Abaqus Script

In [ ]:

```
from abaqus import *
from abaqusConstants import *
import numpy as np

mcfrp = {
    "name": "S-glass/Epoxy", "units": "MPa-mm-Mg", "type": "UD", "fiber": "S-glass",
    "Vf": 0.55, "rho": 2000E-12,
    "description": "Typical UD S-glass/Epoxy from TMM4175",
    "E1": 48000, "E2": 11000, "E3": 11000,
    "v12": 0.3, "v13": 0.3, "v23": 0.4,
    "G12": 4200, "G13": 4200, "G23": 3600,
    "a1": 4e-06, "a2": 2.0e-05, "a3": 2.0e-05,
    "XT": 1300, "YT": 40, "ZT": 40,
    "XC": 800, "YC": 140, "ZC": 140,
    "S12": 70, "S13": 70, "S23": 40,
    "f12":-0.5, "f13":-0.5, "f23":-0.5
}


def matlib(modelname):
    mod = mdb.models[modelname]

    # Definerer material
    mat = mod.Material(mcfrp['name'])
    mat.Density(table=((mcfrp['rho'], ), ))
    mat.Elastic(type=ENGINEERING_CONSTANTS,
                table=((mcfrp['E1'], mcfrp['E2'], mcfrp['E3'],
                        mcfrp['v12'], mcfrp['v13'], mcfrp['v23'],
                        mcfrp['G12'], mcfrp['G13'], mcfrp['G23']), ))
    mat.elastic.FailStress(table=((mcfrp['XT'], mcfrp['XC'], mcfrp['YT'], mcfrp['YC'], mcfrp['S12'], mcfrp['f12'], 0.0), ))


def boxpro(modelname, L, b, h, t, n_spars, esize, applied_mass, profile=1):

    # Define length from edge to point load
    load_pos = (2.0 / 3.0) * L
    # Correct h and b (due to offsetType = MIDDLE_SURFACE)
    b -= t
    h -= t

    # Create model
    mod = mdb.Model(name=modelname, modelType=STANDARD_EXPLICIT)
    matlib(modelname)

    # region Sketch
    if profile == 1:
        # ------- Outer Sketch ----------
        ske = mod.ConstrainedSketch(name='__profile__', sheetSize=200.0)
        ske.rectangle(point1=(-b/2.0, -h/2.0), point2=(b/2.0, h/2.0))


        # ------- Inner Sketch ----------

        # Vertical Spars
        if n_spars > 0:
            spacing = b / (n_spars + 1)
            x_pos = -b/2.0 + spacing
            for _ in range(n_spars):
                ske.Line(point1=(x_pos, h/2.0), point2=(x_pos, -h/2.0))
                x_pos += spacing


        # Calculate total length
        total_length = 2*b + 2*h + n_spars*h
        mass_factor = 550 # factor that will give simple box profile t=1
        t = mass_factor / total_length # update t

    elif profile == 2:
        ske = mod.ConstrainedSketch(name='__profile__', sheetSize=200.0)
        ske.Line(point1=(-b/2.0, h/2.0), point2=(b/2.0, h/2.0))

        o1 = 10.0
        o2 = 40.0

        ske.Line(point1=(-b/2.0 + o1, h/2.0), point2=(-b/2.0 + o2, -h/2.0))
        ske.Line(point1=(b/2.0 - o1, h/2.0), point2=(b/2.0 - o2, -h/2.0))
        ske.Line(point1=(-b/2.0 + o2, -h/2.0), point2=(b/2.0 - o2, -h/2.0))

        # Vertical Spars
        if n_spars > 0:
            spacing = (b - 2*o2) / (n_spars + 1)
            x_pos = -b/2.0 + o2 + spacing
            for _ in range(n_spars):
                ske.Line(point1=(x_pos, h/2.0), point2=(x_pos, -h/2.0))
                x_pos += spacing

        # Calculate total length
        total_length = b + (b-2*o2) + 2*np.sqrt((o2-o1)**2 + h**2) + n_spars*h
```

```python
        mass_factor = 550 # factor that will give simple box profile t=1
        t = mass_factor / total_length # update t

    elif profile == 3:
        pass # Legg til profil her


    # Create part from sketch
    prt = mod.Part(name='Box', dimensionality=THREE_D, type=DEFORMABLE_BODY)
    prt.BaseShellExtrude(sketch=ske, depth=L)
    del mod.sketches['__profile__']


    # Partition for load surface
    wheel_width = 0.1 * b + 0.1 # added small value
    wheel_length = wheel_width * 2
    id = prt.DatumPlaneByPrincipalPlane(principalPlane=YZPLANE, offset=(wheel_width)).id
    prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[id], faces=prt.faces)
    id = prt.DatumPlaneByPrincipalPlane(principalPlane=YZPLANE, offset=(-wheel_width)).id
    prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[id], faces=prt.faces)

    id = prt.DatumPlaneByPrincipalPlane(principalPlane=XYPLANE, offset=(load_pos + wheel_length)).id
    prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[id], faces=prt.faces)
    id = prt.DatumPlaneByPrincipalPlane(principalPlane=XYPLANE, offset=(load_pos - wheel_length)).id
    prt.PartitionFaceByDatumPlane(datumPlane=prt.datums[id], faces=prt.faces)

    """
    # Material and section
    mat = mod.Material(name='Alu')
    mat.Density(table=((2.7E-9, ), ))
    mat.Elastic(table=((70000.0, 0.33), ))
    mod.HomogeneousShellSection(name='Section-shell',
        preIntegrate=OFF, material='Alu', thicknessType=UNIFORM, thickness=t,
        thicknessField='', nodalThicknessField='',
        idealization=NO_IDEALIZATION, poissonDefinition=DEFAULT,
        thicknessModulus=None, temperature=GRADIENT, useDensity=OFF,
        integrationRule=SIMPSON, numIntPts=5)
    region = prt.Set(faces=prt.faces, name='faces-all')
    prt.SectionAssignment(region=region, sectionName='Section-shell', offset=0.0,
        offsetType=MIDDLE_SURFACE, offsetField='', thicknessAssignment=FROM_SECTION)

   """

    # Mesh
    prt.setMeshControls(regions=prt.faces, elemShape=QUAD, technique=STRUCTURED)
    prt.seedPart(size=esize, deviationFactor=0.1, minSizeFactor=0.1)
    prt.generateMesh()


    # Assembly and constraints
    ass = mod.rootAssembly
    ass.DatumCsysByDefault(CARTESIAN)
    ins = ass.Instance(name='Box', part=prt, dependent=ON)
    ass.rotate(instanceList=('Box', ), axisPoint=(0.0, 0.0, 0.0), axisDirection=(0.0, 1.0, 0.0), angle=90.0)
    ass.rotate(instanceList=('Box', ), axisPoint=(0.0, 0.0, 0.0), axisDirection=(1.0, 0.0, 0.0), angle=90.0)
    rf1id = ass.ReferencePoint(point=(0.0, 0.0, 0.0)).id
    rf2id = ass.ReferencePoint(point=(L, 0.0, 0.0)).id
    regionRF1=ass.Set(referencePoints=(ass.referencePoints[rf1id],), name='RF1')
    regionRF2=ass.Set(referencePoints=(ass.referencePoints[rf2id],), name='RF2')
    edges1=ins.edges.getByBoundingBox(xMax=0.0)
    region1 = ass.Set(edges=edges1, name = 'EDGES1')
    edges2=ins.edges.getByBoundingBox(xMin=L)
    region2 = ass.Set(edges=edges2, name = 'EDGES2')
    mod.MultipointConstraint(name='Constraint-1',
        controlPoint=regionRF1, surface=region1, mpcType=BEAM_MPC,
        userMode=DOF_MODE_MPC, userType=0, csys=None)
    mod.MultipointConstraint(name='Constraint-2',
        controlPoint=regionRF2, surface=region2, mpcType=BEAM_MPC,
        userMode=DOF_MODE_MPC, userType=0, csys=None)

    # Steps, BC and loading
    bc1 = mod.DisplacementBC(name='BC1', createStepName='Initial',
        region=regionRF1, u1=SET, u2=SET, u3=SET, ur1=SET, ur2=SET, ur3=SET)
    bc2 = mod.DisplacementBC(name='BC2', createStepName='Initial',
        region=regionRF2, u1=SET, u2=SET, u3=SET, ur1=SET, ur2=SET, ur3=SET)

    mod.BuckleStep(name='Step-Buck', previous='Initial', numEigen=2, vectors=4, maxIterations=700)
    mod.StaticStep(name='Step-Stat', previous='Step-Buck')


    bc1.setValuesInStep(stepName='Step-Buck',
        ur2=FREED, buckleCase=PERTURBATION_AND_BUCKLING)
    bc1.setValuesInStep(stepName='Step-Stat',
        ur2=FREED)
    bc2.setValuesInStep(stepName='Step-Buck',
        u1=FREED, u3=FREED, buckleCase=PERTURBATION_AND_BUCKLING)
    bc2.setValuesInStep(stepName='Step-Stat',
        u1=FREED, u3=FREED)

    # Create load surface
    faces = ins.faces.getByBoundingBox(
        xMin=load_pos-wheel_length, xMax=load_pos+wheel_length,
        zMin=h/2.0,
        yMax=wheel_width, yMin=-wheel_width
```

```python
    )
    region3 = ass.Set(name='CONTACT-SURFACE', faces=faces)

    # Add inertia to load surface
    ass.engineeringFeatures.NonstructuralMass(
        name='Inertia-1', region=region3, units=TOTAL_MASS, magnitude=applied_mass,
        distribution=MASS_PROPORTIONAL)

    # Add gravity
    mod.Gravity(name='Load-1', createStepName='Step-Buck',
        comp3=-9810.0, distributionType=UNIFORM, field='')
    mod.Gravity(name='Load-2', createStepName='Step-Stat',
        comp3=-9810.0, distributionType=UNIFORM, field='')


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


# boxpro(modelname='BP-1', L=1800, b=200, h=75, t=0.5, n_spars=0, esize=18, applied_mass=0.05, profile=2)
boxpro(modelname='BP-2', L=1800, b=200, h=75, t=0.5, n_spars=1, esize=18, applied_mass=0.05, profile=2)
# boxpro(modelname='BP-3', L=1800, b=200, h=75, t=0.5, n_spars=2, esize=18, applied_mass=0.05, profile=2)
```