步骤1，导入PYTHON所使用的ABAQUS各个模块 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 \* import numpy as np 以上就导入了所有在建模及分析过程中可能用到的工具包。

步骤2：根据铝板的几何尺寸建立模型 mdb.models['Model-1'].ConstrainedSketch(name='profile', sheetSize=1.0) mdb.models['Model-1'].sketches['profile'].rectangle(point1=(0.0, 0.0), point2=(0.5, 0.001)) mdb.models['Model-1'].Part(dimensionality=TWO\_D\_PLANAR, name='Part-1', type=DEFORMABLE\_BODY) mdb.models['Model-1'].parts['Part-1'].BaseShell(sketch= mdb.models['Model-1'].sketches['profile']) del mdb.models['Model-1'].sketches['profile'] mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.02, name='profile', sheetSize=1.0, transform= mdb.models['Model-1'].parts['Part-1'].MakeSketchTransform( sketchPlane=mdb.models['Model-1'].parts['Part-1'].faces[0], sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.25, 0.0005, 0.0))) mdb.models['Model-1'].parts['Part-1'].projectReferencesOntoSketch(filter= COPLANAR\_EDGES, sketch=mdb.models['Model-1'].sketches['profile']) mdb.models['Model-1'].sketches['profile'].Line(point1=(-0.235, 0.0005), point2=(-0.235, -0.000500000040978193)) mdb.models['Model-1'].sketches['profile'].VerticalConstraint(addUndoState= False, entity=mdb.models['Model-1'].sketches['profile'].geometry[6]) mdb.models['Model-1'].sketches['profile'].PerpendicularConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].geometry[4], entity2= mdb.models['Model-1'].sketches['profile'].geometry[6]) mdb.models['Model-1'].sketches['profile'].CoincidentConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].vertices[4], entity2= mdb.models['Model-1'].sketches['profile'].geometry[4]) mdb.models['Model-1'].sketches['profile'].CoincidentConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].vertices[5], entity2= mdb.models['Model-1'].sketches['profile'].geometry[2]) mdb.models['Model-1'].sketches['profile'].Line(point1=(-0.23, 0.0005), point2=(-0.23, -0.000500000040978193)) mdb.models['Model-1'].sketches['profile'].VerticalConstraint(addUndoState= False, entity=mdb.models['Model-1'].sketches['profile'].geometry[7]) mdb.models['Model-1'].sketches['profile'].PerpendicularConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].geometry[4], entity2= mdb.models['Model-1'].sketches['profile'].geometry[7]) mdb.models['Model-1'].sketches['profile'].CoincidentConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].vertices[6], entity2= mdb.models['Model-1'].sketches['profile'].geometry[4]) mdb.models['Model-1'].sketches['profile'].CoincidentConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].vertices[7], entity2= mdb.models['Model-1'].sketches['profile'].geometry[2]) mdb.models['Model-1'].sketches['profile'].Line(point1=(0.23, 0.0005), point2=(0.23, -0.000500000040978193)) mdb.models['Model-1'].sketches['profile'].VerticalConstraint(addUndoState= False, entity=mdb.models['Model-1'].sketches['profile'].geometry[8]) mdb.models['Model-1'].sketches['profile'].PerpendicularConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].geometry[4], entity2= mdb.models['Model-1'].sketches['profile'].geometry[8]) mdb.models['Model-1'].sketches['profile'].CoincidentConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].vertices[8], entity2= mdb.models['Model-1'].sketches['profile'].geometry[4]) mdb.models['Model-1'].sketches['profile'].CoincidentConstraint( addUndoState=False, entity1= mdb.models['Model-1'].sketches['profile'].vertices[9], entity2= mdb.models['Model-1'].sketches['profile'].geometry[2])

步骤3，给模型赋予相应的材料属性 mdb.models['Model-1'].Material(name='Material-1') mdb.models['Model-1'].materials['Material-1'].Density(table=((2700.0, ), )) mdb.models['Model-1'].materials['Material-1'].Elastic(table=((70000000000.0, 0.3), )) mdb.models['Model-1'].HomogeneousSolidSection(material='Material-1', name= 'Section-1', thickness=None) mdb.models['Model-1'].parts['Part-1'].SectionAssignment(offset=0.0, offsetField='', offsetType=MIDDLE\_SURFACE, region=Region( faces=mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask( mask=('[#f ]', ), )), sectionName='Section-1', thicknessAssignment= FROM\_SECTION)

步骤4，设置分析步 mdb.models['Model-1'].ExplicitDynamicsStep(improvedDtMethod=ON, name='Step-1', previous='Initial', timePeriod=0.00015) mdb.models['Model-1'].fieldOutputRequests['F-Output-1'].setValues(timeInterval= EVERY\_TIME\_INCREMENT, variables=('U', )) mdb.models['Model-1'].historyOutputRequests['H-Output-1'].setValues( numIntervals=1, rebar=EXCLUDE, region= mdb.models['Model-1'].rootAssembly.allInstances['Part-1-1'].sets['Set-S1'], sectionPoints=DEFAULT, variables=('U1', 'U2', 'U3', 'UR1', 'UR2', 'UR3')) mdb.models['Model-1'].HistoryOutputRequest(createStepName='Step-1', frequency=1 , name='H-Output-2', rebar=EXCLUDE, region= mdb.models['Model-1'].rootAssembly.allInstances['Part-1-1'].sets['Set-S2'], sectionPoints=DEFAULT, variables=('U1', 'U2', 'U3', 'UR1', 'UR2', 'UR3')) mdb.models['Model-1'].historyOutputRequests['H-Output-1'].setValues(frequency= 1)

步骤5，布种、划分网格并导出INPUT文件 mdb.models['Model-1'].rootAssembly.seedPartInstance(deviationFactor=0.1, minSizeFactor=0.1, regions=( mdb.models['Model-1'].rootAssembly.instances['Part-1-1'], ), size=0.0002) mdb.models['Model-1'].rootAssembly.generateMesh(regions=( mdb.models['Model-1'].rootAssembly.instances['Part-1-1'], )) mdb.Job(activateLoadBalancing=False, atTime=None, contactPrint=OFF, description='', echoPrint=OFF, explicitPrecision=SINGLE, historyPrint=OFF, memory=90, memoryUnits=PERCENTAGE, model='Model-1', modelPrint=OFF, multiprocessingMode=DEFAULT, name='Job-1', nodalOutputPrecision=SINGLE, numCpus=1, numDomains=1, parallelizationMethodExplicit=DOMAIN, queue=None, resultsFormat=ODB, scratch='', type=ANALYSIS, userSubroutine='', waitHours=0, waitMinutes=0) mdb.jobs['Job-1'].writeInput(consistencyChecking=OFF)