Abaqus Script

In [ ]:
```
from abaqus import *
from abaqusConstants import *
import regionToolset

def matlib(modelname):
    mod = mdb.models[modelname]
    mat = mod.Material('CFRP')
    mat.Density(table=((1600e-12, ), ))
    mat.Elastic(type=ENGINEERING_CONSTANTS,
        table=((130000.0, 10000.0, 10000.0, 0.28, 0.28, 0.5, 4500.0, 4500.0, 3500.0,), ))
    mat.elastic.FailStress(table=((1400.0, 900.0, 30.0, 120.0, 60.0, -0.5, 0.0), ))


def joint_4(modelname, L, W, e, r, layup, esize, Nx, second_bolt=False):

    # Create model and material
    mod = mdb.Model(name=modelname)
    matlib(modelname)

    # --- Create Plate sketch with one or two bolt holes ---
    ske = mod.ConstrainedSketch(name='__profile__', sheetSize=200.0)
    ske.rectangle(point1=(0.0, 0.0), point2=(L, W))
    # First bolt hole
    ske.CircleByCenterPerimeter(
        center=(e, W/2.0),
        point1=(e + r, W/2.0)
    )
    # Second bolt hole if requested
    if second_bolt:
        ske.CircleByCenterPerimeter(
            center=(2*e, W/2.0),
            point1=(2*e + r, W/2.0)
        )

    prt = mod.Part(name='Plate', dimensionality=THREE_D, type=DEFORMABLE_BODY)
    prt.BaseShell(sketch=ske)
    del mod.sketches['__profile__']

    # --- Partitioning for layup and joints ---
    # Midplane for layup normal axis
    dp = prt.DatumPlaneByPrincipalPlane(principalPlane=XZPLANE, offset=W/2.0)
    prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[dp.id], faces=prt.faces)
    # Partition at x = e
    dp = prt.DatumPlaneByPrincipalPlane(principalPlane=YZPLANE, offset=e)
    prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[dp.id], faces=prt.faces)
    # Partition at x = 2e (second bolt)
    if second_bolt:
        dp = prt.DatumPlaneByPrincipalPlane(principalPlane=YZPLANE, offset=2*e)
        prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[dp.id], faces=prt.faces)

    # Sets and surfaces for layup
    regionAllFaces   = prt.Set(name='faces-all', faces=prt.faces)
    normalAxisRegion = prt.Surface(side1Faces=prt.faces, name='SurfOuter')
    primaryAxisRegion = prt.Set(
        edges=prt.edges.findAt(((e/2.0, 0.0, 0.0),)),
        name='edge-prim-axis'
    )

    # Composite layup
    compLayup = prt.CompositeLayup(
        name='LU', elementType=SHELL, offsetType=MIDDLE_SURFACE
    )
    compLayup.Section()
    compLayup.ReferenceOrientation(
        orientationType=DISCRETE, localCsys=None,
        axis=AXIS_3, stackDirection=STACK_3,
        normalAxisDefinition=SURFACE, normalAxisRegion=normalAxisRegion,
        primaryAxisDefinition=EDGE, primaryAxisRegion=primaryAxisRegion,
        primaryAxisDirection=AXIS_1
    )
    # Add plies
    plyno = 1
    for layer in layup:
        reg = prt.sets[layer['region']]
        compLayup.CompositePly(
            suppressed=False,
            plyName='Ply-' + str(plyno),
            region=reg,
            material=layer['mat'],
            thicknessType=SPECIFY_THICKNESS,
            thickness=layer['thi'],
            orientationType=SPECIFY_ORIENT,
            orientationValue=layer['ori'],
            axis=AXIS_3,
            numIntPoints=3
        )

        plyno += 1
```

```
# Mesh plate
prt.setMeshControls(regions=regionAllFaces.faces, elemShape=QUAD, technique=STRUCTURED)
prt.seedPart(size=esize)
prt.generateMesh()

# --- Assembly ---
ass = mod.rootAssembly
insPlate = ass.Instance(name='Plate', part=prt, dependent=ON)

# Create one Bolt part
ske = mod.ConstrainedSketch(name='__profile__', sheetSize=200.0)
ske.ConstructionLine(point1=(0.0, -100.0), point2=(0.0, 100.0))
ske.Line(point1=(r, -20.0), point2=(r, 20.0))
prtBolt = mod.Part(name='Bolt', dimensionality=THREE_D, type=ANALYTIC_RIGID_SURFACE)
prtBolt.AnalyticRigidSurfRevolve(sketch=ske)
del mod.sketches['__profile__']

# Bolt positions list
bolt_positions = [e]
if second_bolt:
    bolt_positions.append(2*e)

# Loop to create bolt instances, contacts, RPs and BCs
for i, xpos in enumerate(bolt_positions, start=1):
    inst_name = 'Bolt-' + str(i)
    insBolt = ass.Instance(name=inst_name, part=prtBolt, dependent=ON)
    ass.rotate(
        instanceList=(inst_name,),
        axisPoint=(0.0, 0.0, 0.0),
        axisDirection=(1.0, 0.0, 0.0),
        angle=90.0
    )
    ass.translate(
        instanceList=(inst_name,),
        vector=(xpos, W/2.0, 0.0)
    )

    # Contact interaction
    region1 = ass.Surface(side2Faces=insBolt.faces, name='Surface-' + inst_name)
    edges = insPlate.edges.getByBoundingCylinder(
        center1=(xpos, W/2.0, -20.0),
        center2=(xpos, W/2.0,  20.0),
        radius=r
    )
    region2 = ass.Set(edges=edges, name='Edges-Hole-' + inst_name)
    mod.ContactProperty('Contact-properties')
    mod.SurfaceToSurfaceContactStd(
        name='ContactBolt-' + str(i), createStepName='Initial',
        main=region1, secondary=region2,
        sliding=FINITE, thickness=ON,
        interactionProperty='Contact-properties',
        adjustMethod=NONE, initialClearance=OMIT, datumAxis=None, clearanceRegion=None
    )

    # Rigid body & BC
    rp_id = ass.ReferencePoint(point=(xpos, W/2.0, 0.0)).id
    regionRP = ass.Set(name='RP-' + str(i), referencePoints=(ass.referencePoints[rp_id],))
    mod.RigidBody(
        name='Constraint-Bolt-' + str(i),
        refPointRegion=regionRP,
        surfaceRegion=region1
    )
    mod.EncastreBC(
        name='BC-FIX-' + str(i),
        createStepName='Initial',
        region=regionRP
    )


# Support BCs on plate edges
edges = insPlate.edges.getByBoundingBox(xMax=0.0)
reg = ass.Set(name='edges at x=0', edges=edges)
mod.DisplacementBC(
    name='support-z at x=0', createStepName='Initial',
    region=reg, u3=SET
)
edges = insPlate.edges.getByBoundingBox(xMin=L)
reg   = ass.Set(name='edges at x=L', edges=edges)
sur   = ass.Surface(name='surface at x=L', side1Edges=edges)
mod.DisplacementBC(
    name='support-z at x=L', createStepName='Initial',
    region=reg, u3=SET
)

# Load step & outputs
mod.StaticStep(name='Step-1', previous='Initial')
mod.ShellEdgeLoad(
    name='Nx', createStepName='Step-1',
    region=sur, magnitude=Nx,
    directionVector=((0.0, 0.0, 0.0), (1.0, 0.0, 0.0)),
    distributionType=UNIFORM, traction=GENERAL
)
mod.FieldOutputRequest(
```

```
        name='F-Output-2', createStepName='Step-1',
        variables=('S','E','SE','SF','CFAILURE'),
        layupNames=('Plate.LU',), layupLocationMethod=ALL_LOCATIONS,
        rebar=EXCLUDE
    )

    # job = mdb.Job(name=modelname, model=modelname)
    # job.submit()


layup1 = [ {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':-45  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori': 45  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori': 45  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':-45  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'},
           {'mat':'CFRP' , 'ori':  0  , 'thi':0.6, 'region': 'faces-all'} ]



# joint_4(modelname='A4', L=1000, W=100, e=50, r=10, layup=layup1, esize=5, Nx=1000, second_bolt=False)
# joint_4(modelname='A4', L=1000, W=100, e=50, r=10, layup=layup1, esize=5, Nx=1000, second_bolt=True)
joint_4(modelname='A4', L=1000, W=100/2, e=50, r=10, layup=layup1, esize=5, Nx=1000, second_bolt=False)
```