

有限元方法及应用

|  |  |
| --- | --- |
| **题 目：** | **有限元大作业** |
| **班 级：** | **2315班** |
| **专 业：** | **机械工程** |
| **姓 名：** | **樊夫** |
| **学 号：** | **B2302S0112** |
| **指导教师：** | **王 琥** |
| **时 间：** | **2024.2.23** |

**空间8节点六面体单元分析**

**问题描述：**如图1所示的一个空间块体，在右端部受两个集中力F作用，其中的参数为：E=1×1010 Pa,μ=0.25,t=0.2 m, F=1×105 N。基于Matlab，用一个空间8节点六面体单元计算各个节点位移、支座反力以及单元的应力。

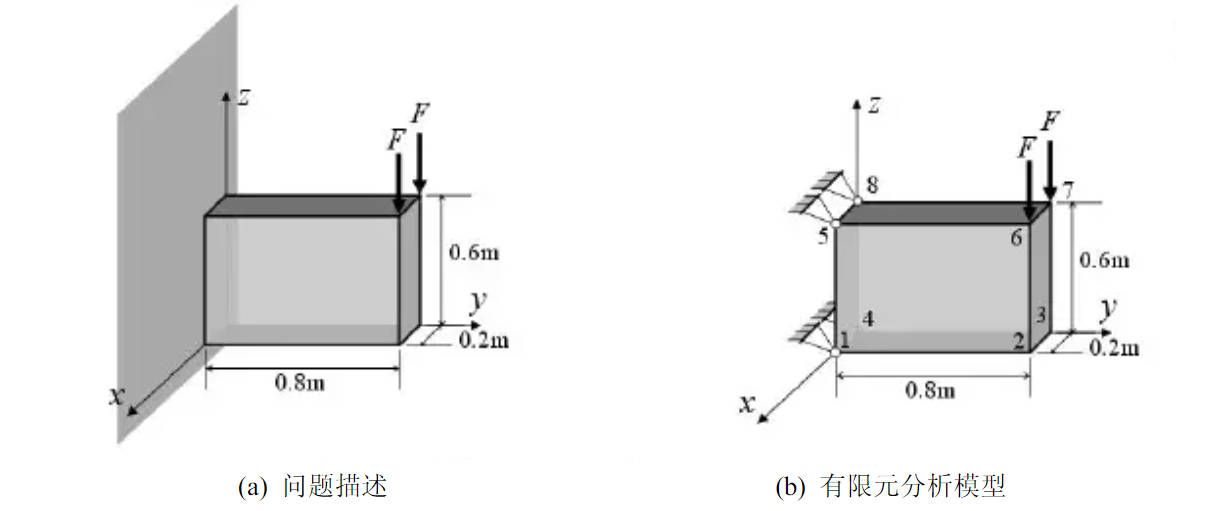


图1 右端部受集中力作用的空间块体

**理论分析：**

（1）结构的离散化与编号

将结构离散为一个8节点六面体单元，节点编号如图1(b)所示，节点的几何坐标见表1。

表1 节点的坐标

|  |  |  |  |
| --- | --- | --- | --- |
| 节点 | x | y | z |
| 1 | 0.2 | 0 | 0 |
| 2 | 0.2 | 0.8 | 0 |
| 3 | 0 | 0.8 | 0 |
| 4 | 0 | 0 | 0 |
| 5 | 0.2 | 0 | 0.6 |
| 6 | 0.2 | 0.8 | 0.6 |
| 7 | 0 | 0.8 | 0.6 |
| 8 | 0 | 0 | 0.6 |

节点位移列阵

总的节点载荷列阵

（2）计算单元的刚度矩阵（以国际标准单位）

首先在MATLAB环境下，输入弹性模量E、泊松比NU，然后针对题中单元节点坐标，调用函数Hexahedral3D8Node\_Stiffness，就可以得到单元的刚度矩阵k1 (24×24)。

（3）建立整体刚度方程

由于该结构共有8个节点，则总共的自由度数为24，因此，结构总的刚度矩阵为KK(24×24)，先对KK清零，然后调用函数Hexahedral3D8Node\_Assembly进行刚度矩阵的组装。由于本题中只用了一个单元，因此总体刚度矩阵KK与单元刚度k1相同。

（4）边界条件的处理及刚度方程求解

由图1(b)可以看出，节点1、节点4、节点5和节点8的三个方向的位移将为零，即。因此，将针对节点2、节点3、节点6和节点7的位移进行求解，这4个节点的位移将对应KK矩阵中的4至9行、4至9列，4至9行、16至21列，16至21行、4至9列，以及16至21行、16至21列，则需从KK(24×24)中提出，置给k，然后生成对应的载荷列阵p，再采用高斯消去法进行求解，即MATLAB中的反斜线符号“\”求解。

（5）支反力的计算

由方程可知，在得到整个结构的节点位移后，由原整体刚度方程就可以计算出对应的支反力；先将上面得到的位移结果与位移边界条件的节点位移进行组合(注意位置关系)，可以得到整体的位移列阵U(24×1)，再代回原整体刚度方程，计算出所有的节点力

P(24×1)，按式的对应关系就可以找到对应的支反力。

（6）各单元的应力计算

先从整体位移列阵U(24×1)中提取出单元的位移列阵，然后，调用计算单元应力的函数

Hexahedral3D8Node\_Stress，就可以得到各个单元的应力分量。

**代码：**

function k = Hexahedral3D8Node\_Stiffness(E,NU,x1,y1,z1,x2,y2,z2,x3,y3,z3,x4,y4,z4,x5,y5,z5,x6,y6,z6,x7,y7,z7,x8,y8,z8)

syms s t n;

N1=(1+s)\*(1-t)\*(1-n)/8;

N2=(1+s)\*(1+t)\*(1-n)/8;

N3=(1-s)\*(1+t)\*(1-n)/8;

N4=(1-s)\*(1-t)\*(1-n)/8;

N5=(1+s)\*(1-t)\*(1+n)/8;

N6=(1+s)\*(1+t)\*(1+n)/8;

N7=(1-s)\*(1+t)\*(1+n)/8;

N8=(1-s)\*(1-t)\*(1+n)/8;

x=N1\*x1+N2\*x2+N3\*x3+N4\*x4+N5\*x5+N6\*x6+N7\*x7+N8\*x8;

y=N1\*y1+N2\*y2+N3\*y3+N4\*y4+N5\*y5+N6\*y6+N7\*y7+N8\*y8;

z=N1\*z1+N2\*z2+N3\*z3+N4\*z4+N5\*z5+N6\*z6+N7\*z7+N8\*z8;

J=[diff(x,s),diff(y,s),diff(z,s);diff(x,t),diff(y,t),diff(z,t);diff(x,n),diff(y,n),diff(z,n)];

Jdet=det(J);

J

Jdet

a=diff(y,t)\*diff(z,n)-diff(z,t)\*diff(y,n);

b=diff(y,s)\*diff(z,n)-diff(z,s)\*diff(y,n);

c=diff(y,s)\*diff(z,t)-diff(z,s)\*diff(y,t);

d=diff(x,t)\*diff(z,n)-diff(z,t)\*diff(x,n);

e=diff(x,s)\*diff(z,n)-diff(z,s)\*diff(x,n);

f=diff(x,s)\*diff(z,t)-diff(z,s)\*diff(x,t);

g=diff(x,t)\*diff(y,n)-diff(y,t)\*diff(x,n);

h=diff(x,s)\*diff(y,n)-diff(y,s)\*diff(x,n);

l=diff(x,s)\*diff(y,t)-diff(y,s)\*diff(x,t);

Ns=[N1,N2,N3,N4,N5,N6,N7,N8];

Bs=sym(zeros(6,3,8));

for i=1:8

Bs(:,:,i)=[a\*diff(Ns(i),s)-b\*diff(Ns(i),t)+c\*diff(Ns(i),n),0,0;0,-d\*diff(Ns(i),s)+e\*diff(Ns(i),t)-f\*diff(Ns(i),n),0;

0,0,g\*diff(Ns(i),s)-h\*diff(Ns(i),t)+l\*diff(Ns(i),n);

-d\*diff(Ns(i),s)+e\*diff(Ns(i),t)-f\*diff(Ns(i),n),a\*diff(Ns(i),s)-b\*diff(Ns(i),t)+c\*diff(Ns(i),n),0;

0,g\*diff(Ns(i),s)-h\*diff(Ns(i),t)+l\*diff(Ns(i),n),-d\*diff(Ns(i),s)+e\*diff(Ns(i),t)-f\*diff(Ns(i),n);

g\*diff(Ns(i),s)-h\*diff(Ns(i),t)+l\*diff(Ns(i),n),0,a\*diff(Ns(i),s)-b\*diff(Ns(i),t)+c\*diff(Ns(i),n)]/Jdet;

end

B=[Bs(:,:,1),Bs(:,:,2),Bs(:,:,3),Bs(:,:,4),Bs(:,:,5),Bs(:,:,6),Bs(:,:,7),Bs(:,:,8)];

B

D=(E/((1+NU)\*(1-2\*NU)))\*[1-NU,NU,NU,0,0,0;NU,1-NU,NU,0,0,0;NU,NU,1-NU,0,0,0;0,0,0,0.5-NU,0,0;0,0,0,0,0.5-NU,0;0,0,0,0,0,0.5-NU];

D

BD=Jdet\*transpose(B)\*D\*B;

z=(int(int(int(BD,n,-1,1),t,-1,1),s,-1,1));

z

k=double(z);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function z = Hexahedral3D8Node\_Assembly(KK,k,i,j,l,m,n,o,p,q)

DOF=[3\*i-2:3\*i,3\*j-2:3\*j,3\*l-2:3\*l,3\*m-2:3\*m,3\*n-2:3\*n,3\*o-2:3\*o,3\*p-2:3\*p,3\*q-2:3\*q];

for n1=1:24

for n2=1:24

KK(DOF(n1),DOF(n2))=KK(DOF(n1),DOF(n2))+k(n1,n2);

end

end

z=KK;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function stress=Hexahedral3D8Node\_Stress(E,NU,x1,y1,z1,x2,y2,z2,x3,y3,z3,x4,y4,z4,x5,y5,z5,x6,y6,z6,x7,y7,z7,x8,y8,z8,u)

syms s t n;

N1=(1+s)\*(1-t)\*(1-n)/8;

N2=(1+s)\*(1+t)\*(1-n)/8;

N3=(1-s)\*(1+t)\*(1-n)/8;

N4=(1-s)\*(1-t)\*(1-n)/8;

N5=(1+s)\*(1-t)\*(1+n)/8;

N6=(1+s)\*(1+t)\*(1+n)/8;

N7=(1-s)\*(1+t)\*(1+n)/8;

N8=(1-s)\*(1-t)\*(1+n)/8;

x=N1\*x1+N2\*x2+N3\*x3+N4\*x4+N5\*x5+N6\*x6+N7\*x7+N8\*x8;

y=N1\*y1+N2\*y2+N3\*y3+N4\*y4+N5\*y5+N6\*y6+N7\*y7+N8\*y8;

z=N1\*z1+N2\*z2+N3\*z3+N4\*z4+N5\*z5+N6\*z6+N7\*z7+N8\*z8;

J=[diff(x,s),diff(y,s),diff(z,s);diff(x,t),diff(y,t),diff(z,t);diff(x,n),diff(y,n),diff(z,n)];

Jdet=det(J);

a=diff(y,t)\*diff(z,n)-diff(z,t)\*diff(y,n);

b=diff(y,s)\*diff(z,n)-diff(z,s)\*diff(y,n);

c=diff(y,s)\*diff(z,t)-diff(z,s)\*diff(y,t);

d=diff(x,t)\*diff(z,n)-diff(z,t)\*diff(x,n);

e=diff(x,s)\*diff(z,n)-diff(z,s)\*diff(x,n);

f=diff(x,s)\*diff(z,t)-diff(z,s)\*diff(x,t);

g=diff(x,t)\*diff(y,n)-diff(y,t)\*diff(x,n);

h=diff(x,s)\*diff(y,n)-diff(y,s)\*diff(x,n);

l=diff(x,s)\*diff(y,t)-diff(y,s)\*diff(x,t);

Ns=[N1,N2,N3,N4,N5,N6,N7,N8];

Bs=sym(zeros(6,3,8));

for i=1:8

Bs(:,:,i)=[a\*diff(Ns(i),s)-b\*diff(Ns(i),t)+c\*diff(Ns(i),n),0,0;0,-d\*diff(Ns(i),s)+e\*diff(Ns(i),t)-f\*diff(Ns(i),n),0;

0,0,g\*diff(Ns(i),s)-h\*diff(Ns(i),t)+l\*diff(Ns(i),n);

-d\*diff(Ns(i),s)+e\*diff(Ns(i),t)-f\*diff(Ns(i),n),a\*diff(Ns(i),s)-b\*diff(Ns(i),t)+c\*diff(Ns(i),n),0;

0,g\*diff(Ns(i),s)-h\*diff(Ns(i),t)+l\*diff(Ns(i),n),-d\*diff(Ns(i),s)+e\*diff(Ns(i),t)-f\*diff(Ns(i),n);

g\*diff(Ns(i),s)-h\*diff(Ns(i),t)+l\*diff(Ns(i),n),0,a\*diff(Ns(i),s)-b\*diff(Ns(i),t)+c\*diff(Ns(i),n)]/Jdet;

end

B=[Bs(:,:,1),Bs(:,:,2),Bs(:,:,3),Bs(:,:,4),Bs(:,:,5),Bs(:,:,6),Bs(:,:,7),Bs(:,:,8)];

D=(E/((1+NU)\*(1-2\*NU)))\*[1-NU,NU,NU,0,0,0;NU,1-NU,NU,0,0,0;NU,NU,1-NU,0,0,0;0,0,0,0.5-NU,0,0;0,0,0,0,0.5-NU,0;0,0,0,0,0,0.5-NU];

w=D\*B\*u;

wcent=subs(w,{s,t,n},{0,0,0})

stress=double(wcent)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%run\_code

clear;clc;

E=1.0e10;NU=0.25;lx=0.2;ly=0.8;lz=0.6;

k1=Hexahedral3D8Node\_Stiffness(E,NU,lx,0,0,lx,ly,0,0,ly,0,0,0,0,lx,0,lz,lx,ly,lz,0,ly,lz,0,0,lz);

KK=zeros(24,24);

KK=Hexahedral3D8Node\_Assembly(KK,k1,1,2,3,4,5,6,7,8);

k=[KK(4:9,4:9),KK(4:9,16:21);KK(16:21,4:9),KK(16:21,16:21)];

p=[0;0;0;0;0;0;0;0;-1e5;0;0;-1e5];

u=k\p

U=[0;0;0;u(1:6);0;0;0;0;0;0;u(7:12);0;0;0];

P=KK\*U

u1=U(1:24);

stress1=Hexahedral3D8Node\_Stress(E,NU,lx,0,0,lx,ly,0,0,ly,0,0,0,0,lx,0,lz,lx,ly,lz,0,ly,lz,0,0,lz,u1)

**代码输出结果：**

各个节点位移（单位为m）：

各个节点支座反力（单位为N）：

单元1的中心点的应力分量为（单位为Pa）：