8H 单元报告

杨昊光　学号：2014011619

December 3, 2016

1. 算法分析

- 位移计算

给出一个 8H 单元，其节点坐标矩阵为 $\boldsymbol{X}=\begin{bmatrix}x_1 & y_1 & z_1\\ x_2 & y_2 & z_2\\ \vdots & \vdots & \vdots\\ x_8 & y_8 & z_8\end{bmatrix}$，每条边的方向上采用 2 点高斯积分，共 8 个高斯点，并采用三线性函数作为形函数。作前处理时，将坐标变换到 ξ, η, ζ 下求形函数的梯度，再对 8 个高斯点作加权求和，得到单元刚度阵：

$$N_i=\frac{1}{8}(1+\xi_i\xi)(1+\eta_i\eta)(1+\zeta_i\zeta),\ i=1,2,...,8$$

$$\boldsymbol{N}=\begin{bmatrix}\boldsymbol{N}_1 & \boldsymbol{N}_2 & \boldsymbol{N}_3 & \boldsymbol{N}_4 & \boldsymbol{N}_5 & \boldsymbol{N}_6 & \boldsymbol{N}_7 & \boldsymbol{N}_8\end{bmatrix},\ \boldsymbol{N}_i=N_i\begin{bmatrix}1 & & \\ & 1 & \\ & & 1\end{bmatrix}$$

$$GN=\begin{bmatrix}\frac{\partial N_1}{\partial\xi} & \frac{\partial N_2}{\partial\xi} & \cdots & \frac{\partial N_8}{\partial\xi}\\ \frac{\partial N_1}{\partial\eta} & \frac{\partial N_2}{\partial\eta} & \cdots & \frac{\partial N_8}{\partial\eta}\\ \frac{\partial N_1}{\partial\zeta} & \frac{\partial N_2}{\partial\zeta} & \cdots & \frac{\partial N_8}{\partial\zeta}\end{bmatrix}$$

$$\boldsymbol{B}=\nabla_s\boldsymbol{N}=\begin{bmatrix}\boldsymbol{B}_1 & \boldsymbol{B}_2 & \boldsymbol{B}_3 & \boldsymbol{B}_4 & \boldsymbol{B}_5 & \boldsymbol{B}_6 & \boldsymbol{B}_7 & \boldsymbol{B}_8\end{bmatrix},\ \boldsymbol{B}_i=\begin{bmatrix}B_{ix} & 0 & 0\\ 0 & B_{iy} & 0\\ 0 & 0 & B_{iz}\\ B_{iy} & B_{ix} & 0\\ 0 & B_{iz} & B_{iy}\\ B_{iz} & 0 & B_{ix}\end{bmatrix},\ i=1,2,...,8$$

$\boldsymbol{B}$ 矩阵的各量由单元 Jacobian 矩阵和形函数对各坐标的导数得到：

$$\boldsymbol{J}^{-1}\cdot GN=\begin{bmatrix}B_{1x} & B_{2x} & \cdots & B_{8x}\\ B_{1y} & B_{2y} & \cdots & B_{8y}\\ B_{1z} & B_{2z} & \cdots & B_{8z}\end{bmatrix},\ \boldsymbol{J}=GN\cdot\boldsymbol{X}^T$$

$\boldsymbol{D}$ 矩阵由本构关系得到，在此认为材料是各向同性的：

$$\sigma_{ij}=2G\varepsilon_{ij}+\lambda\varepsilon_{kk}\delta_{ij}$$

$$\Rightarrow\boldsymbol{D}=\begin{bmatrix}2G+\lambda & \lambda & \lambda & & & \\ \lambda & 2G+\lambda & \lambda & & & \\ \lambda & \lambda & 2G+\lambda & & & \\ & & & 2G & & \\ & & & & 2G & \\ & & & & & 2G\end{bmatrix},\ \text{s.t. }\boldsymbol{\sigma}=\boldsymbol{D}\boldsymbol{\varepsilon},\ \text{where }\boldsymbol{\varepsilon}=\begin{bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \varepsilon_{xy}\\ \varepsilon_{yz}\\ \varepsilon_{zx}\end{bmatrix},\ \boldsymbol{\sigma}=\begin{bmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{bmatrix}.$$

$$\boldsymbol{K}^e=\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}w_iw_jw_k\boldsymbol{B}^T(\xi_i,\eta_j,\zeta_k)\boldsymbol{D}\boldsymbol{B}(\xi_i,\eta_j,\zeta_k)$$

生成单元刚度阵通过 Location Matrix 进行组装，得到全局的刚度阵，再解 $\boldsymbol{K}\boldsymbol{d}=\boldsymbol{f}$.

- 应力恢复

  采用 SPR 方法进行节点应力恢复。对于一个内部节点，若与 8 个 8H 单元相连，则有 $n = 8 \times 8 = 64$ 个高斯点用于应力恢复；对于一个边界面上的节点，则与 4 个单元相连，有 32 个高斯点用于应力恢复；同理，对于边界棱上的节点有 16 个，边界顶点上的节点有 8 个高斯点。为防止过拟合，对于边界面上的节点和内部节点采用完备二阶多项式进行最小二乘逼近，对于边界棱和边界顶点上的节点采用完备一阶多项式进行最小二乘逼近。

  – 对二阶逼近：

$$\boldsymbol{A} = \begin{bmatrix} 1 & x_1 & y_1 & z_1 & x_1y_1 & y_1z_1 & z_1x_1 & x_1^2 & y_1^2 & z_1^2 \\ 1 & x_2 & y_2 & z_2 & x_2y_2 & y_2z_2 & z_2x_2 & x_2^2 & y_2^2 & z_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & y_n & z_n & x_ny_n & y_nz_n & z_nx_n & x_n^2 & y_n^2 & z_n^2 \end{bmatrix}, \boldsymbol{X} = \begin{bmatrix} 1 & x & y & z & xy & yz & zx & x^2 & y^2 & z^2 \end{bmatrix}$$

$$\boldsymbol{C}_{ij} = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \end{bmatrix}^T, \boldsymbol{S}_{ij} = \begin{bmatrix} \sigma_{ij}^{(1)} & \sigma_{ij}^{(2)} & \cdots & \sigma_{ij}^{(n)} \end{bmatrix}^T$$

  最小二乘逼近为解系数矩阵 $\boldsymbol{C}_{ij}$，使得：$\boldsymbol{A}\boldsymbol{C}_{ij} = \boldsymbol{S}_{ij}$, 即$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{C}_{ij} = \boldsymbol{A}^T\boldsymbol{S}_{ij}$。最后解 $\boldsymbol{X}^{(k)}\boldsymbol{C}_{ij} = S_{ij}^{(k)}$ 为第 k 个节点上的应力。

  – 对一阶逼近：
  同理，但只取 1, x, y, z 的项使用。

2. 算法实现

   位移计算部分按照变换将坐标变换到 $[-1,1] \times [-1,1] \times [-1,1]$ 后即可得到形函数和各导数。取得高斯点位置，代入 $\boldsymbol{D}$ 矩阵即可以产生单元刚度阵，调用 COLHT 和 ADDBAN 生成总刚度矩阵并求解。

   SPR 部分，在前处理时保存单元连接矩阵到临时文件中，在后处理中调取，计算每个节点连接的单元数量和编号，以及用于恢复节点位置的信息，生成为 NodeRelationFlag 数组。对每个节点，根据该数组选取逼近的阶次，计算 $\boldsymbol{A}$、$\boldsymbol{S}$ 矩阵，并传入最小二乘子程序 LeastSquare 中，得到系数数组，并代入节点位置信息，得到恢复的节点应力。

3. 算法测试

   Patch Test 选取了 7 单元模型和 8 单元模型进行测试，测试形状为单位立方体，$E = 1000$, $\nu = 0.25$，通过内部确定 1 个或 8 个点来将其分划为 8 个或 7 个单元。施加线性位移场：

$$u_x = 0.001x,\ u_y = 0.002y,\ u_z = 0.003z$$

   得到各点需要施加的外力，并采用最少的边界条件对单元进行测试。测试结果如 data/8H 中各输入输出文件所示。

   测试结果表明，对于线性场，输出误差在机器 ε 量级，单元设计能精确重构线性位移场，为一阶收敛的。

   对 SPR 的测试，其输出结果误差也在机器 ε 量级，因而从算法设计的角度来说是合理的。