

**研究生期末大作业**

课程名称： 有限元方法及应用

研究生姓名： 陈桓隆

学号： S230200160

任课教师： 王琥

目录

[一．问题重述 3](#_Toc159529743)

[1.1问题描述 3](#_Toc159529744)

[1.2理论分析 3](#_Toc159529745)

[二. 悬臂梁有限元分析的过程 4](#_Toc159529746)

[2.1线类型模型分析 4](#_Toc159529747)

[2.1.1前处理 4](#_Toc159529748)

[2.1.2求解 4](#_Toc159529749)

[2.1.3后处理 4](#_Toc159529750)

[2.2实体类型模型分析 5](#_Toc159529751)

[2.2.1前处理 5](#_Toc159529752)

[2.2.2求解 5](#_Toc159529753)

[2.2.3后处理 6](#_Toc159529754)

[三．误差分析 7](#_Toc159529755)

[3.1材料力学手算 7](#_Toc159529756)

[3.2误差分析 7](#_Toc159529757)

[四．附录代码 8](#_Toc159529758)

[4.1代码解释 8](#_Toc159529759)

[4.2代码 8](#_Toc159529760)

[4.2.1线模型代码 8](#_Toc159529761)

[4.2.2实体模型代码 11](#_Toc159529762)

[五.致谢 15](#_Toc159529763)

悬臂梁abaqus有限元建模方法分析

陈桓隆

# 一．问题重述

## 1.1问题描述

如图1所示的悬臂梁尺寸为，单位为mm。杨氏模量E=2.1e5Mpa，泊松比。左端完全固定，右端施加于y方向大小为-100N的集中荷载，分析悬臂梁自由端的变形。

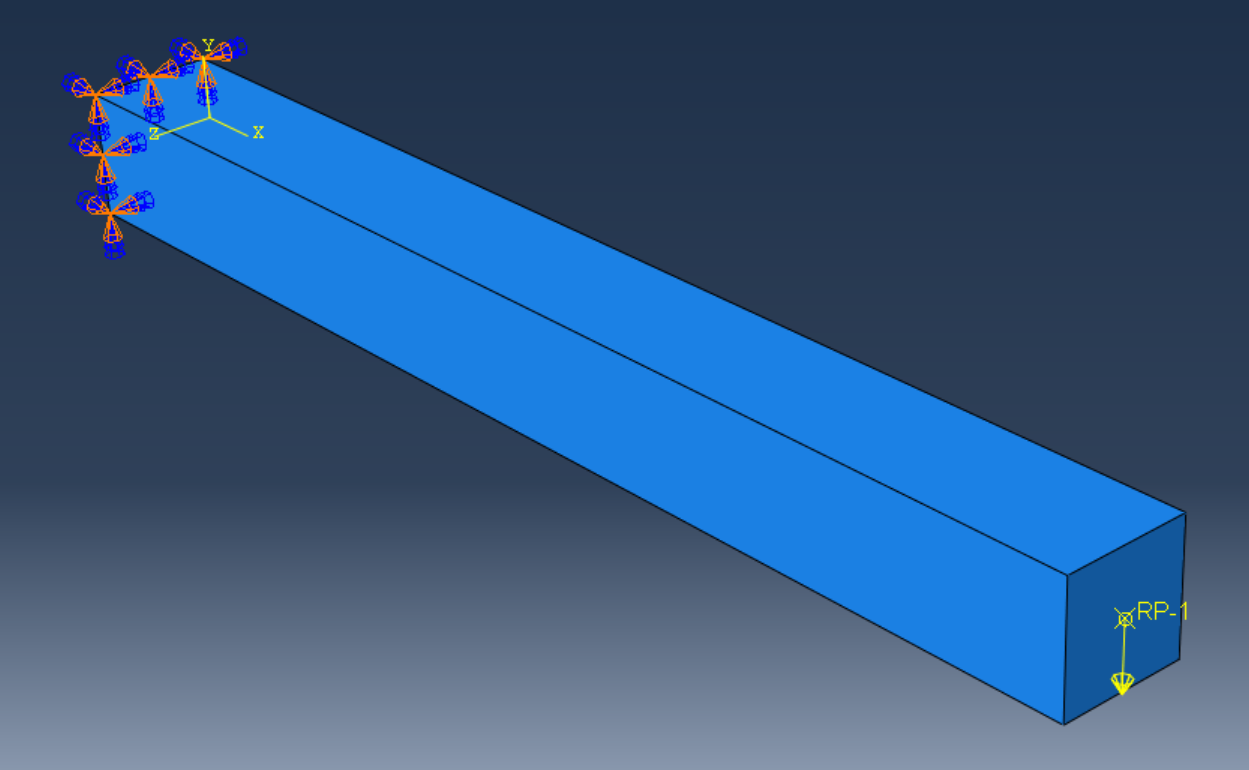


图1.悬臂梁的结构模型

## 1.2理论分析

于abaqus中建立悬臂梁模型，建立采取模型空间为三维，类型为可变形但形状分别采用实体以及线两种不同类型的模型，计算各自最终结果与手算结果进行误差分析，确定两种不同建模方式的可行性。

# 二. 悬臂梁有限元分析的过程

## 2.1线类型模型分析

### 2.1.1前处理

几何模型的构建：采用模型空间为三维，类型为可变性，形状为线的部件类型，得到一根长度为的线。然后创建一个边长为的矩形剖面，以该剖面为截面类型，指派该截面为线的截面。

材料参数的定义：杨氏模量为2.1e5Mpa，泊松比为0.3。

网络系统的构建：采用B31梁单元类型。

### 2.1.2求解

求解器的设定：设置一个静力学通用的分析步，设置保持默认。

边界条件的设定：模型左端点设置为完全固定，右端点施加一个延y方向负方向大小为100N的集中荷载，如图2所示。

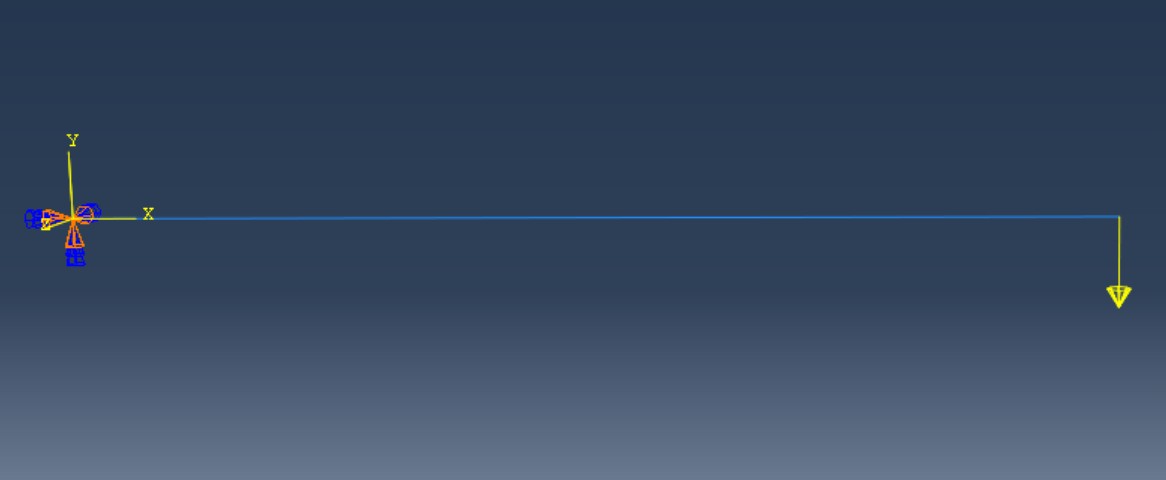


图2.线模型构建

### 2.1.3后处理

创建作业文件xian，提交计算。

打开视图ODB渲染选项，得到图3。



图3.线模型后处理图像

## 2.2实体类型模型分析

### 2.2.1前处理

几何模型的构建：采用模型空间为三维，类型为可变性，形状为实体的部件类型，得到一根长度为，边长为的梁。

材料参数的定义：杨氏模量为2.1e5Mpa，泊松比为0.3。

网络系统的构建：采用C3D8R八结点线性六面体单元，10000个单元节点。

### 2.2.2求解

求解器的设定：设置一个静力学通用的分析步，设置保持默认。

边界条件的设定：模型左端设置为完全固定，右端首先在面中心设置一个参考点，然后将整个面耦合到参考点，最后再参考点施加一个延y方向负方向大小为100N的集中荷载，如图4所示。

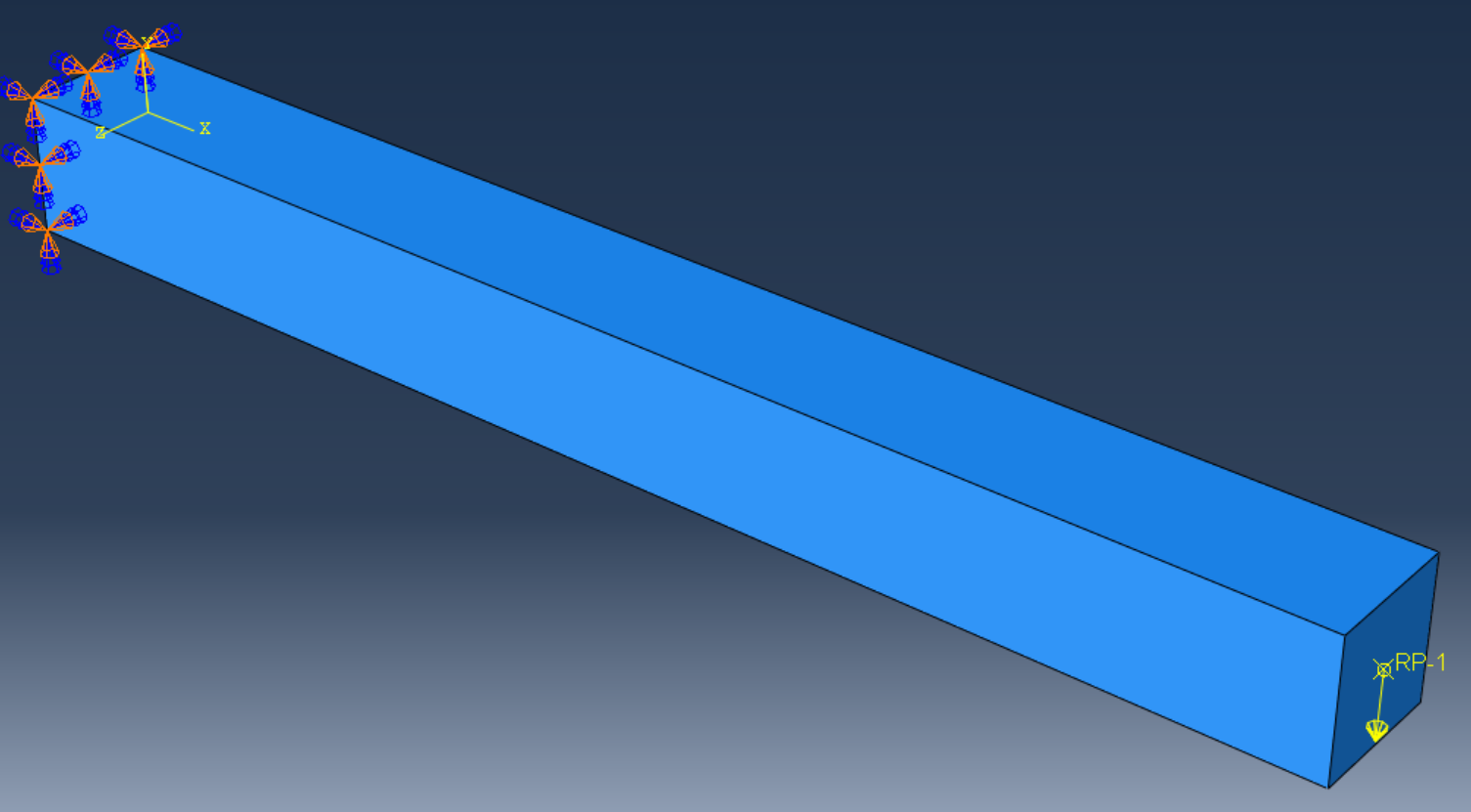


图4.实体模型构建

### 2.2.3后处理

创建作业文件shiti，提交计算，得到图5。

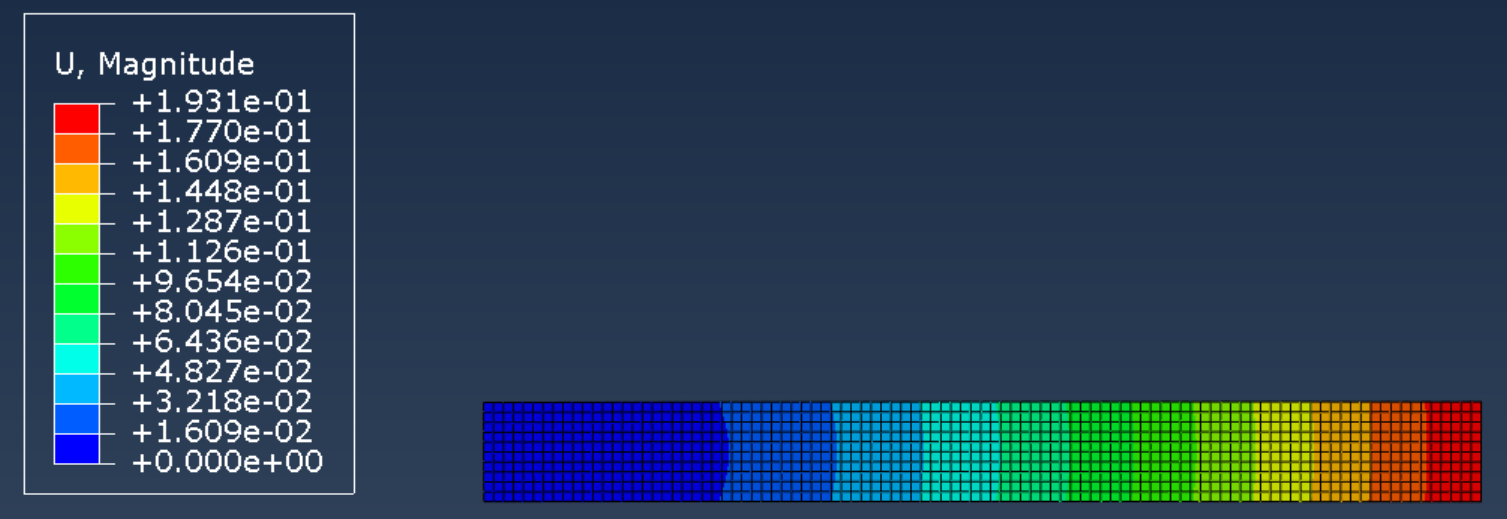


图5.实体模型后处理图像

# 三．误差分析

## 3.1材料力学手算

将问题简化为如图6所示的悬臂梁问题。

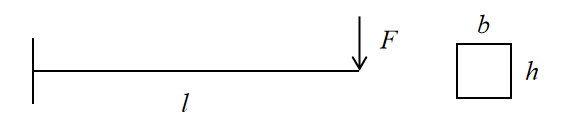


图6.模型简化

由题可知弹性模量，泊松比，根据材料力学计算悬臂梁由集中力计算位移公式：



可得位移

## 3.2误差分析

根据前面几种不同的方式得到的有关最大位移的数据，汇总可得表1。

表1：各种方式误差表（与手算相比）

|  |  |  |  |
| --- | --- | --- | --- |
| 计算方式 | 材料力学手算 | Abaqus线模型 | Abaqus实体模型 |
| 最大位移（mm） | 0.1905 | 0.1919 | 0.1931 |
| 误差 | 0 | 0.735% | 1.365% |

据表1可知，两种不同的Abaqus建模方式与手算所得结果误差均很小，故两种建模方式均合理。

# 四．附录代码

## 4.1代码解释

代码是由abaqus软件附带的宏录制功能自动生成的abaqusMacros.py文件稍加更改后所得，将代码写入py文件后于abaqus中运行即可。

## 4.2代码

### 4.2.1线模型代码

# -\*- coding: mbcs -\*-  
# Do not delete the following import lines  
from abaqus import \*  
from abaqusConstants import \*  
import \_\_main\_\_  
  
  
import section  
import regionToolset  
import displayGroupMdbToolset as dgm  
import part  
import material  
import assembly  
import step  
import interaction  
import load  
import mesh  
import optimization  
import job  
import sketch  
import visualization  
import xyPlot  
import displayGroupOdbToolset as dgo  
import connectorBehavior  
s = mdb.models['Model-1'].ConstrainedSketch(name='\_\_profile\_\_'**,** sheetSize=**200.0**)  
g**,** v**,** d**,** c = s.geometry**,** s.vertices**,** s.dimensions**,** s.constraints  
s.setPrimaryObject(option=STANDALONE)  
s.Line(point1=(**0.0, 0.0**)**,** point2=(**100.0, 0.0**))  
s.HorizontalConstraint(entity=g[**2**]**,** addUndoState=False)  
p = mdb.models['Model-1'].Part(name='Part-1'**,** dimensionality=THREE\_D**,** type=DEFORMABLE\_BODY)  
p = mdb.models['Model-1'].parts['Part-1']  
p.BaseWire(sketch=s)  
s.unsetPrimaryObject()  
p = mdb.models['Model-1'].parts['Part-1']  
session.viewports['Viewport: 1'].setValues(displayedObject=p)  
del mdb.models['Model-1'].sketches['\_\_profile\_\_']  
session.viewports['Viewport: 1'].partDisplay.setValues(sectionAssignments=ON**,** engineeringFeatures=ON)  
session.viewports['Viewport: 1'].partDisplay.geometryOptions.setValues(  
 referenceRepresentation=OFF)  
mdb.models['Model-1'].Material(name='Material-1')  
mdb.models['Model-1'].materials['Material-1'].Elastic(table=((**210000.0, 0.3**)**,** ))  
mdb.models['Model-1'].RectangularProfile(name='Profile-1'**,** a=**10.0,** b=**10.0**)  
mdb.models['Model-1'].BeamSection(name='Section-1'**,** integration=DURING\_ANALYSIS**,** poissonRatio=**0.0,** profile='Profile-1'**,** material='Material-1'**,** temperatureVar=LINEAR**,** consistentMassMatrix=False)  
p = mdb.models['Model-1'].parts['Part-1']  
e = p.edges  
edges = e.getSequenceFromMask(mask=('[#1 ]'**,** )**,** )  
region = regionToolset.Region(edges=edges)  
p = mdb.models['Model-1'].parts['Part-1']  
p.SectionAssignment(region=region**,** sectionName='Section-1'**,** offset=**0.0,** offsetType=MIDDLE\_SURFACE**,** offsetField=''**,** thicknessAssignment=FROM\_SECTION)  
p = mdb.models['Model-1'].parts['Part-1']  
e = p.edges  
edges = e.getSequenceFromMask(mask=('[#1 ]'**,** )**,** )  
region=regionToolset.Region(edges=edges)  
p = mdb.models['Model-1'].parts['Part-1']  
p.assignBeamSectionOrientation(region=region**,** method=N1\_COSINES**,** n1=(**0.0, 0.0,** -**1.0**))  
session.viewports['Viewport: 1'].partDisplay.setValues(sectionAssignments=OFF**,** engineeringFeatures=OFF**,** mesh=ON)  
session.viewports['Viewport: 1'].partDisplay.meshOptions.setValues(  
 meshTechnique=ON)  
p = mdb.models['Model-1'].parts['Part-1']  
p.seedPart(size=**1.0,** deviationFactor=**0.1,** minSizeFactor=**0.1**)  
p = mdb.models['Model-1'].parts['Part-1']  
p.generateMesh()  
a = mdb.models['Model-1'].rootAssembly  
session.viewports['Viewport: 1'].setValues(displayedObject=a)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(  
 optimizationTasks=OFF**,** geometricRestrictions=OFF**,** stopConditions=OFF)  
a = mdb.models['Model-1'].rootAssembly  
a.DatumCsysByDefault(CARTESIAN)  
p = mdb.models['Model-1'].parts['Part-1']  
a.Instance(name='Part-1-1'**,** part=p**,** dependent=ON)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(  
 adaptiveMeshConstraints=ON)  
mdb.models['Model-1'].StaticStep(name='Step-1'**,** previous='Initial')  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(step='Step-1')  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(interactions=ON**,** constraints=ON**,** connectors=ON**,** engineeringFeatures=ON**,** adaptiveMeshConstraints=OFF)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(loads=ON**,** bcs=ON**,** predefinedFields=ON**,** interactions=OFF**,** constraints=OFF**,** engineeringFeatures=OFF)  
a = mdb.models['Model-1'].rootAssembly  
v1 = a.instances['Part-1-1'].vertices  
verts1 = v1.getSequenceFromMask(mask=('[#1 ]'**,** )**,** )  
region = regionToolset.Region(vertices=verts1)  
mdb.models['Model-1'].EncastreBC(name='BC-1'**,** createStepName='Step-1'**,** region=region**,** localCsys=None)  
a = mdb.models['Model-1'].rootAssembly  
v1 = a.instances['Part-1-1'].vertices  
verts1 = v1.getSequenceFromMask(mask=('[#2 ]'**,** )**,** )  
region = regionToolset.Region(vertices=verts1)  
mdb.models['Model-1'].ConcentratedForce(name='Load-1'**,** createStepName='Step-1'**,** region=region**,** cf2=-**100.0,** distributionType=UNIFORM**,** field=''**,** localCsys=None)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(loads=OFF**,** bcs=OFF**,** predefinedFields=OFF**,** connectors=OFF)  
mdb.Job(name='xian'**,** model='Model-1'**,** description=''**,** type=ANALYSIS**,** atTime=None**,** waitMinutes=**0,** waitHours=**0,** queue=None**,** memory=**90,** memoryUnits=PERCENTAGE**,** getMemoryFromAnalysis=True**,** explicitPrecision=SINGLE**,** nodalOutputPrecision=SINGLE**,** echoPrint=OFF**,** modelPrint=OFF**,** contactPrint=OFF**,** historyPrint=OFF**,** userSubroutine=''**,** scratch=''**,** resultsFormat=ODB**,** multiprocessingMode=DEFAULT**,** numCpus=**1,** numGPUs=**0**)  
mdb.jobs['xian'].submit(consistencyChecking=OFF)  
session.mdbData.summary()  
o3 = session.openOdb(name='D:/abaqus/exerciese/fem\_final\_work/chen/xian.odb')  
session.viewports['Viewport: 1'].setValues(displayedObject=o3)  
session.viewports['Viewport: 1'].makeCurrent()  
session.viewports['Viewport: 1'].odbDisplay.display.setValues(plotState=(  
 CONTOURS\_ON\_DEF**,** ))  
session.viewports['Viewport: 1'].odbDisplay.basicOptions.setValues(  
 renderBeamProfiles=ON)  
session.viewports['Viewport: 1'].view.setValues(nearPlane=**313.937,** farPlane=**423.522,** width=**329.056,** height=**147.319,** cameraPosition=(  
 **65.8325, 143.155, 337.314**)**,** cameraUpVector=(-**0.0254501, 0.727746,** -**0.685374**)**,** cameraTarget=(**50.6721,** -**4.65693, 2.99307**))  
session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(  
 variableLabel='U'**,** outputPosition=NODAL**,** refinement=(INVARIANT**,** 'Magnitude')**,** )

### 4.2.2实体模型代码

# -\*- coding: mbcs -\*-  
# Do not delete the following import lines  
from abaqus import \*  
from abaqusConstants import \*  
import \_\_main\_\_  
  
  
import section  
import regionToolset  
import displayGroupMdbToolset as dgm  
import part  
import material  
import assembly  
import step  
import interaction  
import load  
import mesh  
import optimization  
import job  
import sketch  
import visualization  
import xyPlot  
import displayGroupOdbToolset as dgo  
import connectorBehavior  
s = mdb.models['Model-1'].ConstrainedSketch(name='\_\_profile\_\_'**,** sheetSize=**200.0**)  
g**,** v**,** d**,** c = s.geometry**,** s.vertices**,** s.dimensions**,** s.constraints  
s.setPrimaryObject(option=STANDALONE)  
s.Line(point1=(**0.0, 5.0**)**,** point2=(**0.0,** -**5.0**))  
s.VerticalConstraint(entity=g[**2**]**,** addUndoState=False)  
session.viewports['Viewport: 1'].view.setValues(nearPlane=**156.875,** farPlane=**220.249,** width=**256.499,** height=**114.835,** cameraPosition=(  
 **24.2102, 5.36152, 188.562**)**,** cameraTarget=(**24.2102, 5.36152, 0**))  
s.Line(point1=(**0.0,** -**5.0**)**,** point2=(**100.0,** -**5.0**))  
s.HorizontalConstraint(entity=g[**3**]**,** addUndoState=False)  
s.PerpendicularConstraint(entity1=g[**2**]**,** entity2=g[**3**]**,** addUndoState=False)  
s.Line(point1=(**100.0,** -**5.0**)**,** point2=(**100.0, 5.0**))  
s.VerticalConstraint(entity=g[**4**]**,** addUndoState=False)  
s.PerpendicularConstraint(entity1=g[**3**]**,** entity2=g[**4**]**,** addUndoState=False)  
s.Line(point1=(**100.0, 5.0**)**,** point2=(**0.0, 5.0**))  
s.HorizontalConstraint(entity=g[**5**]**,** addUndoState=False)  
s.PerpendicularConstraint(entity1=g[**4**]**,** entity2=g[**5**]**,** addUndoState=False)  
p = mdb.models['Model-1'].Part(name='Part-1'**,** dimensionality=THREE\_D**,** type=DEFORMABLE\_BODY)  
p = mdb.models['Model-1'].parts['Part-1']  
p.BaseSolidExtrude(sketch=s**,** depth=**10.0**)  
s.unsetPrimaryObject()  
p = mdb.models['Model-1'].parts['Part-1']  
session.viewports['Viewport: 1'].setValues(displayedObject=p)  
del mdb.models['Model-1'].sketches['\_\_profile\_\_']  
session.viewports['Viewport: 1'].partDisplay.setValues(sectionAssignments=ON**,** engineeringFeatures=ON)  
session.viewports['Viewport: 1'].partDisplay.geometryOptions.setValues(  
 referenceRepresentation=OFF)  
mdb.models['Model-1'].Material(name='Material-1')  
mdb.models['Model-1'].materials['Material-1'].Elastic(table=((**210000.0, 0.3**)**,** ))  
mdb.models['Model-1'].HomogeneousSolidSection(name='Section-1'**,** material='Material-1'**,** thickness=None)  
p = mdb.models['Model-1'].parts['Part-1']  
c = p.cells  
cells = c.getSequenceFromMask(mask=('[#1 ]'**,** )**,** )  
region = regionToolset.Region(cells=cells)  
p = mdb.models['Model-1'].parts['Part-1']  
p.SectionAssignment(region=region**,** sectionName='Section-1'**,** offset=**0.0,** offsetType=MIDDLE\_SURFACE**,** offsetField=''**,** thicknessAssignment=FROM\_SECTION)  
session.viewports['Viewport: 1'].partDisplay.setValues(sectionAssignments=OFF**,** engineeringFeatures=OFF**,** mesh=ON)  
session.viewports['Viewport: 1'].partDisplay.meshOptions.setValues(  
 meshTechnique=ON)  
p = mdb.models['Model-1'].parts['Part-1']  
p.seedPart(size=**1.0,** deviationFactor=**0.1,** minSizeFactor=**0.1**)  
p = mdb.models['Model-1'].parts['Part-1']  
p.generateMesh()  
a = mdb.models['Model-1'].rootAssembly  
session.viewports['Viewport: 1'].setValues(displayedObject=a)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(  
 optimizationTasks=OFF**,** geometricRestrictions=OFF**,** stopConditions=OFF)  
a = mdb.models['Model-1'].rootAssembly  
a.DatumCsysByDefault(CARTESIAN)  
p = mdb.models['Model-1'].parts['Part-1']  
a.Instance(name='Part-1-1'**,** part=p**,** dependent=ON)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(  
 adaptiveMeshConstraints=ON)  
mdb.models['Model-1'].StaticStep(name='Step-1'**,** previous='Initial')  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(step='Step-1')  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(interactions=ON**,** constraints=ON**,** connectors=ON**,** engineeringFeatures=ON**,** adaptiveMeshConstraints=OFF)  
a = mdb.models['Model-1'].rootAssembly  
v11 = a.instances['Part-1-1'].vertices  
a.DatumPointByMidPoint(point1=v11[**1**]**,** point2=v11[**3**])  
a = mdb.models['Model-1'].rootAssembly  
d11 = a.datums  
a.ReferencePoint(point=d11[**4**])  
a = mdb.models['Model-1'].rootAssembly  
r1 = a.referencePoints  
refPoints1=(r1[**5**]**,** )  
region1=regionToolset.Region(referencePoints=refPoints1)  
a = mdb.models['Model-1'].rootAssembly  
s1 = a.instances['Part-1-1'].faces  
side1Faces1 = s1.getSequenceFromMask(mask=('[#1 ]'**,** )**,** )  
region2=regionToolset.Region(side1Faces=side1Faces1)  
mdb.models['Model-1'].Coupling(name='Constraint-1'**,** controlPoint=region1**,** surface=region2**,** influenceRadius=WHOLE\_SURFACE**,** couplingType=KINEMATIC**,** localCsys=None**,** u1=ON**,** u2=ON**,** u3=ON**,** ur1=ON**,** ur2=ON**,** ur3=ON)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(loads=ON**,** bcs=ON**,** predefinedFields=ON**,** interactions=OFF**,** constraints=OFF**,** engineeringFeatures=OFF)  
session.viewports['Viewport: 1'].view.setValues(nearPlane=**175.393,** farPlane=**257.853,** width=**98.7787,** height=**44.2234,** cameraPosition=(  
 -**40.4632, 138.113, 145.264**)**,** cameraUpVector=(**0.196319, 0.474249,** -**0.858223**)**,** cameraTarget=(**54.3779,** -**2.61895, 3.24109**))  
a = mdb.models['Model-1'].rootAssembly  
f1 = a.instances['Part-1-1'].faces  
faces1 = f1.getSequenceFromMask(mask=('[#4 ]'**,** )**,** )  
region = regionToolset.Region(faces=faces1)  
mdb.models['Model-1'].EncastreBC(name='BC-1'**,** createStepName='Step-1'**,** region=region**,** localCsys=None)  
session.viewports['Viewport: 1'].view.setValues(nearPlane=**164.657,** farPlane=**273.649,** width=**92.7323,** height=**41.5164,** cameraPosition=(  
 **211.314, 98.4504, 115.969**)**,** cameraUpVector=(-**0.539243, 0.685225,** -**0.489575**)**,** cameraTarget=(**48.9493,** -**1.76378, 3.8727**))  
a = mdb.models['Model-1'].rootAssembly  
r1 = a.referencePoints  
refPoints1=(r1[**5**]**,** )  
region = regionToolset.Region(referencePoints=refPoints1)  
mdb.models['Model-1'].ConcentratedForce(name='Load-1'**,** createStepName='Step-1'**,** region=region**,** cf2=-**100.0,** distributionType=UNIFORM**,** field=''**,** localCsys=None)  
session.viewports['Viewport: 1'].assemblyDisplay.setValues(loads=OFF**,** bcs=OFF**,** predefinedFields=OFF**,** connectors=OFF)  
mdb.Job(name='shiti1'**,** model='Model-1'**,** description=''**,** type=ANALYSIS**,** atTime=None**,** waitMinutes=**0,** waitHours=**0,** queue=None**,** memory=**90,** memoryUnits=PERCENTAGE**,** getMemoryFromAnalysis=True**,** explicitPrecision=SINGLE**,** nodalOutputPrecision=SINGLE**,** echoPrint=OFF**,** modelPrint=OFF**,** contactPrint=OFF**,** historyPrint=OFF**,** userSubroutine=''**,** scratch=''**,** resultsFormat=ODB**,** multiprocessingMode=DEFAULT**,** numCpus=**1,** numGPUs=**0**)  
mdb.jobs['shiti1'].submit(consistencyChecking=OFF)  
session.mdbData.summary()  
o3 = session.openOdb(name='D:/abaqus/exerciese/fem\_final\_work/chen/shiti1.odb')  
session.viewports['Viewport: 1'].setValues(displayedObject=o3)  
session.viewports['Viewport: 1'].makeCurrent()  
session.viewports['Viewport: 1'].odbDisplay.display.setValues(plotState=(  
 CONTOURS\_ON\_DEF**,** ))  
session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(  
 variableLabel='U'**,** outputPosition=NODAL**,** refinement=(INVARIANT**,** 'Magnitude')**,** )

# 五.致谢

研一的第一个学期已然结束，感谢王琥老师在有限元方法课程中对我的悉心教导，让我不仅对abaqus等软件的使用更加熟悉，也加深了我对有限元方法的了解。万分感谢！