清華大學

计算动力学

课程报告

林子雄、郭嘉鑫、王婉婷

2022 年 6 月

目 录

第1章 精细积分法文献调研

1.1 文献调研

一般的线性定常动力系统，存在以下的运动方程[1]：

$$\begin{aligned} &\boldsymbol{M\ddot{a}+C\dot{a}+Ka=Q} \\ &\boldsymbol{a}(0)=\boldsymbol{a}_0,\dot{\boldsymbol{a}}(0)=\dot{\boldsymbol{a}}_0 \end{aligned} \tag{1-1}$$

对于这个方程，已经发展出了多种数值解法，例如加权余量法，中心差分法，Houbolt 法，Newmark 法，Wilsonθ 法等 51[1]。但是这些数值方法都存在着两个较大的问题：（1）对时间步长存在敏感性；（2）精度较低。

针对以上的问题，钟万勰教授在 1995 年提出了精细积分法[2]，这种方法避免了由于精细划分造成的数值求解截断误差，可以求出计算机层面的精确解。同时该方法是无条件稳定的，对时间步长不敏感，在求解结构长时间动力学响应时有很大优势。精细积分方法提出后,被广泛应用于结构动力响应、最优控制、随机振动、热传导、波传播、复杂动力弹塑性分析、结构优化设计、刚性问题、动态载荷识别、Maxwell 方程求解、偏微分方程求解等众多领域[3]。

精细积分法主要求解两个问题：(1)矩阵指数迭代求解(2)非齐次项 Duhamel 积分的求解。由于载荷可能随着时间发生变化，因此在积分区间[tk,tk+1]中，钟万勰教授提出通过线性函数近似载荷项[2]，虽然可以解析求解 Duhamel 积分，但需要对状态矩阵 $\boldsymbol{H}$ 求逆，存在丧失计算精度以及出现数值不稳定或者逆矩阵不存在的问题。

因此近些年来，对于精细积分法的发展与改进主要包括以下几个方面[3]：

（1） 矩阵指数的精细积分方法；

（2） 直接积分法数值求解 Duhamel 积分；

（3） 提高计算效率的改进算法；

下面逐一对这些改进做介绍：

1.1.1 经典精细积分法

为了便于介绍近些年来精细积分法的发展与改进，首先介绍经典精细积分法

的求解过程。

对于 n 维的线性定常结构动力学离散方程：

$$\begin{aligned} &\boldsymbol{M\ddot{a} + C\dot{a} + Ka = Q} \\ &\boldsymbol{a}(0) = \boldsymbol{a}_0, \dot{\boldsymbol{a}}(0) = \dot{\boldsymbol{a}}_0 \end{aligned} \tag{1-2}$$

$\boldsymbol{M}$、$\boldsymbol{C}$、$\boldsymbol{K}$ 为 $\boldsymbol{n}\times\boldsymbol{n}$ 矩阵，$\boldsymbol{x}$ 待求，$\boldsymbol{f(t)}$为给定外力。引入对偶变量

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{p} \end{bmatrix},\ \boldsymbol{p} = \boldsymbol{M\dot{a}} + \frac{\boldsymbol{Ca}}{\boldsymbol{2}} \tag{1-3}$$

则可以得到 Hamilton 体系下的动力学方程

$$\dot{\boldsymbol{x}} = \boldsymbol{Hx} + \boldsymbol{R} \tag{1-4}$$

式中

$$\begin{aligned} &\boldsymbol{H} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_3 \\ \boldsymbol{A}_2 & \boldsymbol{A}_4 \end{bmatrix}, \boldsymbol{R} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{Q} \end{bmatrix} \\ &\boldsymbol{A}_1 = \boldsymbol{-M^{-1}C\ /\ 2}, \boldsymbol{A}_2 = \boldsymbol{CM^{-1}C\ /\ 4 - k} \\ &\boldsymbol{A}_3 = \boldsymbol{-CM^{-1}\ /\ 2}, \boldsymbol{A}_4 = \boldsymbol{M^{-1}} \end{aligned} \tag{1-5}$$

式(1-4)对应的齐次方程通解为

$$\boldsymbol{v} = \boldsymbol{exp(H \cdot t)x_0} \tag{1-6}$$

令时间步长为 $\boldsymbol{\Delta t}$，则可以递推得到各个时刻的解

$$\boldsymbol{x_1 = Tx_0, x_2 = Tx_1, \cdots, x_{k+1} = Tx_k} \cdots \tag{1-7}$$

其中 $\boldsymbol{T = e^{H\Delta t}}$ 为矩阵指数。

当存在非齐次项时，有积分形式的解答

$$\boldsymbol{x(t) = e^{H(t-t_0)}x_0 + \int_{t_0}^{t} e^{H(t-\xi)}R(\xi)d\xi} \tag{1-8}$$

时域积分是步进的，即在计算 $\boldsymbol{t_{k+1}}$ 时刻的解时，$\boldsymbol{t_k}$ 以及以前各时刻解答均已得到。在(1-8)式中令 $\boldsymbol{t_0 = t_k, t = t_{k+1}}$，得

$$\boldsymbol{x_{k+1} = Tx_k + \int_{t_k}^{t_{k+1}} e^{H(t_{k+1}-\xi)}R(\xi)d\xi} \tag{1-9}$$

对于矩阵指数 $\boldsymbol{T}$ 的计算，钟万勰教授提出了 $\boldsymbol{2^N}$ 算法，其要点有两个：（1）加法定理；（2）将注意力集中在增量上，而不是全量。

$$\boldsymbol{exp(H \times \Delta t) = [exp(H \times \Delta t\ /\ m)]^m} \tag{1-10}$$

当取 $m = 2^N$,N=20 时，$\Delta\tau = \Delta t / m$ 是非常小的一个时间区间，利用泰勒展开可得

$$\begin{aligned} exp(H\Delta\tau) &= I + H\Delta\tau + \frac{(H\Delta\tau)^2}{2!} + \frac{(H\Delta\tau)^3}{3!} + \frac{(H\Delta\tau)^4}{4!} + \cdots \\ &\approx I + T_a \end{aligned} \tag{1-11}$$

其中

$$T_a = H\Delta\tau + \frac{(H\Delta\tau)^2}{2}[I + \frac{H\Delta\tau}{3} + \frac{(H\Delta\tau)^2}{12}] \tag{1-12}$$

由于 T_a 的元素值非常小，当它与矩阵 I 相加时会由于计算机的舍入误差而丧失精度，为避免出现这一问题，将(1-10)式作以下分解

$$T = [T_a + I]^{2^N} = [T_a + I]^{2^{N-1}} \times [T_a + I]^{2^{N-1}} \tag{1-13}$$

还注意到

$$(I + T_a) \times (I + T_a) \equiv I + (2T_a + T_a \times T_a) \tag{1-14}$$

式(1-13)相当于执行语句

For(iter = 0;iter<N;iter++)

$$T_a = 2T_a + T_a \times T_a \tag{1-15}$$

当循环结束时有

$$T = I + T_a \tag{1-16}$$

式(1-12)，式(1-15)，式(1-16)是矩阵指数的精细积分计算公式，是精细积分法的核心与关键。

对于式(1-9)中的 Duhamel 积分项

$$\int_{t_k}^{t_{k+1}} e^{H(t_{k+1}-\xi)} R(\xi) d\xi \tag{1-17}$$

对于任意形式的载荷 $R(t)$,无法对其进行显示积分，钟万勰教授提出在每一离散时间段 $[t_k, t_{k+1}]$ 上用线性函数近似载荷项[2]

$$R(t) = R_0 + R_1(t - t_k) \tag{1-18}$$

将式(1-18)代入递推式(1-9)并积分，可得

$$x_{k+1} = T[x_k + H^{-1}(R_0 + H^{-1}R_1)] - H^{-1}(R_0 + H^{-1}R_1 + R_1\Delta t) \tag{1-19}$$

此式即为时域精细积分的步进公式。

1.1.2 矩阵指数的精细积分法

1.1.2.1 基于 Taylor 级数近似的精细积分方法

在 1.1.1 经典精细积分方法中，已经介绍了基于 Taylor 级数近似的矩阵指数积分方法，在这里需要注意，这种方法的核心在于（1）加法定理，即 2^N 类算法；（2）将注意力放在增量上，而不是全量上。

对于 $exp(H\Delta\tau)$ 的 Taylor 级数展开式(1-11)，其展开项数记为 M。矩阵指数的加法定理：

$$exp(H\eta) \equiv exp(H\eta / m)^m, m = 2^N \tag{1-20}$$

增大 N 和 M 的取值，矩阵指数的计算就更加精确。钟万勰教授指出，在实际课题中 N=20，M=4 是一个非常保守的选择[4]。

1.1.2.2 基于 Pade 级数近似的精细积分公式

对于矩阵指数 $exp(H\tau)$ 还可以选择使用 Pade 级数进行展开近似。Pade 级数逼近比 Taylor 级数逼近具有更好的精度和稳定性，因此采用 Pade 级数展开来近似初始区段的矩阵指数可以达到更好的精度和效率[3]。对于小区段 $\tau = \eta / m$，可采用 M 项 Pade 级数近似矩阵指数，即

$$exp(H\tau) \approx [D_M(H\tau)]^{-1} N_M(H\tau) \tag{1-21}$$

其中

$$\begin{aligned} N_M(H\tau) &= \sum_{j=0}^{M} \frac{(2M-j)!M!}{(2M)!j!(M-1)!}(H\tau)^j, \\ D_M(H\tau) &= \sum_{j=0}^{M} \frac{(2M-j)!M!}{(2M)!j!(M-j)!}(-H\tau)^j \end{aligned} \tag{1-22}$$

同样为了避免舍入误差，在使用加法定理时，也只计算和存储增量部分，为此将 $N_M(H\tau)$ 和 $D_M(H\tau)$ 表示为如下形式

$$\begin{aligned} N_M(H\tau) &= I + \overline{N}(H\tau), \\ D_M(H\tau) &= I + \overline{D}(H\tau), \end{aligned} \tag{1-23}$$

其中

$$\overline{N}_M(H\tau) = \sum_{j=1}^{M} \frac{(2M-j)!M!}{(2M)!j!(M-1)!}(H\tau)^j,$$
$$\overline{D}_M(H\tau) = \sum_{j=1}^{M} \frac{(2M-j)!M!}{(2M)!j!(M-j)!}(-H\tau)^j \tag{1-24}$$

则可得到：

$$exp(H\tau) \approx [D_M(H\tau)]^{-1} N_M(H\tau) = I + R \tag{1-25}$$

其中

$$R = (I + \overline{D})^{-1}(\overline{N} - \overline{D}) \tag{1-26}$$

然后即可对增量部分使用加法定理。

1.1.2.3 Taylor 级数近似与 Pade 级数展开优劣对比

虽然相同展开项数 M 下，Pade 级数近似在小区段上的精度比 Taylor 级数近似精度更高，但是 Pade 级数的计算中需要进行矩阵求逆，计算量远远大于 Taylor 级数展开，而且求逆计算还容易带来数值上的不稳定；同时对于 Taylor 级数展开，可以通过增加展开项数 M 和分段指数 N 来提高精度。综合比较分析，由于相同精度下，Pade 级数近似的计算量远大于 Taylor 级数，并且对于实际工程问题选取 M=4,N=20 已经足够保守，所以在后续的编程计算中我们采用了 M=4，N=20 的 Taylor 级数展开近似来进行矩阵指数的精细积分计算。

1.1.3 直接积分法数值求解 Duhamel 积分

前面提到，由于精细积分法是一种无条件稳定的算法，因此很适合大步长的快速计算，对于大步长情况下，如式(1-18)所示，在离散时间段$[t_k, t_{k+1}]$上用线性函数近似载荷项的精度会非常低。同时求解积分时需要对矩阵进行求逆。矩阵求逆不仅计算量大，而且数值稳定性不好，在实际工程中还可能存在逆矩阵不存在的情况。因此发展出了状态方程直接积分法，有效地避免了非齐次方程求逆的问题。状态方程直接积分法是指对 Duhamel 积分项采用数值积分的方法求解，目前已经提出了使用梯形公式（代数精度为 1）、Simpson 公式（代数精度为 3）、Newton-Cotes 公式（代数精度为 5）以及 Gauss 积分公式等[5]。文献指出 Gauss 求积公式是比较好的选择，因为其计算精度取决于高斯积分点的数量，理论上该算法可以达到任意高的精度[6]。所以在后续的代码编写中考虑使用 Gauss 求积公式求解 Duhamel 积分项。下面仅对 Gauss 求积公式改进的精细积分法进行简要介绍：

对于式(1-9)中的 Duhamel 积分项,应用高斯积分可得：

$$\begin{aligned}\int_{t_k}^{t_{k+1}} e^{H(t_k-\tau)}R(\tau)d\tau &= \frac{\Delta t}{2}\sum_{i=1}^{n}\omega_i e^{\frac{\Delta t}{2}(1-\xi_i)H}R(t_k+\frac{\Delta t}{2}(1+\xi_i)+O(\Delta t^{2n-1}))\\ &= \frac{\Delta t}{2}\sum_{i=1}^{n}\omega_i T(\frac{\Delta t}{2}(1-\xi_i))R(t_k+\frac{\Delta t}{2}(1+\xi_i))+O(\Delta t^{2n-1})\end{aligned} \quad (1\text{-}27)$$

式中，n 为积分点个数，ξ_i 为积分点坐标，ω_i 为加权系数，$T(\frac{\Delta t}{2}(1-\xi_i))$ 为矩阵指数。将式 (1-27)代入式(1-9)可得到高斯精细积分格式：

$$x_{k+1} = Tx_k + \frac{\Delta t}{2}\sum_{i=1}^{n}\omega_i T(\frac{\Delta t}{2}(1-\xi_i))R(t_k+\frac{\Delta t}{2}(1+\xi_i)) \quad (1\text{-}28)$$

上式计算得到的状态向量 x_{k+1} 的精度，取决于高斯积分点的数目和时间步长 Δt 的大小。当 n 越大时，计算精度越高。因此从理论上来说，该计算格式可达到任意计算精度。

n=3 时为 3 节点的高斯积分公式，代数精度为 5 阶，因此采用 3 节点的高斯积分公式在求解区间上能保证足够的精度，n=3 时，式(1-28)中的参数为：

$$\begin{aligned}\omega_1 &= \frac{8}{9}, \xi_1 = 0\\ \omega_2 &= \frac{5}{9}, \xi_2 = -\sqrt{0.6}\\ \omega_3 &= \frac{5}{9}, \xi_3 = \sqrt{0.6}\end{aligned} \quad (1\text{-}29)$$

1.1.4 提高计算效率的改进算法

精细积分法有着计算精度高，无条件稳定等优点，但是对于大规模有限元问题，其矩阵指数计算过程中的高维矩阵相乘带来极大的计算量，计算成本较高，因此考虑利用矩阵的结构特性，以及矩阵指数的物理意义对算法进行改进从而减少计算量和计算时间。

1.1.4.1 利用非齐次项结构特性减少计算量

通过利用非齐次项的结构特点，对矩阵进行分块计算，减小了矩阵维数，从而能够实现减小计算量，提高计算效率的目的。

当对式 (1-27)Duhamel 积分项采用数值积分时，将会出现以下的矩阵运算[7]：

$$\boldsymbol{D} = \boldsymbol{T}_{2n\times 2n} \times \boldsymbol{R(t)}_{2n\times 1} \tag{1-30}$$

对 $\boldsymbol{T}$ 和 $\boldsymbol{R(t)}$进行如下的分块：

$$\begin{aligned} \boldsymbol{T}_{2n\times 2n} &= [\boldsymbol{T}^{(1)}_{2n\times n} \quad \boldsymbol{T}^{(2)}_{2n\times n}] \\ \boldsymbol{R(t)}_{2n\times 1} &= \begin{bmatrix} \boldsymbol{R}^{(1)}_{n\times 1} \\ \boldsymbol{R}^{(2)}_{n\times 1} \end{bmatrix} \end{aligned} \tag{1-31}$$

由于状态空间方程中的 $\boldsymbol{R(t)}$是一个 $2\boldsymbol{n}\times 1$ 的列向量，并且前 $\boldsymbol{n}$ 行均为零，分块得到的 $\boldsymbol{R}^{(1)}_{n\times 1}$ 是 0，这样可以得到

$$\boldsymbol{D} = \boldsymbol{T}_{2n\times 2n} \times \boldsymbol{R(t)}_{2n\times 1} = \boldsymbol{T}^{(2)}_{2n\times n} \times \boldsymbol{R}^{(2)}_{n\times 1} \tag{1-32}$$

相当于把一个 $2\boldsymbol{n}\times 2\boldsymbol{n}$ 阶矩阵与 $2\boldsymbol{n}$ 阶列向量的乘法降阶为 $2\boldsymbol{n}\times\boldsymbol{n}$ 阶矩阵和 $\boldsymbol{n}$ 阶列向量的乘法。只统计乘法的计算时间，从计算量上来说，式(1-30)需要做 $2\boldsymbol{n}\times 2\boldsymbol{n}$ 次乘法，但式(1-32)仅需要做 $2\boldsymbol{n}\times\boldsymbol{n}$ 次乘法，可以将计算量减半。当计算过程出现式(1-30)的运算较多时，简化计算可以大大提高计算效率。

同时，以上的推导仅利用了非齐次项的结构特性，是等价变形而没有进行任何的近似，因此不会因为简化计算而损失计算精度。

从文献中得知[7]，对于同一个算例，不同数值求积公式在该简化算法下的计算效率提升如下表所示：

表 1-1 简化精细积分法与经典方法计算用时对比

Table 2 The comparison of numerical calculation efficiency

	the time of iterating 5000 steps	the time of iterating 10000 steps	the time of iterating 50000 steps
Trapezium formula	0.34	0.69	3.40
simplified Trapezium formula	0.28	0.56	2.78
Simpson formula[3]	0.51	1.01	5.02
simplified Simpson formula	0.38	0.77	3.78
Cotes formula [4]	0.88	1.75	8.71
simplified Cotes formula	0.62	1.23	6.12
Gauss formula [5]	0.76	1.53	7.61
simplified Gauss formula	0.44	0.87	4.37

从上表中可以看出，该简化算法将 Gauss 求积公式的计算效率提高约 43%。这种

简化算法的优势在于插值点越多，提升的效率越高。

更进一步，一般动力学分析时不一定每个节点上都作用有外载荷，所以外载荷 ***R(t)***中会出现大量的 0 元素，因此可以通过去掉这些 0 元素来进一步降低式(1-30)中矩阵乘法的阶数，减少计算量。

在矩阵运算之前将外载荷是 0 的行对应的部分进行重新分块，根据外载荷有大量 0 元素的特点，假设外载荷不为 0 的部分为 m 行，可以在式(1-31)的基础上继续分块如下[8]：

$$\boldsymbol{T}_{2n\times 2n}=[\boldsymbol{T}^{(1)}_{2n\times n}\quad \boldsymbol{T}^{(2)}_{2n\times(n-m)}\quad \boldsymbol{T}^{(3)}_{2n\times m}]$$
$$\boldsymbol{R(t)}_{2n\times 1}=\begin{bmatrix}\boldsymbol{R}^{(1)}_{n\times 1}\\ \boldsymbol{R}^{(2)}_{(n-m)\times 1}\\ \boldsymbol{R}^{(3)}_{m\times 1}\end{bmatrix} \tag{1-33}$$

去掉 0 项对应的部分，则相应的式(1-30)可改写为：

$$\boldsymbol{D}=\boldsymbol{T}_{2n\times 2n}\times\boldsymbol{R(t)}_{2n\times 1}=\boldsymbol{T}^{(3)}_{2n\times m}\times\boldsymbol{R}^{(3)}_{m\times 1} \tag{1-34}$$

通过上式的改进可以将式(1-32)中的 $2\boldsymbol{n}\times\boldsymbol{n}$ 次乘法进一步减小为 $2\boldsymbol{n}\times\boldsymbol{m}$ 次乘法。该改进方法节约的时间会随着 $\boldsymbol{m}$ 的变化而变化，其极限的情况就是外载荷只有 1 行不为零，所以只有 1 行参加计算，即最大能将矩阵与向量的乘法减小为矩阵和数字的乘法。只考虑矩阵乘法计算，定量分析一个迭代步内 Gauss 求积公式下的计算效率：

式(1-32)一个迭代步内的乘法计算量为：

$$2\boldsymbol{n}\times 2\boldsymbol{n}+3\times(2\boldsymbol{n}\times\boldsymbol{n})+2\boldsymbol{n} \tag{1-35}$$

利用式(1-34)改进后一个迭代步内乘法的计算量为：

$$2\boldsymbol{n}\times 2\boldsymbol{n}+3\times(2\boldsymbol{n}\times\boldsymbol{m})+2\boldsymbol{n} \tag{1-36}$$

改进后和改进前两者的比较为：

$$\frac{2\boldsymbol{n}\times 2\boldsymbol{n}+3\times(2\boldsymbol{n}\times\boldsymbol{m})+2\boldsymbol{n}}{2\boldsymbol{n}\times 2\boldsymbol{n}+3\times(2\boldsymbol{n}\times\boldsymbol{n})+2\boldsymbol{n}}=\frac{4\boldsymbol{n}+6\boldsymbol{m}+2}{10\boldsymbol{n}+2} \tag{1-37}$$

上式中 $\boldsymbol{m}$ 的取值范围为 1-$\boldsymbol{n}$，因此改进方法最多能在式(1-32)简化算法的基础上进一步节约大约 60%的时间。

文献中的算例计算时间对比如下所示[8]：

表 1-2 进一步改进的精细积分格式与简化格式计算时间的比较

	the time of iterating 1000 steps
simplified Trapezium formula[10]	3.42
improved Trapezium formula	2.33
simplified Simpson formula[10]	4.67
improved Simpson formula	2.38
simplified Cotes formula[10]	6.97
improved Cotes formula	2.38
simplified Gauss formula[10]	5.85
improved Gauss formula	2.38

对于 Gauss 求积公式，改进的算法可以进一步节约 59%的计算时间。

1.1.4.2 利用矩阵指数的物理意义简化计算

对于大规模系统，由于系统的自由度很大，计算矩阵指数将非常耗费时间和内存。虽然对于大规模动力系统，其刚度、阻尼和质量矩阵是稀疏矩阵，从而矩阵指数 $\boldsymbol{T}$ 也是稀疏矩阵，但是直接通过以上的精细积分方法计算其矩阵指数，在计算过程中，特别是方程的加法定理的合并过程中，矩阵将逐渐变为满矩阵或非常稠密的矩阵，很难利用矩阵的稀疏性质，从而计算量很大[9]。

考虑一种新的方程形式：

$$\dot{\boldsymbol{x}} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{R}$$
$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{p} \end{bmatrix}, \boldsymbol{H} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{M}^{-1} \\ -\boldsymbol{K} & -\boldsymbol{C}\boldsymbol{M}^{-1} \end{bmatrix} \tag{1-38}$$

式中 $\boldsymbol{p}$ 为动量。

能量在一维杆中的传播速度是有限值 $\sqrt{\boldsymbol{E}/\boldsymbol{\rho}}$，同理，在离散的结构中，虽然其能量的传播速度很难确切得到，但其动力学响应的能量传递速度肯定是有限的，这将对矩阵指数的结构产生很大影响。根据式(1-9)，如果外载荷为零，则方程可写为如下分块形式 ，即

$$\boldsymbol{T} = \exp(\boldsymbol{H}\Delta t) = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{T}_{21} & \boldsymbol{T}_{22} \end{bmatrix} \tag{1-39}$$

方程的解为：

$$\begin{aligned} \boldsymbol{a}_{k+1} &= \boldsymbol{T}_{11}\boldsymbol{a}_k + \boldsymbol{T}_{12}\boldsymbol{p}_k \\ \boldsymbol{p}_{k+1} &= \boldsymbol{T}_{21}\boldsymbol{y}_k + \boldsymbol{T}_{22}\boldsymbol{p}_k \end{aligned} \tag{1-40}$$

由上述方程可以得到矩阵指数元素的物理意义，即 ：$\boldsymbol{T}_{11}$ 的第 $\boldsymbol{i}$ 行第 $\boldsymbol{j}$ 列元素表示第 $\boldsymbol{j}$ 个自由度上给定单位位移，而其他自由度位移为零且所有自由度动量为零时，经过一个积分步长 $\boldsymbol{\Delta t}$ 后，第 $\boldsymbol{i}$ 个自由度的位移响应；$\boldsymbol{T}_{12}$ 的第 $\boldsymbol{i}$ 行第 $\boldsymbol{j}$ 列元素表示第 $\boldsymbol{j}$ 个自由度上给定单位动量，而其他自由度动量为零且所有自由度位移为零时，经过一个积分步长 $\boldsymbol{\Delta t}$ 后，第 $\boldsymbol{i}$ 个自由度的位移响应；$\boldsymbol{T}_{21}$ 的第 $\boldsymbol{i}$ 行第 $\boldsymbol{j}$ 列元素表示第 $\boldsymbol{j}$ 个自由度上给定单位位移,而其他自由度位移为零且所有自由度动量为零时，经过一个积分步长 $\boldsymbol{\Delta t}$ 后，第 $\boldsymbol{i}$ 个自由度的动量响应；$\boldsymbol{T}_{22}$ 的第 $\boldsymbol{i}$ 行第 $\boldsymbol{j}$ 列元素表示第 $\boldsymbol{j}$ 个自由度上给定单位动量，而其他自由度动量为零且所有自由度位移为零时，经过一个积分步长 $\boldsymbol{\Delta t}$ 后，第 $\boldsymbol{i}$ 个自由度的动量响应。由于能量在结构中的传递速度是有限的，假设某个自由度上有扰动，在较小的时间内，必然只能传播到有限的自由度，而不可能传播到所有自由度。根据上面给出的 $\boldsymbol{T}_{11}$,$\boldsymbol{T}_{12}$,$\boldsymbol{T}_{21}$和$\boldsymbol{T}_{22}$ 物理含义，则它们一定是稀疏矩阵。这样，就可以将矩阵指数作为稀疏矩阵计算和存储，从而节约计算量和存储空间[9]。

对于给定矩阵 $\boldsymbol{H}$ 和积分步长 $\boldsymbol{\Delta t}$，原始的精细积分方法按如下过程计算矩阵指数：首先确定 N，然后令 $\boldsymbol{H}' = \boldsymbol{H\Delta t} / \boldsymbol{2}^{\boldsymbol{N}}$，其次按照式(1-12)计算 $\boldsymbol{Ta}$，然后执行循环式(1-15)，最后将 $\boldsymbol{R}$ 矩阵与单位矩阵相加得到矩阵指数。

若采用上述的计算过程，虽然矩阵 $\boldsymbol{H}$ 为稀疏矩阵，但是如果观察根据式(1-15)计算得到的矩阵 $\boldsymbol{R}$，可以发现它可能是一个满矩阵或很不稀疏的矩阵，但仔细观察会发现，此矩阵有很多很小的元素；另一方面，根据上面的分析，此时的矩阵 $\boldsymbol{Ta}$ 相当于很小的时间步长 $\boldsymbol{\Delta t} / \boldsymbol{2}^{\boldsymbol{N}}$ 对应的矩阵指数减去一个单位矩阵，因此按照上面分析的能量传播性质，它一定是非常稀疏的矩阵。因此，此时 $\boldsymbol{Ta}$ 矩阵中非常小的非零元素是计算误差，它们理论上应该为零，因此需要将它们判断出来，并令它们为零，从而将 $\boldsymbol{Ta}$ 转换为稀疏矩阵存储。具体过程是：从上面分析给出的矩阵指数的物理意义可知，$\boldsymbol{R}$ 矩阵的四块子矩阵物理意义不同，因此将 $\boldsymbol{Ta}$

分为Ta_{11},Ta_{12},Ta_{21}和Ta_{22}四块 ，假设a_{11}为矩阵Ta_{11}中绝对值最大的元素，并给定一个误差标准，如$\varepsilon = 10^{-25}$，则Ta_{11}中的元素如果满足其绝对值小于$a_{11} \cdot \varepsilon$则认为此元素应该为零，根据此原则可将Ta_{11}稀疏存储。同样，可将Ta_{12}、Ta_{21}和Ta_{22}稀疏化[9]。以上过程定义为一个矩阵的稀疏化。

对于大规模动力系统，选取一个合理的时间步长，某一处的扰动经过一个时间步长后，其影响只是局部化的，不会传播到整个结构，因此系统对应的矩阵指数一定是稀疏矩阵，则可将精细积分方法作如下改进，即对于给定矩阵 H 和积分步长 Δt，可给出如下的快速精细积分算法(FPIM)，来计算矩阵指数。

（1） 由于 H 为稀疏矩阵，将 H 按照稀疏矩阵存储

（2） 计算 $T_a = H\Delta\tau + \frac{(H\Delta\tau)^2}{2}[I + \frac{H\Delta\tau}{3} + \frac{(H\Delta\tau)^2}{12}]$

（3） 将 Ta 矩阵稀疏化

（4） 执行如下语句，即

for iter =1:N

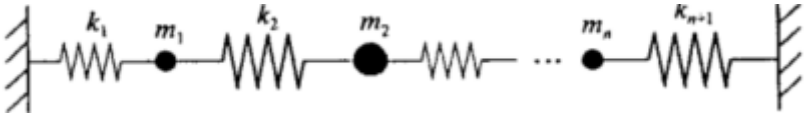
$Ta = Ta*（Ta+2*I）$

将 Ta 稀疏化；

end

（5） 得到矩阵 Ta 后，将其与单位矩阵 I 相加。

上述计算过程与原始精细积分方法相比，只是增加了矩阵 Ta 的稀疏化过程，但是这样的处理将极大地提高计算效率。通过文献中的算例可以发现进行矩阵元素清零的快速精细积分法可以极大的提高大规模稀疏矩阵的计算速度[9]。



	FPIM	ODE45	PIM	Newmark
CPU time(s)	18.3	1162.7	412.5	440.4

图 1-1 多种计算方法计算时间对比

第2章 扩展程序框架

2.1 扩展程序功能

本次大作业中所使用的 STAPMAT 程序是基于 MATLAB 语言编写的有限元问题求解程序，具有模块化程度高、可修改性强、适应性好等多种优点。本次大作业的主要任务是对 STAPMAT 程序进行拓展：(1)在静力学部分增添一种线性三维实体单元和一种高阶三维实体单元；(2)增加特征值求解功能；(3)增加时间积分求解功能，包括模态叠加法、速度 Verlet 法、广义α法、使用高斯积分的精细积分法、线性近似载荷的精细积分法。由此需要对 STAPMAT 的文件读入部分，单元计算部分和整体求解部分进行修改和开发。

2.2 程序输入文件格式

本次大作业使用 ABAQUS 进行前处理，前期在 ABAQUS 建模和网格划分，之后通过宏文件进行格式化输出，生成 STAPMAT 程序的输入文件。由于添加了单元和求解模块，因此添加了相关输入格式，修改后的输入文件格式如下：

（1）标题行

列	变量	意义
1-80	HED(80)	标题，用于对所求问题进行简单的描述

（2）控制行

列	变量	意义
1-5	NUMNP	节点总数：如果为 0 则程序终止运行
6-10	NUMEG	单元组总数，每个单元组只包含相同类型的单元
11-15	NUMEM	单元总数
15-20	NLCASE	载荷工况数
20-25	MODEX	求解模式，等于 0 时只做数据检查，

		等于 1 时进行求解， 等于 2 时使用稀疏矩阵左除求解， 等于 3 时同时求解特征值 等于 4 时求解时间积分（且只求解第一个载荷工况）
26-30	DSTIME	求解时间（静力学可不写）
31-35	DSMETH	时间积分方法（静力学可不写）： 1-模态叠加；2-速度 Verlet；3-广义α； 4-精细积分法 1；5-精细积分法 2

注：时间积分方法中，精细积分法 1 使用高斯积分，其特点为可以将时间步长取较大而不影响计算结果的精度，但由于需要计算高斯点上的值，计算速度慢；精细积分法 2 中，使用线性函数近似载荷项，其计算速度较快，但需要取较小的时间步长。因此，当只关心结构在某一时刻的响应时，可以使用精细积分法 1，时间步长可以只取一步；当关心结构在一个时间段内的响应时，取较小的时间步长，选择精细积分法 2，可以较快的计算出结果。

（3）节点数据

列	变量	意义
1-5	N	节点号
6-10	ID(1,N)	x-平动方向边界条件代码（0-自由，1-固定）
11-15	ID(2,N)	y-平动方向边界条件代码（0-自由，1-固定）
16-20	ID(3,N)	z-平动方向边界条件代码（0-自由，1-固定）
21-39	X	X 坐标
40-57	Y	Y 坐标
58-75	Z	Z 坐标

（4）载荷数据：

共输入 NLCASE 组载荷数据

（a）载荷数据控制行

列	变量	意义
1-5	LL	载荷工况号，必须按顺序输入所有载荷工况数据
6-10	NLOAD	本工况中集中载荷的个数

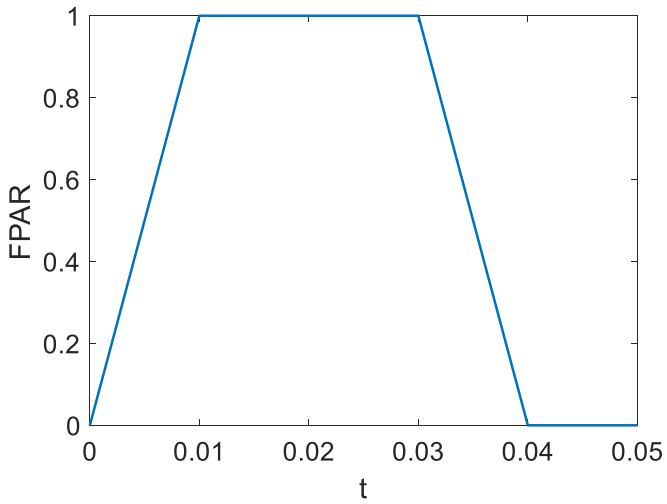
11-15	DLTYPE	动力学载荷类型；1-分段线性；2-正弦函数 如为静力学问题可不写
16-20	DLPAR(1)	动载荷系数参数(静力学问题可不写)：t_1 / t_1
21-25	DLPAR(2)	动载荷系数参数(静力学问题可不写)：t_2 / ω
26-30	DLPAR(3)	动载荷系数参数(静力学问题可不写)：t_3 / φ
31-35	DLPAR(4)	动载荷系数参数(静力学问题可不写)：t_4 / t_4

注：在动力学计算中，为了增加施加随时间变化的载荷，输入文件增加了动载荷类型和动载荷系数两类参数。在程序 GetDynamicsPAR.m 中，可以根据动载荷类型和动载荷系数参数，计算得到某一时刻的动载荷系数。该时刻的动载荷系数乘节点载荷，即为该时刻的载荷。

对于分段线性的动载荷类型，其四个动载荷系数参数均为时间，分别为 t_1,t_2,t_3,t_4，其意义为：动载荷系数从 0 开始增加的时刻、线性增加到 1 的时刻、从 1 开始线性减小的时刻、线性减小到 0 的时刻。

对于正弦函数的动载荷类型，其四个动载荷系数为t_1,ω,φ,t_4，分别表示：载荷系数从 0 开始变为正弦函数的时刻、正弦函数的圆频率、正弦函数的相位、载荷系数从正弦函数变为 0 的时刻。

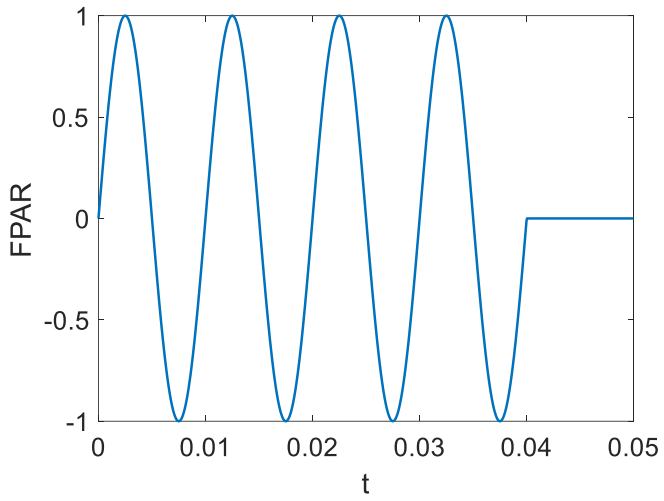
例如：求解时间为 0 到 0.05s，如果载荷类型为分段线性，动载荷系数参数



为 0，0.01，0.03，0.04，则动载荷系数随时间的变化如下图：

图 2.1 线性动载荷系数样例

如果求解时间为 0 到 0.05s，载荷类型为正弦函数，动载荷系数参数为 0，



628，0，0.04，则动载荷系数随时间的变化如下图：

图 2.2 正弦函数动载荷样例

（b）各工况载荷数据

列	变量	意义
1-5	NOD	集中载荷作用的节点号
6-10	IDIRN	载荷作用方向（1-x 方向，2-y 方向，3-z 方向）
11-20	FLOAD	载荷值

第3章 静力学理论、实现与验证

3.1 单元理论

3.1.1 单元基本理论

有限元中将节点位移作为基本变量，使用插值求得任一点的位移、应变和应力。假设一个单元有 m 个节点，则单元节点位移为

$$\boldsymbol{a}^e=\left[u_1,v_1,w_1,u_2,v_2,w_2,...,u_m,v_m,w_m\right]^T_{(m\times3)\times1} \tag{3-1}$$

其中位移使用形函数插值：

$$u=\sum_{i=1}^{m}N_i u_i \qquad v=\sum_{i=1}^{m}N_i v_i \qquad w=\sum_{i=1}^{m}N_i w_i \tag{3-2}$$

即

$$\begin{bmatrix} u & v & w \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{N} \end{bmatrix}_{3\times(m\times 3)} \boldsymbol{a}^e_{(m\times 3)\times 1} \tag{3-3}$$

形函数矩阵为

$$\begin{bmatrix} \boldsymbol{N} \end{bmatrix}_{3\times(m\times 3)} = \begin{bmatrix} \boldsymbol{I}N_1 & \boldsymbol{I}N_2 & ..... & \boldsymbol{I}N_m \end{bmatrix}_{3\times(m\times 3)} \tag{3-4}$$

这里 $\boldsymbol{I}$ 为三阶单位阵，即

$$\boldsymbol{I}=\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}$$

$N_i(i=1,2,...,m)$ 是单元位移模式的插值基函数，也称为形函数。

C3D8 单元在自然坐标下的节点编号如下：

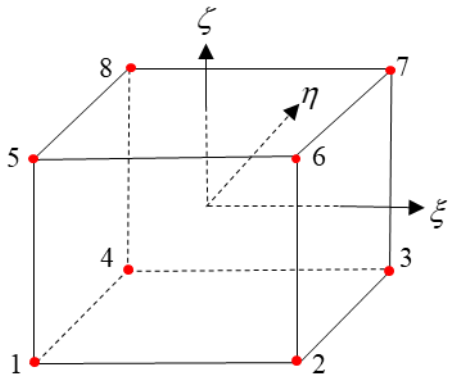


图 3.1 C3D8 单元节点编号

形函数为：

$$N_i(\xi,\eta,\zeta)=\frac{1}{8}(1+\xi_i\xi)(1+\eta_i\eta)(1+\zeta_i\zeta) \qquad i=1,...,8 \tag{3-5}$$

C3D20 单元在自然坐标下的节点编号如下：

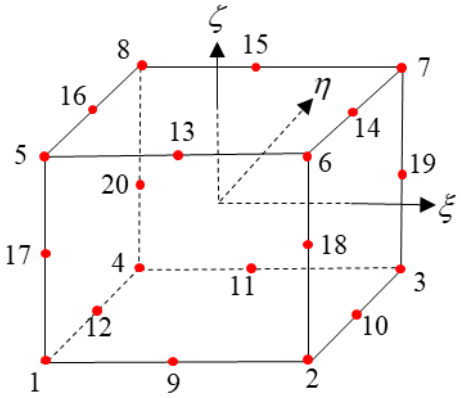


图 3.2 C3D20 单元节点编号

C3D20 单元的形函数分为角节点和棱内节点：

角节点 $(i=1,2,3,4,5,6,7,8)$

$$N_i=\frac{1}{8}(1+\xi_i\xi)(1+\eta_i\eta)(1+\zeta_i\zeta)(\xi_i\xi+\eta_i\eta+\zeta_i\zeta-2) \tag{3-6}$$

棱内节点 $(i=9,11,15,13)$

$$N_i=\frac{1}{4}(1-\xi^2)(1+\eta_i\eta)(1+\zeta_i\zeta) \tag{3-7}$$

棱内节点 $(i=10,12,16,14)$

$$N_i=\frac{1}{4}(1+\xi_i\xi)(1-\eta^2)(1+\zeta_i\zeta) \tag{3-8}$$

棱内节点 $(i=17,18,19,20)$

$$N_i=\frac{1}{4}(1+\xi_i\xi)(1+\eta_i\eta)(1-\zeta^2) \tag{3-9}$$

根据弹性力学的相关理论，三维实体单元的应变分量为

$$\boldsymbol{\varepsilon}=\begin{bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{bmatrix}=\begin{bmatrix}\dfrac{\partial u}{\partial x}\\ \dfrac{\partial v}{\partial y}\\ \dfrac{\partial w}{\partial z}\\ \dfrac{\partial u}{\partial y}+\dfrac{\partial v}{\partial x}\\ \dfrac{\partial v}{\partial z}+\dfrac{\partial w}{\partial y}\\ \dfrac{\partial w}{\partial x}+\dfrac{\partial u}{\partial z}\end{bmatrix}=\boldsymbol{B}\boldsymbol{a}^e \tag{3-10}$$

其中 $\boldsymbol{B}$ 为单元的几何矩阵，即为如下形式：

$$\boldsymbol{B}=[B_1 \quad B_2 \quad \ldots \quad B_m]_{6\times(m\times 3)} \tag{3-11}$$

每个子矩阵的形式为

$$[B_i]=\begin{bmatrix}\dfrac{\partial N_i}{\partial x} & 0 & 0\\ 0 & \dfrac{\partial N_i}{\partial y} & 0\\ 0 & 0 & \dfrac{\partial N_i}{\partial z}\\ \dfrac{\partial N_i}{\partial y} & \dfrac{\partial N_i}{\partial x} & 0\\ 0 & \dfrac{\partial N_i}{\partial z} & \dfrac{\partial N_i}{\partial y}\\ \dfrac{\partial N_i}{\partial z} & 0 & \dfrac{\partial N_i}{\partial x}\end{bmatrix}_{6\times 3} \tag{3-12}$$

由于形函数 N_i 是基于局部坐标得到的， $\boldsymbol{B}$ 矩阵中每个元素的计算将涉及到全局坐标与局部坐标间的坐标转换和相应的雅克比矩阵 $\boldsymbol{J}$

$$\begin{bmatrix}\dfrac{\partial N_i}{\partial x}\\ \dfrac{\partial N_i}{\partial y}\\ \dfrac{\partial N_i}{\partial z}\end{bmatrix}=\boldsymbol{J}^{-1}\begin{bmatrix}\dfrac{\partial N_i}{\partial \xi}\\ \dfrac{\partial N_i}{\partial \eta}\\ \dfrac{\partial N_i}{\partial \zeta}\end{bmatrix},\ \boldsymbol{J}\equiv\frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)}=\begin{bmatrix}\dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} & \dfrac{\partial z}{\partial \xi}\\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} & \dfrac{\partial z}{\partial \eta}\\ \dfrac{\partial x}{\partial \zeta} & \dfrac{\partial y}{\partial \zeta} & \dfrac{\partial z}{\partial \zeta}\end{bmatrix} \tag{3-13}$$

代入坐标转换关系与位移模式，可以得到雅克比矩阵的计算公式为

$$\boldsymbol{J}\equiv\frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)}=\begin{bmatrix}\sum_{i=1}^{m}\frac{\partial N_i}{\partial\xi}x_i & \sum_{i=1}^{m}\frac{\partial N_i}{\partial\xi}y_i & \sum_{i=1}^{m}\frac{\partial N_i}{\partial\xi}z_i\\ \sum_{i=1}^{m}\frac{\partial N_i}{\partial\eta}x_i & \sum_{i=1}^{m}\frac{\partial N_i}{\partial\eta}y_i & \sum_{i=1}^{m}\frac{\partial N_i}{\partial\eta}z_i\\ \sum_{i=1}^{m}\frac{\partial N_{ii}}{\partial\zeta}x_i & \sum_{i=1}^{m}\frac{\partial N_{ii}}{\partial\zeta}y_i & \sum_{i=1}^{m}\frac{\partial N_i}{\partial\zeta}z_i\end{bmatrix}=\begin{bmatrix}\frac{\partial N_1}{\partial\xi} & \frac{\partial N_2}{\partial\xi} & \cdots & \frac{\partial N_m}{\partial\xi}\\ \frac{\partial N_1}{\partial\eta} & \frac{\partial N_2}{\partial\eta} & \cdots & \frac{\partial N_m}{\partial\eta}\\ \frac{\partial N_1}{\partial\zeta} & \frac{\partial N_2}{\partial\zeta} & \cdots & \frac{\partial N_m}{\partial\zeta}\end{bmatrix}\begin{bmatrix}x_1 & y_1 & z_1\\ x_2 & y_2 & z_2\\ \vdots & \vdots & \vdots\\ x_m & y_m & z_m\end{bmatrix}$$

(3-14)

对于三维问题，单元应力应变间的本构方程如下

$$\boldsymbol{\sigma}=\begin{bmatrix}\sigma_x & \sigma_y & \sigma_z & \tau_{xy} & \tau_{yz} & \tau_{zx}\end{bmatrix}^{\boldsymbol{T}}=\boldsymbol{D\varepsilon} \tag{3-15}$$

其中弹性矩阵 $\boldsymbol{D}$ 为常数阵，由材料参数计算得到

$$\boldsymbol{D}=\frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\begin{bmatrix}1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & & & & & \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & & & & & \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & & & & & \\ & & & \frac{1-2\nu}{2(1-\nu)} & & & & \\ & & & & \frac{1-2\nu}{2(1-\nu)} & & & \\ & & & & & \frac{1-2\nu}{2(1-\nu)} & & \end{bmatrix} \tag{3-16}$$

根据变分原理推导与上述推导，可以计算出单元刚度阵为

$$\boldsymbol{K^e}=\int_{\Omega^e}\boldsymbol{B^T DB}dV=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\boldsymbol{B^T DB}\left|\boldsymbol{J}\right|d\xi d\eta d\zeta \tag{3-17}$$

同理，单元协调质量阵为

$$\boldsymbol{M^e}=\int_{\Omega^e}\rho\boldsymbol{N^T N}dV=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\rho\boldsymbol{N^T N}\left|\boldsymbol{J}\right|d\xi d\eta d\zeta \tag{3-18}$$

在程序中，使用高斯积分计算单元刚度阵和协调质量阵。

为了计算速度更快，改写程序中采用了集中质量阵 $\tilde{\boldsymbol{M}}$，在计算得到单元协调质量阵后，使用对角元素放大法的集中质量阵，即将协调质量阵的对角元 $\boldsymbol{M}_{rr}$ 乘一个放大系数 α 作为集中质量阵的对角元，即

$$\tilde{M}_{rr}=\alpha M_{rr} \tag{3-19}$$

其中α满足

$$\sum_r M_{rr} = \int_V \rho dV \tag{3-20}$$

3.1.2 应力磨平

在计算单元刚度阵和质量阵时，已经求出了高斯积分点上的几何矩阵$\boldsymbol{B}$和形函数$\boldsymbol{N}$，因此在计算出单元节点位移后，可以直接根据上述公式求出高斯积分点上的应力。但我们希望得到节点上的应力，可以通过高斯积分点上的应力插值得到。本次大作业主要参照王勖成《有限单元法》中的单元应力磨平方法，该方法的基本思想是用形函数将节点上的应力插值到高斯积分点上，具体如下：

$$\begin{Bmatrix} \boldsymbol{\sigma}_I \\ \boldsymbol{\sigma}_{II} \\ \vdots \\ \boldsymbol{\sigma}_L \end{Bmatrix} = \begin{bmatrix} \boldsymbol{N}_1(\mathrm{I}) & \boldsymbol{N}_2(\mathrm{I}) & \cdots & \boldsymbol{N}_m(\mathrm{I}) \\ \boldsymbol{N}_1(\mathrm{II}) & \boldsymbol{N}_2(\mathrm{II}) & \cdots & \boldsymbol{N}_m(\mathrm{II}) \\ \vdots & \vdots & \cdots & \vdots \\ \boldsymbol{N}_1(\boldsymbol{L}) & \boldsymbol{N}_2(\boldsymbol{L}) & \cdots & \boldsymbol{N}_m(\boldsymbol{L}) \end{bmatrix} \begin{Bmatrix} \boldsymbol{\sigma}_1^* \\ \boldsymbol{\sigma}_2^* \\ \vdots \\ \boldsymbol{\sigma}_m^* \end{Bmatrix} \tag{3-21}$$

其中等式左侧为高斯积分点上的应力，右侧为高斯积分点上的形函数组成的矩阵乘节点上的应力列向量。在应力磨平时，高斯积分点上的应力和形函数的值已知，只需要用高斯积分点上的应力左除高斯积分点上的形函数组成的矩阵，即可得到节点上的应力，由于一个节点可能属于多个单元，因此最后还需要取平均值，得到应力磨平结果。

需要注意的是，由于计算时需要左除，如果要得到正确的结果，需要保证高斯积分点上的形函数组成的矩阵需要行数大于等于列数，即高斯积分点的个数需要大于等于节点数。因此这种应力磨平方法不适用与减缩积分的情况。对于 C3D8 单元的全积分格式，使用 8 个高斯积分点求 8 个节点的应力；对于 C3D20 单元的全积分格式，使用 27 个高斯积分点求 20 个节点的应力；两种单元均可正确计算。

3.1.3 后处理

使用 Tecplot 作为后处理软件，在程序中创建了两个输出文件，分别以_anim.DAT 和_curv.DAT 结尾，其中包含了单元节点坐标信息和节点位移、应力应变等信息，可被 Tecplot 读取。

3.2 单元实现与静力学算例验证

3.2.1 C3D8 单元实现与静力学验证

C3D8 单元的实现在文件夹.\SRC\Mechanics\C3D8 中，其中文件的主要功能见下表：

子程序名称	子程序功能描述
ReadC3D8.m	读取 in 文件，建立单元信息
C3D8Stiff.m	计算单元刚度阵和集中质量阵
C3D8NJB.m	计算形函数、Jacobi 矩阵、几何矩阵
C3D8Stress.m	计算高斯积分点上的应力应变，应力磨平

表 3.1 C3D8 单元程序实现

为了验证 C3D8 单元程序的正确性，先使用一个单元的单轴拉伸做简单验证，结构如下：

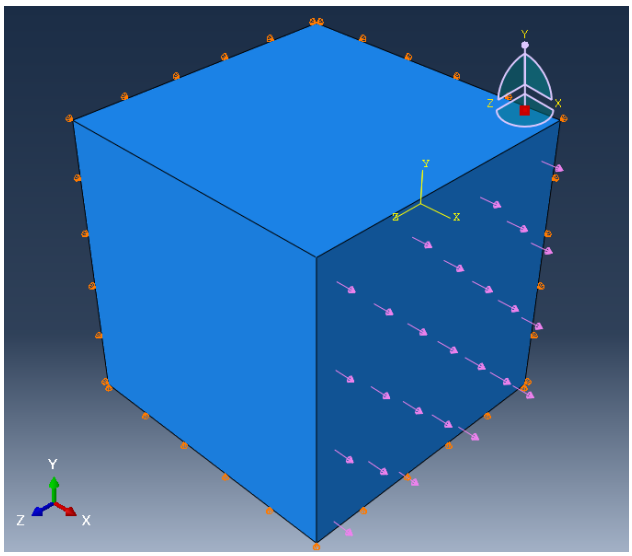


图 3.3 C3D8 胞元单轴拉伸模型

如图所示，模型为正方体，划分一个单元，约束了三个面的法向位移，并在一个面给均匀拉伸载荷，计算结果如下图：

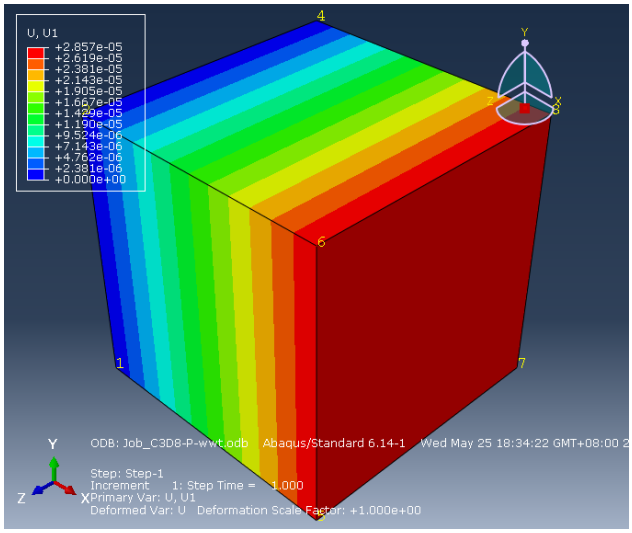


图 3.4 C3D8 胞元单轴拉伸计算结果

对比 STAPMAT 和 ABAQUS 计算得到的每个自由度上的位移，结果如图：

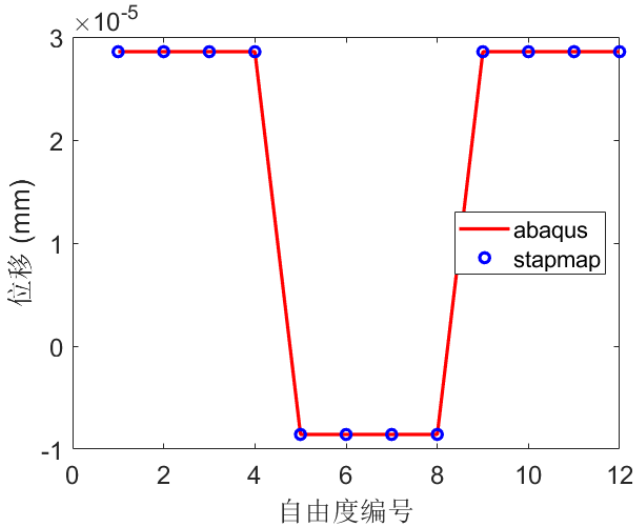


图 3.5 C3D8 胞元单轴拉伸位移结果比较

从图中可以看到，两者结果完全相同。

进一步比较了节点 6 上的应力和应变：

对比项	ABAQUS	STAPMAT
E11	14.2857E-06	1.428571e-05
S11	3.000000E+00	3.00000

E22	-4.28571E-06	-4.285714e-06
E33	-4.28571E-06	-4.285714e-06

表 3.2 C3D8 胞元单轴拉伸应力应变结果比较

综上，对于 C3D8 单元胞元的单轴拉伸问题，STAPMAT 结果与 ABAQUS 结果完全相同，初步验证了单元的正确性。

接下来讨论悬臂梁的纯弯曲问题，模型和载荷如下，共 40 个单元：

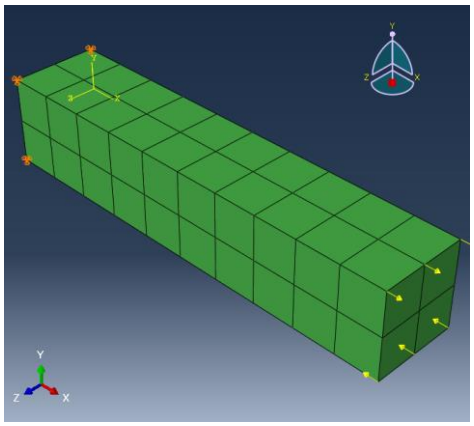


图 3.6 C3D8 单元梁弯曲问题

取梁上表面中线上节点的挠度作比较：

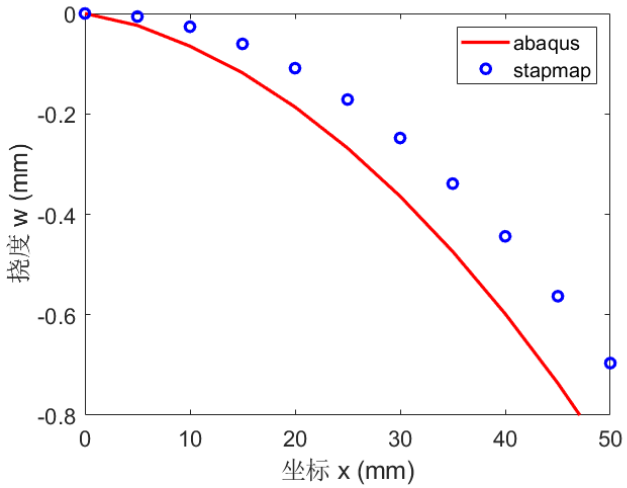


图 3.7 C3D8 单元梁挠度结果比较

从图中可以看到，STAPMAT 的挠度计算结果比 ABAQUS 小，这是因为 C3D8 单元为线性单元，在使用全积分计算纯弯曲问题时，会出现剪切锁死的现象。其原

因如图所示：

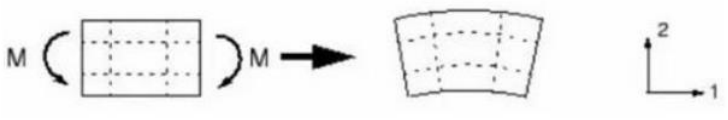


图 3.8 高阶单元纯弯曲

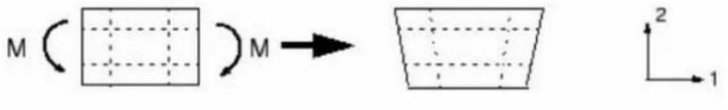


图 3.9 线性单元纯弯曲

材料力学中，梁在纯弯曲载荷下只有正应变，没有剪应变，高阶单元能够实现这种变形，如图 3.8 所示。线性单元的边都是直线，因此在纯弯曲时，单元会出现剪应变，如图 3.9 所示，这产生了名义上的剪应力，增加了刚度。在 ABAQUS 中，对应这种情况做了处理，减小了剪切锁死带来的刚度增加，但 STAPMAT 没有增加这方面的功能，因此挠度计算结果会偏小。

为了减弱剪切锁死，可以采用减缩积分、加密网格等方法。将网格加密后（320 个单元）的模型和计算结果如下：

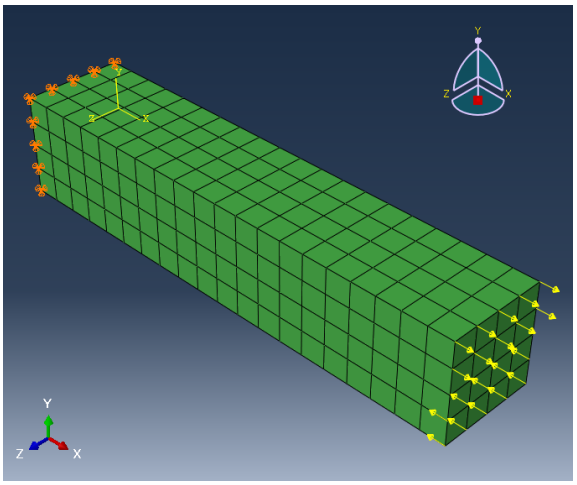


图 3.10 C3D8 梁弯曲问题——加密网格

梁中线上节点挠度分布：

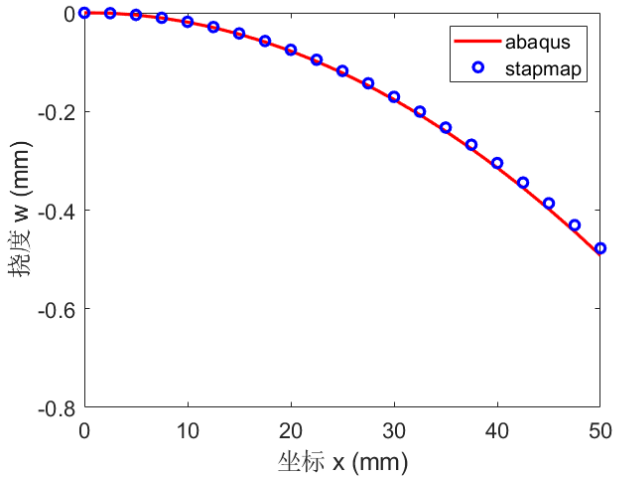


图 3.11 C3D8 梁弯曲挠度分布——加密网格

从图中可以看到加密网格后 STAPMAT 计算的挠度略小于 ABAQUS 计算结果，但差距很小，说明加密网格可以减小剪切锁死带来的影响，同时也验证了程序的正确性。

3.2.2 C3D20 单元实现与静力学验证

C3D20 单元的实现在文件夹.\SRC\Mechanics\C3D20 中，其中文件的主要功能见下表：

子程序名称	子程序功能描述
ReadC3D20.m	读取 in 文件，建立单元信息
C3D20Stiff.m	计算单元刚度阵和集中质量阵
C3D20NJB.m	计算形函数、Jacobi 矩阵、几何矩阵
C3D20Stress.m	计算高斯积分点上的应力应变，应力磨平

表 3.3 C3D20 单元程序实现

为了验证 C3D20 单元程序的正确性，先使用一个单元的单轴拉伸做简单验证，结构如下

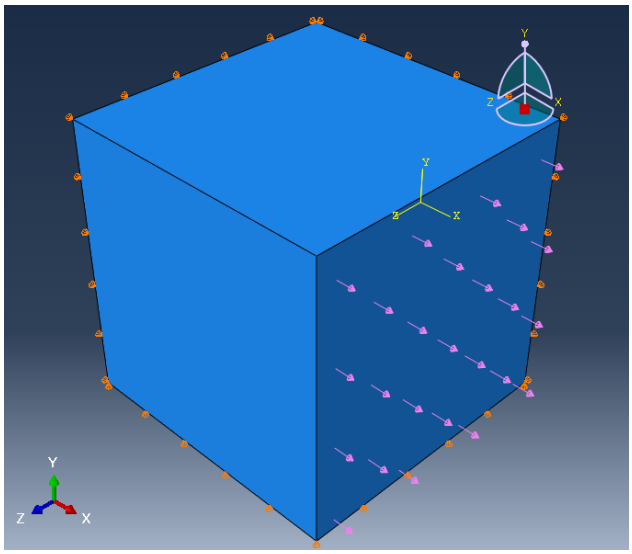


图 3.12 C3D20 胞元单轴拉伸模型

如图所示，模型为正方体，划分一个单元，约束了三个面的法向位移，并在一个面给均匀拉伸载荷。

STAPMAT 和 ABAQUS 计算得到的节点位移结果如下：

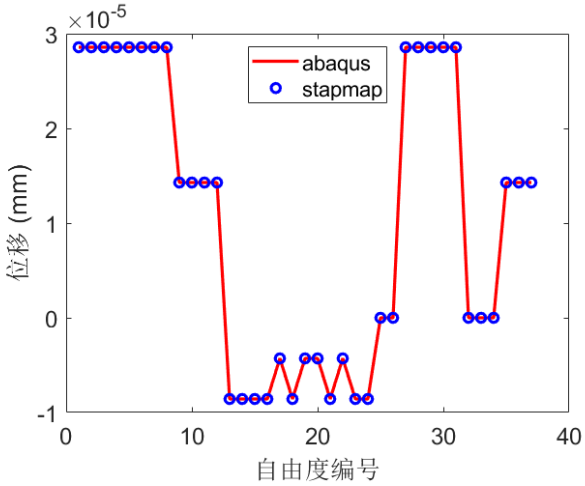


图 3.13 C3D20 胞元单轴拉伸位移结果

从图中可以看到二者位移结果完全相同，节点 6 上的应力应变计算结果如下：

对比项	Abaqus	Stapmat
E11	14.2857E-06	1.428571e-05
S11	3.000000E+00	3.00000

E22	-4.28571E-06	-4.285714e-06
E33	-4.28571E-06	-4.285714e-06

表 3.4 C3D20 胞元单轴拉伸应力音变结果对比

从表中可以看到，该节点上的应力应变结果也相同，验证了程序的正确性。

进一步，使用悬臂梁的纯弯曲问题做验证，模型如下，先划分单元数为 40：

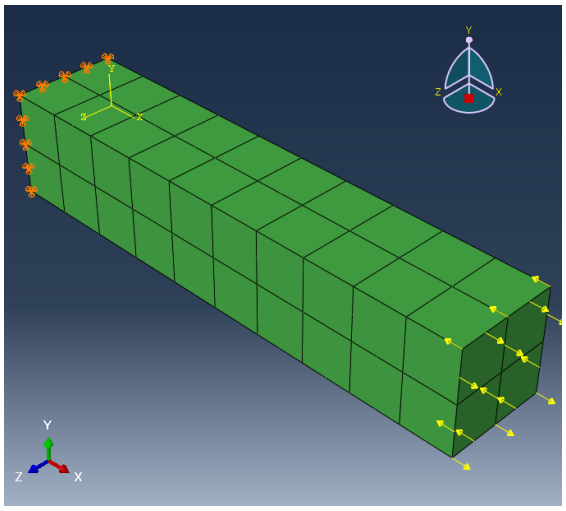


图 3.14 C3D20 悬臂梁纯弯曲模型

取梁上表面中线上节点的挠度做对比：

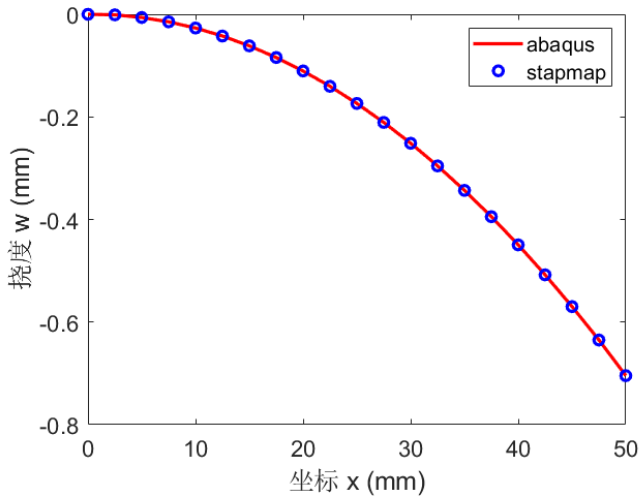


图 3.15 C3D20 梁纯弯曲结果比较

从图中可以看到，STAPMAT 的计算结果与 ABAQUS 完全吻合。

同时，也对加密网格（32 个单元）后的结果做了对比：

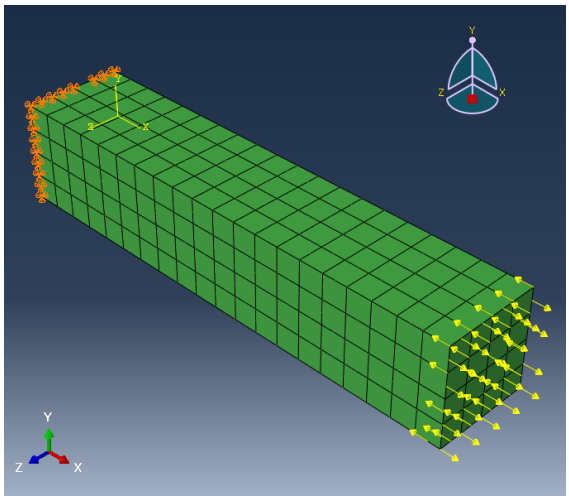


图 3.16 C3D20 悬臂梁纯弯曲问题——加密网格

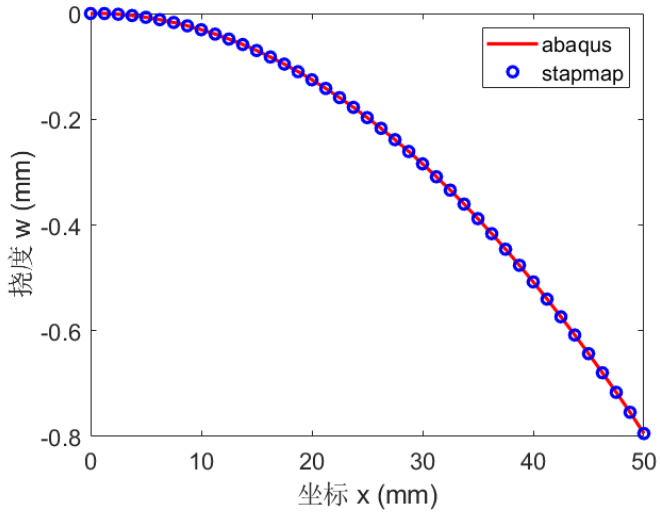


图 3.17 C3D20 悬臂梁纯弯曲结果比较——加密网格

从图中可以看到，加密网格后，计算结果同样吻合。验证了 C3D20 单元的正确性。

第4章 特征值理论、实现与验证

4.1 特征值求解理论

在动力学问题中，系统振动的固有频率和阵型是一个重要的问题。求解固有频率和阵型即为求解刚度阵 $\boldsymbol{K}$ 和质量阵 $\boldsymbol{M}$ 的广义特征值问题，其中特征根对应圆频率的平方，特征向量对应阵型。本次大作业使用了带正交化的逆迭代法求系统前 EIG_NUM 阶特征值。算法如下：

for i = 1:EIG_NUM

1. 选取初始迭代向量 x_1，计算 $y_1 = \boldsymbol{M}x_1$， $k=1$

2. 使用已经求得的前 j 阶特征向量正交化：

$$x_k = x_k - \sum_{m=1}^{j} \alpha_m \varphi_m, \quad \alpha_m = \varphi_m^T \boldsymbol{M} x_k$$

3. 解方程 $\boldsymbol{K}\overline{x}_{k+1} = y_k$

4. 计算 $\overline{y}_{k+1} = \boldsymbol{M}\overline{x}_{k+1}$

5. 计算瑞利商

$$\rho\left(\overline{x}_{k+1}\right) = \frac{\overline{x}_{k+1}^T y_k}{\overline{x}_{k+1}^T \overline{y}_{k+1}}$$

6. 判断

$$\frac{\left|\rho\left(\overline{x}_{k+1}\right) - \rho\left(\overline{x}_k\right)\right|}{\rho\left(\overline{x}_{k+1}\right)} \le tol\,?$$

-成立：$\lambda_i = \rho\left(\overline{x}_{k+1}\right)$, $\varphi_i = \dfrac{\overline{x}_{k+1}^T}{\left(\overline{x}_{k+1}^T \overline{y}_{k+1}\right)^{1/2}}$

-不成立：令 $k = k+1$，转向 2

end

4.2 特征值程序实现与算例验证

在程序中，文件 Eigenvalue.m 实现了特征值求解，该文件读取结构的刚度阵和质量阵，并使用 4.1 中带正交化的逆迭代法求解结构的前六阶固有频率和阵型。

为了验证程序的正确性，计算了 C3D20 胞元的固有频率，约束与静力学验证中的算例相同。模型和计算结果如下：

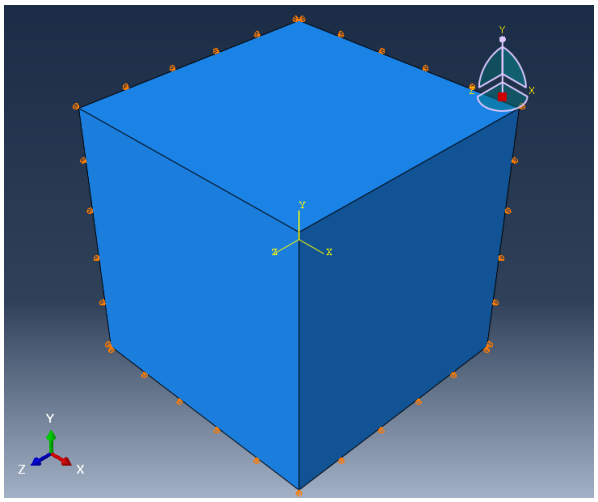


图 4.1 C3D20 胞元模型

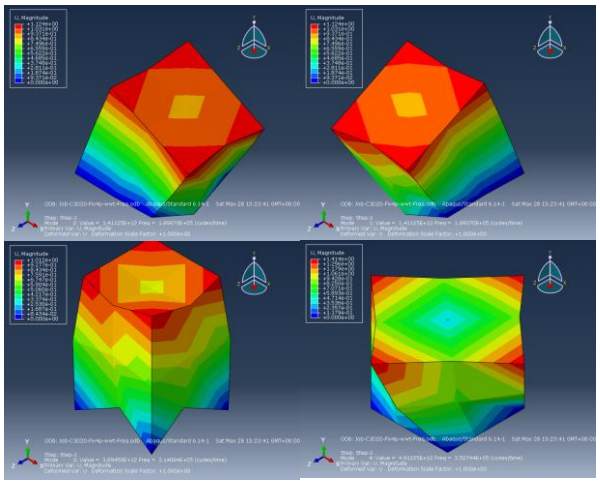


图 4.2 C3D20 胞元阵型计算结果

对比项[Hz]	ABAQUS	STAPMAT
Freq1	5.73507e+05	5.735151e+05
Freq2	5.73507e+05	5.735152e+05

Freq3	5.78180e+05	5.781631e+05
Freq4	7.16340e+05	7.163402e+05
Freq5	7.16340e+05	7.163404e+05
Freq6	9.17184e+05	9.171836e+05

表 4.1 C3D20 胞元固有频率结果对比

从以上图表可以看到，STAPMAT 计算结果与 AQAQUS 相同，验证了程序的正确性。

第5章 时间积分理论、实现与验证

5.1 改进的精细积分法求解结构动力学方程

基于文献调研的成果，在经典精细积分法的基础上，利用 Taylor 级数展开求解矩阵指数；使用三节点的 Gauss 求积公式对 Duhamel 积分项直接求解。同时利用非齐次项的结构特性以及矩阵指数的物理意义减小计算量，提高计算效率。在本节中将对代码内容进行详细介绍。

此外，在程序编写过程中遇到了一些有讨论价值的问题，将在本节中对这些问题进行介绍和原因分析。

5.1.1 改进精细积分法的代码实现

5.1.1.1 基于 Taylor 级数展开近似的矩阵指数精细积分求解

根据文献中的结论，对于实际工程问题，矩阵指数使用 Taylor 级数展开到 4 阶，N 取 20 是非常保守的做法，可以足够保证精度[4]。因此在代码编写中也采用这个取法。首先根据加法定理，可以将积分区间$\left[t_k, t_{k+1}\right]$进一步细分$2^N$份

$$T = exp(H \times \Delta t) = [exp(H \times \Delta t / m)]^m, m = 2^N \tag{5-1}$$

在每一份小的积分区间上对矩阵指数进行泰勒展开到四阶

$$exp(H\Delta\tau) \approx I + H\Delta\tau + \frac{(H\Delta\tau)^2}{2!} + \frac{(H\Delta\tau)^3}{3!} + \frac{(H\Delta\tau)^4}{4!} = I + T_a \tag{5-2}$$

$$\Delta\tau = \frac{\Delta t}{2^N}, T_a = H\Delta\tau + \frac{(H\Delta\tau)^2}{2}[I + \frac{H\Delta\tau}{3} + \frac{(H\Delta\tau)^2}{12}] \tag{5-3}$$

由于 Ta 为小量矩阵，与单位阵 I 相加后会成为尾数，如果直接利用(5-1)式求解矩阵 T 则会带来很大的误差。所以使用循环求解增量矩阵，伪代码如下所示：

For(iter = 0;iter<N;iter++)

$T_a = 2T_a + T_a \times T_a;$

end

$T = I + T_a;$

将矩阵指数的求解封装为函数：[T] = MatrixIndex(H,dert_t,N)，供求解 Duhamel 积分调用。

5.1.1.2 三节点 Gauss 积分公式求解 Duhamel 积分

根据文献调研结果，采用三节点 Gauss 积分公式对 Duhamel 积分进行直接求解，三节点 Gauss 积分公式代数精度为 5 阶。3 节点 Gauss 精细积分格式如下所示：

$$x_{k+1} = Tx_k + \frac{\Delta t}{2}\sum_{i=1}^{3}\omega_i T(\frac{\Delta t}{2}(1 - \xi_i))R(t_k + \frac{\Delta t}{2}(1 + \xi_i)) \tag{5-4}$$

其中：

$$\omega_1 = \frac{8}{9}, \xi_1 = 0$$

$$\omega_2 = \frac{5}{9}, \xi_2 = -\sqrt{0.6}$$

$$\omega_3 = \frac{5}{9}, \xi_3 = \sqrt{0.6}$$

同样，Duhamel 积分求解代码也封装为函数：[result]=integralGass(tk,tk1,H)。

5.1.1.3 利用非齐次项结构特点减小计算量

根据增广后的载荷向量 $R(t)$第一个非零元素的位置对矩阵指数 T 和载荷向量 R 进行分块。通过函数 find_unzero(R)返回载荷向量 R 的第一个非零元行号：

flag = find_unzero(R) ，因此矩阵 ***T*** 和向量 ***R*** 的乘积就可以降阶为：

T * R = T(: ,flag : 2*n) * R(flag : 2*n,:)

伪代码为：

```
For(iter = 1;iter<=3;iter++)
    flag = find_unzero(R(t));
    result = result + T(: ,flag : 2*n) *  R(flag : 2*n,:);
end
```

5.1.1.4 利用矩阵指数 *T* 的稀疏性减小计算量

在矩阵指数 ***T*** 求解时，每做一次矩阵乘法就对 ***Ta*** 矩阵的误差项进行清零，即恢复矩阵 ***Ta*** 的稀疏性。同时使用稀疏矩阵的存储方式和乘法算法可以显著的节省内存空间，加快计算速度。具体的求解流程如下：

（1）由于 ***H*** 为稀疏矩阵，将 ***H*** 按照稀疏矩阵存储；

（2）计算 $\boldsymbol{H}^{'} = \boldsymbol{H}\frac{\boldsymbol{\eta}}{\boldsymbol{2^N}}$

（3）计算 $\boldsymbol{T_a} = \boldsymbol{H\Delta\tau} + \frac{\boldsymbol{(H\Delta\tau)^2}}{\boldsymbol{2}}[\boldsymbol{I} + \frac{\boldsymbol{H\Delta\tau}}{\boldsymbol{3}} + \frac{\boldsymbol{(H\Delta\tau)^2}}{\boldsymbol{12}}]$

（4）将 ***Ta*** 矩阵稀疏化

（5）执行如下语句，即

```
for iter =1:N
    Ta = Ta*（Ta+2*I）;
    将 Ta 稀疏化；
end
```

（6）***Ta = Ta+I***;

由于 ***Ta*** 矩阵各部分的物理意义不同，因此稀疏化需要分块进行。***Ta*** 矩阵稀疏化的基本原理为：首先给定一个阈值，在 ***Ta*** 矩阵的迭代运算过程中，将小于阈值的元素判定为计算误差，进行归零操作，从而保证求解过程中 ***Ta*** 矩阵的稀疏性，***Ta*** 矩阵分块归零算法的流程如下所示：

（1）给定阈值 $\boldsymbol{\varepsilon}$=1E-25；

（2）***Ta*** 矩阵分块：

$$T_a = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}$$

（3）找到每块绝对值最大的元素：$a_{11}, a_{12}, a_{21}, a_{22}$；

（4）每块的归零阈值 $\varepsilon_{ii} = a_{ii} \times \varepsilon$；

（5）分块归零；

（6）再组装为整体 ***Ta*** 矩阵输出；

5.1.1.5 矩阵乘法算法的动态选择

在实际算例求解过程中我们发现，当矩阵的填充率超过 1%后，稀疏矩阵的专用乘法算法优势快速丧失，甚至计算时间超过了满矩阵乘法算法；同时从物理意义上考虑，由于扰动的传播，***T*** 矩阵填充率上升是必然的趋势；并且 ***T*** 矩阵的填充率跟具体问题有关，不同问题填充率的变化范围可能很大。所以需要编写能够根据填充率动态选择矩阵乘法算法的程序来适应不同填充率下 ***T*** 矩阵的迭代求解。

伪代码如下所示：

```
for iter =1:N
    if(Ta 的填充率<1%）
        使用稀疏矩阵乘法计算 Ta = Ta*（Ta+2*I）;
    else
        使用满矩阵乘法计算 Ta = Ta*（Ta+2*I）;
        同时之后的迭代计算全部使用满矩阵算法；
    end
    将 Ta 稀疏化；
end
```

当矩阵填充率小于 1%时使用稀疏矩阵乘法算法；当矩阵填充率超过 1%后，以后的矩阵乘法全部使用满矩阵算法。这样就可以选择最适合当前矩阵填充率的乘法算法，从而保证对于不同的求解问题或者不同的迭代步，算法始终保持在效率最高的状态，使得求解程序更具有普适性。

5.1.2 精细积分法求解程序编写过程中的问题研究

在实际算例的求解过程中，我们遇到了这样的问题：$\boldsymbol{T}$ 矩阵在循环迭代求解时中出现数据越界，结果全为 NaN 的情况。针对这一问题，对其原因进行分析。

Hamilton 体系下的动力学方程为：

$$\dot{\boldsymbol{x}} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{R}$$

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_3 \\ \boldsymbol{A}_2 & \boldsymbol{A}_4 \end{bmatrix}, \boldsymbol{R} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{Q} \end{bmatrix} \tag{5-5}$$

$$\boldsymbol{A}_1 = -\boldsymbol{M}^{-1}\boldsymbol{C} / \boldsymbol{2}, \boldsymbol{A}_2 = \boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{C} / \boldsymbol{4} - \boldsymbol{K}$$

$$\boldsymbol{A}_3 = -\boldsymbol{C}\boldsymbol{M}^{-1} / \boldsymbol{2}, \boldsymbol{A}_4 = \boldsymbol{M}^{-1}$$

以一个具体的算例为例来说明结果出现 NaN 的原因,考虑一个如图 5.1 所示的悬臂梁，材料选择为钢材，单位制选择为：t，N，mm。

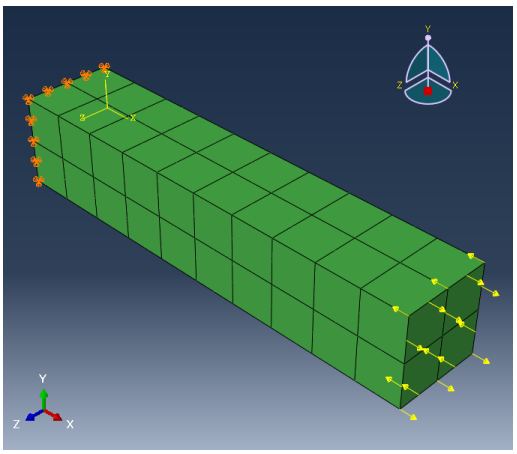


图 5.1 算例模型示意图

在该单位制下材料密度 $\boldsymbol{\rho = 7.8 \times 10^{-9} t / mm^3}$，杨氏模量 $\boldsymbol{E = 2 \times 10^5 N / mm^2}$。所以 $\boldsymbol{M}$ 矩阵元素为10^{-9}量级，$\boldsymbol{M}^{-1}$ 元素为10^9量级，K 矩阵元素为10^5量级。因此可以发现，是由于 $\boldsymbol{H}$ 矩阵中元素过大造成 $\boldsymbol{T}$ 矩阵迭代过程中的中间数据超过了计算机能存储的最大数，所以最终结果全部为 NaN。

为了寻求这一问题的解决办法，我们考虑系统动力学方程的另一种表达形式：

$$\dot{\boldsymbol{x}} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{R}$$

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -\boldsymbol{M}^{-1}\boldsymbol{K} & -\boldsymbol{M}^{-1}\boldsymbol{C} \end{bmatrix}, \boldsymbol{R} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{M}^{-1}\boldsymbol{Q} \end{bmatrix} \tag{5-6}$$

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{a} \\ \dot{\boldsymbol{a}} \end{bmatrix}$$

对于上方提到的算例，如果采用这种方程形式求解，$\boldsymbol{H}$ 矩阵的元素将会达到 10^{14} 量级，进一步加剧数据越界的问题。

下面对 $\boldsymbol{M}^{-1}\boldsymbol{K}$ 进行量纲分析：

如果采用 t，N，mm 的单位制，$\boldsymbol{M}^{-1}\boldsymbol{K}$ 的单位为：

$$\frac{\boldsymbol{N}\times\boldsymbol{mm}^{-2}}{\boldsymbol{t}\times\boldsymbol{mm}^{-3}}=\frac{\boldsymbol{N}\times\boldsymbol{mm}}{\boldsymbol{t}} \tag{5-7}$$

如果采用国际单位制，即 kg，N，m 的单位制，$\boldsymbol{M}^{-1}\boldsymbol{K}$ 的单位为：

$$\frac{\boldsymbol{N}\times\boldsymbol{m}^{-2}}{\boldsymbol{kg}\times\boldsymbol{m}^{-3}}=\frac{\boldsymbol{N}\times\boldsymbol{m}}{\boldsymbol{kg}} \tag{5-8}$$

这两个单位之间的换算关系为：

$$\frac{\boldsymbol{N}\times\boldsymbol{m}}{\boldsymbol{kg}}=\boldsymbol{10}^{6}\times\frac{\boldsymbol{N}\times\boldsymbol{mm}}{\boldsymbol{t}} \tag{5-9}$$

这说明式(5-7)为小单位，式(5-8)为大单位。对于相同的模型，如果采用 t，N，mm 的单位制，则 $\boldsymbol{M}^{-1}\boldsymbol{K}$ 的数值会非常大，而如果采用国际单位制就可以显著改善这一问题。

综合以上分析说明，要合理选择单位制，使得程序运算的中间结果不过大或者过小，这样有利于保证计算的稳定性。

5.2 其他时间积分方法

5.2.1 模态叠加法

模态叠加法使用了阵型的正交性，假设系统的质量阵为 $\boldsymbol{M}$，刚度阵为 $\boldsymbol{K}$，阵型矩阵和谱矩阵分别为 $\boldsymbol{\Phi}$ 和 $\boldsymbol{\Lambda}$。模态叠加法将直角坐标 x 变换到阵型坐标 a 下：

$$x=\boldsymbol{\Phi}a$$

将运动方程 $\boldsymbol{M}\ddot{x}+\boldsymbol{C}\dot{x}+\boldsymbol{K}x=\boldsymbol{Q}$ 转化为：

$$\boldsymbol{\Phi}^T\boldsymbol{M}\boldsymbol{\Phi}\ddot{a}+\boldsymbol{\Phi}^T\boldsymbol{C}\boldsymbol{\Phi}\dot{a}+\boldsymbol{\Phi}^T\boldsymbol{K}\boldsymbol{\Phi}a=\boldsymbol{\Phi}^T\boldsymbol{Q}$$

由于阵型的正交性，当使用瑞利阻尼时，上式中质量阵、阻尼阵和刚度阵均会被对角化，即在阵型坐标系下，方程解耦，变为 n 个二阶常微分方程：

$$\ddot{a}_i+2\omega_i\zeta_i\dot{a}_i+\omega_i^2a_i=r_i$$

每个常微分方程都相当于一个单自由度系统振动方程，在 MATLAB 中调用 ode45 函数求解。

在本大作业的程序中，因为已经求得了系统前六阶模态（见第 4 章），因此模态叠加法也只是用前六阶模态参与计算。这是因为在载荷频率较低时，系统高频响应的幅值很小，因此只用低阶模态计算得到的响应已经非常准确，从而能够大幅提高计算效率。在算例分析中，发现模态叠加法的计算速度很快，且计算结果与精细积分方法相比，没有高频波动，整体符合的很好。

5.2.2 速度 Verlet 法

速度 Verlet 法是一种常用的显示积分格式。假设系统的位矢为 a，其算法为：

1. 根据初始条件 $a_0,\dot{a}_0$，利用 0 时刻的运动方程求解 $\ddot{a}_0$；

2. 对每一个时间步 $t(t=\Delta t,2\Delta t,3\Delta t,...)$

(1) 计算 $t+\frac{1}{2}\Delta t$ 时刻的速度 $\dot{a}_{t+\frac{1}{2}\Delta t}$：

$$\dot{a}_{t+\frac{1}{2}\Delta t}=\dot{a}_t+\frac{1}{2}\ddot{a}_t$$

(2) 计算 $t+\Delta t$ 时刻的位矢 $a_{t+\Delta t}$：

$$a_{t+\Delta t}=a_t+\dot{a}_{t+\frac{1}{2}\Delta t}$$

(3) 由 $t+\Delta t$ 时刻的运动方程计算 $t+\Delta t$ 时刻的加速度：

$$\ddot{a}_{t+\Delta t}=\boldsymbol{M}\backslash\left(\boldsymbol{Q}_{t+\Delta t}-\boldsymbol{K}a_{t+\Delta t}\right)$$

(4) 更新 $t+\Delta t$ 时刻的速度：

$$\dot{a}_{t+\Delta t}=\dot{a}_{t+\frac{1}{2}\Delta t}+\frac{1}{2}\ddot{a}_{t+\frac{1}{2}\Delta t}\Delta t$$

中心差分法形式简单，计算速度快，但它是条件稳定的，其时间步长需要小

于临界时间步长，即

$$\Delta t \le \Delta t_{cr} = \frac{T_n}{\pi}\left(\sqrt{1+\zeta^2}-\zeta\right)$$

其中T_n为系统的最小固有周期。当系统自由度很高时，最高阶固有频率很高，最小固有周期很小，因此时间步长必须很小才能保证结果稳定。在计算算例时发现，其结果与精细积分法相同，但如果自由度很高，可能会导致步长过长，存储位矢、速度、加速度的数组过大使得内存不足。

5.2.3 广义α法

广义α法的位矢计算公式为：

$$\hat{\boldsymbol{K}} a_{t+\Delta t} = \hat{\boldsymbol{Q}}_{t+\Delta t}$$

其中

$$\hat{\boldsymbol{K}} = c_k \boldsymbol{K} + c_0 \boldsymbol{M} + c_1 \boldsymbol{Q}$$

$$\hat{\boldsymbol{Q}}_{t+\Delta t} = \boldsymbol{Q}_{t+(1+\alpha_f)\Delta t} - \alpha_f \boldsymbol{K} a_t + \boldsymbol{M}\left(c_0 a_t + c_2 \dot{a}_t + c_3 \ddot{a}_t\right) + \boldsymbol{C}\left(c_1 a_t + c_4 \dot{a}_t + c_5 \ddot{a}_t\right)$$

$$c_k = 1-\alpha_f, c_0 = \frac{1-\alpha_m}{\beta \Delta t^2}, c_1 = \frac{c_k \gamma}{\beta \Delta t}, c_2 = \Delta t c_0$$

$$c_3 = \frac{c_2 \Delta t}{2} - 1, c_4 = c_k \frac{\gamma}{\beta} - 1, c_5 = c_k\left(\frac{\gamma}{2\beta} - 1\right)\Delta t$$

取谱半径$\rho_\infty = 1$，$\alpha_f = \frac{\rho_\infty}{\rho_\infty + 1}$，$\alpha_m = \frac{2\rho_\infty - 1}{\rho_\infty + 1}$，时间步长取已经求得的第六阶固有频率对应的周期的 0.8 倍。计算结果发现，这种参数选择的结果与精细积分法相同，且计算速度较快。

5.3 算例验证

一、算例 1：

算例 1 采取选自文献的七自由度系统

图 1 所示为一七自由度体系，其运动方程为

$$M\ddot{X} + C\dot{X} + KX = F(t) \qquad (21a)$$

$$X(0) = 0 \quad \dot{X}(0) = 0 \qquad (21b)$$

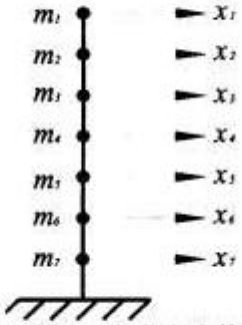


图 1　七自由度体系

Fig.1　A seven DOF system

其中，质量矩阵、阻尼矩阵、刚度矩阵分别为

$$M = \begin{bmatrix} 153.0 & & & & & & \\ & 170.0 & & & & & \\ & & 170.0 & & & & \\ & & & 170.0 & & & \\ & & & & 170.0 & & \\ & & & & & 170.0 & \\ & & & & & & 183.0 \end{bmatrix}$$

$$C = \begin{bmatrix} 3510 & -5780 & 1830 & 425 & 121 & 29.6 & 8.74 \\ -5780 & 13500 & -8950 & 1090 & 215 & 69.4 & 16.5 \\ 1830 & -8950 & 14500 & -8740 & 1160 & 231 & 78.5 \\ 425 & 1090 & -8740 & 14600 & -8720 & 1170 & 251 \\ 121 & 215 & 1160 & -8740 & 14600 & -8710 & 1230 \\ 29.6 & 69.4 & 231 & 1170 & -8710 & 14600 & -8430 \\ 8.74 & 16.5 & 78.5 & 251 & 1230 & -8430 & 13900 \end{bmatrix}$$

$$K = 10^4 \begin{bmatrix} 204 & -350 & 111 & 25.7 & 7.30 & 1.79 & 0.529 \\ -350 & 808 & -542 & 66.2 & 13 & 4.2 & 1 \\ 111 & -542 & 870 & -529 & 70.3 & 14 & 4.75 \\ 25.7 & 66.2 & -529 & 874 & -528 & 70.7 & 15.2 \\ 7.30 & 13 & 70.3 & -528 & 875 & -527 & 74.6 \\ 1.79 & 4.20 & 14 & 70.7 & -527 & 876 & -510 \\ 0.529 & 1 & 4.75 & 15.2 & 74.6 & -510 & 833 \end{bmatrix}$$

计算结果如下：

七自由度体系在动力载荷作用下的顶层位移反应

时间/s	0.2	0.6	1.0	最大相对误差
精确解	0.004269	0.011925	0.002571	0
精细积分	0.004270	0.011920	0.002572	0.042%

二、算例 2

算例 2 沿用静力学分析中的悬臂梁结构，采用 40 个 C3D8 单元，力载荷改为恒力情况。

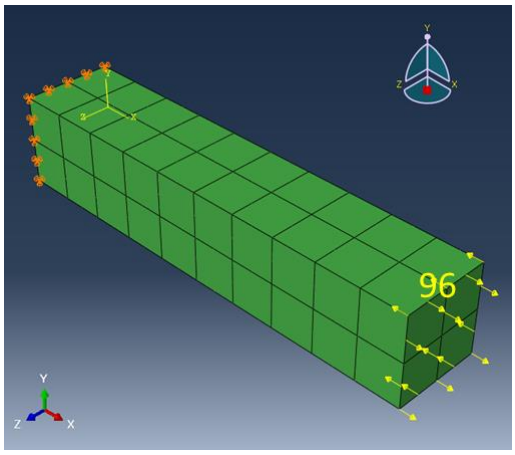


图 5.2 算例 2 网格划分

对比第 96 号节点在 y 方向的位移情况，各不同方法的计算结果如下：

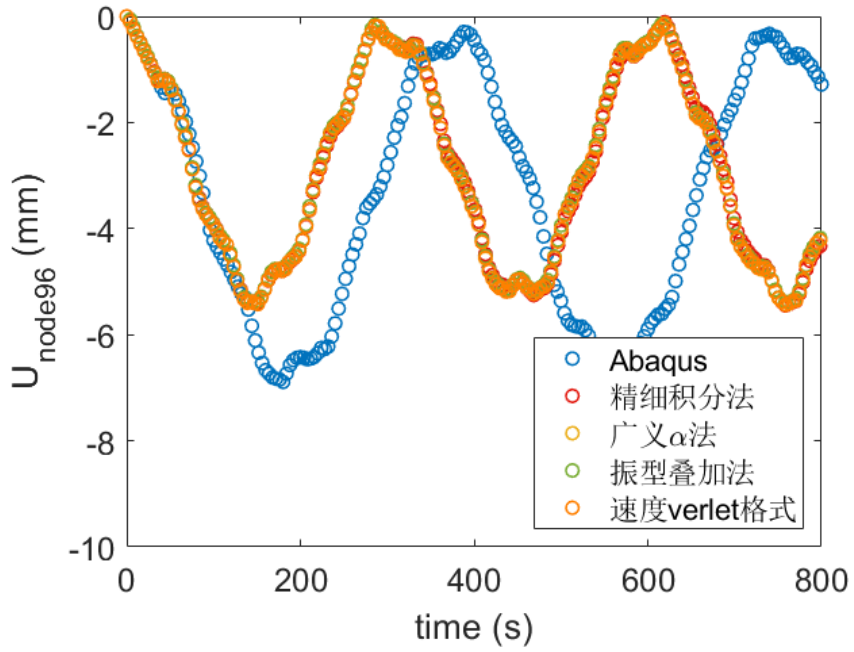


图 5.3 算例 2 各方法计算结果对比

分析：可以看到处 abaqus 的计算结果外，其余结果的吻合度较高。将 abaqus 的计算的刚度阵和质量阵输出后，发现主要是 abaqus 的刚度阵相差较多。

三、算例 3

算例 3 中将算例 2 的单元类型改为 C3D20

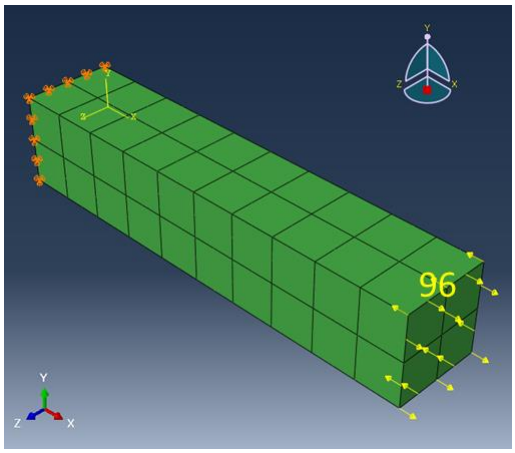


图 5.4 算例 3 网格划分

对比第 96 号节点在 y 方向的位移情况，各不同方法的计算结果如下，

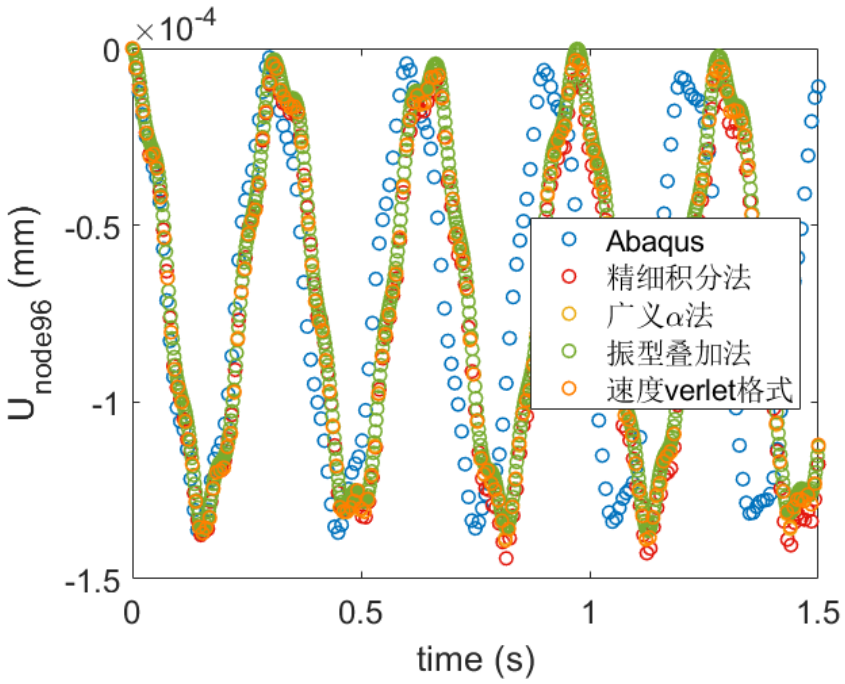


图 5.5 算例 3 各方法计算结果对比

分析：可以看到，相比于 C3D8 单元，C3D20 单元 abaqus 采用的质量阵与其余方法采用的质量阵相差较小。

四、算例四

算例四将算例三中的网格加密至 320 个单元。

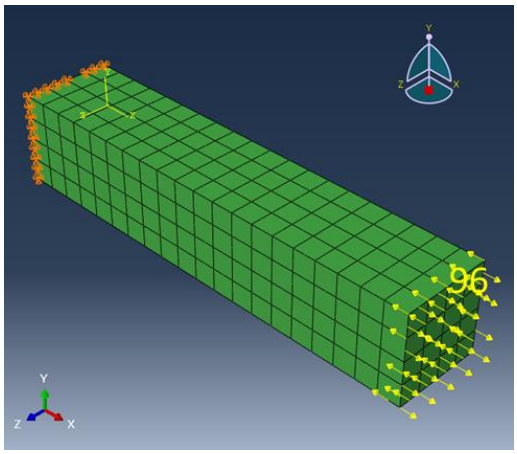


图 5.6 算例 4 网格划分

依旧对比第 96 号节点 y 轴方向的位移的时程曲线，对比各不同方法：

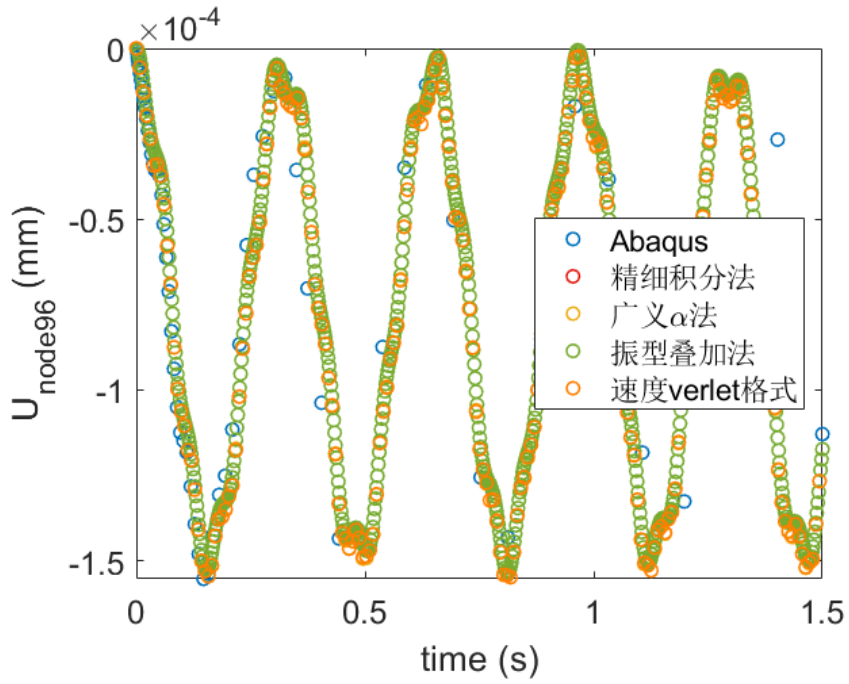


图 5.7 算例 4 各方法计算结果对比

分析：可以看到各方法的吻合度随着网格密度的增大而有所提升。

第6章 自选结构分析

6.1 自选结构介绍与模型简化

本次大作业选取实验室某振动模态实验结构，该自选结构的外形及尺寸见图 6.1 图 6.2，该结构的四根立柱材料为钢，三块平板材料为铝。

图 6.1 自选结构实物图

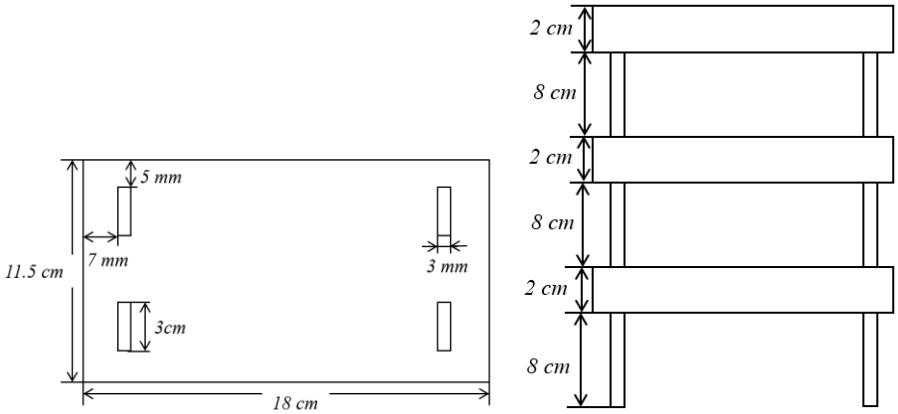


图 6.2 自选结构尺寸图

6.2 模态分析

该模型在理论上可以简化为三质点三弹簧结构，也可以简化为梁结构，如图 6.3 所示为梁结构的原型和前三阶模态。

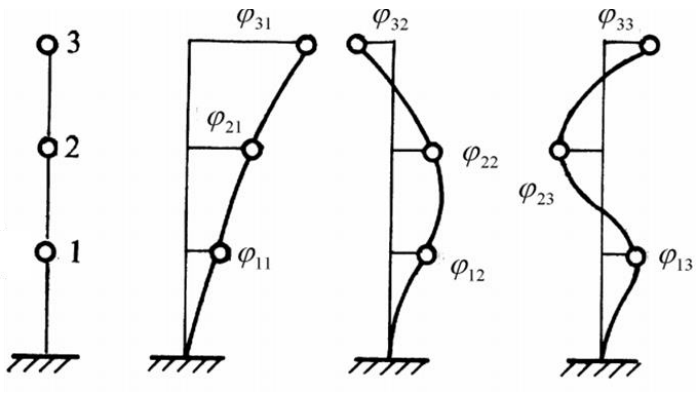


图 6.3 梁模型的原型和前三阶模态

该模型使用 C3D20 单元，如图 6.4 所示，主要加密的是钢片轴向的网格，每一段是 12 个单元。整体模型一共为 315 个单元，3065 个节点。

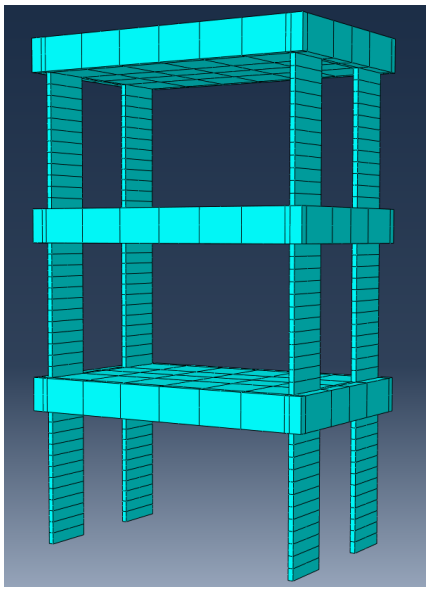


图 6.4 自选结构网格分布

如图 6.5 所示，为计算的模态结果，经过和简化后的梁机构的模态对比可知，构型一致。

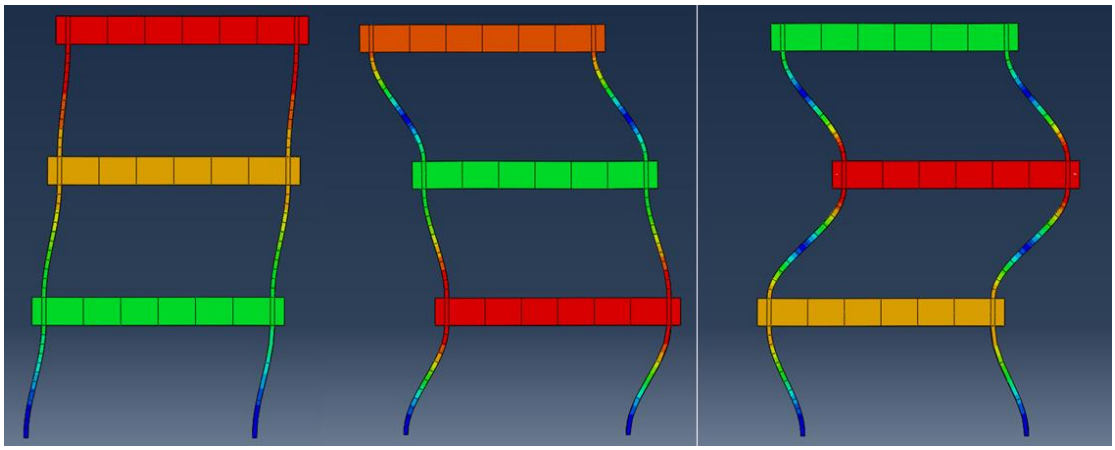


图 6.5 abaqus 仿真模态结果

模态	Stapmat 计算结果（Hz）	Abaqus 仿真结果（Hz）
1	68.71	72.87
2	193.56	205.36
3	281.55	298.79

表 6.1 自选结构特征值计算结果与仿真结果比较

6.3 动力学分析

对自选结构的动力学分析使用了两种载荷工况。如图 6.6 所示，为施加载荷的作用位置，即顶部第一块铝板短边面沿 x 轴负方向施加均布载荷。分析顶部第 254 号节点沿载荷方向的位移的时程曲线。

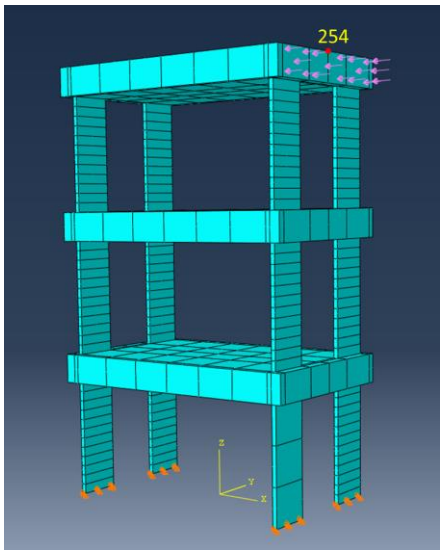


图 6.6 载荷施加区域与方向

恒定载荷

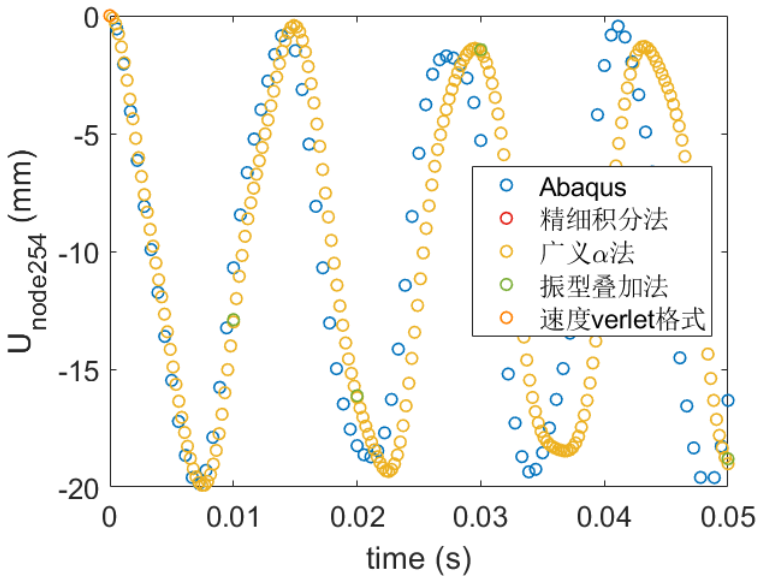


图 6.7 第 254 号节点的时程曲线(恒定载荷)

正弦载荷

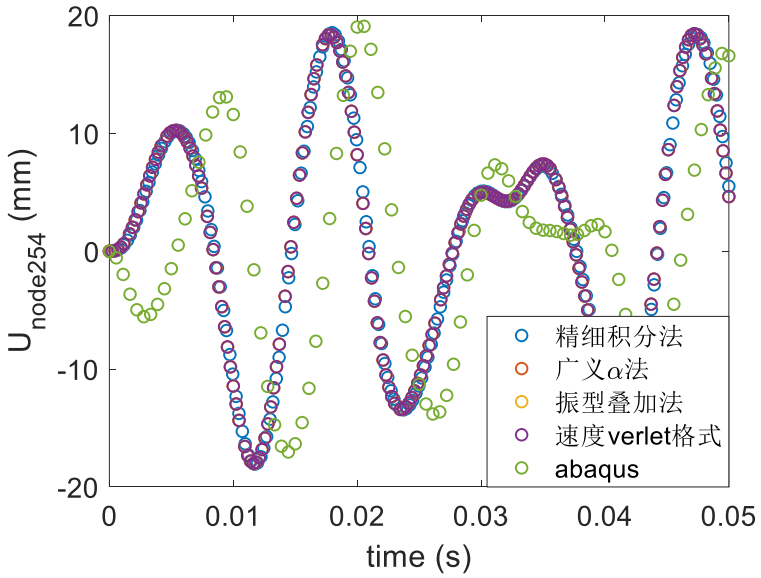


图 6.8 第 254 号节点的时程曲线(正弦载荷)

分析：可以看到除 abaqus 之外，其余方法的计算结果较为吻合。位移幅值也近乎相似，主要区别在于频率。频率相差的原因也在于质量阵与 abaqus 的不同。

第7章 总结与体会

7.1 总结与分工

在本次大作业中，我们完成了以下工作：

文献调研模块：

1： 调研精细积分法的改进方案 （郭嘉鑫）

程序改写模块：

1： 增加一种高阶三维实体单元：一种线性单元、一种高阶单元:

增加 C3D8 和 C3D20 单元，计算位移（林子雄）

应力磨平，Abaqus 仿真验证，导出输入文件（王婉婷）

2：增加特征值求解的功能（林子雄）

3：增加时间积分的功能

编写精细积分法求解结构动力学响应的基本求解代码（郭嘉鑫）

编写精细积分法算法提速代码，以及动态选择乘法算法的改进代码（郭嘉鑫）

增加广义 alpha、振型叠加、速度 verlet 法，代码合并（林子雄）

Abaqus 仿真验证，导出 KCMQ 矩阵和 stapmat 输入文件（王婉婷）

4： 结果输出成后处理软件可以读取的形式（王婉婷）

实际问题分析模块：

1：选择生活或工程中一个较复杂结构，采用三维实体单元，用 STAP 程序对其进行瞬态响应分析（林子雄&郭嘉鑫&王婉婷）

7.2 收获与体会

郭嘉鑫：这是我第一次接触有限元程序的编写，对于力学专业的学生，有限元分析和编程能力是最为重要。通过完成这次大作业，一方面夯实了我的有限元理论基础，从最底层的物理和算法上理解了有限元问题；另一方面锻炼了我的程序编写能力，在 debug 的过程中编程能力得到了飞速的提升。《计算动力学》这门课程是动力学与控制专业学生最应当扎实学习的课程，后续我也一定会进一步加强理论基础和编程能力训练。

王婉婷：经过这次大作业的锻炼，首先是加深了对理论知识的理解，比如高斯积分阶次对节点应力磨平结果的影响、单元内节点编号与总体节点编号的对应关系、低阶单元与高阶单元适用的不同问题类型、时间积分总时长和时间步长的量级估计等等；其次是锻炼了对算例管理的能力，我在本次大作业中主要是负责 abaqus 与 stapmat 计算结果验证的工作，不同 model 和 job 的命名如何清晰有条理，不同输入文件之间格式的转化，不同输出文件的结果对比等，都在随着大作业的进程中不断完善；最后是对 abaqus command 和 tecplot 的使用的熟练程度增加了，在这之前我主要使用 abaqus cae 界面，但在阅读 abaqus 手册后发现 cae 支持的功能不如 abaqus command 多，比如导出刚度阵质量阵等操作仅支持在 command 中实现，所以经过大作业使得对各类文本型输入文件的格式有了基本的了解，不再依赖于图形界面。总的来说，还是颇有收获的。

林子雄：在这次大作业中，我的主要工作为添加单元、计算模态，并添加了几种在课上讲过的时间积分方法，因此编写了大量程序，编程过程中也出现了很

多问题，但最终发现大多数问题都是由于理论理解不够深入造成的，解决这些问题的过程，使我不仅加深了对已经学习过的理论的理解，同时为了实现不同的功能，也学习了一些之前不了解的理论知识。同时，大量的编程工作也提高了编程能力。总之，我在这次大作业中既加强了理论知识，又提高了编程能力，收获颇丰。

参考文献

[1] 张雄, 王天舒, 刘岩. 计算动力学（第 2 版）[M]. 清华大学出版社, 2015.

[2] 钟万勰.暂态历程的精细计算方法[J].计算结构力学及其应用，1995，12(1):1-6.

[3] 高强,谭述君,钟万勰. 精细积分方法研究综述[J]. 中国科学（技术科学）,2016,46(12):1207-1218.

[4] 谭述君,吴志刚,钟万勰. 矩阵指数精细积分方法中参数的自适应选择[J]. 力学学报,2009,41(6):961-966.

[5] 储德文,王元丰. 精细直接积分法的积分方法选择[J]. 工程力学,2002,19(6):115-119.

[6] 汪梦甫,周锡元. 结构动力方程的高斯精细时程积分法[J]. 工程力学,2004,21(4):13-16.

[7] 张继锋,邓子辰,胡伟鹏. 结构动力方程精细直接积分法的简化计算[J]. 动力学与控制学报,2008,6(2):107-111.

[8] 张继锋,邓子辰,徐方暖,等. 一种新的改进精细直接积分法[J]. 动力学与控制学报,2015(4):241-245.

[9] 高强,吴锋,张洪武,等. 大规模动力系统改进的快速精细积分方法[J]. 计算力学学报,2011,28(4):493-498.

[10] 谭述君,钟万勰. 非齐次动力方程 Duhamel 项的精细积分[J]. 力学学报,2007,39(3):374-381.

[11] 徐建新,郭巧荣,卿光辉. 可分型指数矩阵的快速精细积分法[J]. 动力学与控制学报,2010,8(1):24-28.

[12] 吴泽艳,王立峰,武哲. 大规模动力系统高精度增维精细积分方法快速算法[J]. 振动与冲击,2014(2):188-192.

[13] 张森文,曹开彬. 计算结构动力响应的状态方程直接积分法[J]. 计算力学学报,2000,17(1):94-97,118.

[14] 顾元宪,陈飚松,张洪武. 结构动力方程的增维精细积分法[J]. 力学学

报,2000,32(4):447-456.