MECH 151L: Finite Element Theory and Applications
Lab Two: Python Scripting in Abaqus

German Markaryan

October 23, 2025

Abstract

The experiment goal is to investigate the usage of a Python script with Abaqus CAE in order to perform multiple finite element analyses with a 40 mm by 40 mm plate with a central circular hole diameter that ranges from 2 mm to 12 mm with a step of 2 mm. The boundary conditions for all six experiments are the same. Finite element results were compared with theoretical results that were determined analytically. Analysis showed that both results follow the same trend. However, the difference between those two solutions is growing as the diameter of the hole increases. This means that the finite element solution is likely accumulating higher error due to the smaller element density on the edge in comparison to the small diameter.

1. Introduction

Abaqus CAE is a powerful instrument for Finite Element Analysis that allows setting up a simulation using a graphical user Interface. Abaqus also supports running Python scripts, which is a perfect way to execute simulations if the parameters must be changed repeatedly. In this experiment, the radius of the hole in a square part will be changed 6 times.

Bodies can sometimes have sudden geometrical features such as holes, fillets, grooves, and cracks that become stress raisers. These stress raisers influence the stress distribution inside the body in a significant way. The stresses near the sudden geometrical features are much higher in comparison with the far-field nominal stresses. These concentration factors can be calculated using the equation below:

$$K_t = \frac{\sigma_{max}}{\sigma_{nom}}$$

where K_t is a stress concentration σ_{nom} is a nominal stress in a far-field and σ_{max} is a maximum stress near the stress raiser. In this experiment, a plate with a circular hole under uniaxial tension is considered, for which K_t = 3 and σ_{nom} = σ_∞. The following equation defines a stress concentration factor for the used geometry:

$$K_t = 3 - 3.14\left(\frac{d}{W}\right) + 3.667\left(\frac{d}{W}\right)^2 - 1.527\left(\frac{d}{W}\right)^3$$

where d is the diameter of the hole and W is the length of the square side. The nominal stress is given by:

$$\sigma_{nom} = \frac{P}{t(W-d)} = \frac{W}{(W-d)}\sigma_\infty$$

where t is the plate thickness and P is the applied force. The following equation provides the value for the far-field stress:

$$\sigma_\infty = \frac{P}{tW}$$

For all simulations, some parameters are kept constant, such as W = 40 mm, t = 1 mm, and P = 400 N.
The equations above are going to play a necessary role in the comparison of theoretical data with simulation results.

For the part of the experiment evolving around simulations, the Python script was changed to execute six simulations with different radius values for the hole. The mesh size for the simulation was chosen to be 0.25 mm, so the results of the experiments will pass convergence validation. The diameter was changed from 2 to 12 mm with a step change of 2 mm.

2. Results and Discussion

The first part of the experiment was to execute a Python script for the largest and smallest diameter holes considered and provide a picture of the mesh. Figure 1 shows a simulation with a hole diameter of 2 mm, and Figure 2 shows a simulation with a hole diameter of 12 mm. Both meshes were properly partitioned.

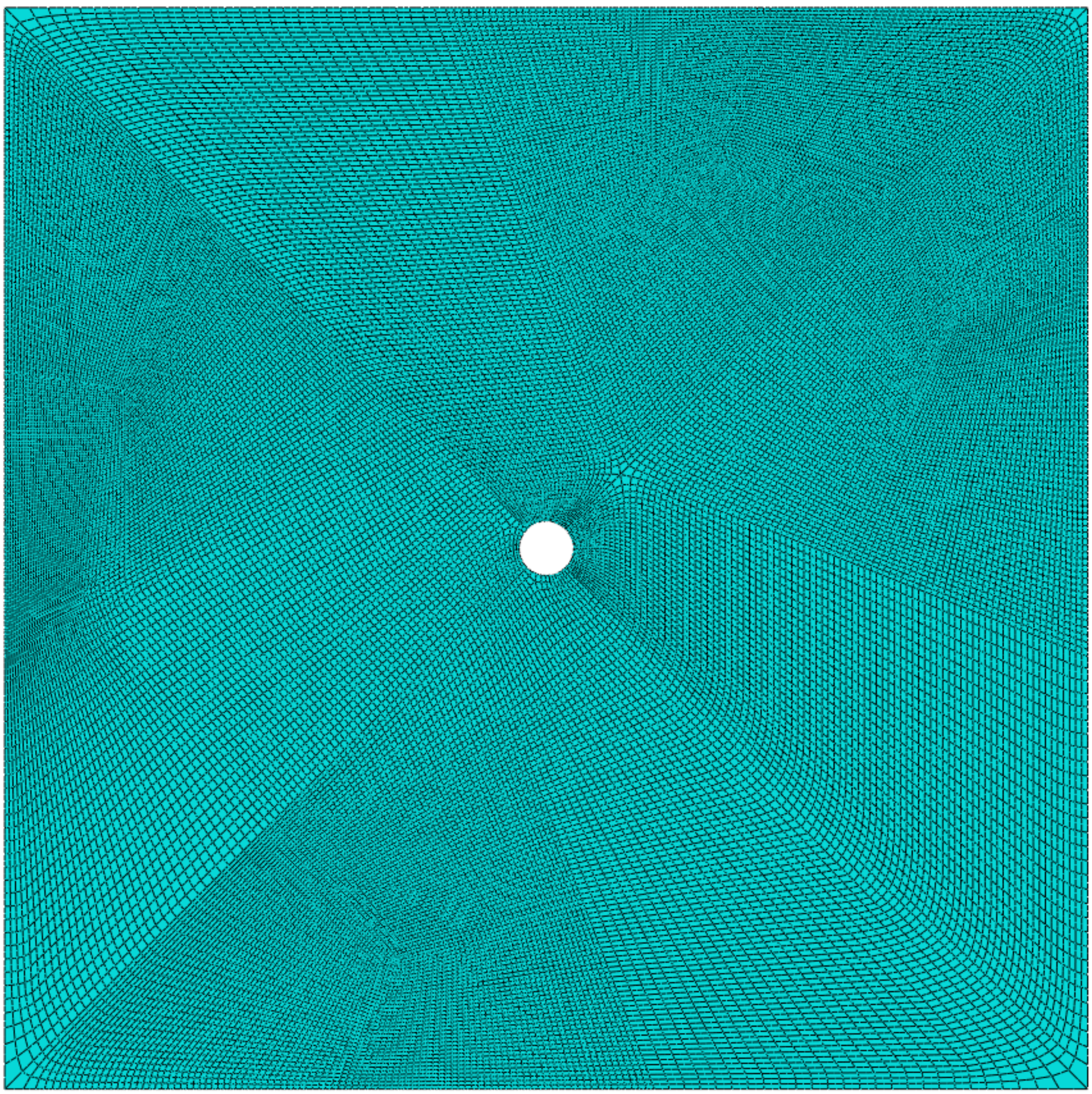

Figure 1: The mesh with an element size of 0.25mm of a 40 mm by 40 mm square plate with a 2 mm diameter hole.

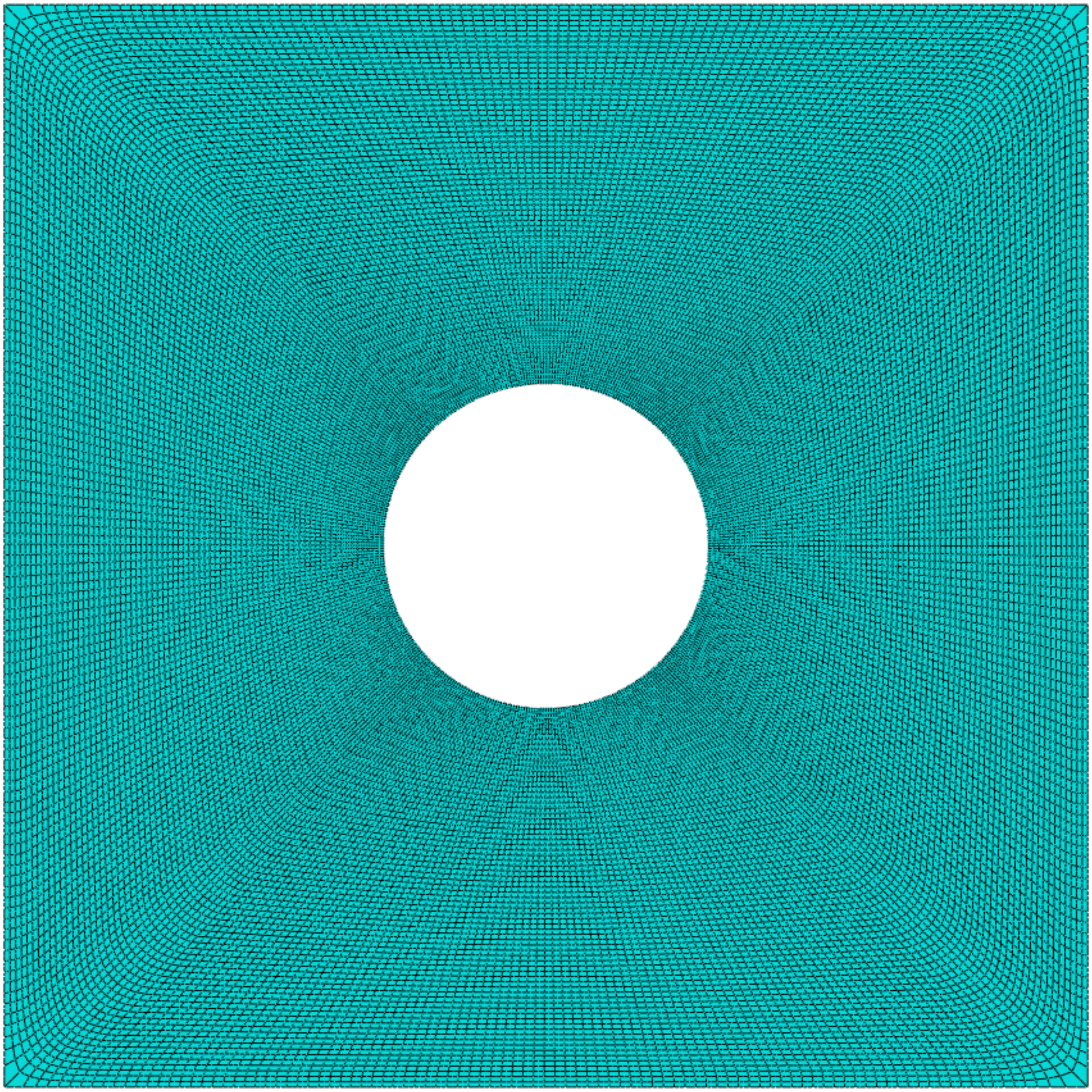

Figure 2: The mesh with an element size of 0.25mm of a 40 mm by 40 mm square plate with a 12 mm diameter hole.

The second part of the experiment aims to determine a counterplot of S22 for the simulations of the specimen with a hole diameter of 2 mm, as shown in Figure 3, and a hole diameter of 12 mm, as shown in Figure 4. An increase in the diameter of the hole influenced the result in two ways. An increase in the stress magnitude in Figure 4 is also expected and happened because more material was removed from the specimen, which means less material is left to resist the stress. General high stress concentration location with correction on size difference looks similar to what is to be expected for simulation of the object with the same general form.

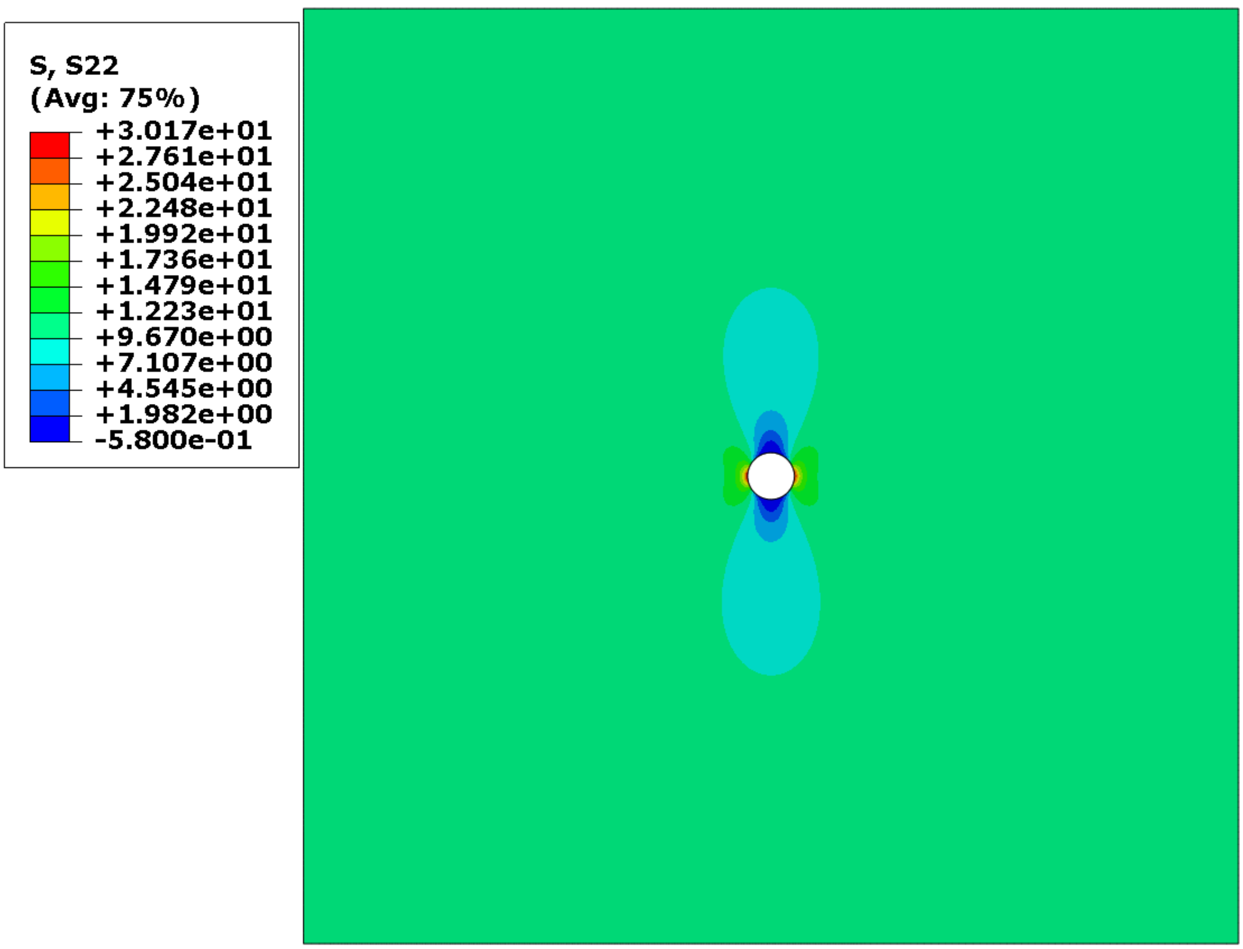

Figure 3: The normal stress in the y-direction S22 (MPa) for a 40 mm by 40 mm square plate with a 2 mm diameter hole under 10 traction. Maximum stress occurs on the left and right sides of the hole boundaries.

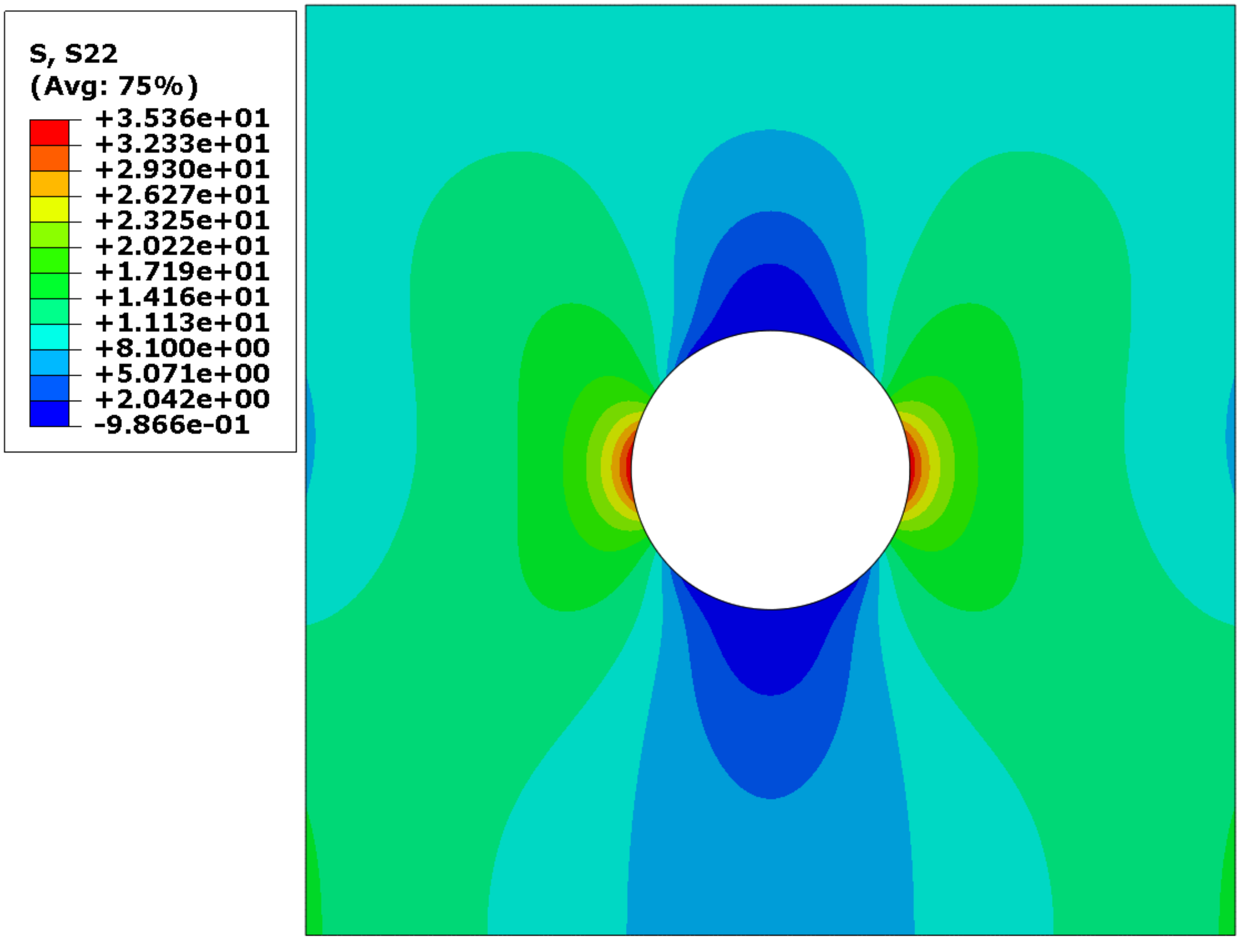

Figure 4: The normal stress in the y-direction S22 (MPa) for a 40 mm by 40 mm square plate with a 12 mm diameter hole under 10 traction. Maximum stress occurs on the left and right sides of the hole boundaries.

In the third part of the experiment, the nodal averaged maximum values of S22 from the contour were collected in Table 1 to calculate $\frac{\sigma_{max}}{\sigma_{nom}}$ as a function of $\frac{d}{W}$. Nodal values of stress for this problem are preferred over integration point because they provide the exact stress values of the nodes without any error, which is not the case for integration point value that provides values with error. To calculate $\frac{\sigma_{max}}{\sigma_{nom}}$ as a function of $\frac{d}{W}$, following equation will be used:

$$K_t = \frac{\sigma_{max}}{\sigma_{nom}} = \frac{\sigma_{max}}{\frac{W}{(W-d)}\sigma_\infty} = 3 - 3.14\left(\frac{d}{W}\right) + 3.667\left(\frac{d}{W}\right)^2 - 1.527\left(\frac{d}{W}\right)^3$$

$$\frac{\sigma_{max}}{\sigma_{nom}} = (3 - 3.14\left(\frac{d}{W}\right) + 3.667\left(\frac{d}{W}\right)^2 - 1.527\left(\frac{d}{W}\right)^3)(1 - \frac{d}{W})^{-1}$$

To find the finite element results, values of diameter from 2 mm to 12 mm will be used with an increment of 2 mm, and the data will be recorded in Table 2.

Table 1: Maximum S22 values for each hole radius.

Hole Radius (mm)	S22 (MPa)
1	3.017E+01
2	3.041E+01
3	3.114E+01
4	3.214E+01
5	3.354E+01
6	3.536E+01

To find values for the finite element results, σ_{nom}= 10 MPa. Final and intermediate values for finite element results are included in Table 2.

Table 2: Comparison of theoretical and FE stress values for varying d/w ratios.

$\frac{d}{W}$	$\frac{\sigma_{max}}{\sigma_\infty}$	**FE results for σ_{max} (MPa)**	**FE results for $\frac{\sigma_{max}}{\sigma_\infty}$**
0.05	3.0021	3.017E+01	3.0170
0.1	3.0235	3.041E+01	3.0410
0.15	3.0663	3.114E+01	3.1140
0.2	3.1331	3.214E+01	3.2140
0.25	3.2271	3.354E+01	3.3540
0.3	3.3526	3.536E+01	3.5360

For the final part of the experiment, the stress concentration factor should be plotted against $\frac{d}{W}$. The concentration factor is determined using the equation below:

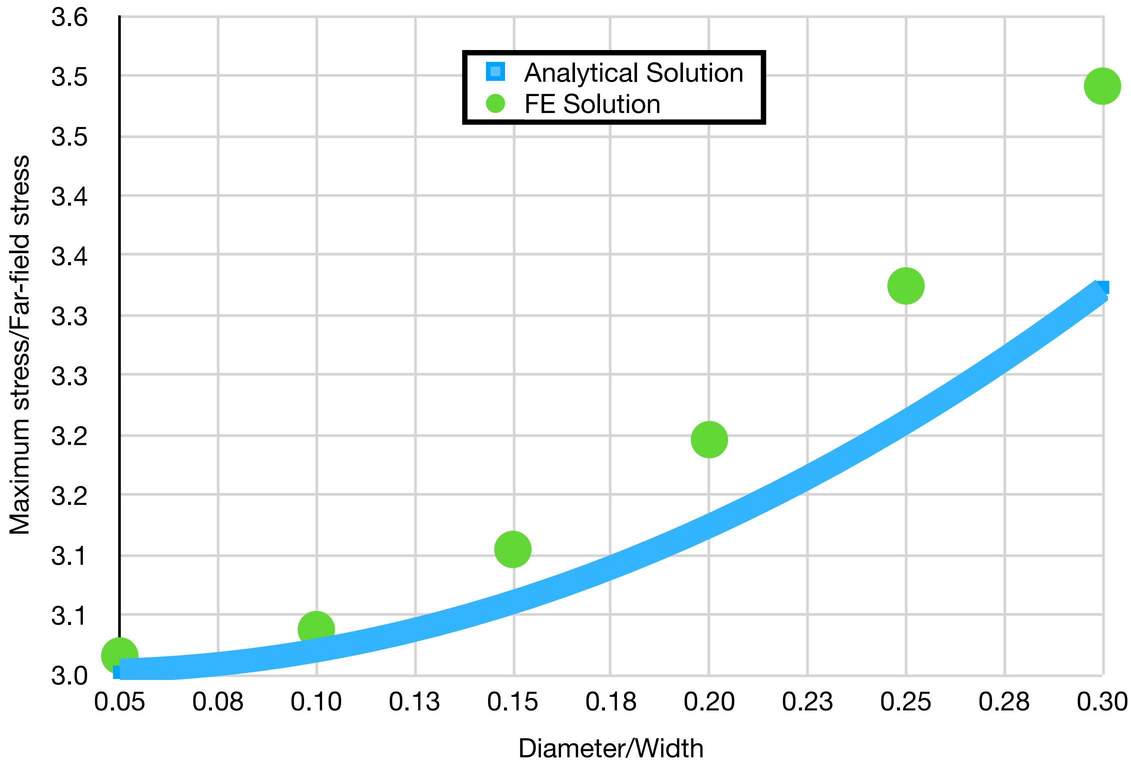

Figure 5: $\frac{\sigma_{max}}{\sigma_{\infty}}$ plotted against $\frac{d}{W}$ for a mm by 40 mm square plate with a hole under traction = 10 MPa. The solid blue line represents the Analytical/Theoretical solution, while green dots represent output from finite element analysis using Abaqus simulation.

$$K_t = 3 - 3.14\left(\frac{d}{W}\right) + 3.667\left(\frac{d}{W}\right)^2 - 1.527\left(\frac{d}{W}\right)^3$$

Values of the concentration factor for both theoretical and finite element results are calculated and entered in Table 3.

Table 3: Values of concentration factor for theoretical and finite element results.

$\frac{d}{W}$	$\frac{\sigma_{max}}{\sigma_{nom}}$	FE results for σ_{max} (MPa)	FE results for σ_{nom} (MPa)	FE results for $\frac{\sigma_{max}}{\sigma_{nom}}$
0.05	2.8520	3.017E+01	10.5263	2.8662
0.1	2.7211	3.041E+01	11.1111	2.7369
0.15	2.6064	3.114E+01	11.7647	2.6469
0.2	2.5065	3.214E+01	12.5000	2.5712
0.25	2.4203	3.354E+01	13.3333	2.5155
0.3	2.3468	3.536E+01	14.2857	2.4752

Theoretical calculations correlate with the results of FEA using Abaqus CAE and Python script. The difference between two are smallest at the minimum diameter and get higher as diameter of the hole increases.

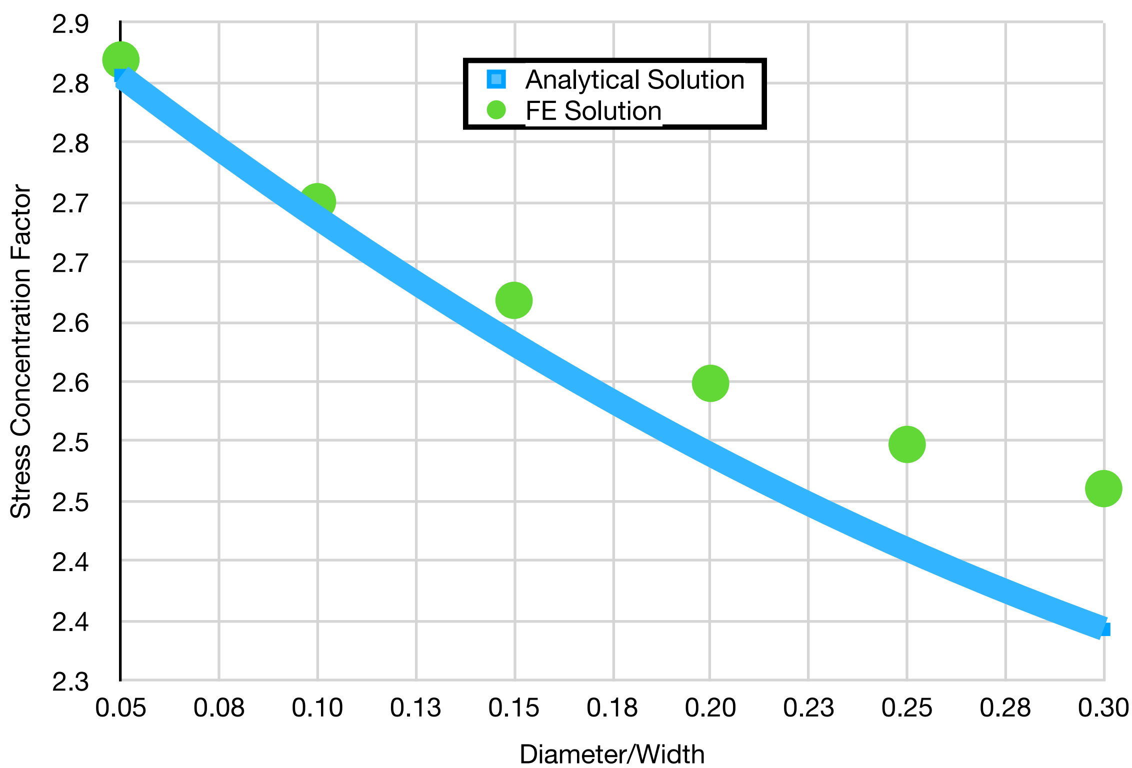

Figure 6: The stress concentration factor K_t plotted against $\frac{d}{W}$ for a 40 mm by 40 mm square plate with a hole under traction = 10 MPa. The solid blue line represents the Analytical/Theoretical solution, while green dots represent output from finite element analysis using Abaqus simulation.

3. Conclusion

This experiment confirmed that repeatable FEA can be executed effectively using Python scripts. During the experiment, the hole diameter was changed six times, which yielded results needed for the analysis of the relationship between the stress concentration factor and hole size. According to calculations and analysis, theoretical values follow the same trend as finite element results. At the minimum values, the difference is minimal, but it starts growing as the diameter of the hole gets larger.

Appendix

```
#--------------------------------------------------------------------------------------------------------
# German Markaryan
# 10/16/2025
#
# MECH 151L: Lab 2
# 2D Plate with center hole under tension displacement BCs. Set up for parameter study of hole size.

#6 diameter: 2; 4; 6; 8; 10; 12;

#Setup----------------------------------------------------------------------------------------------------
Mdb()
pathName = "Z:/dcengr/My Documents/MECH151L/Lab 2/"
os.chdir(pathName)
#Includes-------------------------------------------------------------------------------------------------
from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
#Parameter------------------------------------------------------------------------------------------------
plateHeight = 40
plateWidth = 40
holeRad = 6
youngsMod = 200e3
poissonRatio = 0.3
eps = 1.0e-2
seedSize = 0.25
traction = 10.0

subPath = pathName + "pH" + str(plateHeight) + "-pW" + str(plateWidth) + "_hR" + str(int(holeRad*100))
+"_sS"+str(int(seedSize*100))+"/"
if not os.path.exists(subPath):
   os.makedirs(subPath)
os.chdir(subPath)
#Create
Part-----------------------------------------------------------------------------------------------------
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(-plateWidth/2.0, -plateWidth/2.0),
   point2=(plateWidth/2.0, plateWidth/2.0))
mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
   0.0, 0.0), point1=(holeRad, 0.0))
mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='plate', type=
   DEFORMABLE_BODY)
```

```
mdb.models['Model-1'].parts['plate'].BaseShell(sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']
#
#mdb.models['Model-1'].ConstrainedSketch(gridSpacing=2.82, name='__profile__',
#    sheetSize=113.13, transform=
#    mdb.models['Model-1'].parts['plate'].MakeSketchTransform(
#    sketchPlane=mdb.models['Model-1'].parts['plate'].faces.findAt((8.2699,
#    -6.98557, 0.0), (0.0, 0.0, 1.0)), sketchPlaneSide=SIDE1,
#    sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
#mdb.models['Model-1'].parts['plate'].projectReferencesOntoSketch(filter=
#    COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
#del mdb.models['Model-1'].sketches['__profile__']

#Create
Material---------------------------------------------------------------------------------------------------------
mdb.models['Model-1'].Material(name='steel')
mdb.models['Model-1'].materials['steel'].Elastic(table=((youngsMod, poissonRatio),
    ))
#Section
Assignment---------------------------------------------------------------------------------------------------------

mdb.models['Model-1'].HomogeneousSolidSection(material='steel', name=
    'plateSection', thickness=None)
mdb.models['Model-1'].parts['plate'].SectionAssignment(offset=0.0, offsetField=
    '', offsetType=MIDDLE_SURFACE, region=Region(
    faces=mdb.models['Model-1'].parts['plate'].faces.findAt(((-plateWidth/2.0
    + eps, -plateHeight/2.0 + eps ,0.0), (0.0, 0.0, 1.0)), )), sectionName='plateSection',
thicknessAssignment=FROM_SECTION)

#Instances----------------------------------------------------------------------------------------------------------

mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='plate-1', part=
    mdb.models['Model-1'].parts['plate'])

# Save by gmarkaryan on 2025_10_16-12.39.49; build 2021 2020_03_06-06.50.37 167380


#Step-------------------------------------------------------------------------------------------------------------

mdb.models['Model-1'].StaticStep(name='stepTension', previous='Initial')

#Create Sets and
Surfaces---------------------------------------------------------------------------------------------------------
#might need to fix below:
mdb.models['Model-1'].parts['plate'].Set(edges=
    mdb.models['Model-1'].parts['plate'].edges.findAt(((0.0, -plateHeight/2.0, 0.0), )),
    name='bottomEdge')
mdb.models['Model-1'].parts['plate'].Surface(name='topEdge', side1Edges=
    mdb.models['Model-1'].parts['plate'].edges.findAt(((0.0, plateHeight/2.0, 0.0), )))
mdb.models['Model-1'].rootAssembly.regenerate()

#BCs---------------------------------------------------------------------------------------------------------------
```

```
mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName=
    'stepTension', distributionType=UNIFORM, fieldName='', fixed=OFF,
    localCsys=None, name='BC-1', region=
    mdb.models['Model-1'].rootAssembly.instances['plate-1'].sets['bottomEdge'],
    u1=UNSET, u2=0.0, ur3=0.0)
mdb.models['Model-1'].Pressure(amplitude=UNSET, createStepName='stepTension',
    distributionType=UNIFORM, field='', magnitude=-traction, name='Load-1', region=
    mdb.models['Model-1'].rootAssembly.instances['plate-1'].surfaces['topEdge'])

#Partition into 4
Quadrants------------------------------------------------------------------------------------------------------

mdb.models['Model-1'].ConstrainedSketch(gridSpacing=2.82, name='__profile__',
    sheetSize=113.13, transform=
    mdb.models['Model-1'].parts['plate'].MakeSketchTransform(
    sketchPlane=mdb.models['Model-1'].parts['plate'].faces.findAt((-plateWidth/2.0
    + eps, -plateHeight/2.0 + eps ,0.0), (0.0, 0.0, 1.0)), sketchPlaneSide=SIDE1,
    sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
mdb.models['Model-1'].parts['plate'].projectReferencesOntoSketch(filter=
    COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
mdb.models['Model-1'].sketches['__profile__'].Line(point1=(plateWidth/2.0, plateHeight/2.0), point2=
    (-plateWidth/2.0, -plateHeight/2.0))
mdb.models['Model-1'].sketches['__profile__'].Line(point1=(-plateWidth/2.0, plateHeight/2.0),
    point2=(plateWidth/2.0, -plateHeight/2.0))
mdb.models['Model-1'].parts['plate'].PartitionFaceBySketch(faces=
    mdb.models['Model-1'].parts['plate'].faces.findAt(((plateWidth/2.0, plateHeight/2.0, 0.0),
    )), sketch=mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']

#Create
Mesh------------------------------------------------------------------------------------------------------------

mdb.models['Model-1'].parts['plate'].setMeshControls(algorithm=MEDIAL_AXIS,
    elemShape=QUAD, regions=mdb.models['Model-1'].parts['plate'].faces.findAt(((
    0, plateHeight/2.0-eps, 0.0), ), ((0, -plateHeight/2.0+eps, 0.0), ), ((plateWidth/2.0-eps,
   0, 0.0), ), ((-plateWidth/2.0+eps, 0, 0.0), ), ))
mdb.models['Model-1'].parts['plate'].setElementType(elemTypes=(ElemType(
    elemCode=CPS8R, elemLibrary=STANDARD), ElemType(elemCode=CPS6M,
    elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['plate'].faces.findAt(((0, plateHeight/2.0-eps, 0.0), ), ((0, -plateHeight/
2.0+eps, 0.0), ),
    ((plateWidth/2.0-eps,
   0, 0.0), ), ((-plateWidth/2.0+eps, 0, 0.0), ), ), ), )
mdb.models['Model-1'].parts['plate'].seedPart(deviationFactor=0.1,
    minSizeFactor=0.1, size=seedSize)
mdb.models['Model-1'].parts['plate'].generateMesh()
mdb.models['Model-1'].rootAssembly.regenerate()

#Create/Submit
Job-------------------------------------------------------------------------------------------------------------

mdb.Job(atTime=None, contactPrint=OFF, description='', echoPrint=OFF,
    explicitPrecision=SINGLE, getMemoryFromAnalysis=True, historyPrint=OFF,
```

```
    memory=90, memoryUnits=PERCENTAGE, model='Model-1', modelPrint=OFF,
    multiprocessingMode=DEFAULT, name='job_2DHolePlate', nodalOutputPrecision=
    SINGLE, numCpus=1, numGPUs=0, queue=None, resultsFormat=ODB, scratch='',
    type=ANALYSIS, userSubroutine='', waitHours=0, waitMinutes=0)
mdb.jobs['job_2DHolePlate'].submit(consistencyChecking=OFF)
mdb.jobs['job_2DHolePlate'].waitForCompletion()
mdb.saveAs(pathName=subPath + 'job_2DHolePlate.cae')

#Write
Report------------------------------------------------------------------------------------------------------

session.viewports['Viewport: 1'].setValues(displayedObject=None)
o1 = session.openOdb(name=subPath + 'job_2DHolePlate.odb')
session.viewports['Viewport: 1'].setValues(displayedObject=o1)
session.viewports['Viewport: 1'].odbDisplay.display.setValues(plotState=(
    CONTOURS_ON_DEF, ))
odb = session.odbs[subPath + 'job_2DHolePlate.odb']
session.writeFieldReport(fileName='stressReport.rpt', append=ON,
    sortItem='Element Label', odb=odb, step=0, frame=1,
    outputPosition=INTEGRATION_POINT, variable=(('S', INTEGRATION_POINT, ((
    INVARIANT, 'Mises'), (COMPONENT, 'S11'), (COMPONENT, 'S22'), (COMPONENT,
    'S33'), (COMPONENT, 'S12'), )), ))
```