/rho/9.81

    res=ResultSet(ps, "Centrifugal␣Pump", "CFD␣Results")
    res.insert("Q", ScalarResult(Qfinal, "Torque", "", "Nm"))
    res.insert("P", ScalarResult(Qfinal*2.*math.pi*rpm/60.,
                   "Power", "", "W"))
    res.insert("delta_p", ScalarResult(delta_p,
                    "Pressure␣increase", "", "Pa"))
    res.insert("H", ScalarResult(H, "Head", "", "m"))
    res.insert( "forceConvergence",
        Chart( "iter", "F",
            [PlotCurve(f_vs_t[:,0], f_vs_t[:,3], "axialForce",
            "w␣l␣t␣’$F_{ax}$’")], "", "", "") )
    res.insert( "torqueConvergence",
        Chart( "iter", "Q",
            [PlotCurve(f_vs_t[:,0], f_vs_t[:,9], "torque",
             "w␣l␣t␣’$Q$’")], "", "", "" ) )
    return res
```


Create New Analysis
Available Analyses
OK
Cancel
Uncategorized
Wind Engineering
Generic Analyses
Hydrodynamic Bearing
Car Aerodynamics
Centrifugal Pump
Centrifugal Pump
Validation Cases
Propeller Design
Electric Motor Design
Structural Analysis
Ship Resistance CFD
General Postprocessing
Rocket Engine
Magnetic Sensor Design


Insight Workbench - [Centrifugal Pump]
Analysis
Input
Run
Output
Save parameters...
Merge parameters...
Run
Kill
Parameter Name
Current Valu
Parameters
Q
0.0001
bladeAngle
15
np
6
rpm
1000
execution directory
.
bladeAngle
[deg] Blade sweep angle
Value: 15
Apply

Thank you for your attention!

Dr.-Ing. Hannes Kröger
Email: hannes.kroeger@silentdynamics.de
Tel.: +49 381 36 77 98 53

`http://silentdynamics.de`
`http://sourceforge.net/projects/insightcae`