2025 年 第 25 卷 第 10 期
2025, 25(10): 04265-09

科 学 技 术 与 工 程
Science Technology and Engineering

ISSN 1671—1815
CN 11—4688/T

DOI:10.12404/j.issn.1671-1815.2403703
引用格式:董正方, 王佳豪, 李运华, 等. 考虑约束的 CSC 格式总体矩阵存储算法[J]. 科学技术与工程, 2025, 25(10): 4265-4273.
Dong Zhengfang, Wang Jiahao, Li Yunhua, et al. CSC format global matrix storage algorithm considering constraints[J]. Science Technology and Engineering, 2025, 25(10): 4265-4273.

考虑约束的 CSC 格式总体矩阵存储算法

董正方[1], 王佳豪[1], 李运华[1*], 程纬[2], 王君杰[2]
(1. 河南大学建筑工程学院, 开封 475004; 2. 同济大学桥梁工程系, 上海 200092)

摘 要 为了提高有限元分析效率,减少内存消耗,对总体矩阵的存储算法进行研究。将约束条件统一处理为单元层面的矩阵分块,后续以刚度矩阵为例,确定了约束条件下刚度矩阵非零元素分布规律,并提出了适用于二维、三维有限元刚度矩阵非零元素数量计算方法及相应计算公式,同时检验了正确性;推导了一维等带宽存储与原始方阵地址对应关系,并将其作为辅助数组应用于 CSC 格式总体矩阵集成过程中,提高了总体矩阵组集效率,避免了节点编号优化难题。最后,以北京市房山线长稻区间四跨单线梁桥验证所提出算法的正确性与实用性,结果表明:该算法分析时总体矩阵组集时间减少 30%、存储空间节省 68% 以上、保证了计算精度的同时线性方程组计算效率提升超过 71%。

关键词 有限单元法; 刚度矩阵; 非零元素; 压缩存储; Cholesky 分解
中图法分类号 TU311.41; **文献标志码** A

CSC Format Global Matrix Storage Algorithm Considering Constraints

DONG Zheng-fang[1], WANG Jia-hao[1], LI Yun-hua[1*], CHENG Wei[2], WANG Jun-jie[2]
(1. School of Civil Engineering and Architecture, Henan University, Kaifeng 475004, China;
2. Department of bridge engineering, Tongji University, Shanghai 200092, China)

[**Abstract**] In order to improve the efficiency of finite element analysis and reduce memory consumption, the storage algorithm of the overall matrix was studied. The constraints were uniformly processed as matrix partitions at the element level. Subsequently, taking the stiffness matrix as an example, the distribution pattern of non-zero elements under the constrained condition was determined. A calculation method and corresponding formulas suitable for the number of non-zero elements in the stiffness matrix of two-dimensional and three-dimensional finite elements were proposed, and the correctness was verified. The correspondence between one-dimensional equal bandwidth storage and the original square matrix address was deduced, and it was applied as an auxiliary array in the process of integrating the overall matrix in CSC format, which has improved the efficiency of the overall matrix assembly and avoided the difficulty of node numbering optimization. Finally, the correctness and practicability of the proposed algorithm were verified by using thefour-span single-line girder bridge in the Changdao section of the Fangshan Line in Beijing. The results show that the overall matrix assembly time is reduced by 30%, the storage space is saved by more than 68%, and the calculation efficiency of linear equations is improved by more than 71% while ensuring the calculation accuracy.

[**Keywords**] finite element method; stiffness matrix; non-zero elements; compressed storage; cholesky decomposition

随着工程技术的快速发展和数值计算方法的不断研究,有限元法逐渐成为数值仿真的主流方法。近年来,工程规模不断扩大,所需要分析的工程问题日益复杂,人们必须考虑内存资源占用与计算时间成本的影响,如何进一步提高有限元分析效率已成为人们关注的热点。有限元分析最终归结为求解一个大型线性方程组,其求解效率取决于线性方程组的解法和储存结构[1-3]。现有的一些存储方法,如等带宽存储、变带宽存储等都受到节点编号的影响,若节点编号设置不合理则会增加有限元分析时间和内存开销;若对节点编号进行优化,则会耗费一定的时间[4-5]。

目前,节省内存开销和提高分析效率主要有两个方面。一方面是减少存储数量,只存储非零元素。通常根据 ID 表法、细胞元和广义相邻节点对的概念确定总体刚度矩阵中的非零元素数量和位置[6-8];这些方法避免了节点编号优化的复杂问题,但需根据单元模型的拓扑结构形成索引数组,增加

收稿日期: 2024-05-19; **修订日期**: 2025-01-13
基金项目: 中国博士后科学基金(2024M750780);中建六局科技研发计划(CSCEC6B-2024-Z-7)
第一作者: 董正方(1980—),男,汉族,河南滑县人,博士,教授。研究方向:有限元软件;桥梁抗震。E-mail:dzf@henu.edu.cn。
* **通信作者**: 李运华(1980—),男,汉族,河南项城人,硕士,高级实验师。研究方向:材料学。E-mail:24561591@qq.com。

了计算时间;并且这些方法适用性较差,仅适用于实体单元、特定单元分析以及没有复杂约束的模型。另一方面是改进压缩存储方法和技术,如一维变带宽、三元组、十字链表法[9]等。然而,随着模型复杂程度的增加,这些方法无法满足工程计算人员对内存资源的要求。因此,列压缩(compressed sparse column, CSC)和行压缩(compressed sparse row,CSR)成为主流[10]。为了进一步优化内存开销,研究人员又提出了多种新的压缩存储技术,包括改进一维变带宽存储、MSR(modified sparse row)、改进 S(B)CSR[11][streamed (block) compressed sparse row]和负号 CSR 存储[12]等;后续随着计算机技术的发展,并行计算、机器学习与存储技术及矩阵运算的结合也成为实现高效率存储与高效率计算的主要手段[13-16];尽管如此,但上述方法仍有不足,如改进 S(B)CSR 在稀疏矩阵向量乘法(sparse matrix-vector multiplication, SpMV)和稀疏矩阵处理上表现优异,但局限性强,需要特殊的工具与方法;负号 CSR 存储目前只适用于三维有限元;而与并行计算、机器学习结合的方法在稀疏矩阵处理上仍有欠缺,主要在 SpMV 计算上表现优异等。

为提升有限元分析的效率并降低其在分析过程中的内存需求,现从自由度的角度出发,对受约束条件影响的总体矩阵进行分块,消除所有约束自由度,缩减矩阵规模;后续以刚度矩阵为例,根据其组成特点,提出一种适用于二维、三维刚度矩阵非零元素确定方法,并给出通用计算公式;将等带宽移植原则与 CSC 存储相结合,仅存储总体刚度矩阵中的非零元素,可以高效地直接集成 CSC 压缩格式的总体刚度矩阵。

1 总体矩阵非零元素确定

1.1 考虑约束的总体矩阵分块

在有限元分析中,结构的约束形式及其处理方式对于在模拟实际物理行为时至关重要。常见的约束形式及其处理方式如表 1 所示。

对于复杂的结构形式及不同的边界条件或约束形式,节点位移形态迥异,为了有效地处理这类问题,钟万勰[25]提出了位移规格数的概念,根据节点位移自由度将其分为 5 种规格。为了明确辨析地震作用的影响,对几何可动自由度进一步分类为独立自由度与地震动自由度、将受主从约束及非线性约束影响的自由度归类为从自由度,同时将受 Dirichlet 边界限制的自由度细分为固定自由度与指定位移自由度,共 5 种规格。

表 1 约束形式及其处理方式

Table1 Constraint forms and their handling methods

类型	主要形式	处理方式
Dirichlet 边界[17]	固定约束	直接消去法
	指定位移	对角元素改 1,其他元素置 0
地震动作用	地震作用的施加(一致激励、非一致激励、人工边界地震动输入)	本质上是一种运动着的边界约束条件,直接代入计算
主从约束	单级约束,(刚性约束、柔性约束)多级约束[18],时变系数约束	将从节点特性转换到主节点上[18-19],通常使用罚函数法[20]、拉格朗日乘子法[21]
非线性约束	大位移大变形问题,曲线或曲面滑动支承,接触碰撞问题	转换为单级主从约束方程;使用罚函数法、拉格朗日乘子法处理等[22-24]

由表 1 可知,处理固定、指定位移、从自由度等约束时,通常采用直接消去法、罚函数法、拉格朗日乘子法等在总体层面对矩阵进行处理。然而,对于大型矩阵,在总体层面进行处理时会出现工作繁琐、计算量大的情况。针对于此,采用在单元层面统一将约束条件处理转换为单元层面的矩阵分块与总体矩阵分块,在装配总体矩阵时将所有约束自由度消除。假设总体矩阵为对称矩阵,单元的全局位移自由度向量为

$$\boldsymbol{\delta}_e = \{u_1 \quad u_2 \quad \cdots \quad u_n \quad\} \tag{1}$$

式(1)中:u_1、u_2、…、u_n 分别为单元自由度 1、2、…、n 的位移。

考虑一般情况,式(1)中位移元素包含了独立自由度、约束自由度(包括固定、指定位移、从自由度)和地震动自由度,经过多次行变换,即左乘转换矩阵 $\boldsymbol{H}^{\mathrm{T}}$,且 $\boldsymbol{H}^{\mathrm{T}} = \boldsymbol{H}^{-1}$,将自由度向量分类排序为

$$\boldsymbol{\delta}_e = \boldsymbol{H}\{\boldsymbol{\delta}_e^{\mathrm{I}} \quad \boldsymbol{\delta}_e^{\mathrm{C}} \quad \boldsymbol{\delta}_e^{\mathrm{G}}\}^{\mathrm{T}} \tag{2}$$

式(2)中:$\boldsymbol{\delta}_e^{\mathrm{I}}$、$\boldsymbol{\delta}_e^{\mathrm{C}}$、$\boldsymbol{\delta}_e^{\mathrm{G}}$ 分别单元的独立、约束和地震动自由度位移向量。

对于复杂的约束条件方程,一般的约束位移自由度可以表示为

$$\begin{aligned} u_j^{\mathrm{C}} = {} & (\alpha_{j1}u_1^{\mathrm{I}} + \alpha_{j2}u_2^{\mathrm{I}} + \cdots + \alpha_{ji}u_i^{\mathrm{I}}) + \\ & (\beta_{j1}u_1^{\mathrm{G}} + \beta_{j2}u_2^{\mathrm{G}} + \cdots + \beta_{ji}u_i^{\mathrm{G}}) + \\ & (\gamma_{j1}u_1^{\mathrm{P}} + \gamma_{j2}u_2^{\mathrm{P}} + \cdots + \gamma_{ji}u_i^{\mathrm{P}}) + \\ & (\eta_{j1}u_1^{\mathrm{F}} + \eta_{j2}u_2^{\mathrm{F}} + \cdots + \eta_{ji}u_i^{\mathrm{F}}) + v_j^{\mathrm{C}} \end{aligned} \tag{3}$$

式(3)中:u_i^{I}、u_i^{G}、u_i^{P}、u_i^{F} 分别为与约束位移自由度 j 相关的独立、地震动、固定约束及指定位移约束自由度位移;α_{ji}、β_{ji}、γ_{ji}、η_{ji} 为相应自由度关系系数,其中 $i = 1,2,\cdots,i$;v_j^{C} 为非其次项,这里是常数。固定

约束部分为零,所以,约束自由度有以下转换方式。

$$\boldsymbol{\delta}_j^{\mathrm{C}} = \begin{bmatrix} u_1^{\mathrm{C}} \\ u_2^{\mathrm{C}} \\ \vdots \\ u_k^{\mathrm{C}} \end{bmatrix} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1i} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2i} \\ \vdots & \vdots & & \vdots \\ \alpha_{k1} & \alpha_{k2} & \cdots & \alpha_{ki} \end{bmatrix} \begin{bmatrix} u_1^{\mathrm{I}} \\ u_2^{\mathrm{I}} \\ \vdots \\ u_k^{\mathrm{I}} \end{bmatrix} + \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1i} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2i} \\ \vdots & \vdots & & \vdots \\ \beta_{k1} & \beta_{k2} & \cdots & \beta_{ki} \end{bmatrix} \begin{bmatrix} u_1^{\mathrm{G}} \\ u_2^{\mathrm{G}} \\ \vdots \\ u_k^{\mathrm{G}} \end{bmatrix} + \begin{bmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1i} \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2i} \\ \vdots & \vdots & & \vdots \\ \gamma_{k1} & \gamma_{k2} & \cdots & \gamma_{ki} \end{bmatrix} \begin{bmatrix} u_1^{\mathrm{P}} \\ u_2^{\mathrm{P}} \\ \vdots \\ u_k^{\mathrm{P}} \end{bmatrix} + \begin{bmatrix} v_1^{\mathrm{C}} \\ v_2^{\mathrm{C}} \\ \vdots \\ v_k^{\mathrm{C}} \end{bmatrix}$$

$$= \boldsymbol{A}\bar{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{I}} + \boldsymbol{B}\bar{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{G}} + \boldsymbol{C}\bar{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{P}} + \boldsymbol{V}^{\mathrm{C}} \tag{4}$$

式(4)中:$\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$ 分别为约束独立自由度关系矩阵、约束地震动自由度关系矩阵和约束指定位自由度关系矩阵。

将式(4)代入式(2),单元位移自由度向量可表示为

$$\boldsymbol{\delta}_{\mathrm{e}} = \boldsymbol{H}\begin{bmatrix} \boldsymbol{\delta}_{\mathrm{e}}^{\mathrm{I}} \\ \boldsymbol{\delta}_{\mathrm{e}}^{\mathrm{C}} \\ \boldsymbol{\delta}_{\mathrm{e}}^{\mathrm{G}} \end{bmatrix} = \boldsymbol{H}\begin{bmatrix} \boldsymbol{I}_{\mathrm{I}} & \boldsymbol{O} \\ \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{O} & \boldsymbol{I}_{\mathrm{G}} \end{bmatrix}\begin{bmatrix} \bar{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{I}} \\ \bar{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{G}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{C}\bar{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{P}} + \boldsymbol{V}^{\mathrm{C}} \\ \boldsymbol{O} \end{bmatrix} \tag{5}$$

式(5):$\boldsymbol{O}$ 为零矩阵;$\boldsymbol{I}_{\mathrm{I}}$、$\boldsymbol{I}_{\mathrm{G}}$ 均为 eye 矩阵(不一定是方阵)。

将式(5)对时间求一阶导数与二阶导数,即

$$\begin{cases} \dot{\boldsymbol{\delta}}_{\mathrm{e}} = \boldsymbol{H}\begin{bmatrix} \dot{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{I}} \\ \dot{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{C}} \\ \dot{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{G}} \end{bmatrix} = \boldsymbol{H}\begin{bmatrix} \boldsymbol{I}_{\mathrm{I}} & \boldsymbol{O} \\ \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{O} & \boldsymbol{I}_{\mathrm{G}} \end{bmatrix}\begin{bmatrix} \dot{\bar{\boldsymbol{\delta}}}_{\mathrm{e}}^{\mathrm{I}} \\ \dot{\bar{\boldsymbol{\delta}}}_{\mathrm{e}}^{\mathrm{G}} \end{bmatrix} \\ \ddot{\boldsymbol{\delta}}_{\mathrm{e}} = \boldsymbol{H}\begin{bmatrix} \ddot{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{I}} \\ \ddot{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{C}} \\ \ddot{\boldsymbol{\delta}}_{\mathrm{e}}^{\mathrm{G}} \end{bmatrix} = \boldsymbol{H}\begin{bmatrix} \boldsymbol{I}_{\mathrm{I}} & \boldsymbol{O} \\ \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{O} & \boldsymbol{I}_{\mathrm{G}} \end{bmatrix}\begin{bmatrix} \ddot{\bar{\boldsymbol{\delta}}}_{\mathrm{e}}^{\mathrm{I}} \\ \ddot{\bar{\boldsymbol{\delta}}}_{\mathrm{e}}^{\mathrm{G}} \end{bmatrix} \end{cases} \tag{6}$$

对于动力问题,考虑瞬时势能泛函,即

$$\Pi_{\mathrm{e}} = \frac{1}{2}\boldsymbol{\delta}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{K}_{\mathrm{e}}\boldsymbol{\delta}_{\mathrm{e}} - \boldsymbol{\delta}_{\mathrm{e}}^{\mathrm{T}}(\boldsymbol{P}_{\mathrm{e}} - \boldsymbol{M}_{\mathrm{e}}\ddot{\boldsymbol{\delta}}_{\mathrm{e}} - \boldsymbol{C}_{\mathrm{e}}\dot{\boldsymbol{\delta}}_{\mathrm{e}}) \tag{7}$$

式(7)中:$\boldsymbol{K}_{\mathrm{e}}$、$\boldsymbol{M}_{\mathrm{e}}$、$\boldsymbol{C}_{\mathrm{e}}$ 分别为单元刚度、质量、阻尼矩阵;$\boldsymbol{P}_{\mathrm{e}}$ 为单元荷载向量。

将式(5)、式(6)代入式(7),根据动力最小势能原理[26]可得引入了约束条件后的单元刚度、质量、阻尼矩阵为

$$\begin{cases} \bar{\boldsymbol{K}}_{\mathrm{e}} = \begin{bmatrix} \boldsymbol{I}_{\mathrm{I}}^{\mathrm{T}} & \boldsymbol{A}^{\mathrm{T}} & \boldsymbol{O}^{\mathrm{T}} \\ \boldsymbol{O}^{\mathrm{T}} & \boldsymbol{B}^{\mathrm{T}} & \boldsymbol{I}_{\mathrm{G}}^{\mathrm{T}} \end{bmatrix}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{K}_{\mathrm{e}}\boldsymbol{H}\begin{bmatrix} \boldsymbol{I}_{\mathrm{I}} & \boldsymbol{O} \\ \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{O} & \boldsymbol{I}_{\mathrm{G}} \end{bmatrix} \\ \bar{\boldsymbol{C}}_{\mathrm{e}} = \begin{bmatrix} \boldsymbol{I}_{\mathrm{I}}^{\mathrm{T}} & \boldsymbol{A}^{\mathrm{T}} & \boldsymbol{O}^{\mathrm{T}} \\ \boldsymbol{O}^{\mathrm{T}} & \boldsymbol{B}^{\mathrm{T}} & \boldsymbol{I}_{\mathrm{G}}^{\mathrm{T}} \end{bmatrix}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{C}_{\mathrm{e}}\boldsymbol{H}\begin{bmatrix} \boldsymbol{I}_{\mathrm{I}} & \boldsymbol{O} \\ \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{O} & \boldsymbol{I}_{\mathrm{G}} \end{bmatrix} \\ \bar{\boldsymbol{M}}_{\mathrm{e}} = \begin{bmatrix} \boldsymbol{I}_{\mathrm{I}}^{\mathrm{T}} & \boldsymbol{A}^{\mathrm{T}} & \boldsymbol{O}^{\mathrm{T}} \\ \boldsymbol{O}^{\mathrm{T}} & \boldsymbol{B}^{\mathrm{T}} & \boldsymbol{I}_{\mathrm{G}}^{\mathrm{T}} \end{bmatrix}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{M}_{\mathrm{e}}\boldsymbol{H}\begin{bmatrix} \boldsymbol{I}_{\mathrm{I}} & \boldsymbol{O} \\ \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{O} & \boldsymbol{I}_{\mathrm{G}} \end{bmatrix} \end{cases} \tag{8}$$

将矩阵按自由度类型分块,即

$$\begin{cases} \bar{\boldsymbol{K}}_{\mathrm{e}} = \begin{bmatrix} \boldsymbol{K}_{\mathrm{II}}^{\mathrm{e}} & \boldsymbol{K}_{\mathrm{IG}}^{\mathrm{e}} \\ \boldsymbol{K}_{\mathrm{GI}}^{\mathrm{e}} & \boldsymbol{K}_{\mathrm{GG}}^{\mathrm{e}} \end{bmatrix} \\ \bar{\boldsymbol{C}}_{\mathrm{e}} = \begin{bmatrix} \boldsymbol{C}_{\mathrm{II}}^{\mathrm{e}} & \boldsymbol{C}_{\mathrm{IG}}^{\mathrm{e}} \\ \boldsymbol{C}_{\mathrm{GI}}^{\mathrm{e}} & \boldsymbol{C}_{\mathrm{GG}}^{\mathrm{e}} \end{bmatrix} \\ \bar{\boldsymbol{M}}_{\mathrm{e}} = \begin{bmatrix} \boldsymbol{M}_{\mathrm{II}}^{\mathrm{e}} & \boldsymbol{M}_{\mathrm{IG}}^{\mathrm{e}} \\ \boldsymbol{M}_{\mathrm{GI}}^{\mathrm{e}} & \boldsymbol{M}_{\mathrm{GG}}^{\mathrm{e}} \end{bmatrix} \end{cases} \tag{9}$$

式(9)中:$\boldsymbol{K}_{\mathrm{II}}^{\mathrm{e}}$ 为单元刚度矩阵中独立自由度模块;$\boldsymbol{K}_{\mathrm{GG}}^{\mathrm{e}}$ 为单元刚度矩阵中地震动自由度模块;$\boldsymbol{K}_{\mathrm{IG}}^{\mathrm{e}}$、$\boldsymbol{K}_{\mathrm{GI}}^{\mathrm{e}}$ 为单元刚度矩阵中地震动自由度对独立自由度影响模块。单元阻尼矩阵、单元质量矩阵与单元刚度矩阵分块相同。

$\bar{\boldsymbol{K}}_{\mathrm{e}}$、$\bar{\boldsymbol{C}}_{\mathrm{e}}$、$\bar{\boldsymbol{K}}_{\mathrm{e}}$ 都是方阵,阶数为单元独立自由度数与地震动自由度数总和。将各个单元矩阵块分类组装可以得到总体矩阵为

$$\begin{cases} \boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{\mathrm{II}} & \boldsymbol{K}_{\mathrm{IG}} \\ \boldsymbol{K}_{\mathrm{GI}} & \boldsymbol{K}_{\mathrm{GG}} \end{bmatrix} \\ \boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_{\mathrm{II}} & \boldsymbol{C}_{\mathrm{IG}} \\ \boldsymbol{C}_{\mathrm{GI}} & \boldsymbol{C}_{\mathrm{GG}} \end{bmatrix} \\ \boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_{\mathrm{II}} & \boldsymbol{M}_{\mathrm{IG}} \\ \boldsymbol{M}_{\mathrm{GI}} & \boldsymbol{M}_{\mathrm{GG}} \end{bmatrix} \end{cases} \tag{10}$$

式(10)中:$\boldsymbol{K}$、$\boldsymbol{M}$、$\boldsymbol{C}$ 分别为总体刚度、质量、阻尼矩阵。

对于静力问题,不考虑惯性力和阻尼力的影响,刚度矩阵表达式与动力问题相同。

综上所述,可以看出无论是静力问题还是动力问题,经过约束处理后,只有独立自由度、地震动自由度对单元矩阵和总体矩阵有贡献,一定程度上减少了总体矩阵的规模及相应存储需求。并且从刚度矩阵、质量矩阵、阻尼矩阵计算方式与分块形式上可以看出,3 种矩阵原理相同,后续以刚度矩阵为例。

1.2　非零元素确定

文献[7]采用“广义相邻节点对”确定了三维实体单元的非零子矩阵数量与位置,本文研究对其进

行推广。在同一单元内,任意两个节点是相关的,进而任意两个自由度也相关。单元刚度矩阵的每个矩阵数据可由这些“相关自由度”所确定。“相关自由度”数量与总体刚度矩阵上三角部分存储的非零元素数量一致,且一一对应。所以,通过分析自由度的关联性,可以准确地确定刚度矩阵非零元素的数量。一旦确定了非零元素的数据量,即可精确地分配总体矩阵的存储空间,这对于总体矩阵存储算法的实现具有重要意义。

由刚度矩阵特性可知,单元之间共同节点只影响矩阵数值,不会改变矩阵非零元素数量。因此,当模型由单个单元组成时,非零元素数量完全取决于该单元的自由度数量;如果考虑模型由多个单元组成时,单元与单元之间有共节点,非零元素数量将由单元与单元之间增加自由度数量确定。可得具体计算公式为

$$K_{\mathrm{II}}^{\mathrm{num}}=\begin{cases}\sum_{i=1}^{N}\dfrac{D_i(D_i+1)}{2}-\\ \quad\sum_{i=2}^{N}\dfrac{[D_i-n_{i(i-1)}^{D}][D_i-n_{i(i-1)}^{D}+1]}{2}, & N\neq 1\\ \dfrac{D_i(D_i+1)}{2}, & N=1\end{cases}\tag{11}$$

$$K_{\mathrm{GG}}^{\mathrm{num}}=\begin{cases}\sum_{i=1}^{N}\dfrac{G_i(D_i+1)}{2}-\\ \quad\sum_{i=2}^{N}\dfrac{[G_i-n_{i(i-1)}^{G}][G_i-n_{i(i-1)}^{G}+1]}{2}, & N\neq 1\\ \dfrac{G_i(G_i+1)}{2}, & N=1\end{cases}\tag{12}$$

$$K_{\mathrm{IG}}^{\mathrm{num}}=\begin{cases}\sum_{i=1}^{N}\dfrac{G_iD_i}{2}-\\ \quad\sum_{i=2}^{N}\dfrac{[D_i-n_{i(i-1)}^{D}][G_i-n_{i(i-1)}^{G}]}{2}, & N\neq 1\\ G_iD_i, & N=1\end{cases}\tag{13}$$

$$K_{\mathrm{Num}}=\begin{cases}\sum_{i=1}^{N}\dfrac{Y_i(Y_i+1)}{2}-\\ \quad\sum_{i=2}^{N}\dfrac{[Y_i-n_{i(i-1)}][Y_i-n_{i(i-1)}+1]}{2}, & N\neq 1\\ \dfrac{Y_i(Y_i+1)}{2}, & N=1\end{cases}\tag{14}$$

$$K_{\mathrm{Num}}=K_{\mathrm{II}}^{\mathrm{num}}+K_{\mathrm{GG}}^{\mathrm{num}}+K_{\mathrm{IG}}^{\mathrm{num}}\tag{15}$$

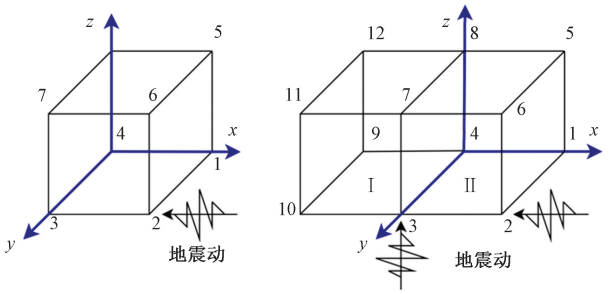
式中:$K_{\mathrm{xx}}^{\mathrm{num}}$ 为各子块非零元素数量;K_{Num} 为总体刚度矩阵的非零刚度矩阵元素数量;D_i、G_i 与 Y_i($Y_i=D_i+G_i$)为第 i 个单元的独立自由度、地震动自由度与或总自由度数量;$n_{i(i-1)}^{D}$、$n_{i(i-1)}^{G}$ 与 $n_{i(i-1)}$[$n_{i(i-1)}=n_{i(i-1)}^{D}+n_{i(i-1)}^{G}$]为第 i 个单元相较于前个 $i-1$ 单元增加的独立自由度、地震动自由度或总自由度数量;N 为单元数量。

为验证式(11)~式(15)的正确性,建立如图 1 所示的八节点六面体单元模型。各参数取值如表 2 所示。

由式(11)计算可得,$K_{\mathrm{II}}^{\mathrm{num1}}=276$、$K_{\mathrm{GG}}^{\mathrm{num1}}=1$、$K_{\mathrm{IG}}^{\mathrm{num1}}=3$、$K_{\mathrm{num1}}=300$;$K_{\mathrm{II}}^{\mathrm{num2}}=463$、$K_{\mathrm{GG}}^{\mathrm{num1}}=3$、$K_{\mathrm{IG}}^{\mathrm{num2}}=56$、$K_{\mathrm{num2}}=522$。文献[7]中,对于图 1 所示模型计算的非零元素数量分别为 300 和 522,上述计算结果与文献[7]完全一致。

进一步,为验证式(11)~式(15)的通用性,建立梁单元与平面单元连接的二维模型,如图 2 所示。各参数取值如表 3 所示。

由式(11)计算可得,$K_{\mathrm{II}}^{\mathrm{num}}=33$、$K_{\mathrm{GG}}^{\mathrm{num}}=3$、$K_{\mathrm{IG}}^{\mathrm{num}}=14$、$K_{\mathrm{num}}=50$。可以看出公式的计算结果与图 2 非零元素总数相同,验证了公式的正确性与通用性。



(a) 模型1 (b) 模型2

图 1 八节点六面体单元模型

Fig. 1 Eight-node hexahedral element model

表 2 模型各参数取值

Table 2 Parameter values of the model

模型	N	单元	Y_i	D_i	G_i	$n_{i(i-1)}$	$n_{i(i-1)}^{D}$	$n_{i(i-1)}^{G}$
1	1	1	24	23	1	—	—	—
2	2	1	24	23	1	12	12	1
		2	24	22	2			

表 3 模型各参数取值

Table 3 Parameter values of the model

N	单元	Y_i	D_i	G_i	$n_{i(i-1)}$	$n_{i(i-1)}^{D}$	$n_{i(i-1)}^{G}$
3	1	3	3	0	4	3	1
	2	6	5	1			
	3	8	6	2	4	3	1

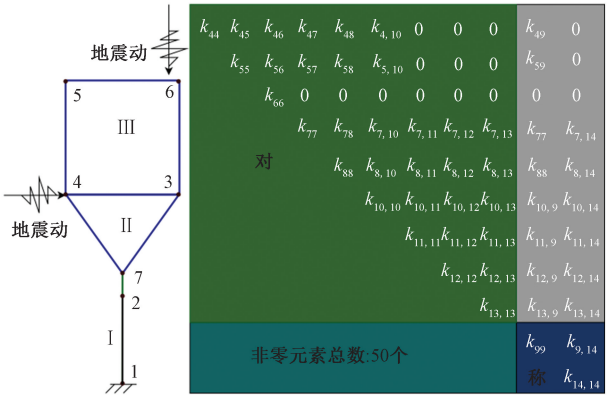


图 2　不同单元连接模型

Fig. 2　Different element connection model

2　CSC 格式总体刚度矩阵直接集成

2.1　等带宽一维数组定位原理

等带宽存储是有限元中常用的矩阵存储格式。如图 3 所示,矩阵按照等带宽存储时,矩阵数组中的每一行仍是原来 U_{BW} 个元素,只是矩阵的主对角元素都移到了矩阵数组中的第 1 列。

用 i、j 表示方阵形式的行号和列号,k_{ij} 表示矩阵元素。按等带宽存储后,行号记为 r_{ow},列号记为 c_{ol}。此时,方阵形式与等带宽形式的二维矩阵元素对应关系为

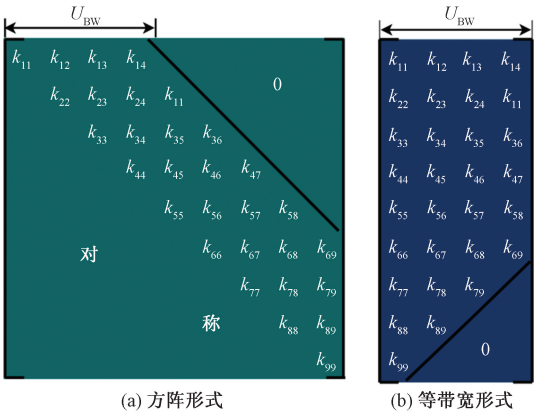
$$\begin{cases} r_{ow} = i \\ c_{ol} = j - i + 1 \end{cases} \tag{12}$$

将等带宽矩阵数组以一维数组 $\boldsymbol{B}$ 表示,即

$$\boldsymbol{B} = \{k_{11} \quad k_{12} \quad \cdots \quad 0 \quad k_{99} \quad 0 \quad 0 \quad 0\} \tag{13}$$

按等带宽一维存储形式储存后,在一维数组中的偏移记为 o_{ff}。此时,方阵形式与等带宽形式的一维矩阵元素对应关系为

$$o_{ff} = U_{BW}(r_{ow} - 1) + c_{ol} = U_{BW}(i - 1) + j - i + 1 \tag{14}$$



(a) 方阵形式　　(b) 等带宽形式

图 3　等带宽存储

Fig. 3　Equiband width storage

只保存非零元素,此时矩阵元素 k_{ij}、元素位置 N_{umber} 与偏移位置 o_{ff} 的对应关系如图 4 所示。

可以看出,方阵中的每一个矩阵元素在等带宽一维存储形式下都有一个独立 o_{ff} 与 N_{umber},可以通过 o_{ff} 来找寻方阵中的某一矩阵元素 k_{ij} 与元素位置 N_{umber};相反,也可以通过方阵中的某一矩阵元素 k_{ij} 精准定位元素位置 N_{umber} 与偏移信息 o_{ff}。

因此,可以通过遍历单元刚度矩阵信息计算偏移信息 o_{ff},并根据 o_{ff} 大小进行排序获取位置信息 N_{umber},通过 o_{ff} 与 N_{umber} 的唯一性实现矩阵元素定位、排序以及总体矩阵的集成。为减少计算的工作量且保证偏移信息的唯一性,该算法不再计算带宽 U_{BW},将 U_{BW} 取模型自由度个数。

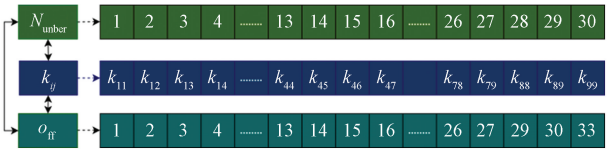


图 4　等带宽存储元素位置对应关系

Fig. 4　Equibandwidth storage element location mapping

2.2　CSC 存储

该方法采用三个一维数组存储矩阵数据,即(R_{ow},$C_{ol.\,off}$,V_{alue})。V_{alue} 数组存储矩阵中的非零元素,矩阵按列划分为 n 个段,每段包含该列的所有非零元素,长度为矩阵非零元素个数;R_{ow} 记录矩阵非零元素相应的行号,按列划分为 n 段,与 V_{alue} 的长度相同,数组的每段都是矩阵行中非零元素的行索引;$C_{ol.\,off}$ 为矩阵表示某列的起始元素在 V_{alue} 中的偏移位置,长度为矩阵的阶数加一。

以一个包含 11 个非零元素的 6×6 二维稀疏矩阵 $\boldsymbol{A}$ 为例,根据上述存储方法,该矩阵的 CSC 存储如图 5 所示。

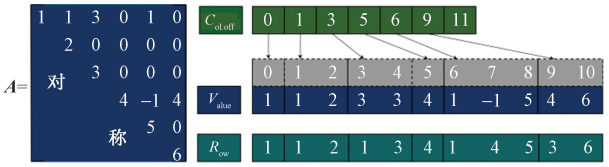


图 5　矩阵 $\boldsymbol{A}$ 及 CSC 储存格式

Fig. 5　Matrix $\boldsymbol{A}$ and CSC storage format

2.3　刚度矩阵的集成

一般情况下,在刚度矩阵生成时,首先需要根据单元的拓扑结构来确定非零子矩阵的数量和位置,生成索引数组,根据刚度矩阵的生成规则和索引数组,将刚度矩阵数据对号入座进行组合。

本文研究将等带宽元素定位原理与 CSC 格式存储相结合,不需要根据单元的拓扑结构来形成索引数组;在单元刚度矩阵层面,将单元刚度矩阵

的每个非零元素数据直接集成到总体刚度矩阵中。

CSC 格式的总体刚度矩阵的集成主要分为两阶段,第一阶段与第二阶段相继作用完成 CSC 格式的矩阵组装:①将单元刚度矩阵每块矩阵数据无序放入总刚度矩阵一维数组中,形成等带宽辅助数组,并获取总体刚度矩阵数值、列高、每个数据的行号。等带宽辅助数组起定位作用;②根据等带宽一维定位辅助数组将无序的数组进行排列,形成最终的 CSC 存储格式。

K_{II}、K_{GG}、K_{IG} 或 K_{GI} 块集成原理相同,以 K_{II} 块集成为例,第一阶段组集流程如表 4 所示。表 4 中 $K_{\mathrm{II}}^{\mathrm{num}}$ 数组长度,即非零元素数量;D_i 为模型自由度数量;$v_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$c_{\mathrm{ol.\,height}}\_\boldsymbol{K}_{\mathrm{II}}[D_i]$、$r_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$ 为刚度矩阵的数值、行号及列高数组;$B_{\mathrm{and}}\_\boldsymbol{K}_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$ 为等带宽辅助数组(第二阶段组集完成后消除),该数组是一个特殊 N_{umber} 的二叉树结构,每一个 o_{ff} 数据对应一个编号,可以根据 o_{ff} 数据索引出 N_{umber} 编号,并且该数组不允许出现重复 o_{ff} 数据。

对所有单元遍历完毕即完成 K_{II} 块的组装,得到临时数组 $r_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$v_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$c_{\mathrm{ol.\,height}}\_\boldsymbol{K}_{\mathrm{II}}[D_i]$、$B_{\mathrm{and}}\_\boldsymbol{K}_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$。第二阶段组集流程如表 5 所示。$R_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$V_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$C_{\mathrm{ol.\,off}}\_\boldsymbol{K}_{\mathrm{II}}[D_i+1]$ 为 CSC 存储格式的行号、数值、列偏移数组。

表 4 第一阶段实现流程

Table 4 The first phase implements the process

第一阶段:
1. 初始化
1-1. 初始化 K_{II} 块、$v_{\mathrm{alue}}\_K_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$r_{\mathrm{ow}}\_K_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$c_{\mathrm{ol.\,height}}\_K_{\mathrm{II}}[D_i]$、$B_{\mathrm{and}}\_K_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$;
1-2. 临时变量 N_{um} 初始化为 0。(记录元素位置)。
2. 遍历 $E_0(1,1,N_{\mathrm{E}})$
2-1. 获取当前单元 $E_{\mathrm{LEM}}(E_0)_i$;
2-2. 获取当前单元独立自由度个数 D_i 及相应 K_{II} 子块;
2-3. 判断单元是否循环完毕,若循环完毕,集成结束;若循环没有完毕,进入操作 3。
3. 遍历 K_{II} 子块
3-1. 获取子块矩阵元素 v_{alue}、r_{ow}、c_{ol};
3-2. 判断 r_{ow} 是否小于 c_{ol},若小于进行 3-3,若大于返回 3-1;
3-3. 计算 $o_{\mathrm{ff}}=U_{\mathrm{BW}}(i-1)+j-i+1$,并判断 o_{ff} 是否存在于 $B_{\mathrm{and}}\_\boldsymbol{K}_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$ 中,若存在,根据 o_{ff} 索引出 N_{umber},将 $v_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{umber}}]$ 进行数值叠加;若不存在,存储相应数据:$v_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{um}}]$、$r_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{um}}]=r_{\mathrm{ow}}$、$c_{\mathrm{ol.\,height}}\_\boldsymbol{K}_{\mathrm{II}}[c_{\mathrm{ol}}]++$、$B_{\mathrm{and}}\_\boldsymbol{K}_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$(默认第一次遍历不存在,直接存储即可)。
4. 单元循环完毕,返回操作操作 2

表 5 第二阶段实现流程

Table 5 The second phase implements the process

第二阶段:
1. 初始化
1-1. 临时变量 N 初始化为 0。(记录循环次数);初始化 $R_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$V_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$C_{\mathrm{ol.\,off}}\_\boldsymbol{K}_{\mathrm{II}}[D_i+1]$。
2. 遍历 $B_{\mathrm{and}}\_\boldsymbol{K}_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$
2-1. 获取当前 o_{ff} 对应的编号 N_{umber};
2-2. 获取 $r_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{umber}}]$、$v_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{umber}}]$;
2-3. $R_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[N]=r_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{umber}}]$、$V_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[N]=v_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[N_{\mathrm{umber}}]$;判断 N 是否小于 D_i+1,若小于,将 $c_{\mathrm{ol.\,height}}\_\boldsymbol{K}_{\mathrm{II}}[D_i]$ 的前 N 个数值进行叠加并赋值给数组 $C_{\mathrm{ol.\,off}}\_\boldsymbol{K}_{\mathrm{II}}[N+1]$。
2-4. 变量 $N++$,返回操作 2-1。

对 $B_{\mathrm{and}}\_\boldsymbol{K}_{\mathrm{II}}[o_{\mathrm{ff}},N_{\mathrm{umber}}]$ 遍历结束即形成按顺序排列的 $R_{\mathrm{ow}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$V_{\mathrm{alue}}\_\boldsymbol{K}_{\mathrm{II}}[K_{\mathrm{II}}^{\mathrm{num}}]$、$C_{\mathrm{ol.\,off}}\_\boldsymbol{K}_{\mathrm{II}}[D_i+1]$ 数组。对于 K_{GG}、K_{IG} 或 K_{GI} 的集成,将上述集成过程的块替换为其他相应子块数据即可,并且在实际组装过程中,所有子块可以同时进行,所以该方法对单元遍历完成之后即可完成总体刚度矩阵的组装,减少了矩阵数据约束处理工作,避免了节点编号优化的难题。

3 算例分析

以北京市轨道交通房山线长稻区间四跨单线梁桥为研究对象,大桥设计桥长 120 m,跨径均为 30 m,桥面净宽 12.5 m,该区间桥梁采用简支箱梁,T 型独柱桥墩和钻孔灌注桩(4 桩或 5 桩)基础,桥面铺设单向轨道,桥墩编号从左到右为 0#~4#。全桥有限元模型如图 6 所示,主箱梁、桥墩和桩采用梁单元;桩土作用采用土弹簧;支座、轨道与桥面连接采用两点弹簧单元;支座与箱梁、桩基与承台、轨道与桥面连接弹簧的下端均采用主从约束,目的使其位移保持一致。

为了验证本文算法正确性、实用性,将非零元素计算方法、组集方法与 CSC 存储方法应用于自行开发的有限元分析程序 ADUTS 中。开发平台为

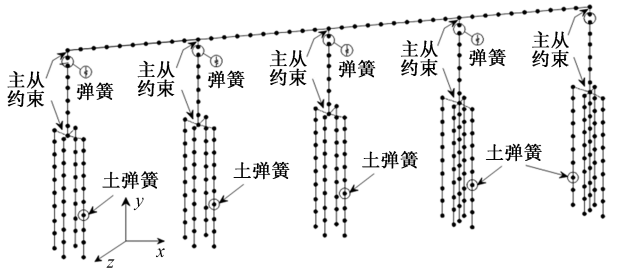


图 6 有限元模型示意图

Fig. 6 Diagram of finite element model

Visual Studio 2017、计算机配置为 CPU 为 Intel(R) Core(TM) i7-10700、运行内存 16 GB。从桩基础 X 方向输入图 7 所示的地震动时程,对全桥模型进行动力线性分析,通过比较组集效率、从内存占用和计算效率 3 个方面,将本文提出的组集方法、存储算法与 ID 表组集法、变带存储方法进行比较;为确保计算精度,将两种存储方法的计算结果与 OpenSees 软件进行对比分析。

对于组集效率方面,在不同的节点和自由度数量下,对采用不同组集方法的组集效率进行比较。结果如图 8 所示。

辅助数组法相较于 ID 表组集法效率提升 30% 左右,原因是辅助数组法减少了构建拓扑结构索引数组步骤;然而,随着模型的增大,提升幅度会逐渐降低,这是由于 ID 表形成时间相较于整个非零元素组集过程所占的时间比重相对较小。

在内存占用方面,采用双精度浮点型保存刚度矩阵数据,每个数据占用 8 个字节,整型保存行、列数组,每个数据占用 4 个字节;对于计算效率方面,以采用 Cholesky 分解法求解整个动力时程分析为例,时间间隔为 0.02 s,共持续 61.86 s,计算 3 093 步。分别对比变带宽存储方法与 CSC 存储方法的内存占用与求解时间,结果如图 9 和图 10 所示。

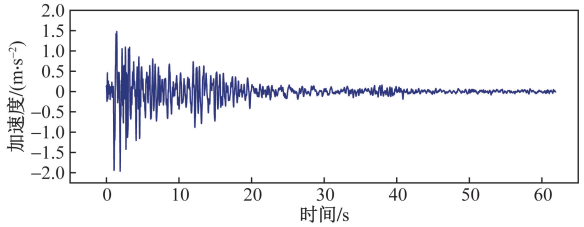


图 7　E1 地震加速度时程

Fig. 7　E1 earthquake acceleration time history

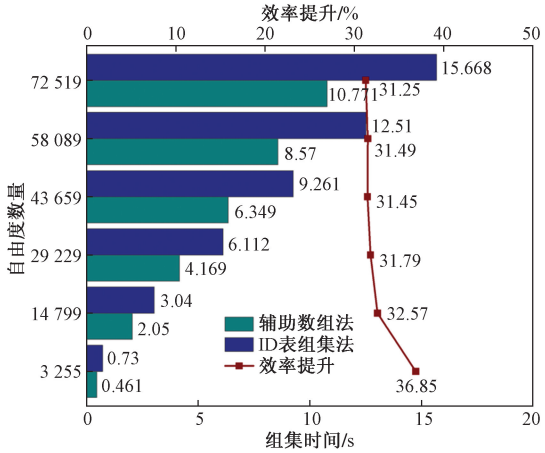


图 8　不同组集方法计算效率对比

Fig. 8　Comparison of computational efficiency of different group set methods

由图 9 和图 10 可以看出,CSC 存储在内存占用和求解时间显著低于变带存储。这是由于 CSC 存储只存储矩阵中的非零元素,从而有效减少了所需存储的数据量;对于变带存储,存储的数据量与节点编号有关,若节点编号设置不合理,则会存储大量刚度矩阵中不必要的零元素。而且,在进行 Cholesky 分解以求解线性方程组的过程中,求解效率与存储的数据量大小密切相关,数据量越大,所需的分解时间越长。

在计算精度方面,提取整个动力分析过程 2#墩顶节点整体坐标系 X 方向位移和各墩顶地震作用下的最大反应与 OpenSees 进行对比,最大反应由表 6给出,位移曲线如图 11 所示。

通过对比分析,可以发现两种存储方法计算的

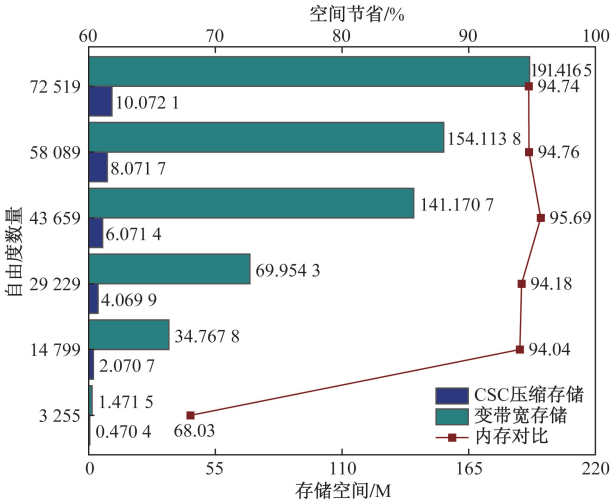


图 9　不同存储方法内存占用对比

Fig. 9　Comparison of memory usage of different storage methods

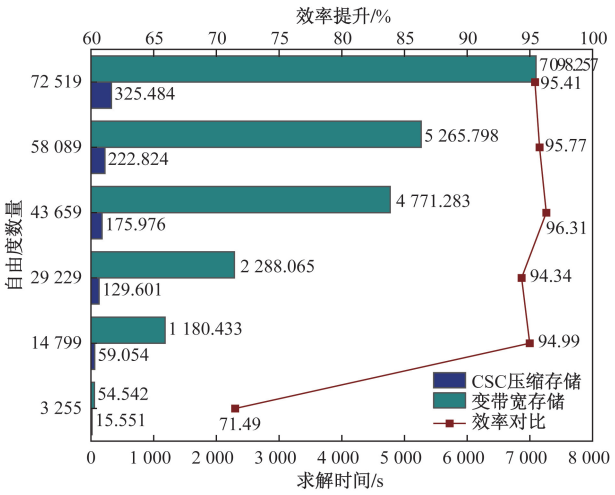


图 10　不同存储方法求解时间对比

Fig. 10　Comparison of solution time of different storage methods

结果与 OpenSees 吻合度较高;以 OpenSees 为基准,墩顶最大反应误差最大值为 0.027%,均在合理范围内。

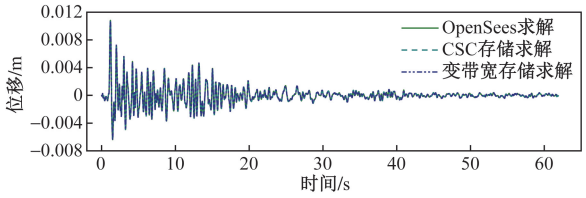


图 11 2#墩顶位移计算结果曲线

Fig. 11 2# pier top displacement calculation result curve

表 6 地震作用下的最大反应

Table 6 The maximum response to an earthquake

位置	OpenSees/cm	CSC 存储/cm	变带宽存储/cm	误差 1/%	误差 2/%
0#墩顶	1.079 30	1.079 00	1.079 00	0.027 0	0.027 0
1#墩顶	1.079 10	1.079 07	1.079 07	0.002 7	0.002 7
2#墩顶	1.081 44	1.081 41	1.081 41	0.002 7	0.002 7
3#墩顶	1.084 60	1.084 57	1.084 57	0.002 7	0.002 7
4#墩顶	1.102 53	1.102 48	1.102 48	0.004 5	0.004 5

4 结论

(1)经过约束处理后,只有独立自由度、地震动自由度总体矩阵有贡献,可以减少总体矩阵的规模及相应存储需求。

(2)基于自由度与刚度矩阵非零元素之间的关系,确定刚度矩阵非零元素的分布规律,提出二维、三维有限元刚度矩阵非零元素数量计算方法,给出各分块矩阵与总体刚度矩阵非零元素数量计算公式。

(3)通过等带宽定位原理与 CSC 格式总体矩阵集成相结合,可以在单元层面直接将每个矩阵元素集成到总体矩阵中,组集效率提高 30%,避免了节点编号的优化的难题。

(4)与变带宽存储相比,CSC 存储空间节省在 68% 以上;在采用直接法 Cholesky 分解来求解有限元线性方程组时,采用 CSC 存储进行求解可以显著提升线性方程组的求解效率,同时保证了计算精度;与变带宽存储相比,效率提升在 71% 以上,并且节点数量越多,效率提升越明显。

参 考 文 献

[1] Bakari A I, Dahiru I A. Comparison of Jacobi and Gauss-Seidel iterative methods for the solution of systems of linear equations[J]. Asian Research Journal of Mathematics, 2018, 8(3): 1-7.

[2] 纪国良, 丁勇, 周曼, 等. 工程计算中大型稀疏矩阵存储方法研究[J]. 数值计算与计算机应用, 2018, 39(3): 217-230.
Ji Guoliang, Ding Yong, Zhou Man, et al. Research on the storage method of large-scale sparse matrix in engineering calculation[J]. Journal on Numerical Methods and Computer Applications, 2018, 39(3): 217-230.

[3] Czech Technical University in Prague, Czech Technical University in Prague. Evaluation of different approaches to solution of the direct solution of large, sparse systems of linear equations[J]. Advanced Materials Research, 2017, 4442: 97-101.

[4] 姜涛, 王安麟, 朱灯林. 有限元结点编号的综合带宽优化算法[J]. 机械设计, 2005, 22(11): 3-6.
Jiang Tao, Wang Anlin, Zhu Denglin. Synthetic bandwidth optimization algorithm of finite element node numbering[J]. Journal of Machine Design, 2005, 22(11): 3-6.

[5] 吴鸿庆. 结构有限元分析[M]. 北京: 中国铁道出版社, 2009.
Wu Hongqing. Finite element analysis of structure[M]. Beijing: China Railway Publishing House, 2009.

[6] 姚松, 田红旗. 有限元刚度矩阵的压缩存贮及组集[J]. 中南大学学报(自然科学版), 2006, 37(4): 826-830.
Yao Song, Tian Hongqi. Compression storage scheme and assembly procedure of global stiffness matrix finite element analysis[J]. Journal of Central South University(Science and Technology Edition), 2006, 37(4): 826-830.

[7] 王忠雷, 赵国群, 马新武. 三维有限元刚度矩阵的压缩存储算法[J]. 材料科学与工艺, 2012, 20(2): 96-100, 107.
Wang Leizhong, Zhao Guoqun, Ma Xinwu. Compressed storage algorithm of 3D-FEM stiffness matrix[J]. Materials Science and Technology, 2012, 20(2): 96-100, 107.

[8] 刘尧喜, 唐进元, 周炜, 等. 扩展有限元刚度矩阵的 CSR 存储实现[J]. 机械强度, 2019, 41(6): 1384-1390.
Liu Yaoxi, Tang Jinyuan, Zhou Wei, et al. CSR storage method of extended finite element stiffness matrix[J]. Journal of Mechanical Strength, 2019, 41(6): 1384-1390.

[9] 周张兰. 基于十字链表与三元组表的稀疏矩阵压缩存储实例研究[J]. 软件导刊, 2017, 16(11): 22-25.
Zhou Zhanglan. Study on the compressed storage to sparse matrix based on cross list and tiriple table[J]. Software Guide, 2017, 16(11): 22-25.

[10] Liu C, Ye J, Ma Y. Storage and solving of large sparse matrix linear equations[C]//Fourth International Conference on Computational and Information Sciences. New York: IEEE, 2012: 673-677.

[11] Guo D H, Gropp W. Applications of the streamed storage format forsparse matrix operations[J]. The International Journal of High Performance Computing Applications, 2014, 28(1): 3-12.

[12] Pei D M, Meng F J, Wang H L. An improved compression and storage algorithm of the stiffness matrix of 3D-FEM based on rider binary classification and negative sign CSR[J]. International Journal of u- and e-Service(Science and Technology Edition), 2015, 8(8): 215-224.

[13] Xing L Y, Wang Z S, Ding Z Z, et al. An efficient sparse stiffness matrix vector multiplication using compressed sparse row storage format on AMD GPU[J]. Concurrency and Computation: Practice and Experience, 2022, 34(23). DOI: 10.1002/CPE.7186.

[14] Saak J, Schulze J. Diagonally-addressed matrix nicknack: how to improve SpMV performance[J]. PAMM, 2023, 23(4): e202300228.

[15] Bian H D, Huang J Q, Dong R T, et al. A simple and efficient

storage format for SIMD-accelerated SpMV[J]. Cluster Computing, 2021, 24(4): 1-18.

[16] 刘丽, 陈长波. 带状稀疏矩阵乘法及高效 GPU 实现[J]. 计算机应用, 2023, 43(12): 3856-3867.
Liu Li, Liu Changbo. Band sparse matrix multiplication and efficient GPU implementation[J]. Journal of Computer Applications, 2023, 43(12): 3856-3867.

[17] Jendele L, Červenka J. On the solution of multi-point constraints-application to FE analysis of reinforced concrete structures[J]. Computers & Structures, 2009, 87(15/16): 970-980.

[18] Boungard J, Wackerfuß J. Consideration of nonlinear multipoint constraints in finite element analyses based on a master-slave elimination scheme operating at the global level[J]. PAMM, 2023, 22(1): e202200311.

[19] 张军锋, 温珺博, 张金鹏, 等 主从约束在有限元计算中的实现方式和过程[J]. 水利与建筑工程学报, 2023, 21(2): 195-200.
Zhang Junfeng, Wen Junbo, Zhang Jinpeng, et al. Realization method and process of master-slave constraint in finite element method[J]. Journal of Water Resources and Architectural Engineering, 2023, 21(2): 195-200.

[20] Quyen V T B, Tien D N, Dung N N, et al. Penalty function method for imposing nonlinear multi freedom and multi node constraints in finite element analysis of frame systems[C]//IOP Conference Series: Materials Science and Engineering. Hanoi: IOP Publishing, 2020: 1621-1629.

[21] Ahn J G, YangH I, Kim J G. Multipoint constraints with lagrange multiplier for system dynamics and its reduced-order modeling[J]. AIAA Journal, 2020, 58(1): 385-401.

[22] Tsuta T, Yamaji S. Finite element analysis of contact problem theory and practice in finite element structural analysis[M]. Tokyo: University of Tokyo Press, 1973.

[23] 韩学川, 陶连金, 张宇, 等. 不同类型地震波作用下一体化车站结构地震响应分析[J]. 科学技术与工程, 2021, 21(4): 1506-1514.
Han Xuechuan, Tao Lianjin, Zhang Yu, et al. Seismic response of integrated station structures under different types of seismic waves[J]. Science Technology and Engineering, 2021, 21(4): 1506-1514.

[24] Peng X Y, Yu P C, Zhu H, et al. Proposal of a coupled DDA-SPH method incorporating a new contact algorithm for soil-structure interaction simulations in geotechnical hazards[J]. Computers and Geotechnics, 2023, 164. DOI: 10.1016/J.COMPGEO.2023.105849.

[25] 钟万勰. 计算结构力学微机程序设计[M]. 北京: 水利电力出版社, 1986.
Zhong Waixie. Computer programming for computational structural mechanics[M]. Beijing: Water Conservancy and Electric Power Press, 1986.

[26] 唐松花, 罗迎社, 周筑宝. 线弹性动力学中的最小势能原理(含最小余能原理)[J]. 动力学与控制学报, 2005, 3(1): 36-40.
Tang Songhua, Luo Yingshe, Zhou Zhubao. Minimum potential energy principle in linear elastic dynamics (including minimum complementary energy principle)[J]. Journal of Dynamics and Control, 2005, 3(1): 36-40.