ANNUAL REPORT 2012

UIUC, August 16, 2012

CON1D Mold Geometry Calibration: "Offset Method"

Inwho Hwang
(BSME Student)

Lance C. Hibbeler
(Ph.D. Student)

Department of Mechanical Science and Engineering
University of Illinois at Urbana-Champaign

Introduction

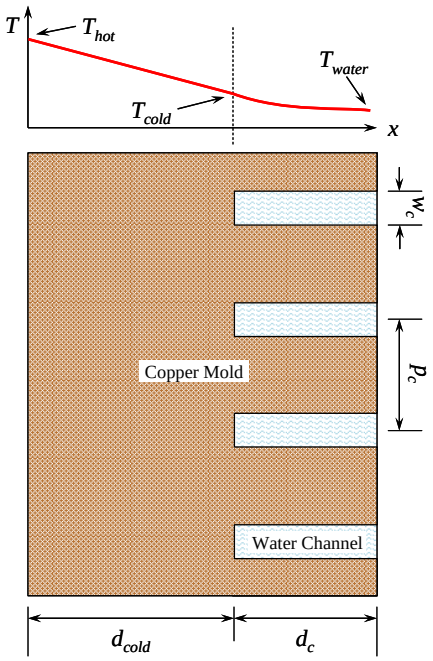
- The simplified mold geometry in CON1D can be calibrated using analytical techniques and heat transfer FEM models to provide increased accuracy at practically no cost
- This calibration has been automated using a Python script and ABAQUS

CON1D's 1D Mold Model

- Given
  - Hot face heat flux q_{hot}
  - Cold face convection h_{cold}, T_{water}
  - Thermal conductivity k
- Conduction equation gives:

$$T = T_{water} + q_{hot}\left(\frac{1}{h_{cold}} + \frac{L-x}{k_{mold}}\right)$$

- h_{cold} is a function of channel geometry, water properties, and water speed



Water Channel Geometry
Width and Depth

- The simulated rectangular channels and the actual water channels must have identical
  - Cross-sectional area: correct amount of water

$$w_c d_c = A_{c,actual}$$

  - Hydraulic diameter: correct convection behavior

$$2w_c d_c / (w_c + d_c) = D_{h,actual}$$

- Two equations and two variables, solved:

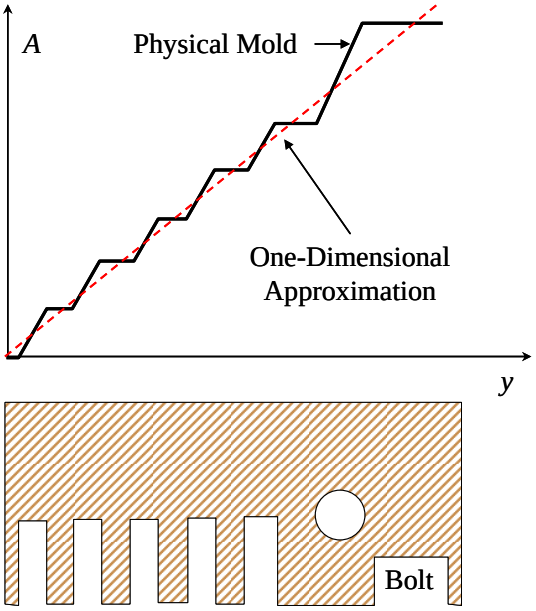
$$w_c, d_c = A_{c,actual}/D_{h,actual} \pm \sqrt{\left(A_{c,actual}/D_{h,actual}\right)^2 - A_{c,actual}}$$

Use average A_c and D_h for the mold

Water Channel Geometry
Pitch

- Water channel pitch is determined by plotting cumulative water channel area as a function of distance from the centerline and performing least-squares
  - Slope used to determine pitch

$$A_{cumulative,simulated}(x) = \frac{A_{c,actual}}{p_c} x$$



Calibrating for 3D Effects

- The accuracy of a 3D finite-element model can be given to CON1D by calibrating the mold thickness and thermocouple locations
- Manipulating 1D temperature solution gives
  - Calibrated cold face position

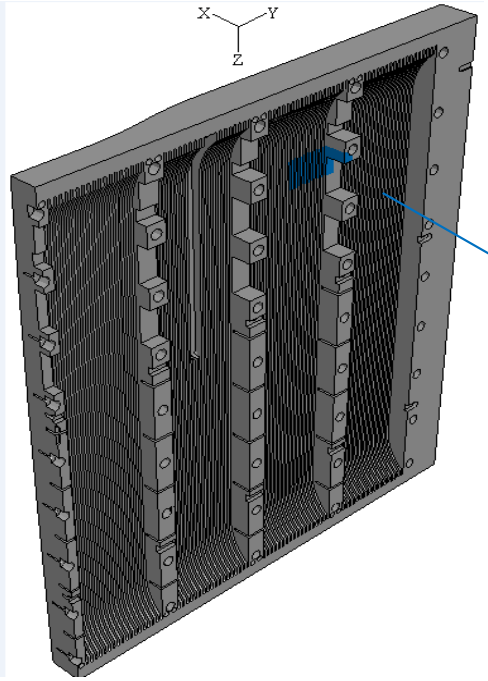
$$d'_{cold} = \frac{k}{q}\left(T_{hot,3D} - T_{cold,3D}\right)$$

  - Calibrated thermocouple position

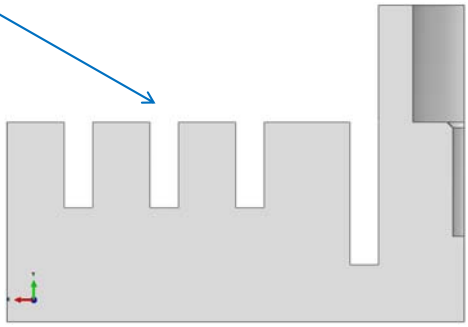
$$d'_{TC} = \frac{k}{q}\left(T_{hot,3D} - T_{TC,3D}\right)$$

Underlined terms taken from 3D FE model; *k* and *q* must match model

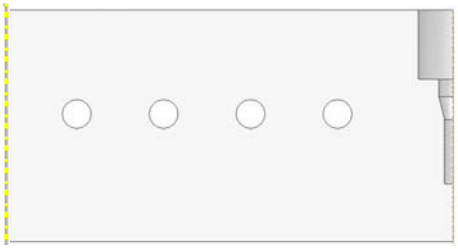
3D Mold Model



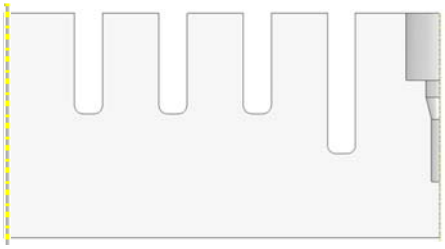
- The script exploits symmetry and models a small piece of the bigger casting mold



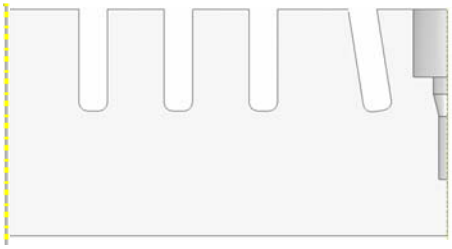
4 Mold Types



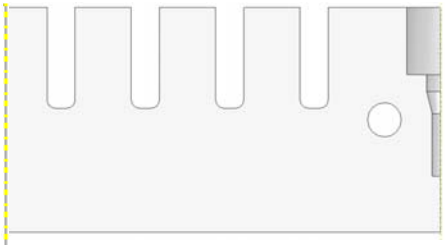
Type 1 has circular channels



Type 2 has rectangular channels and a deep channel



Type 3 has rectangular channels and a slanted channel



Type 4 has rectangular channels and a single circular channel

```
35  type = 1        # Type 1 is all circular channels, Type 2~4 has rectangular channels along with an oddball channel.
36                  # Type 2 has a deep channel. Type 3 has a slant channel. Type 4 has a circle channel.
```

General Dimensions: Domain size

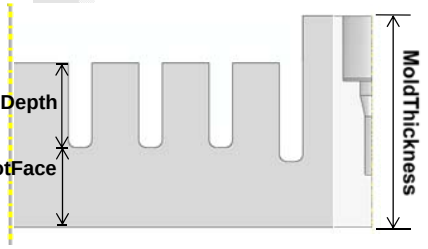
Script Input:

```
11   MoldWidth =          77.5
12   MoldThickness =      40.0
13   MoldLength =         50.0
```

Example mold with bolt rib

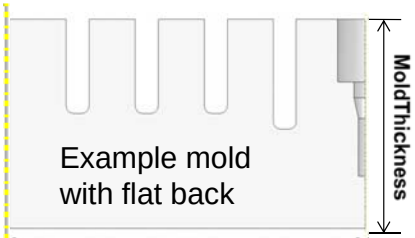
```
17   ChannelDepthFromHotFace =    17.0
44   ChannelDepth = 18.0
```

Top

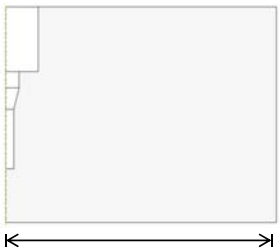


ChannelDepthFromHotFace + ChannelDepth = MoldThickness
makes a flat back mold

Front

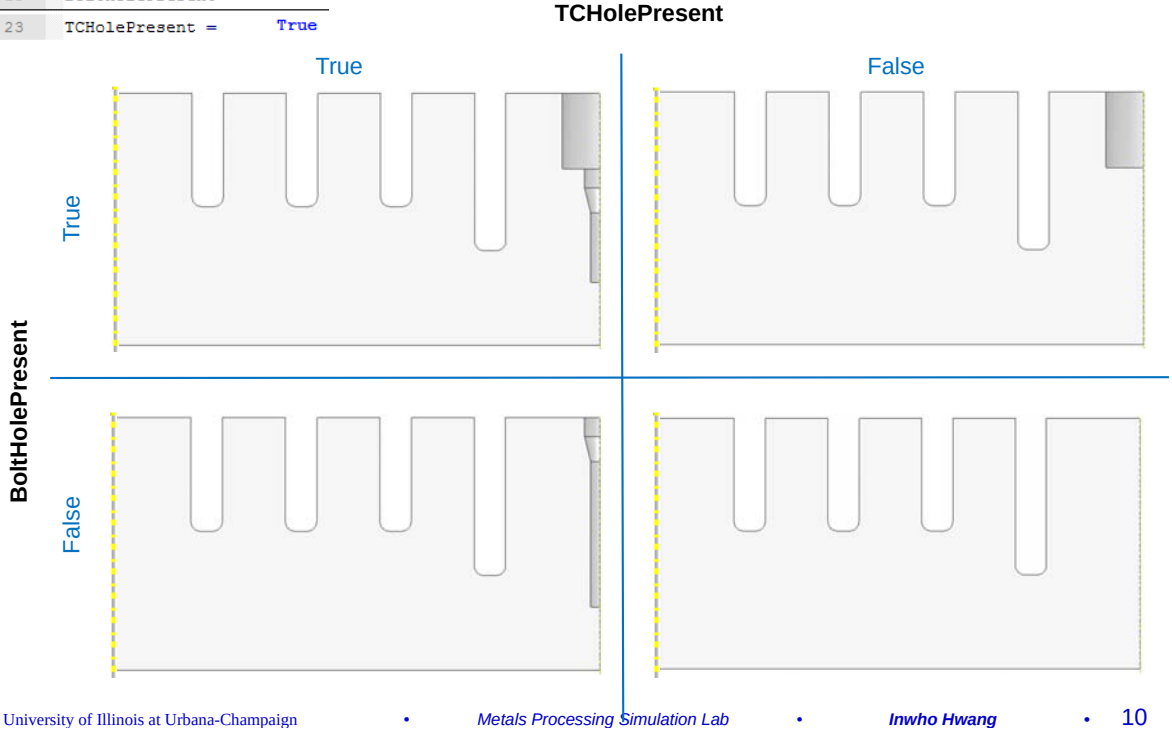


Side

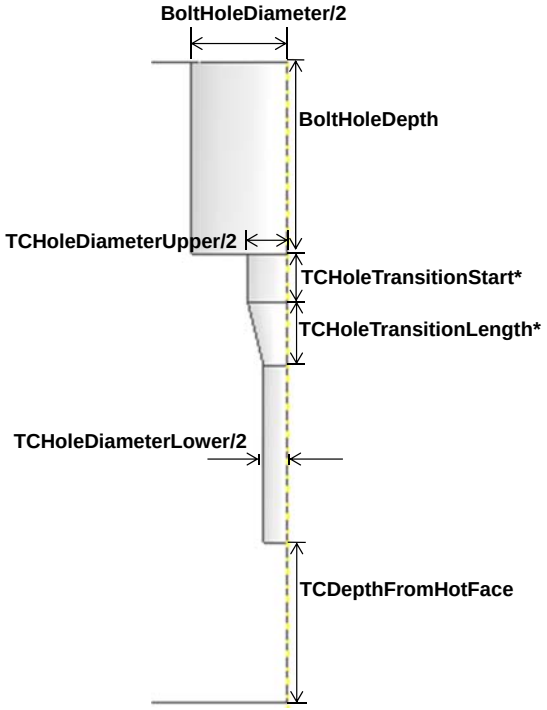


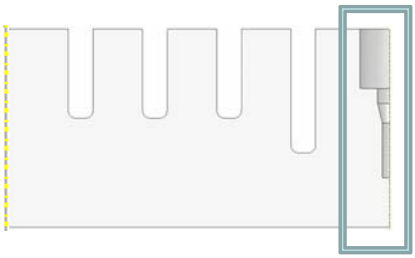
General Dimensions: Are TC and bolt present?

```
19   BoltHolePresent =    True
23   TCHolePresent =      True
```



General Dimensions: bolt & TC

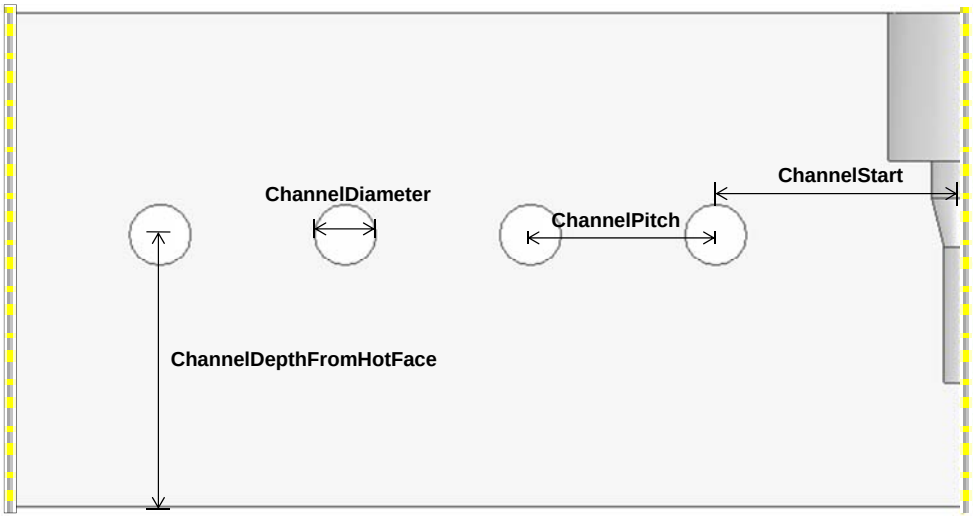




```
20  BoltHoleDiameter =          12.0
21  BoltHoleDepth =             12.0

24  TCHoleDiameterUpper =        5.0
25  TCHoleDiameterLower =        3.0
26  TCHoleTransitionStart =      3.0
27  TCHoleTransitionLength =     4.0
28  TCDepthFromHotFace =        10.0
```

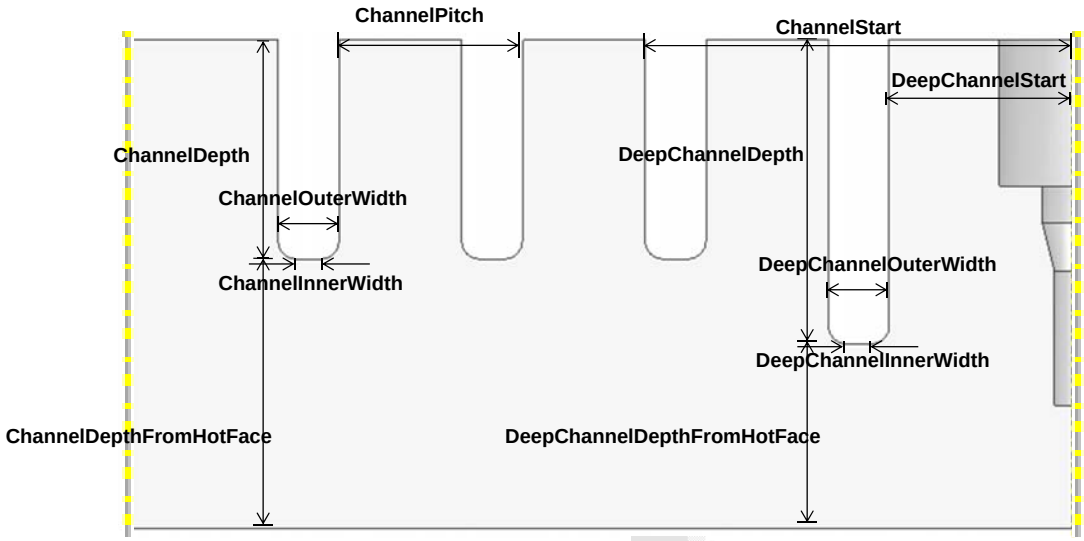
Starred (*) values can equal zero.

Type 1 Dimensions




```
15  ChannelStart =              20.0
16  ChannelPitch =              15.0
17  ChannelDepthFromHotFace =   22.0

38  # Type 1 input
39  ChannelDiameter = 5.0
```

Type 2 Dimensions

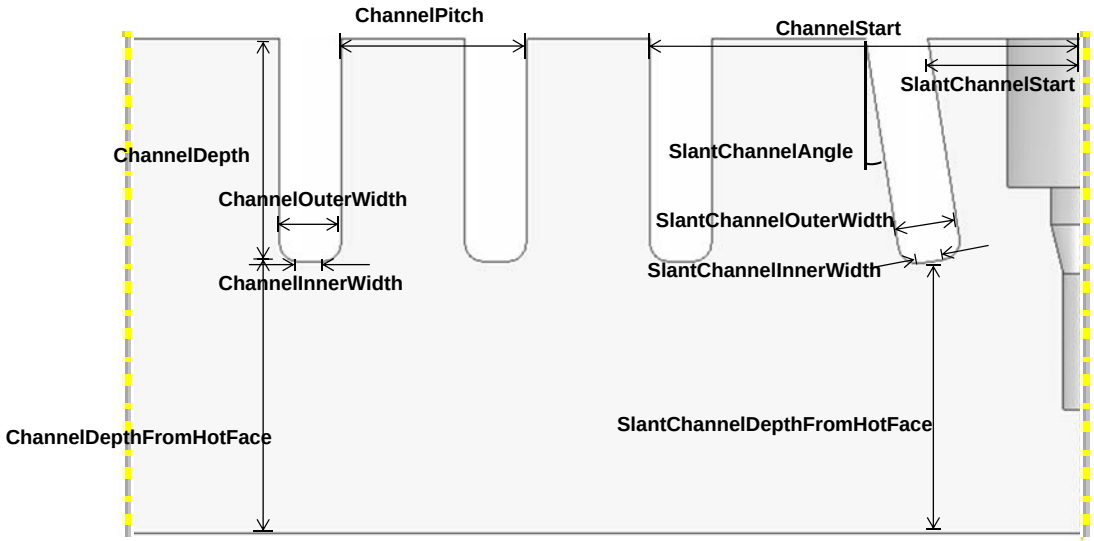



```
15  ChannelStart =                  20.0
16  ChannelPitch =                  15.0
17  ChannelDepthFromHotFace =       22.0
```

```
41  # Type 2~4 inputs
42  ChannelOuterWidth = 5.0
43  ChannelInnerWidth = 2.0
44  ChannelDepth = 18.0
```

```
46  # Type 2 input
47  DeepChannelStart = 15.0
48  DeepChannelDepthFromHotFace = 15.0
49  DeepChannelOuterWidth = 5.0
50  DeepChannelInnerWidth = 2.0
```


Type 3 Dimensions

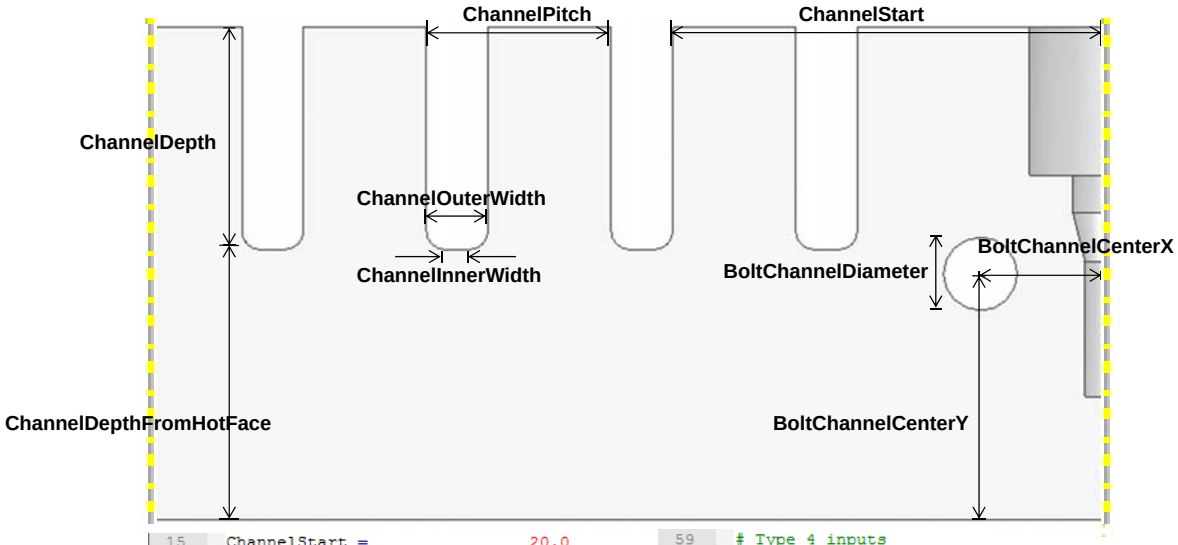



```
15  ChannelStart =                  20.0
16  ChannelPitch =                  15.0
17  ChannelDepthFromHotFace =       22.0
```

```
41  # Type 2~4 inputs
42  ChannelOuterWidth = 5.0
43  ChannelInnerWidth = 2.0
44  ChannelDepth = 18.0
```

```
52  # Type 3 input
53  SlantChannelStart = 15.0
54  SlantChannelDepthFromHotFace = 19.0
55  SlantChannelOuterWidth = 5.0
56  SlantChannelInnerWidth = 2.0
57  SlantChannelAngle = pi/20
```

Type 4 Dimensions



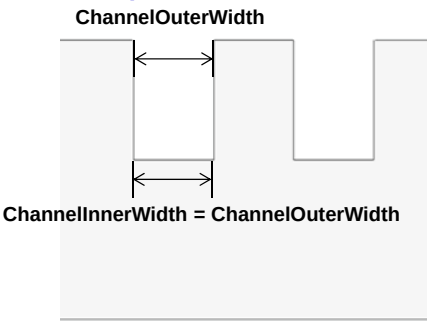

```
15  ChannelStart =                  20.0
16  ChannelPitch =                  15.0
17  ChannelDepthFromHotFace =       22.0
```

```
41  # Type 2~4 inputs
42  ChannelOuterWidth = 5.0
43  ChannelInnerWidth = 2.0
44  ChannelDepth = 18.0
```

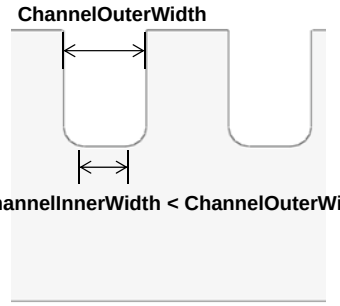
```
59  # Type 4 inputs
60  BoltChannelCenterX = 10.0
61  BoltChannelCenterY = 20.0
62  BoltChannelDiameter = 6.0
```


Type 2~4 Channel Dimensions

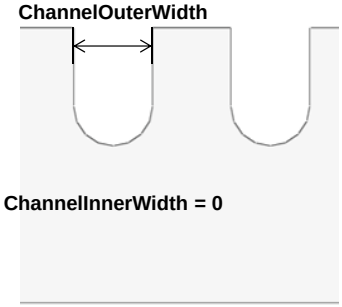
Square Slot Roots



General Radiused Slots



Round Slot Roots

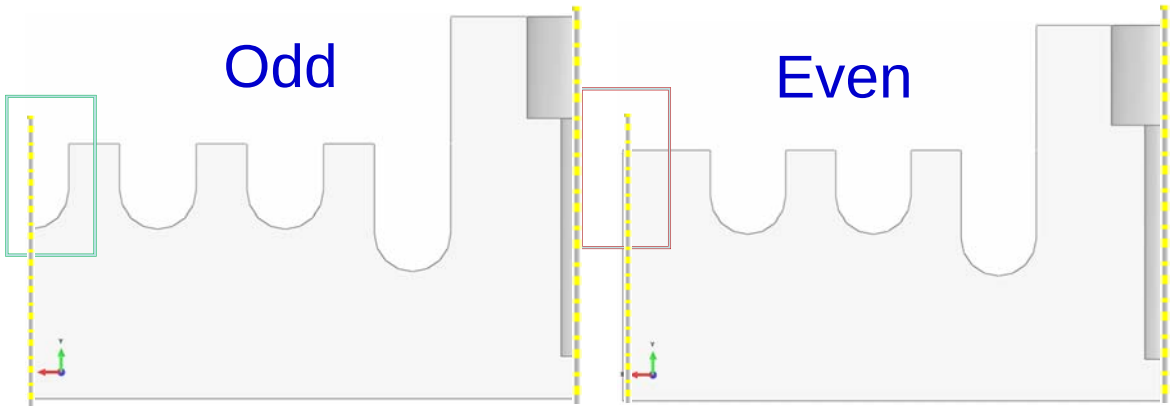



```
41  # Type 2~4 inputs
42  ChannelOuterWidth = 5.0
43  ChannelInnerWidth = 2.0
```

```
46  # Type 2 input
49  DeepChannelOuterWidth = 5.0
50  DeepChannelInnerWidth = 2.0
```

```
52  # Type 3 input
55  SlantChannelOuterWidth = 5.0
56  SlantChannelInnerWidth = 2.0
```

Odd vs Even Number of Slots



True False

```
31  # Would you like to create a half channel / Does your mold have an odd number of channels between bolt holes? [True or False]
32  HalfChannel = True
```

HalfChannel

If there is an odd number of channels and the symmetry line cuts the last channel in half, make HalfChannel = True to model the (last) half channel on the symmetry line

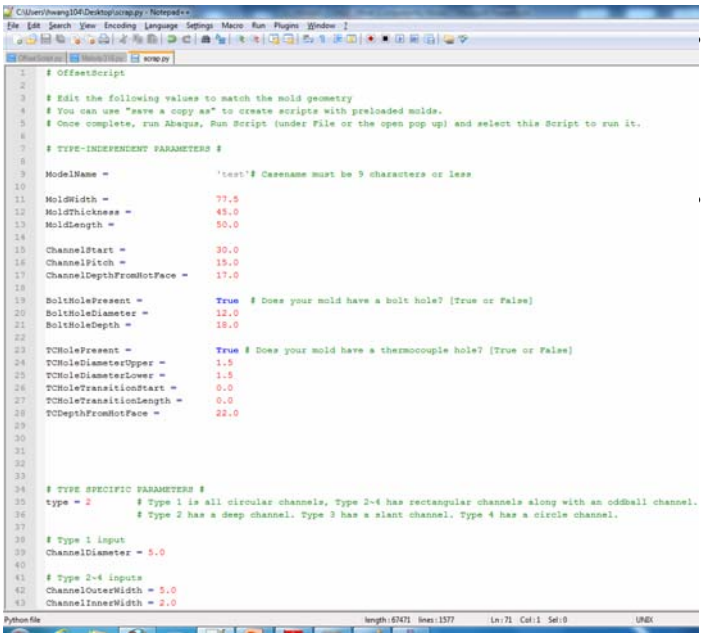
Input data: echo to screen?

```
67  # Would you like the input values printed along with the CON1D parameters? [True or False]
68  EchoInput = True
```

```
Echoing Input Values
ModelName                       =  Corus

MoldWidth                       =  106.25
MoldThickness                   =  55.5
MoldLength                      =  87.5

ChannelStart                    =  18.75
ChannelPitch                    =  10.0
ChannelDepthFromHotFace         =  20.0

BoltHoleDiameter                =  22.0
BoltHoleDepth                   =  20.5

TCHoleDiameterUpper               =  6.0
TCHoleDiameterLower               =  4.0
TCHoleTransitionStart             =  0.0
TCHoleTransitionLength            =  1.0
TCDepthFromHotFace                =  15.0

Type =  4


Type 4 parameters:
BoltChannelCenterX          =  9.0
BoltChannelCenterY          =  26.0
BoltChannelDiameter         =  12.0

ChannelOuterWidth           =  5.0
ChannelInnerWidth           =  0.0
ChannelDepth                =  15.0
```

If EchoInput is True, then the script will print out the relevant input for the mold.

Execution: Input data into script



```
# OffsetScript

# Edit the following values to match the mold geometry
# You can use "save a copy as" to create scripts with preloaded molds.
# Once complete, run Abaqus, Run Script (under File or the open pop up) and select this Script to run it.

# TYPE-INDEPENDENT PARAMETERS #

ModelName =                 'test'# Casename must be 9 characters or less

MoldWidth =                 77.5
MoldThickness =             45.0
MoldLength =                50.0

ChannelStart =              30.0
ChannelPitch =              15.0
ChannelDepthFromHotFace =   17.0

BoltHolePresent =           True  # Does your mold have a bolt hole? [True or False]
BoltHoleDiameter =          12.0
BoltHoleDepth =             18.0

TCHolePresent =             True # Does your mold have a thermocouple hole? [True or False]
TCHoleDiameterUpper =       1.5
TCHoleDiameterLower =       1.5
TCHoleTransitionStart =     0.0
TCHoleTransitionLength =    0.0
TCDepthFromHotFace =        22.0



# TYPE SPECIFIC PARAMETERS #
type = 2        # Type 1 is all circular channels, Type 2-4 has rectangular channels along with an oddball channel.
                # Type 2 has a deep channel. Type 3 has a slant channel. Type 4 has a circle channel.

# Type 1 input
ChannelDiameter = 5.0

# Type 2-4 inputs
ChannelOuterWidth = 5.0
ChannelInnerWidth = 2.0
```

Open the Offset Python File in a text editor. (Notepad++ is free and highly recommended)

Insert the mold geometry by editing the values corresponding to the dimensions shown in the schematics

Execution: enter data for all molds

```
12  ChannelStart =              30.0
13  ChannelPitch =              15.0
14  ChannelDepthFromHotFace =   20.0
15
16  BoltHolePresent =           True
17  BoltHoleDiameter =          12.0
18  BoltHoleDepth =             12.0
19
20  TCHolePresent =             True
21  TCHoleDiameterUpper =       4.0
22  TCHoleDiameterLower =       4.0
23  TCHoleTransitionStart =     0.0
24  TCHoleTransitionLength =    0.0
25  TCDepthFromHotFace =        5.0
```

→

```
15  ChannelStart =              30.0
16  ChannelPitch =              15.0
17  ChannelDepthFromHotFace =   22.0
18
19  BoltHolePresent =           False
20  BoltHoleDiameter =          12.0
21  BoltHoleDepth =             12.0
22
23  TCHolePresent =             True
24  TCHoleDiameterUpper =       5.0
25  TCHoleDiameterLower =       3.0
26  TCHoleTransitionStart =     3.0
27  TCHoleTransitionLength =    4.0
28  TCDepthFromHotFace =        10.0
```

Change the above (red data and blue choices), to match your mold geometry.

Execution: mold type specific data

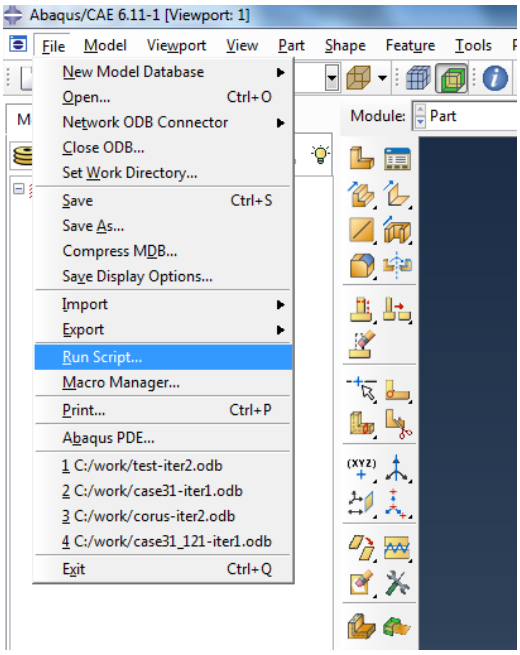
```
28  ######################################
29  # OPTIONAL / TYPE SPECIFIC PARAMETERS #
30  ######################################
31  # These next group of parameter inputs a
32  # E.g. ChannelDiameter only needs to be
33
34  # Type 1 input
35  ChannelDiameter = 5.0
36
37  # Type 2~4 inputs
38  ChannelOuterWidth = 9.0
39  ChannelInnerWidth = 0.0
40  ChannelDepth = 10.0
41
42  # Type 2 input
43  DeepChannelStart = 15.0
44  DeepChannelDepthFromHotFace = 15.0
45  DeepChannelOuterWidth = 9.0
46  DeepChannelInnerWidth = 0.0
47
48  # Type 3 input
49  SlantChannelStart = 15.0
50  SlantChannelDepthFromHotFace = 10.0
51  SlantChannelAngle = pi/10  # the angle
52
53  # Type 4 inputs
54  BoltChannelCenterX = 9.0
55  BoltChannelCenterY = 26.0
56  BoltChannelDiameter = 12.0
57
```

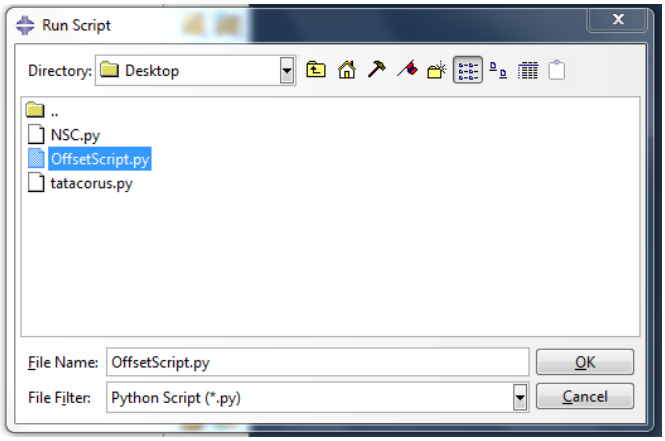
With Type specific parameters, only the values corresponding to the desired type will be considered (other values can be ignored).

E.g. If the mold type was type 2, the only sections that the script will read are the Type 2~4 inputs, and the Type 2 input

Reads these values and ignores the rest

Execution: Run Abaqus

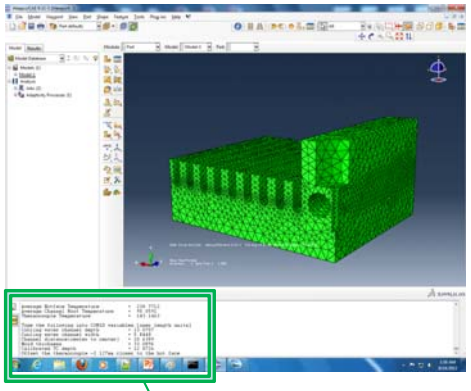




Open Abaqus, and under the File menu click Run Script (left)

Then find and load the OffsetScript and click OK (above).

Execution:
Output results for CON1D calibration



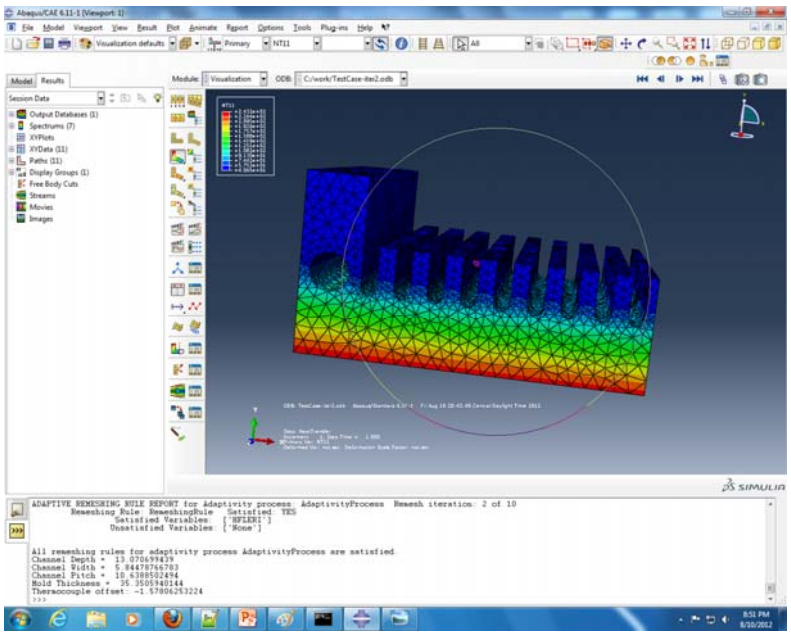
If all the input values are valid, the script will automatically make a mesh and solve with ABAQUS. If not it will state which dimension is invalid.

When complete, Abaqus will present the final mesh and print out the desired values, including the thermocouple offset value.

Typical mesh = ~200,000 tetrahedrons
(for 3% heatflux error target on mesh adaptivity

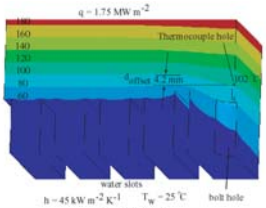
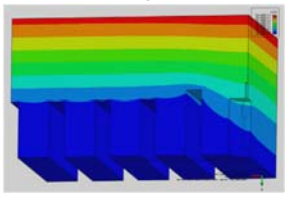
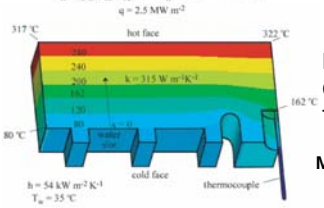
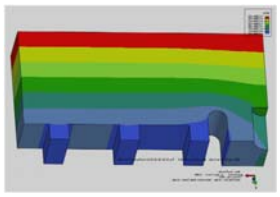
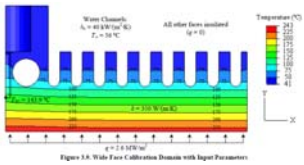
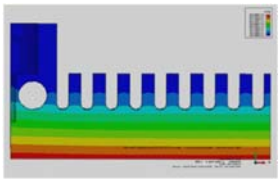
```
Average Hotface Temperature          =  190.8364
Average Channel Root Temperature     =  61.5485
Thermocouple Temperature             =  101.9632

Type the following into CON1D variables [user length units]
Cooling water channel depth          = 21.9644
Cooling water channel width          = 4.2493
Channel distance(center to center)   = 18.3487
Mold thickness                       = 48.8563
Calibrated TC depth                  = 18.4856
i.e. Offset the thermocouple 3.514mm closer to the hot face
```


Execution: 3D detailed results



Because the OffsetScript creates the mold and analyses the effects of the heat loads, after running the script the user can load the .odb file for further analysis.

Validation Cases

Temperature Values (°C):	Abaqus	Calibrated CON1D	Script Results:
			
Hotface:	190.84	191.42	
Coldface:	61.55	62.09	
Thermocouple:	101.96	102.49	
M. Langeneckert MS Thesis Fig 3.16			
			
Hotface:	319.63	319.19*	
Coldface:	86.78	86.36*	
Thermocouple:	161.85	161.37*	
M. Langeneckert MS Thesis Fig 3.1			
			
Hotface:	238.77	238.07	
Coldface:	90.06	89.28	
Thermocouple:	143.15	142.40	
L. Hibbeler MS Thesis Fig 3.9			

*With applied convection coefficient on TC in script of 60 kW/m^{2}K

Conclusion

- We have developed a method to calibrate CON1D to have the accuracy of a 3D FEA model
- This is implemented in a user friendly Python Script that will soon be available

Acknowledgements

- Continuous Casting Consortium Members (ABB, ArcelorMittal, Baosteel, Tata Steel, Goodrich, Magnesita Refractories, Nucor Steel, Nippon Steel, Postech/ Posco, SSAB, ANSYS-Fluent)
- Ron O'Malley, Junya Iwasaki, Melody Langeneckert
- Dassault Systemes (ABAQUS parent company)

- More information:

  L.C. Hibbeler, M.M. Langeneckert, J. Iwasaki, I. Hwang, R.J. O'Malley, and B.G. Thomas, "Calibration of Thermal Models of Continuous Casting of Steel." *AISTech 2012*.