MatLab编程思路

**绘图阶段**

在绘图阶段先考虑第一个要求：用参数n来控制网格密度。

我的思路是：1阶正方形对应4个节点2个单元、2阶正方形对应9个节点8个单元、3阶正方形对应16个节点18个单元……易得到规律：节点数=(正方形阶数+1)2；单元数=2(正方形阶数)2。

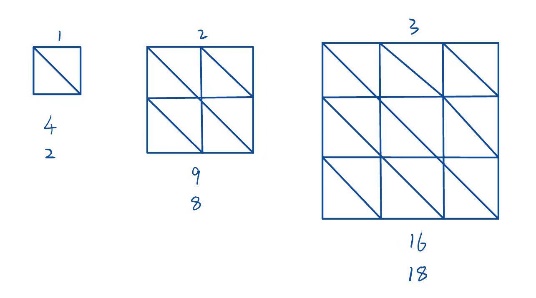


图1 三角形阶数与节点数单元数的关系

对节点的编号可以按照先循环行后循环列的方式一个一个编过去，由于需要根据网格密度的变化而变化，所以使用相关的变量进行运算。

单元的编号则需要区分上三角形和下三角形分开进行写入（第一次这里上三角形是逆时针编号，下三角形成了顺时针编号导致后面运算出了问题）把节点的坐标和单元的序号信息同时写入txt文件，后面绘图的时候直接读取文件里面的信息就会方便不少（写入txt文件还有一个好处就是可以快捷的查询错误并改正，不会影响后面读取文件代码的正确性）。

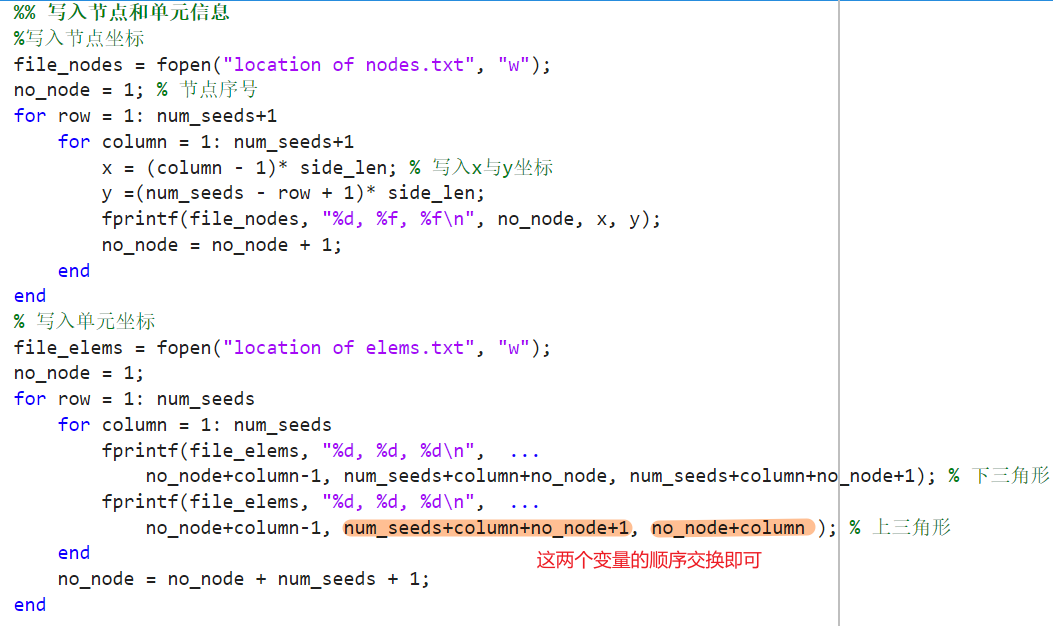


图2 写入节点、单元信息

通过txt文件中的节点坐标信息进行绘图，通过计算三角形单元的形心坐标对单元进行编号。

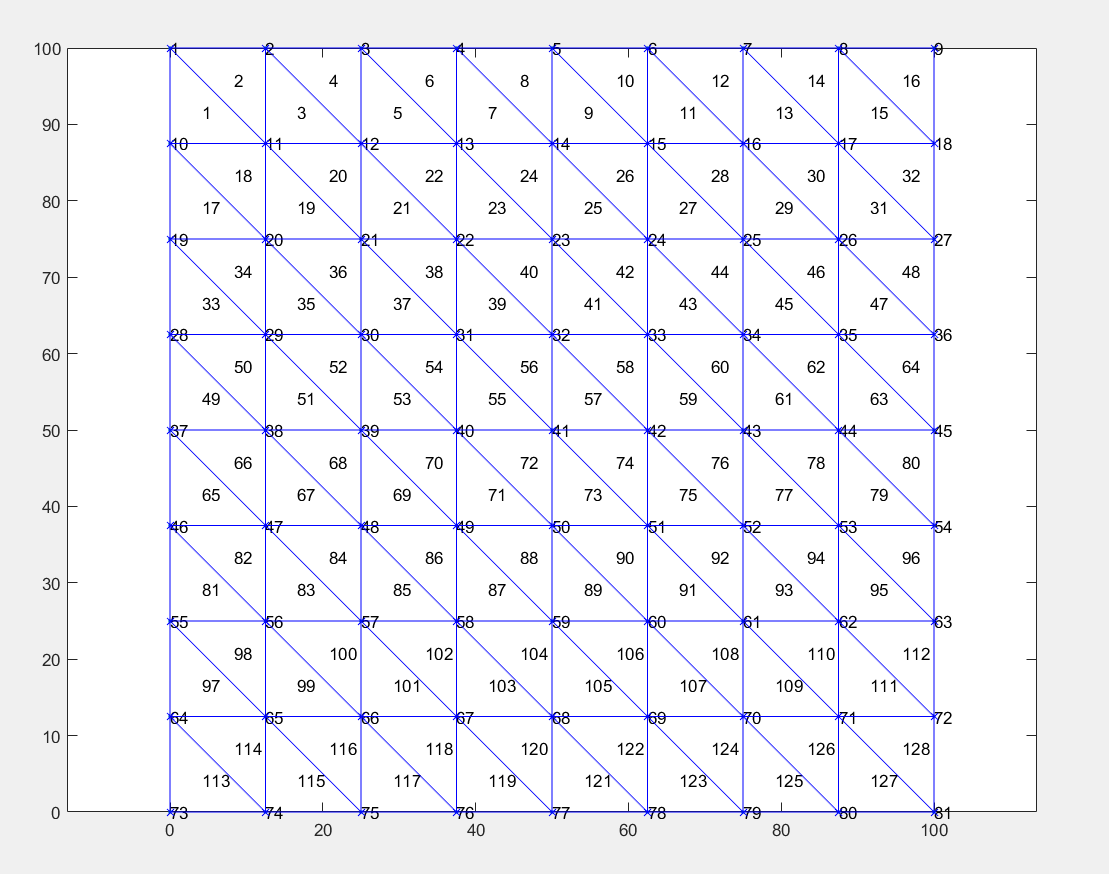


图3 n=8时网格图

**计算阶段**

在计算之前先定义弹性模量、泊松比、集中荷载、刚度矩阵和荷载矩阵。

在这里可以定义一个类，使得刚度矩阵和荷载矩阵这种较大的数据后面计算时在函数间调用不需要再复制一份占用内存，而是直接调用。

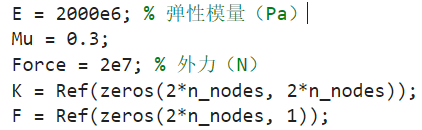


图4 定义各项

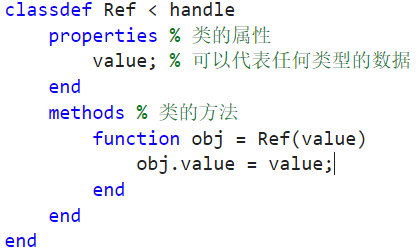


图5 类的定义

读取边界条件信息：这里采用的思路和前面输入节点和单元信息类似，直接在txt文件内编辑边界条件信息，避免修改代码时导致错误。

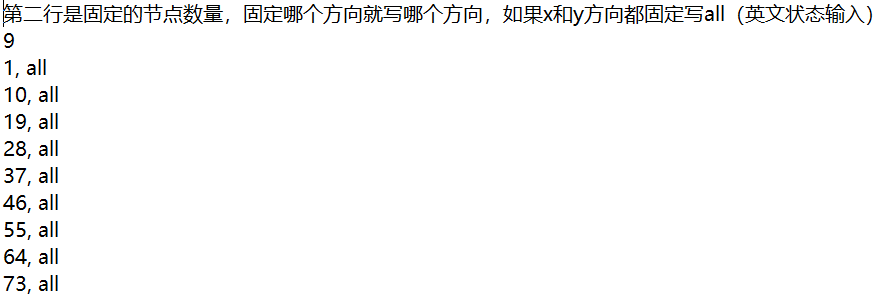


图6 txt文件内的边界条件信息

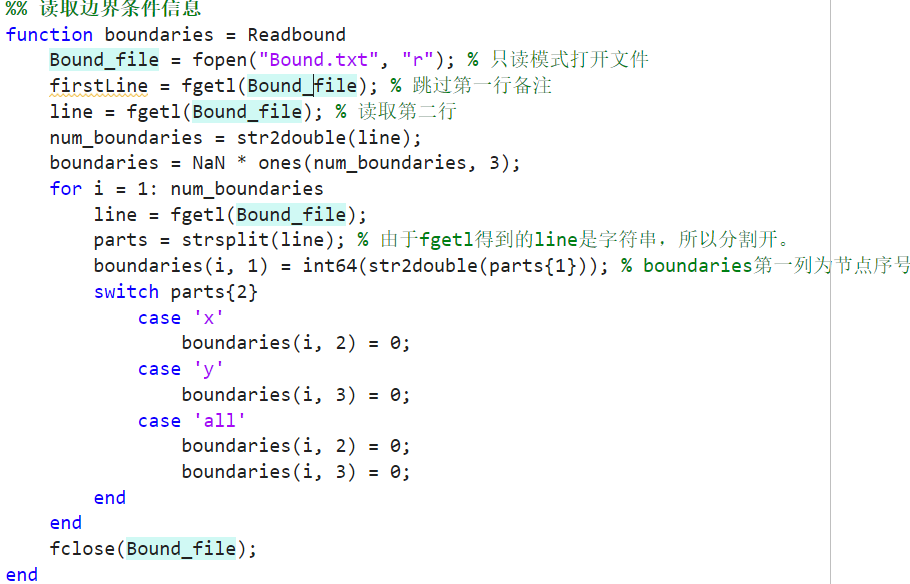


图7 读取边界条件信息

计算局部刚度矩阵并组装：使用for循环一个一个提取组成单元的节点号，再根据节点号索引到单元的x坐标和y坐标并赋值给变量用来计算局部刚度矩阵，再将这几个节点的刚度组装到整体刚度矩阵的对应位置（用原本的节点号在整体刚度矩阵中索引，用1、2、3在局部刚度矩阵中索引）。

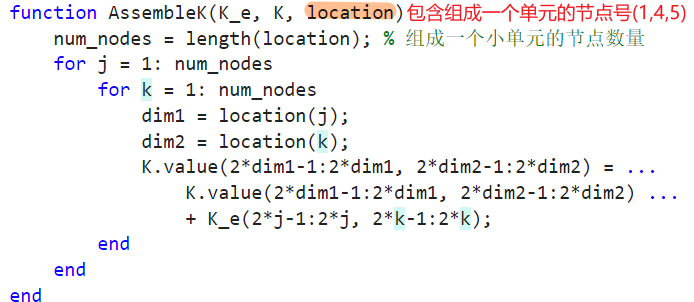
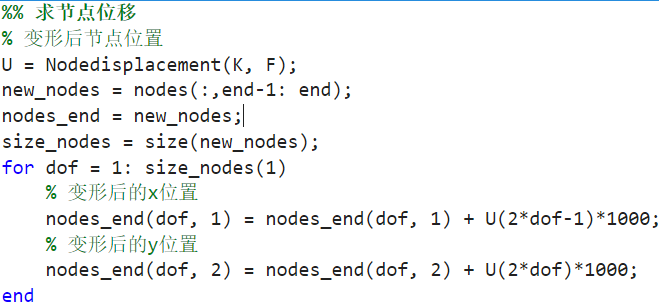


图8 装配至整体刚度矩阵

边界条件的设置就采用书上的“置1法”，对应位置的行列都为0，对角线元素为1。然后在F矩阵的对应位置施加对应大小的力，即可对节点的位移进行计算，计算成功后将节点的位移后坐标储存在nodes\_end内。



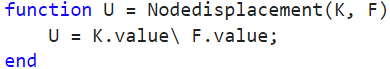


图9 计算节点位移后位置

您今天上课讲的：把需要设置边界条件的那几个节点在K矩阵和F矩阵中的对应行列删掉这种方法。由于当时忽略了单元节点的输入需要逆时针方向，导致运算时经常出现矩阵奇异的问题，我以为是矩阵中0太多了导致它不可逆，便歪打正着的也想到了这个方法，但运行后发现问题还是没有解决。顺便也把代码附在下面。

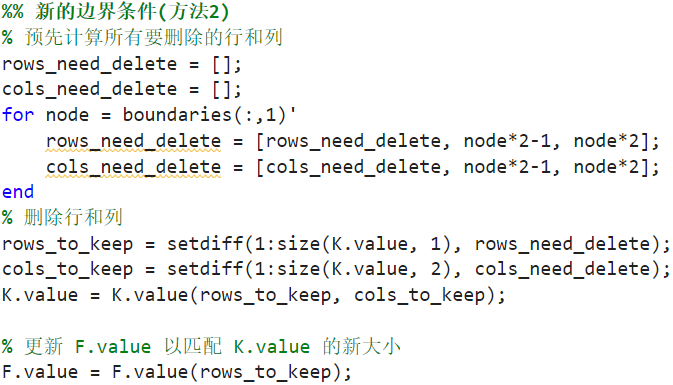


图10 删去对应行列

**位移计算完成后阶段**

获得位移后坐标即可绘制出变形后的图形，前面在画完网格图后可以使用hold on命令保持图像。这里可以不使用Figure命令，把位移后的图像画在一张图上方便对比。

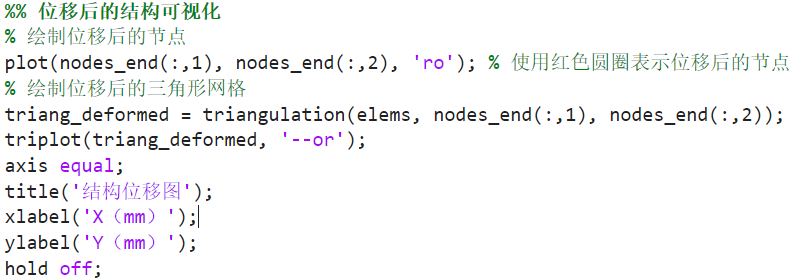


图11 位移后结构图的绘制

计算应力应变：由于应力和应变的计算过程基本相同，只有D矩阵不一样，所以将D换成Need\_D这个参数，根据不同的计算过程输入不同Need\_D进行计算。

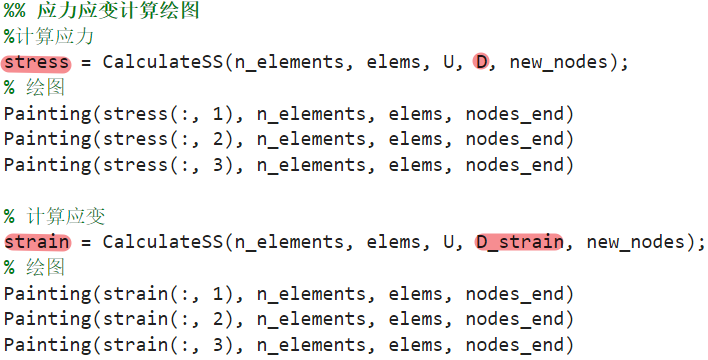


图12 计算并绘制应力应变

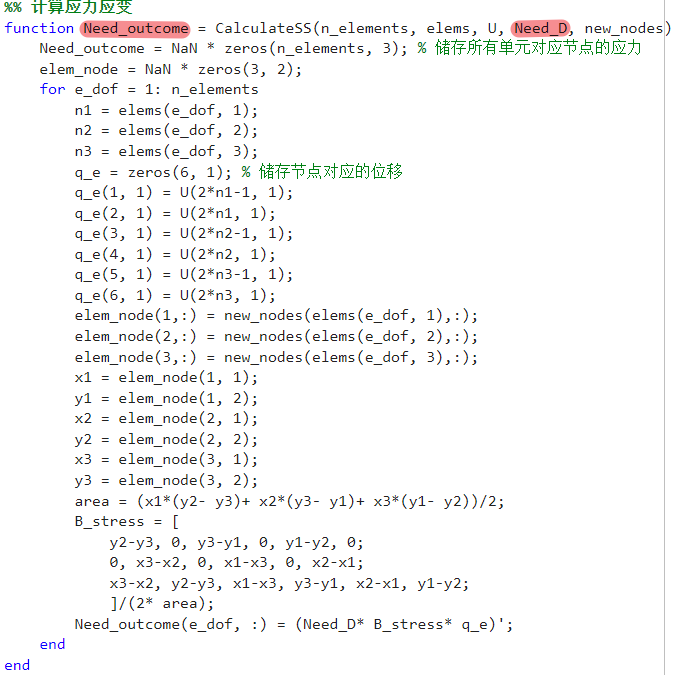
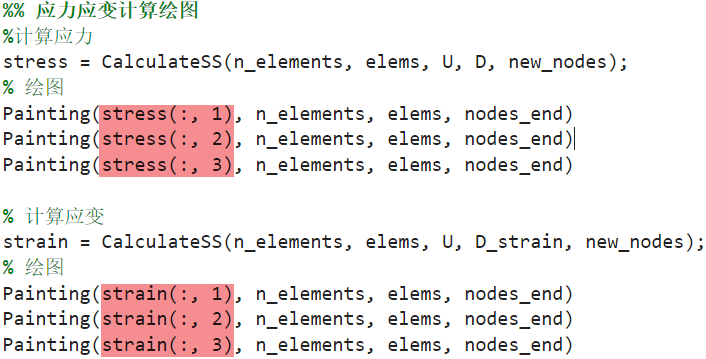


图13 应力应变计算过程

应力应变图的绘制也是相似的过程，因此只需要把传入的数据当作一个参数暴露出来，绘制不同的图只需修改这个参数即可。



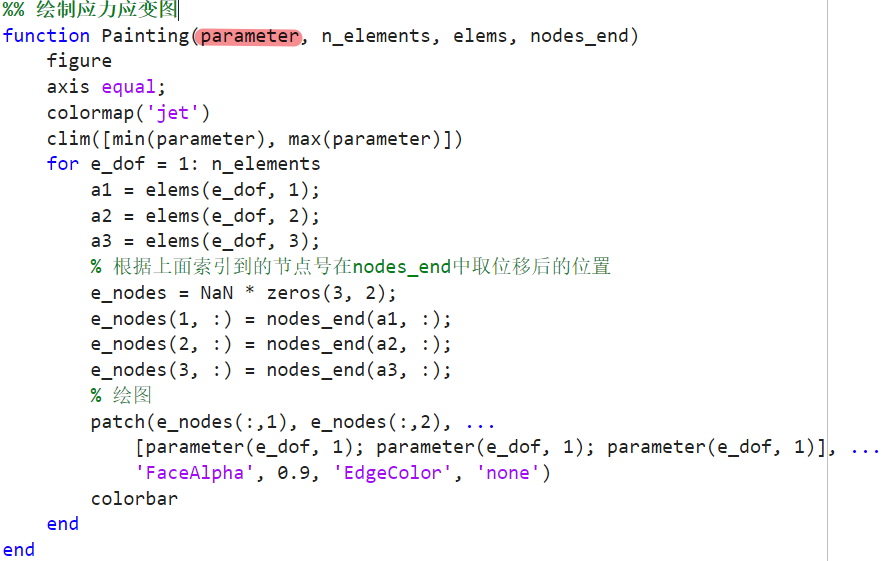


图14 应力应变绘制过程

**最终运行结果展示**

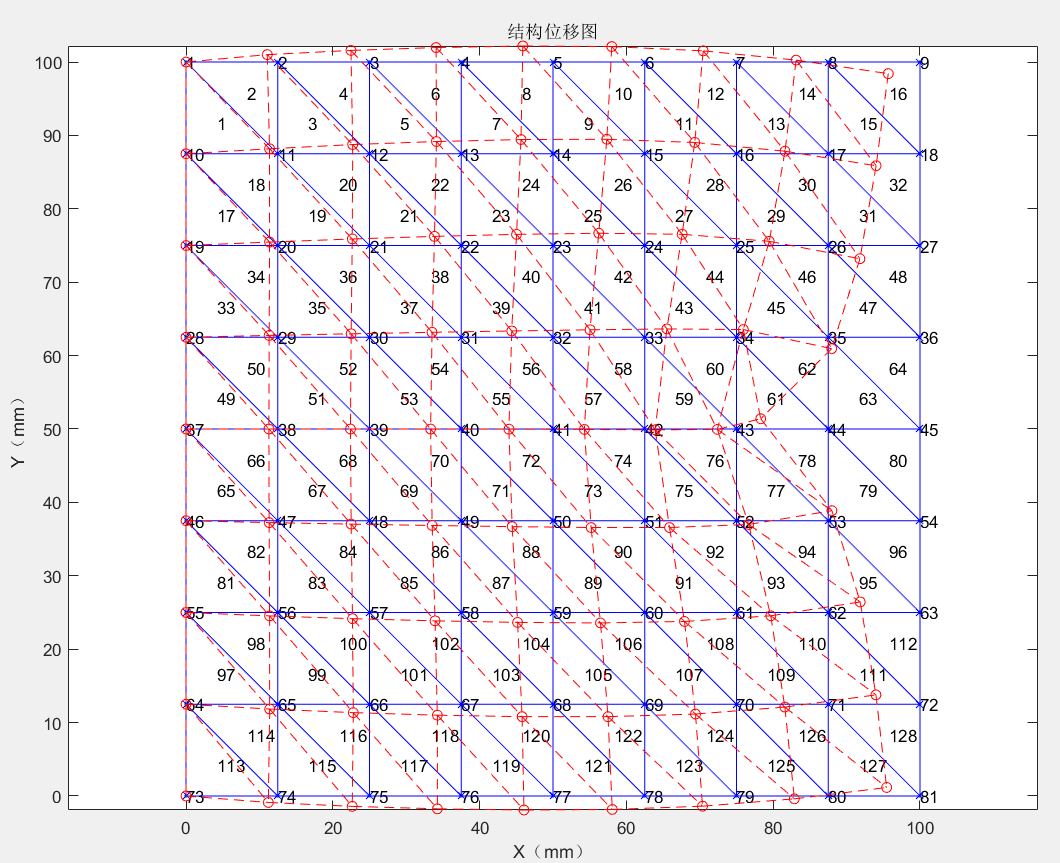


图15 结构位移图

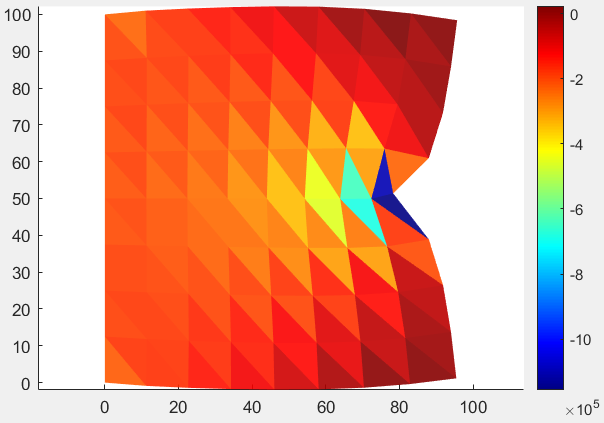


图16 x方向应力

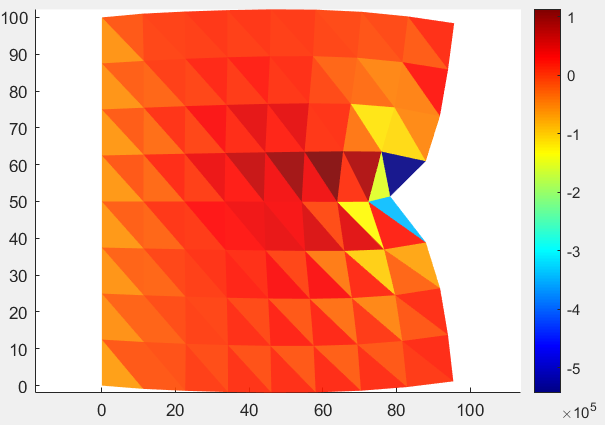


图17 y方向应力

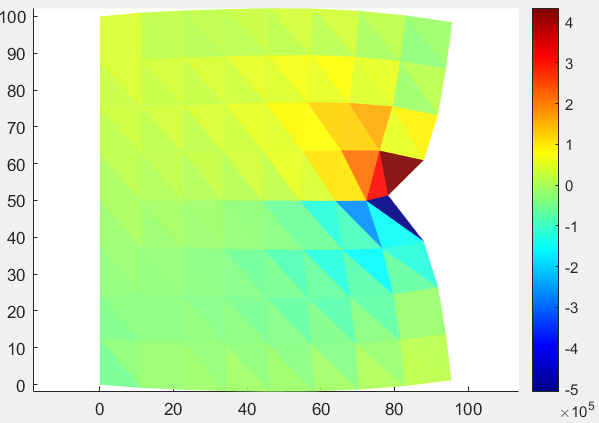


图18 切向应力

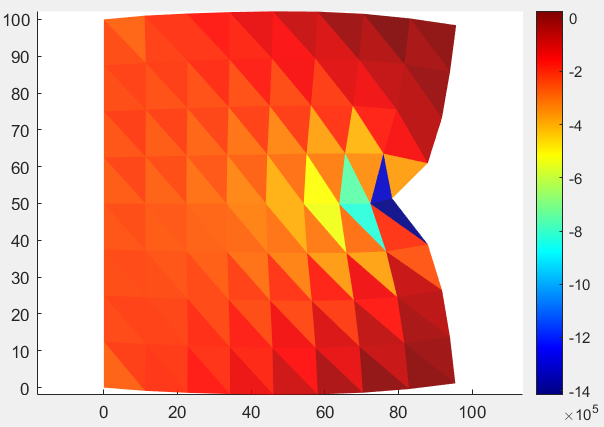


图19 x方向应变

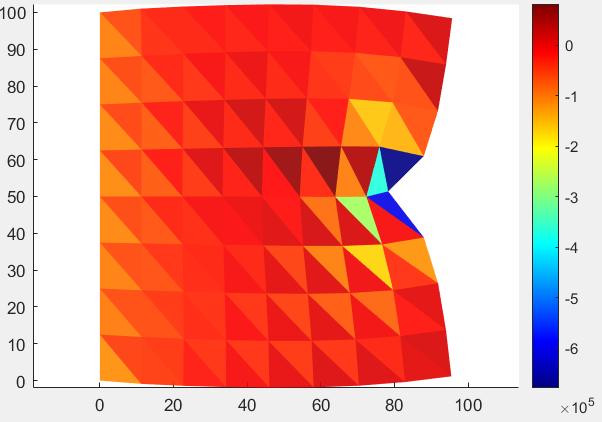


图20 y方向应变

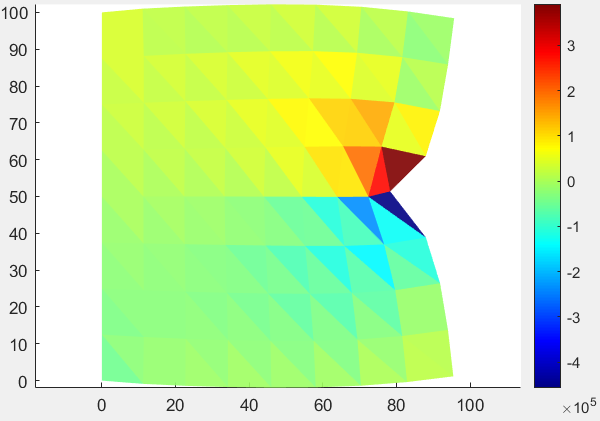


图21 切向应变