现代有限元软件方法

崔俊芝 梁 俊 著

国防工业出版社

F G Y C B S

现代有限元软件方法

崔俊芝　梁　俊　著

国防工业出版社
·北京·

(京)新登字 106 号

图书在版编目(CIP)数据

现代有限元软件方法/崔俊芝,梁俊著. —北京:国防工业出版社,1995.4
ISBN 7-118-01405-2

Ⅰ.现… Ⅱ.①崔… ②梁… Ⅲ.有限元-软件工程 Ⅳ.TP311.5

中国版本图书馆 CIP 数据核字(94)第 15959 号

国防工业出版社出版发行
(北京市海淀区紫竹院南路 23 号)
(邮政编码 100044)
北京四季青印刷厂印刷
新华书店经售
*
开本 787×1092 1/16 印张 9¼ 210 千字
1995 年 5 月第 1 版 1995 年 5 月北京第 1 次印刷
印数:1—3000 册 定价:12.00 元

序

有限元软件是和有限元方法同时诞生，并且随着有限元方法和计算技术的发展而迅速发展的。据不完全统计，40年来已经研制了数百个通用的有限元软件和数千个专用的有限元软件，它们在大中型科学和工程应用软件中占据相当大的比重。有限元软件是使用有限元方法解决水工、土建、桥梁、机械、电机、冶金、造船、飞机、宇航、核能、地震、物探、气象、渗流、水声、物理学、力学以及国防工程等各种科学和工程问题的关键，已经使有限元方法转化为直接推动科技进步和社会发展的生产力，使之发挥了巨大的经济和社会效益。

40年来，有限元软件经历了有限元分析软件、有限元分析与结构设计软件、有限元分析＋CAD软件、有限元分析＋CAD＋专家系统、智能性结构分析系统和集成化有限元软件环境等发展阶段。到目前为止，有限元软件的分析功能已经相当完备，其CAD功能已达到实用程度，但是它在人工智能和专家系统方面的功能仅仅是初步的，远未达到实用程度。

从有限元软件的体系结构、实现方法和应用技术来看，现代有限元软件是一个多学科的、综合技术的集成化产品，除了有限元方法外，数据管理，用户接口，图形加工与管理，人工智能和专家系统，面向对象的软件设计，以及现代软件工程环境的方法和技术，在有限元软件研制中均占有重要的地位，它们和有限元方法相结合，已经形成了一个特殊的称之为有限元软件技术的研究领域。在思维方法、表现形式和知识结构上，这个领域与有限元方法有明显区别，并且随着计算技术的发展它仍在迅速发展。

自1977年《有限元程序设计》[①]一书出版以来，有限元软件方法已经发生了巨大变化，但是至今尚无一本贡献于这一领域的专著。填补空白，推动有限元软件的进步，就是我们决心写作本书的真实意图。

本书是在作者从事有限元软件方法研究和软件研制的基础上完成的，为使读者对有限元软件方法有一个全面的了解，帮助读者研制或更新有限元软件，我们没有局限于自己的研究成果，而是尽力全面地吸取有限元软件方法的新成果。另一方面，由于有限元软件方法是一个综合性的技术领域，涉及到计算方法和软件技术的众多分支，限于篇幅不可能进行全面而详尽的叙述，因此我们紧紧抓住几个关键技术，突出重点加以叙述。

全书共分八章，第一、二两章是对有限元方法及其软件技术的概要介绍和评述，目的是给读者提供预备性知识，第三至八章分六个专题分别叙述现代有限元软件的关键方法，对于每个专题分别介绍了若干先进的实用方法和发展中的新方法。本书的取材，特别是实用方法多数是以作者的研究和软件实践为基础的。

值得指出，本书的作者绝非两人，本书是在两个国家自然科学基金项目和攀登计划项

---

① Hinton E, D. Owen R J. Finite Element Programming. Academic Press. 1977

目的支持下、在项目的进行中完成的，项目的参加人都直接或间接地对本书做出了贡献，他们也应该是本书的作者，在此我们向为本书做出了贡献的专家、教授和国家自然科学基金委员会力学处的同志们表示感谢，感谢他们卓有成效的支持和帮助。

由于作者水平所限，不妥之处在所难免，敬请读者批评指正。

崔俊芝

1994 年 8 月

内 容 简 介

本书是一本全面系统地论述现代有限元软件方法的专著，全书论述了研制一个优秀的有限元软件所需要的各种方法和技术，包括：数据管理技术、用户接口技术、图形加工与管理技术、人工智能和专家系统方法、符号演算与软件自动化技术、面向对象的软件技术以及现代软件工程环境的方法和技术。书中尽力全面地吸取了有限元软件方法的新成果。

全书共分八章，第一二两章是对有限元方法及其软件方法的概要介绍和评述，目的是给读者提供预备性知识，第三～八章分六个专题分别叙述现代有限元软件的关键方法，对于每个专题分别介绍了若干先进的实用方法和发展中的新方法。本书的取材，特别是实用方法多数是以作者的研究和软件实践为基础的。

本书可供计算数学、计算力学以及其他工程计算与软件技术研究者使用，也可作为高等院校相关专业师生的教学参考书。

目　录

第一章　有限元方法

40 年来，有限元方法经历了诞生、发展和完善的三个历史时期，已经拥有十分丰富的方法形态，并且由于其算法过程的通用性，及其对材料组合和几何拓扑的适应性，已经在相当广泛的领域得到有效的应用，成为求解各种固体力学和结构工程问题的最有效的数值计算方法，受到了各行各业众多的科学和工程计算专家的重视。就有限元方法形态而言，除了最早诞生的基于极小势能原理的位移有限元模式外，还发展了基于余能原理的应力平衡模式，基于广义势能原理的位移杂交模式和基于广义余能原理的应力杂交模式，基于 H-W 混合变分原理的混合有限元模式，以及各种各样的特殊有限元方法形态，例如边界有限元法，有限条法，无限元法，半解析有限元法；近年来又发展了综合有限元、有限条与半解析法于一体的，容许任意曲线剖分的有限元线法。这些名目繁多的有限元模式和方法形态，大大提高了有限元方法求解各类科学和工程问题的能力和效率。但是在现有的有限元模式中，最实用、有效和应用最广的，仍然是最早发展起来的基于极小势能原理的位移有限元方法。其他模式和方法形态均因为在某些情况下存在明显缺陷，而未被广泛采用，其理论价值远高于其实用价值。

有限元方法是针对求解固体力学静平衡问题而创立的。现代有限元方法应该包括静态有限元方法、瞬态有限元方法以及支持有限元分析的前后处理算法，它们是一个内容极为广泛的方法集合体。主要的有限元模式和方法形态都是针对静平衡问题的，因此静态有限元方法是有限元方法的基础，瞬态有限元方法是前者的直接推广和发展，不同之处只是增加了如何处理时间离散化方程的算法；有限元分析的前后处理算法是在拓广经典有限元方法的应用和开发有限元软件的过程中，逐步发展起来的数值的和非数值的方法，它们对有限元方法的有效应用和形成现代有限元软件具有至关重要的作用。

§1.1　静态有限元方法

有限元方法是将连续体离散化的一种近似方法，其理论基础是变分原理、连续体剖分与分片插值，即首先找到欲求解的数学物理问题的变分表示，对于固体力学问题而言是写出其总能量表示式，然后将问题的定解区域剖分成有限个小单元体的集合，对于二维连续体，用点、线、面将其分成有限个三角形或四边形的集合，在每个单元体上选择有限个节点并在每个节点上选定有限个待求的广义节点位移，然后在每个单元上以选定的全部节点上的节点广义位移为参数近似地插值整个单元上的连续位移，进而将插值位移代入能量表达式，运用变分原理得到以广义节点位移为未知量的离散化有限元方程，求解有限元方程得到每个节点上的广义节点位移，再在每个单元上使用节点广义位移插值求得各种欲求的物理量，例如位移、应变、应力等。本节简要叙述静态问题的有限元离散化过程；有限

元方程的存储方法与解法。

1.1.1 离散化过程

下面以平面弹性问题为例简要叙述有限元离散化过程。

1. 能量表示

$$\mathrm{Min}W = \frac{1}{2}t\int_{\Omega}\sigma^{T}\varepsilon d\Omega - \int_{s_\sigma}U^{T}Pds - \int_{\Omega}U^{T}Fd\Omega \tag{1.1}$$

其中

$$U = \begin{bmatrix} u \\ v \end{bmatrix}, \quad P = \begin{bmatrix} p_x \\ p_y \end{bmatrix}, \quad F = \begin{bmatrix} f_x \\ f_y \end{bmatrix} \tag{1.2}$$

$$\varepsilon = \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial v}{\partial y} \\ \dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x} \end{bmatrix}, \quad \sigma = \begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{bmatrix} = D\varepsilon \tag{1.3}$$

在各向同性的平面应力情况下：

$$D = \frac{E}{1-\upsilon^2}\begin{bmatrix} 1 & \upsilon & \\ \upsilon & 1 & \\ & & \dfrac{1-\upsilon}{2} \end{bmatrix} \tag{1.4}$$

Ω 为定解区域，在 Ω 的边界 $\partial\Omega = S_\sigma + S_u$ 上，U 满足边界条件：

a)位移边界条件

$$u = \bar{u}, v = \bar{v}, \text{在 } S_u \text{ 上} \tag{1.5}$$

b) 外力边界条件

$$\begin{aligned} p_x &= \sigma_x\cos(n,x) + \tau_{xy}\cos(n,y) \\ p_y &= \tau_{xy}\cos(n,x) + \sigma_y\cos(n,y) \end{aligned} \tag{1.6}$$

其中 n 为外法向方向。后者为自然边界条件，它不对边界位移产生直接约束。

式(1.1)～(1.5)给出了平面弹性问题的变分描述，与它们等价的偏微分方程平面问题是式(1.3)、(1.4)、(1.5)、(1.6)与

$$\left.\begin{aligned} \frac{\partial\sigma_x}{\partial x} + \frac{\partial\tau_{xy}}{\partial y} &= f_x \\ \frac{\partial\tau_{xy}}{\partial x} + \frac{\partial\sigma_y}{\partial y} &= f_y \end{aligned}\right. \quad \Omega_0 \tag{1.7}$$

2. 有限元离散化

对于图 1.1 所示的平面弹性体 Ω，用线段将其分成 n_2 个单元，$\Omega = \sum_{e=1}^{n_2}\Omega_e$，假若我们选定任意三角形为基本单元，且具体剖分 Ω 的形式如图 1.1 所示；在每个三角形上，我们选定顶点为节点，将全部节点按一定顺序编号，n_0 表示节点总数；对每个节点选定节点位移为 $U_i = (u_i, v_i)^T$。显然全部节点位移组成平面弹性问题的未知参数向量 $U = (u_1, v_1, \cdots)^T$。

对于任意一个三角形单元 Ω_e 而言，它所关联的节点为 i_1, i_2, i_3，其节点位移为 $U_e =$

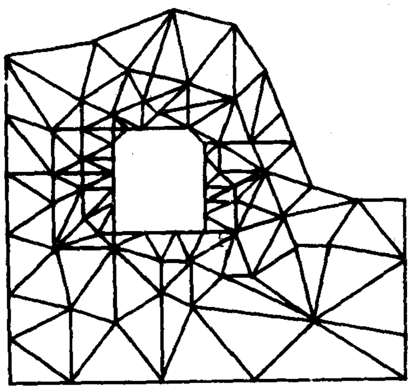

图 1.1 一个平面弹性体的单元剖分

$(u_{i_1},v_{i_1},u_{i_2},v_{i_2},u_{i_3},v_{i_3})^T$。以三个节点上的节点位移为基本参数，采用三角形线性插值，可以插出整个单元上的位移，

$$u_e(x,y)=\sum_{i=i_1}^{i_3}N_i(x,y)u_i$$
$$v_e(x,y)=\sum_{i=i_1}^{i_3}N_i(x,y)v_i \tag{1.8}$$

其中

$$N_i(x,y)=\frac{1}{2A}(a_i+b_ix+c_iy)$$
$$a_i=x_jy_m-x_my_j,b_i=y_j-y_m,c_i=x_m-x_j$$
$$(i=i_1,i_2,i_3,\quad j=i_2,i_3,i_1,\quad m=i_3,i_1,i_2) \tag{1.9}$$
$$2A=\begin{vmatrix}1 & x_{i_1} & y_{i_1}\\ 1 & x_{i_2} & y_{i_2}\\ 1 & x_{i_3} & y_{i_3}\end{vmatrix}=c_3b_2-b_3c_2$$

A 表示三角形面积，详细公式推导请参考[1]。于是每个单元上的位移函数可以表示为

$$U_e(x,y)=N(x,y)U_e \tag{1.10}$$

其中：

$$N(x,y)=[N_1,N_2,N_3]$$
$$N_i=N_i(x,y)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

通过插值位移表示，可以得到单元上的应变表示：

$$\varepsilon_e=BU_e$$
$$B=[B_1,B_2,B_3]$$
$$B_i=\frac{1}{2A}\begin{bmatrix}b_i & 0\\ 0 & c_i\\ c_i & b_i\end{bmatrix},\quad i=1,2,3 \tag{1.11}$$

从而可以确定单元上的应力表示：

$$\sigma_e=D\varepsilon_e=DBU_e \tag{1.12}$$

将 U_e,ε_e 和 σ_e 代入总能量表示中,可以得到以节点位移为基本参数的单元 Ω_e 上的离散形式的总能量表示:

$$W_e = \frac{1}{2}U_e^T(t\int_{\Omega_e} B^T DBd\Omega)U_e - U_e^T\int_{s_\sigma \cap \partial\Omega_e} N^T(x,y)Pds - U_e^T\int_{\Omega_e} N^T(x,y)Fd\Omega$$

当我们对每个单元 Ω_e 完成上述离散化表示之后,便可以很容易得到整个区域 Ω 上的以全部节点位移 U 为参数的离散化的总势能表示:

$$\begin{aligned} W &= \sum_{e=1}^{n_2} W_e \\ &= \sum_{e=1}^{n_2}\left\{\frac{1}{2}U_e^T(t\int_{\Omega_e} B^T DBd\Omega)U_e - U_e^T\int_{s_\sigma \cap \partial\Omega_e} N^T(x,y)Pds \right. \\ &\quad \left. - U_e^T\int_{\Omega_e} N^T(x,y)Fd\Omega\right\} \end{aligned} \tag{1.13}$$

它是一个以全部节点位移为基本参数的二次泛函,其极小化方程为

$$KU = R \tag{1.14}$$

称之为有限元方程,其中 K 称为总刚度矩阵,R 称为总节点荷载,

$$K = \sum_{e=1}^{n_2} K_e, R = \sum_{e=1}^{n_2}(P_e + F_e) \tag{1.15}$$

$$\begin{aligned} K_e &= t \cdot \int_{\Omega_e} B^T DBd\Omega \\ P_e &= \int_{s_\sigma \cap \partial\Omega_e} N^T PdS \\ F_e &= \int_{\Omega_e} N^T Fd\Omega \end{aligned} \tag{1.16}$$

式中,K_e 称为单元刚度矩阵;P_e 和 F_e 分别表示由作用于单元上的分布体力和表面力所产生的等效节点力。

对于线弹性问题,有限元方程(1.14)为节点位移的线性方程组,求解该方程组便可以得到全部节点位移;再返回到每个单元上,利用式(1.8)、(1.11)和(1.12)便可以求得每个单元上的位移、应变和应力,完成有限元分析的全部工作。

由上述离散化过程可知,对于有限元分析而言,一旦确定了单元形态,即单元形状、节点、节点广义位移和插值形式,剩下的工作就是如何根据总能量表示计算单元刚度矩阵 K_e、等效节点力 P_e 和 F_e,以及如何组装它们形成总体刚度矩阵 K 和总节点荷载 R,进而求解有限元方程。

1.1.2 有限元方程的存储方法

有限元方法是一种将连续问题离散化,近似地求解数学物理问题的方法,如果想求得高精度的近似解,必须采用精细的单元剖分或采用多节点广义位移的高次元,总之必然会遇到处理高达数万阶、数十万阶的大型有限元方程组的问题。因此,如何组装单元刚度矩阵形成总体刚度矩阵并存储起来,以准备求解,就成为处理有限元方程的第一个问题,特别是当方程组阶数达到数十万阶时。为了获得有效的存储方法,必须注意有限元方程组,

即总刚度矩阵的下述基本特征：

(1)对称正定性或半定性。对于经典的基于极小势能原理的有限元方程，其总刚度矩阵总是对称的，在施加了边界位移条件或消除了刚体位移后，它又是正定的。利用对称性，可以只存储总刚度矩阵的一半。

(2)大型稀疏性。当要求提高计算结果的精度时或遇到大型复杂组合结构时，有限元方程的阶数可能达到数十万阶。因为一个节点上的节点广义位移只和与该节点在同一单元上的节点广义位移相关，而和不与该节点在同一单元上的节点广义位移无关，这样在总刚度矩阵的每一行里只有少量的非零元素。非零元素密度有时只有千分之一到数万分之一。因此应尽量采用少存储零元素的方法。

(3)采用适当的节点编号可以使非零元素排列具有很强的规则性，例如带性、变带性、子结构加边型等。

在有限元分析过程中，占用内外存最大的部分是组装存储有限元方程并求解之，即存储总刚度矩阵、总质量矩阵和总荷载列阵，有效地存储总刚度矩阵并合理地实施内外存调度，是有限元分析软件中的一个关键技术。总刚度矩阵中非零元素的分布常见的有如下几种：等带宽、变带宽、子结构等带宽、子结构变带宽，如图 1.2 所示。针对总刚度矩阵及其非零元素的分布特征，前人已经分别构造出不同的存储方法，以适应于不同的求解方法。还有一些组装和求解结合在一起的方法，例如波前法存储格式、多重子结构解法。下面介绍四种最常用的存储方法。

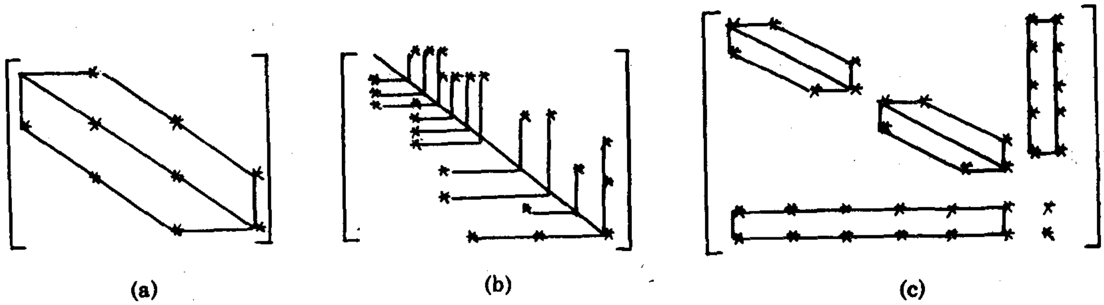

图 1.2 总刚度矩阵中非零元素的分布特征

1. 变带型下半带按行存储。这种方法有效地利用了总刚度矩阵的对称性，只存储总刚度矩阵的一半，还利用了非零元素的稀疏性和非零元素分布的变带型特征。这种存储方法是把总刚度矩阵中每行从第一个非零元素至对角线元素之间的元素按行顺序存储。当总刚度矩阵阶数很大且带外零元素很多时，会导致节省大量的存储空间。例如，对于矩阵

$$\begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ 0 & a_{32} & a_{33} & 0 & 0 \\ 0 & a_{42} & 0 & a_{44} & 0 \\ 0 & a_{52} & 0 & 0 & a_{55} \end{bmatrix} \tag{1.17}$$

其中 $a_{ij}=a_{ji}$ 实际存储的元素为

$$A:a_{11},a_{21},a_{22},a_{32},a_{33},a_{42},0,a_{44},a_{52},0,0,a_{55} \tag{1.18}$$

为了快速查找和调用带内的每个元素，实现有效的内外存管理，与实际存储的带内元素相对应地必须附加一个存储格式信息 INFO，指出每个对角线元素的序号。INFO 的逻辑结构为

$$\boxed{\text{INFO}} \overset{\text{def}}{=\!=} \boxed{\ \big|\ I=1,N} \text{—}\big|\ \text{IAII}\quad a_{ii}\text{在数组 A 中的序号} \tag{1.19}$$

对于上面的矩阵 A，INFO 的内容是

$$\text{INFO}:1,3,5,8,12$$

按此种存储方法，元素 $a_{ij}(i \geqslant j)$处在数组 A 的第 $\text{INFO}(i)-i+j$ 的位置上，第 i 行的第一个非零元素的列标号是

$$I1 = i-(\text{INFO}(i)-\text{INFO}(i-1))+1 \tag{1.20}$$

当 $j<I1$ 时，$a_{ij}=0$。

2. 下半带紧凑存储。这种存储方法充分利用了总刚度矩阵的对称性和稀疏性，是按行序号将矩阵中每行的非零元素从第一个元素到对角线元素按列序号顺序放在数组 A 中。对于上述矩阵 A，实际存放的元素是

$$A:a_{11},a_{21},a_{22},a_{32},a_{33},a_{42},a_{44},a_{52},a_{55} \tag{1.21}$$

按这种存储方法，非零元素的信息存放在另一个数组 INFO 中，它按行顺序存放着每行对角线元素在数组中的序号，以及每行非零元素的列标号。INFO 的逻辑结构是

$$\boxed{\text{INFO}} \overset{\text{def}}{=\!=} \left[\begin{array}{l} \boxed{\ \big|\ I=1,N} \text{—}\big|\ \text{IAII}\quad a_{ii}\text{在数组 A 中的序号} \\ \boxed{\ \big|\ I=1,NA} \text{—}\big|\ \text{JAIJ}\quad \text{非零元素 } a_{ij}\text{的列标号} \end{array}\right. \tag{1.22}$$

其中 NA 表示总刚度矩阵中下半带非零元素总数。对应于式(1.21)，INFO 的内容是：

$$\text{INFO}:1,3,5,7,9,1,1,2,2,3,2,4,2,5$$

按此种存储方法，可知矩阵中第 i 行第一个非零元素的列标号为

$$I1 = \text{INFO}(N+\text{INFO}(i-1)+1) \tag{1.23}$$

当 $j<I1$ 时，$a_{ij}=0$。

3. 变带型全带宽存储。由混合有限元模式和杂交有限模式导出的有限元方程，其总刚度矩阵往往是非对称的，但它们仍然是高度稀疏的，一般呈现变带型。对于这种情况，没有对称性可以利用，但可以采用变带型全带宽存储，即按行号顺序存储每一行第一个非零元素到最后一个非零元素之间的所有元素。对于式(1.17)所示的矩阵 A，实际存储的元素是

$$A:a_{11},a_{12},a_{21},a_{22},a_{23},a_{24},a_{25},a_{32},a_{33},a_{42},0,a_{44},a_{52},0,0,a_{55} \tag{1.24}$$

对于这种存储方法，非零元素的存储信息存放在另一个数组 INFO 中，它存放着每行第一个非零元素和对角元素在总刚度矩阵数组中的序号，其逻辑结构为

$$\boxed{\quad\text{INFO}\quad} \overset{\text{def}}{=\!=} \boxed{\ \big|\ I=1,N} \text{—}\left[\begin{array}{ll} \text{IAIJ1} & a_{ij1}\text{ 在 A 数组中的序号} \\ \text{IAII} & a_{ii}\text{ 在 A 数组中的序号} \end{array}\right. \tag{1.25}$$

对应于式(1.21)，INFO 为：

$$\text{INFO}:1,1,3,4,8,9,10,12,13,16 \tag{1.26}$$

在实际操作总刚度矩阵时，很容易根据这种存储方法，找到每个非零元素。第 i 行的第一个和最后一个非零元素的列标号是

$$
\begin{aligned}
I1 &= i - (\mathrm{INFO}(2i) - \mathrm{INFO}(2i-1)) \\
I2 &= i + (\mathrm{INFO}(2i+1) - \mathrm{INFO}(2i)) - 1
\end{aligned}
\tag{1.27}
$$

当 $j<I1$ 和 $j>I2$ 时，$a_{ij}=0$。

4. 全紧凑存储，对于非对称且高度稀疏的总刚度矩阵，最节省的存储方法是只存储非零元素。总刚度矩阵数组中的元素是按行序号顺序存放每行的非零元素，每行内部的顺序是按非零元素的列序号由小到大逐次排列。对于式(1.17)所示的矩阵 A，按这种存储方法的数组 A 为

$$
A: a_{11}, a_{12}, a_{21}, a_{22}, a_{23}, a_{24}, a_{25}, a_{32}, a_{33}, a_{42}, a_{44}, a_{52}, a_{55} \tag{1.28}
$$

对于这种存储方法，非零元素的信息存储在另两个整型数组 INFO1 和 INFO2 中，INFO1 顺序存放着所有非零元素的列标号，它和总刚度矩阵数组一一对应；INFO2 顺序存放着每行第一个非零元素与对角线元素在总刚度矩阵数组中的位置。例如，对应于(1.25)，INFO1 和 INFO2 分别为

$$
\mathrm{INFO1}: 1,2,1,2,3,4,5,2,3,2,4,2,5 \tag{1.29}
$$

$$
\mathrm{INFO2}: 1,3,4,8,9,10,11,12,13 \tag{1.30}
$$

INFO1 的逻辑结构为

$$
\boxed{\ \mathrm{INFO1}\ } \overset{\text{def}}{=\!=} \boxed{\ \ \boxed{I = 1,\mathrm{NA}}} \text{——}|\ \mathrm{IAIJ}\quad \text{非零元素 } a_{ij} \text{ 在 A 数组中的列标号} \tag{1.31}
$$

其中 NA 为非零元素总数。INFO2 的逻辑结构为

$$
\boxed{\ \mathrm{INFO2}\ } \overset{\text{def}}{=\!=} \boxed{\ \ \boxed{I = 1,\mathrm{N}}} \text{——}\left[\begin{array}{ll} \mathrm{IAIJI} & \text{第 i 行第一个非零元素在 A 数组中的序号} \\ \mathrm{IAIJ} & a_{ii} \text{ 在 A 数组中的序号} \end{array}\right. \tag{1.32}
$$

按照这种存储方法，由 INFO2 可以很容易获得每一行第一个非零元素在总刚度矩阵数组中的序号与列标号，逐次可以找到所有非零元素。

对于大型复杂的组合结构，为了有限元分析之便，有时需要将整个结构人为地分解成若干个子结构，其总刚度矩阵的一般形式如图 1.2(c)所示。对于每个子结构可以采用上述四种方法之一存储之。

1.1.3 解法

求解有限元方程是有限元分析的关键环节，对于大型复杂结构的有限元分析而言，它占去了主要计算量，一般占 3/4 以上。因此应该选择高效能的求解有限元方程的方法。求解有限元方程的方法主要有两种：直接法与迭代法。有效的直接法应该是在保证精度的前提下减少零元素参与运算。迭代法应该是提高收敛速度，以减少迭代的次数。鉴于有限元方程的阶数一般都很高，在求解过程中很难将整个有限元方程放在计算机内存处理，因此求解过程必然伴随着有效的数据组织与管理，解法的选择必须考虑到总刚度矩阵的存储结构，实行合理的数据调度和管理。下面介绍两个与上节存储方法相匹配的解法。

1. LDL^T 分解法。这是一种与变带型下半带按行存储格式相适应的一次求解多列右端项的直接解法。假定要求解的有限元方程是

$$
AU = B \tag{1.33}
$$

其中 A 是 N×N 阶矩阵，具有对称正定性和变带型特征，已采用变带型下半带按行存储。

LDL^T分解法的计算过程如下：

(1)分解的计算过程如下：

$$A = LDL^T$$

其中

$$L = \begin{bmatrix} 1 & & & & & \\ l_{21} & 1 & & & & \\ & & \ddots & & & \\ & l_{ij} & & 1 & & \\ & & & & \ddots & \\ 0 & & & & & 1 \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & & & & \\ & d_2 & & & & \\ & & \ddots & & & \\ & & & d_i & & \\ & & & & \ddots & \\ & & & & & d_N \end{bmatrix} \qquad (1.34)$$

具体分解的步骤为：

a) $d_1 = a_{11}$

b)对于 $i=2,3,\cdots,N, j=I_1, I_1+1,\cdots,i-1$ 计算

$$\tilde{a}_{ij} = a_{ij} - \sum_{k=IJ}^{j-1} \tilde{a}_{ik} l_{jk}$$
$$d_i = a_{ii} - \sum_{k=I1}^{i-1} \tilde{a}_{ik} l_{ik} \qquad (1.35)$$

对于 $j=I1, I1+1,\cdots,i-1$ 计算

$$l_{ij} = \tilde{a}_{ij}/d_j \qquad (1.36)$$

其中 I1 为第 i 行第一个非零元素的列标号，IJ=Max(I1,J1)。

(2)正代与回代

对于 $k=1,2,\cdots,M$ 计算

$$\text{a) 计算} \quad b_{ik} = b_{ik} - \sum_{j=I1}^{i-1} l_{ij} \cdot b_{jk}, i = 1,2,\cdots,N;$$
$$\text{b) 计算} \quad b_{ik} = b_{ik}/d_i, i = 1,2,\cdots,N; \qquad (1.37)$$
$$\text{c) 计算} \quad b_{ik} = b_{ik} - \sum_{j=i+1} l_{ji} \cdot b_{jk}, i = N-1, N-2,\cdots,1;$$

2. 共轭斜量法。在求解大型有限元方程的迭代法中，共轭斜量法是最有效的方法之一。迭代法不像 LDL^T 分解法那样，可以一次求解多列右端项，迭代法一次只能求解一列右端项，因此下面假定 B 是单列的，文献中可以找到的共轭斜量法的计算过程是：

(1)任取初始近似值 $U^{(0)}$，计算

$$r^{(0)} = B - AU^{(0)}$$
$$D_{k-1} = (r^{(0)}, r^{(0)}) \qquad (1.38)$$

同时令 $P^{(1)} = r^{(0)}$。

(2)对于 $k=1,2,\cdots$，按下述次序计算 $U^{(k)}, r^{(k)}, P^{(k)}$，

$$z = AP^{(k)}$$
$$\alpha_k = D_{k-1}/(P^{(k)}, z)$$
$$U^{(k)} = U^{(k-1)} + \alpha_k P^{(k)}$$
$$r^{(k)} = r^{(k-1)} - \alpha_k z \qquad (1.39)$$
$$D_k = (r^{(k)}, r^{(k)})$$
$$\beta_k = D_k/D_{k-1}$$

$$P^{(k+1)} = r^{(k)} + \beta_k P^{(k)}$$

$$D_{k-1} = D_k$$

由计算过程可以看出，当矩阵阶数 N 很大时，由于矩阵 A 的元素不可能同时放在内存，故必然会遇到频繁的内外存交换。例如计算 $Z=AP^{(k)}$，要反复调用 A 的元素，则会大量消耗计算时间。因此设计共轭斜量法计算过程的核心是如何针对矩阵 A 的存储结构，设计最有效的矩阵与向量乘积的计算。

下面给出的计算

$$z(i) = \sum_{j=I1}^{i} a_{ij} p_j^{(k)} + \sum_{j=i+1}^{n} a_{ji} p_j^{(k)}, \quad i = 1,2,\cdots,n \tag{1.40}$$

的过程是针对下半带紧凑存储设计的。它尽量减少了 a_{ij} 的寻址和内外存的交换：

(1)计算

$$z(1) = a_{11} p_1^{(k)} \tag{1.41}$$

(2) 计算

$$\begin{aligned} z(1) &= z(1) + a_{21} p_2^{(k)} \\ z(2) &= a_{21} p_1^{(k)} + a_{22} p_2^{(k)} \\ &\cdots \end{aligned} \tag{1.42}$$

(3)对于 $j=I1, I1+1, \cdots, i-1$，计算

$$z(j) = z(j) + a_{ij} p_i^{(k)} \tag{1.43}$$

(4) 计算

$$z(i) = \sum_{j=I1}^{i} a_{ij} p_j^{(k)} \tag{1.44}$$

由上述计算过程可以看出，对于每行非零元素的一次调入内存和每个非零元素的一次寻址，要使用两次，从而减少了大约一半的寻找非零元素和内外存交换的时间，提高了计算效率。

值得指出，针对特定的求解方法设计特殊的存储总刚度矩阵的数据结构，或者针对特定的总刚度矩阵的存储结构设计特殊的计算过程，是设计最有效的求解算法的妙诀之一。

§1.2 瞬态有限元方法

上节简要叙述了与时间变量无关的静态有限元方法，但是在科学和工程分析领域，经常遇到与时间相关的问题需要处理。例如，动荷载作用下的运动平衡问题，强迫振动问题，瞬态热传导问题和自由振动问题等。对于这类问题，有限元方法也是十分有效的。经过众多学者的研究，已经发展了一系列分析瞬态问题的有限元方法；特别是发展了许多十分有效的求解瞬态有限元方程的方法。由于求解瞬态有限元方程的方法和求解稳态有限元方程的方法有重大区别，故本节在引入瞬态有限元离散化方程之后，重点叙述求解瞬态有限元方程的有效方法。。

1.2.1 瞬态有限元方程

对于结构工程分析问题，瞬态问题的引入是由于在物体上施加了与时间相关的瞬态荷载，由于物体运动状态发生变化出现了惯性加速度力，以及由于外界环境对运动着的物