

国外计算机科学经典教材

MATLAB Guide
to Finite Elements

MATLAB
有限元分析与应用

(德) P.I. Kattan 著
韩来彬 译



清华大学出版社

国外计算机科学经典教材

MATLAB 有限元分析与应用

(德) P.I.Kattan 著

韩来彬 译

清华大学出版社

北 京

内 容 简 介

本书采用当今在工程和工程教育方面非常流行的数学软件 MATLAB 来进行有限元的分析和应用。本书由简单到复杂,循序渐进地介绍了各种有限元及其分析与应用方法。书中提供了大量取自机械工程、土木工程、航空航天工程和材料科学的示例和习题,具有很高的工程应用价值。本书特别强调对 MATLAB 的交互应用,书中的每个示例都以交互的方式求解,使读者很容易就能把 MATLAB 用于有限元分析和应用。另外,本书还提供了大量免费资源。

本书示例丰富,语言简洁准确,可操作性强,可以作为高等院校相关专业学生的教材或参考书,对相关领域的工程技术人员也有很大的参考价值。

P I Kattan

MATLAB Guide to Finite Elements

EISBN: 3-540-43874-2

Copyright© 2003 by Springer-Verlag Berlin Heidelberg

All rights reserved For sale in the People's Republic of China only

Chinese simplified language edition published by Tsinghua University Press

本书中文简体字版由施普林格出版公司授权清华大学出版社出版。未经出版者书面许可,不得以任何方式复制或抄袭本书内容。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

北京市版权局著作权合同登记号 图字:01-2003-3639

图书在版编目(CIP)数据

MATLAB 有限元分析与应用/(德)卡坦著;韩来彬译 —北京:清华大学出版社,2004

书名原文:MATLAB Guide to Finite Elements

(国外计算机科学经典教材)

ISBN 7-302-08290-1

I.M II.①卡·②韩· III 公司-有限元分析-ly 用程序, MATLAB-教材 IV 0241.82

中国版本图书馆 CIP 数据核字(2004)第 019132 号

出 版 者:清华大学出版社 地 址:北京清华大学学研大厦

<http://www.tup.com.cn> 邮 编:100084

社 总 机:010-62770175 客户服务:010-62776969

组稿编辑:曹 康

文稿编辑:王晓娜

封面设计:康 博

版式设计:康 博

印 装 者:北京鑫海金澳胶印有限公司

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:21.25 字数:544 千字

版 次:2004 年 4 月第 1 版 2004 年 4 月第 1 次印刷

书 号:ISBN 7-302-08290-1/TP·5977

印 数:1~4000

定 价:40.00 元

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770175-3103 或(010)62795704

前 言

本书适用于酷爱有限元和 MATLAB 的有关人员。我们将使用流行的计算机软件包 MATLAB 作为矩阵计算器进行有限元分析。除了某些手动操作(特别是边界条件设置)外, 主要使用 MATLAB 解决冗长的矩阵计算问题。本书着重强调有限元方法的步骤。读者可能会发现, 本书中并没有使用现成的、类似黑盒子的 MATLAB 程序, 而是利用 MATLAB 对有限元习题的求解步骤进行详细的检验。线弹性结构力学的问题贯穿全书。本书的重点不在于大量的计算和编程, 而在于学习使用有限元方法进行计算并理解有限元基本概念。除了 MATLAB, 第 12、13 和 14 章还用到了 MATLAB 的 Symbolic Math 工具箱。

本书讨论的有限元类型包括: 弹簧元, 杆元, 二维和三维桁架元, 平面、空间梁和刚架元, 用于平面应力和平面应变问题的二维弹性元, 以及三维立体元。每一章只讨论一种类型的有限元。每一章的开始都会首先简单介绍该元的基本方程, 并随后利用示例阐述所用到的 MATLAB 函数的用法。然后在每一章的结束提供许多用于学生练习的练习题。

本书主要是为那些首次接触有限元分析的学生而编写的, 是有限元方法入门课程主教材的补充教材。由于有限元的计算通常包括矩阵和矩阵的操作, 因此同学们通常会用一个基于矩阵的软件包(如 MATLAB)进行这些计算。事实上, MATLAB 这个词的意思是矩阵实验室(MATrix LABoratory)。本书的主要特色在于: 共分为 15 个章节, 各章节都有很好的阐述并相互关联。

本书第 1 章包括了一个简短的关于 MATLAB 的使用指南, 概述了有限元分析方法的 6 个步骤。以后每章都系统地应用了这 6 个步骤。

我们为本书提供了一个免费资源文件(请登录 www.tupwk.com.cn 站点的下载页面), 其中包含了本书用到的 75 个 MATLAB 函数(M 文件)。这些函数组成了 MATLAB 有限元工具箱 (MATLAB Finite Element Toolbox), 该工具箱主要用于结构力学分析。它所提供的 MATLAB 函数设计简单, 使用方便。这些函数均在 MATLAB 5.0、5.3 和 6.1 版本中成功测试过, 并将继续用于后续版本。另外, 该资源文件中还提供了解答手册。该手册包含了所有习题的详细解答。

本书强调 MATLAB 的交互应用, 每个示例都以与 MATLAB 交互对话的方式解决, 并没有提供任何现成可用的子程序。

在本书的最后提供了所有习题的答案。

在此谨向在本书撰写过程中给予合作和帮助的 Springer-Verlag 出版公司(尤其是 Thomas Ditzinger 博士)致以谢意。并要特别感谢我的家人, 没有他们的支持和鼓励, 我是不可能完成本书的。尤其感谢 Nicola Kattan, 书中大部分插图都是由他绘制而成。

Peter I. Kattan

pkattan@nol.com.jo

2002 年 2 月

目 录

第 1 章 引言	1
1.1 有限元方法的步骤	1
1.2 用于有限元分析的 MATLAB 函数	2
1.3 MATLAB 指南	4
第 2 章 弹簧元	10
2.1 基本方程	10
2.2 用到的 MATLAB 函数	11
2.3 习题	23
第 3 章 线性杆元	24
3.1 基本方程	24
3.2 用到的 MATLAB 函数	25
3.3 习题	38
第 4 章 二次杆元	39
4.1 基本方程	39
4.2 用到的 MATLAB 函数	40
4.3 习题	52
第 5 章 平面桁架元	53
5.1 基本方程	53
5.2 用到的 MATLAB 函数	54
5.3 习题	76
第 6 章 空间桁架元	77
6.1 基本方程	77
6.2 用到的 MATLAB 函数	78
6.3 习题	91
第 7 章 梁元	92
7.1 基本方程	92
7.2 用到的 MATLAB 函数	93
7.3 习题	111

第 8 章 平面刚架元	113
8.1 基本方程	113
8.2 用到的 MATLAB 函数	114
8.3 习题	141
第 9 章 网格元	144
9.1 基本方程	144
9.2 用到的 MATLAB 函数	145
9.3 习题	158
第 10 章 空间刚架元	159
10.1 基本方程	159
10.2 用到的 MATLAB 函数	160
10.3 习题	182
第 11 章 线性三角形元	183
11.1 基本方程	183
11.2 用到的 MATLAB 函数	184
11.3 习题	207
第 12 章 二次三角形元	209
12.1 基本方程	209
12.2 用到的 MATLAB 函数	211
12.3 习题	231
第 13 章 双线性四边形元	232
13.1 基本方程	232
13.2 用到的 MATLAB 函数	234
13.3 习题	260
第 14 章 二次四边形元	262
14.1 基本方程	262
14.2 用到的 MATLAB 函数	264
14.3 习题	286
第 15 章 线性四面体(立体)元	287
15.1 基本方程	287
15.2 用到的 MATLAB 函数	290
15.3 习题	315
参考文献	317
习题答案	319

第1章 引言

这个简短的引言分为两部分，第一部分是对于有限元方法步骤的概括介绍，第二部分是 MATLAB 的简略使用指南。

1.1 有限元方法的步骤

有许多关于有限元分析的优秀教材，比如在参考文献[1-18]中列出的那些书目。因此，本书不准备对有限元理论或有限元方程进行详细地阐述和推导。每一章只总结概括主要的方程，这些章节都附有示例来说明这些方程。此外，全书只讨论线弹性结构力学的问题。

有限元方法用于解决工程问题的数值计算过程。本书假定所有的行为都是线弹性行为。虽然本书的问题都与结构工程相关，但有限元方法也同样适用于工程的其他领域。本书中使用有限元方法解决问题共包括 6 个步骤。对于有限元分析的 6 个步骤阐述如下：

(1) 离散化域——这个步骤包括将域分解成单元和节点。对于像桁架和刚架这类离散系统，已经离散化，这一步就不需要了。此处获得的结果应该已经是精确的。然而，对于连续系统，如板壳，这一步就变得至关重要，因为它只能得到近似的结果。因此解决方案的精确度取决于所使用的离散化方法。本书中，我们将手动完成这一步(对连续系统)。

(2) 写出单元刚度矩阵(element stiffness matrices)——写出域内每个单元的单元刚度矩阵。在本书中，这个步骤通过 MATLAB 实现。

(3) 集成整体刚度矩阵(global stiffness matrices)——这一步用直接刚度法(direct stiffness approach)实现。在本书中，该步骤借助于 MATLAB 实现。

(4) 引入边界条件——诸如支座(supports)、外加载荷(applied loads)和位移(displacements)等。本书中手动实现这一步骤。

(5) 解方程——这一步骤分解整体刚度矩阵并用高斯消去法求解方程组。在本书中，在用高斯消去法实现求解部分的时候需要手动分解矩阵。

(6) 后处理——得到额外的信息，如支反力、单元节点力和单元应力。本书中这一步骤通过 MATLAB 实现。

从以上步骤可以看出，解决问题的过程结合使用了 MATLAB 和某些有限的手动操作。使用手动操作只是实现离散化域(步骤 1)、设置边界条件(步骤 4)和分解整体刚度矩阵(步骤 5)。可以看出，所有冗长、反复的计算都由 MATLAB 完成。

1.2 用于有限元分析的 MATLAB 函数

我们为本书提供的免费资源文件(请登录 www.tupwk.com.cn 站点的下载页面)中包含了 75 个 MATLAB 函数(m 文件), 主要是由本书的作者专门为书中的有限元分析而编写的。它们组成了 MATLAB 有限元工具箱(MATLAB Finite Element Toolbox)。下面列出了所有这些函数。对于每个函数的详细使用方法, 读者可以参考每一章的内容。

SpringElementStiffness(k)

SpringAssemble(K, k, i, j)

SpringElementForces(k, u)

LinearBarElementStiffness(E, A, L)

LinearBarAssemble(K, k, i, j)

LinearBarElementForces(k, u)

LinearBarElementStresses(k, u, A)

QuadraticBarElementStiffness(E, A, L)

QuadraticBarAssemble(K, k, i, j, m)

QuadraticBarElementForces(k, u)

QuadraticBarElementStresses(k, u, A)

PlaneTrussElementLength(x₁, y₁, x₂, y₂)

PlaneTrussElementStiffness(E, A, L, theta)

PlaneTrussAssemble(K, k, i, j)

PlaneTrussElementForce(E, A, L, theta, u)

PlaneTrussElementStress(E, L, theta, u)

PlaneTrussInclinedSupport(T, i, alpha)

SpaceTrussElementLength(x₁, y₁, z₁, x₂, y₂, z₂)

SpaceTrussElementStiffness(E, A, L, thetax, thetay, thetaz)

SpaceTrussAssemble(K, k, i, j)

SpaceTrussElementForce(E, A, L, thetax, thetay, thetaz, u)

SpaceTrussElementStress(E, L, thetax, thetay, thetaz, u)

BeamElementStiffness(E, I, L)

BeamElementAssemble(K, k, i, j)

BeamElementForces(k, u)

BeamElementShearDiagram(f, L)

BeamElementMomentDiagram(f, L)

PlaneFrameElementLength(x₁, y₁, x₂, y₂)

PlaneFrameElementStiffness(E, A, I, L, theta)

PlaneFrameAssemble(K, k, i, j)

PlaneFrameElementForces(E, A, I, L, theta, u)

PlaneFrameElementAxialDiagram(f, L)

PlaneFrameElementShearDiagram(f, L)

PlaneFrameElementMomentDiagram(f, L)

PlaneFrameInclinedSupport(T, i, alpha)

GridElementLength(x₁, y₁, x₂, y₂)

GridElementStiffness(E, G, I, J, L, theta)

GridAssemble(K, k, i, j)

GridElementForces(E, G, I, J, L, theta, u)

SpaceFrameElementLength(x₁, y₁, z₁, x₂, y₂, z₂)

SpaceFrameElementStiffness(E, G, A, I_y, I_z, J, x₁, y₁, z₁, x₂, y₂, z₂)

SpaceFrameAssemble(K, k, i, j)

SpaceFrameElementForces(E, G, A, I_y, I_z, J, x₁, y₁, z₁, x₂, y₂, z₂, u)

SpaceFrameElementAxialDiagram(f, L)

SpaceFrameElementShearZDiagram(f, L)

SpaceFrameElementShearYDiagram(f, L)

SpaceFrameElementTorsionDiagram(f, L)

SpaceFrameElementMomentZDiagram(f, L)

SpaceFrameElementMomentYDiagram(f, L)

LinearTriangleElementArea(x_i, y_i, x_j, y_j, x_m, y_m)

LinearTriangleElementStiffness(E, NU, t, x_i, y_i, x_j, y_j, x_m, y_m, p)

LinearTriangleAssemble(K, k, i, j, m)

LinearTriangleElementStresses(E, NU, t, x_i, y_i, x_j, y_j, x_m, y_m, p, u)

LinearTriangleElementPStresses(sigma)

QuadTriangleElementArea(x₁, y₁, x₂, y₂, x₃, y₃)

QuadTriangleElementStiffness(E, NU, t, x₁, y₁, x₂, y₂, x₃, y₃, p)

QuadTriangleAssemble(K, k, i, j, m, p, q, r)

QuadTriangleElementStresses(E, NU, t, x₁, y₁, x₂, y₂, x₃, y₃, p, u)

QuadTriangleElementPStresses(sigma)

BilinearQuadElementArea(x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄)

BilinearQuadElementStiffness(E, NU, t, x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄, p)

BilinearQuadElementStiffness2(E, NU, t, x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄, p)

BilinearQuadAssemble(K, k, i, j, m, n)

BilinearQuadElementStresses(E, NU, x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄, p, u)

BilinearQuadElementPStresses(sigma)

QuadraticQuadElementArea(x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄)

QuadraticQuadElementStiffness(E, NU, t, x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄, p)

QuadraticQuadAssemble(K, k, i, j, m, p, q, r, s, t)

QuadraticQuadElementStresses(E, NU, x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄, p, u)

QuadraticQuadElementPStresses(sigma)

```
TetrahedronElementVolume(x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4)
TetrahedronElementStiffness(E, NU, x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4)
TetrahedronAssemble(K, k, i, j, m, n)
TetrahedronElementStresses(E, NU, x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4, u)
TetrahedronElementPStresses(sigma)
```

1.3 MATLAB 指南

本部分给出 MATLAB 的一个简短使用指南。详细内容可以查询参考文献[19-27]中列出的精品书或是参考文献[28-35]中列出的大量网上免费 MATLAB 应用指南。该指南并没有全面介绍 MATLAB，而是描述了在本书中使用到的 MATLAB 的一些基本命令。

在指南中，我们假定您在系统中已经成功地运行了 MATLAB 开发环境，并且已经准备好在 MATLAB 提示符(其标识为“>>”)下输入命令。如下所示，输入标量和简单的运算是很简单的：

```
>> 3*4+5

ans =

    17

>> cos(30*pi/180)

ans =

    0.8660

>> x=4

x =

     4

>> 2/sqrt(3+x)

ans =

    0.7559
```

如果不让 MATLAB 输出运算数据，可以在命令行的结尾处输入分号，如下面的示例。如果命令行结尾处没有使用分号，MATLAB 将显示输出内容：

```
>> y=32;  
>> z=5;  
>> x=2*y-z;  
>> w=3*y+4*z  
w =
```

116

MATLAB 区分大小写,也就是说用小写字母定义的变量和用大写字母定义的变量是不相同的。仔细看下面分别使用 x 和 X 作变量的示例:

```
>> x=1
```

```
x =
```

```
1
```

```
>> X=2
```

```
X =
```

```
2
```

```
>> x
```

```
x =
```

```
1
```

使用 `help` 命令可以获得所有 MATLAB 命令的详细用法。下面的示例给出了用 `help` 命令获取 `inv` 命令的方法。

```
>>help inv
```

```
INV      Matrix inverse.
```

```
INV(X) is the inverse of the square matrix X.
```

```
A warning message is printed if X is badly scaled or  
nearly singular.
```

```
See also SLASH, PINV, COND, CONDEST, NNLS, LSCOV.
```

```
Overloaded methods
```

```
help sym/inv.m
```

```
help zpk/inv.m
```

```
help tf/inv.m
```

```
help ss/inv.m
```

```
help lti/inv.m
help frd/inv.m
```

下面的示例展示了如何输入矩阵并实现简单的矩阵运算。

```
>> x=[1 2 3; 4 5 6; 7 8 9]
```

```
x =
```

```
1     2     3
4     5     6
7     8     9
```

```
>> y=[2; 0; -3]
```

```
y =
```

```
2
0
-3
```

```
>> w=x*y
```

```
w =
```

```
-7
-10
-13
```

现在，让我们求解下面的联立代数方程组。

$$\begin{bmatrix} 2 & -1 & 3 & 0 \\ 1 & 5 & -2 & 4 \\ 2 & 0 & 3 & -2 \\ 1 & 2 & 3 & 4 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 3 \\ 1 \\ -2 \\ 2 \end{Bmatrix} \quad (1.1)$$

使用高斯消去法计算上述方程组。在 MATLAB 中，使用反斜线符号 “\” 可以实现这一计算方法。如下所示：

```
>> A=[2 -1 3 0; 1 5 -2 4; 2 0 3 -2; 1 2 3 4]
```

```
A =
```

```
2     -1     3     0
1      5    -2     4
2      0     3    -2
1      2     3     4
```

```
>> b=[3; 1; -2; 2]
```

```
b =
```

```
3
1
-2
2
```

```
>> x=A\b
```

```
x =
```

```
1.9259
-1.8148
-0.8889
1.5926
```

显然，方程组的解是 $x_1=1.9259$, $x_2=-1.8184$, $x_3=-0.8889$, $x_4=1.5926$ 。另一种方法是用矩阵 A 的逆矩阵进行求解，也可以得到相同的答案。其求解过程如下：

```
>> x=inv(A)*b
```

```
x =
```

```
1.9259
-1.8148
-0.8889
1.5926
```

值得注意的是，用逆矩阵方法通常要比用高斯消去法花的时间长，尤其对大系统来说更是如此。在本书中，我们将主要使用高斯消去法(也就是使用反斜线符号“\”的方法)。

现在看下面的 5×5 矩阵 D:

```
>> D=[1 2 3 4 5; 2 4 6 8 9; 2 4 6 2 4; 1 1 2 3 -2; 9 0 2 3 1]
```

```
D =
```

```
1    2    3    4    5
2    4    6    8    9
2    4    6    2    4
1    1    2    3   -2
9    0    2    3    1
```

我们可以从矩阵中提取第 2~4 行的第 3~5 列作为子矩阵，如下所示：

```
>> E=D(2:4, 3:5)
```

E =

6	8	9
6	2	4
2	3	-2

提取 D 的第 3 列作为子矩阵，如下所示：

```
>> F=D(1:5, 3)
```

F =

3
6
6
2
2

提取 D 的第 2 行作为子矩阵，如下所示：

```
>> G=D(2, 1:5)
```

G =

2	4	6	8	9
---	---	---	---	---

提取矩阵 D 中的第 4 行第 3 列元素，如下所示：

```
>> H=D(4, 3)
```

H =

2

最后，为了绘制函数 $y = f(x)$ 的图形，需要定义足够多的 x 和 y 值，然后用 MATLAB 命令 `plot(x, y)` 绘制函数图形。下面给出了一个简单的示例：

```
>> x=[1 2 3 4 5 6 7 8 9 10]
```

x =

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

```
>> y=x.^2
```

```
y =
```

```
1    4    9   16   25   36   49   64   81  100
```

```
>> plot(x, y)
```

图 1-1 表示用 MATLAB 绘制得到的图形。通常用一个分开的图形窗口显示绘制的图形。图中没有为 x 和 y 坐标轴的标注题头。

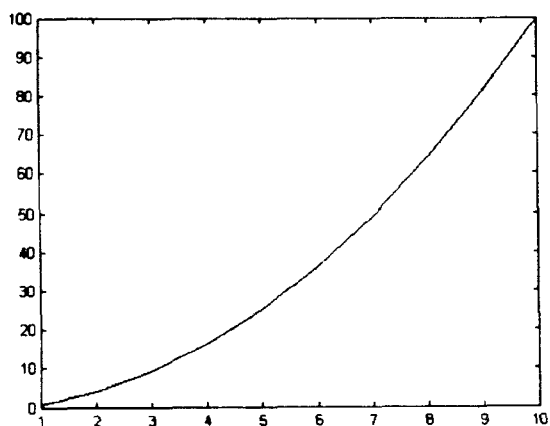


图 1-1 使用 MATLAB Plot 绘图

第2章 弹 簧 元

2.1 基本方程

弹簧元(spring element)是总体和局部坐标一致的一维有限元。值得注意的是, 弹簧元是可用的最简单的有限元。每个弹簧元都有两个节点(node), 如图 2-1 所示。这里将弹簧的刚度定义为 k 。因此, 单元刚度矩阵给定如下(参见文献[1]、[8]和[18]):

$$k = \begin{bmatrix} k & -k \\ -k & k \end{bmatrix} \quad (2.1)$$

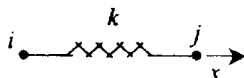


图 2-1 弹簧元

显而易见, 由于弹簧元只有两个自由度——每个节点有一个自由度, 因此弹簧元的刚度矩阵是一个 2×2 矩阵。因此, 如果一个弹簧元系统有 n 个节点, 那么整体刚度矩阵 K 的大小将是 $n \times n$ (这是因为每个节点只有一个自由度)。用直接刚度法得到单元刚度矩阵 $k_i (i=1, 2, 3, \dots, n)$, 再将单元刚度矩阵集成整体刚度矩阵, 就可以得到整体刚度矩阵 K 。例如, 一个联接系统中节点 4 和节点 5 的弹簧元的单元刚度矩阵 k , 可以将它的行和列加到整体刚度矩阵 K 中的第 4、5 行和第 4、5 列, 从而将单元刚度集成到整体刚度。一个名为 *SpringAssemble* 的 MATLAB 函数专门用于这一目的。后面的示例将会详细介绍该函数的用法。

一旦得到整体刚度矩阵 K , 就可以列出以下方程组:

$$[K]\{U\} = \{F\} \quad (2.2)$$

式中, U 是结构节点位移矢量, F 是结构节点载荷矢量。在这一步中, 边界条件被手动赋值给矢量 U 和 F 。这样, 矩阵方程(2.2)可以用分解和高斯消去法求解。一旦求得未知的位移和支反力, 就可以用下式求得单元的节点力, 如下所示:

$$\{f\} = [k]\{u\} \quad (2.3)$$

式中 f 是 2×1 的单元节点力矢量, u 是 2×1 的单元节点位移矢量。

2.2 用到的 MATLAB 函数

弹簧元用到的 3 个 MATLAB 函数是：

SpringElementStiffness(k)——该函数计算弹性系数为 k 的弹簧的单元刚度矩阵。它返回 2×2 的单元刚度矩阵 k 。

SpringAssemble(K, k, i, j)——该函数将弹簧联接节点 i (最左端)和 j (最右端)的单元刚度矩阵 k 集成为整体刚度矩阵 K 。每集成一个单元，该函数将返回 $n \times n$ 的整体刚度矩阵 K 。

SpringElementForces(k, u)——该函数根据单元刚度矩阵 k 和单元节点位移矢量 u 计算单元节点力矢量。它返回 2×1 的单元节点力矢量 f 。

下面列出了每个函数的 MATLAB 源代码：

```
function y = SpringElementStiffness(k)
%SpringElementStiffness      This function returns the element stiffness
%                             matrix for a spring with stiffness k.
%                             The size of the element stiffness matrix
%                             is 2x2.
y = [k -k; -k k];
```

```
function y = SpringAssemble(K, k, i, j)
%SpringAssemble      This function assembles the element stiffness
%                    matrix k of the spring with nodes i and j into the
%                    global stiffness matrix K.
%                    This function returns the global stiffness matrix K
%                    after the element stiffness matrix k is assembled.
K(i, i) = K(i, i) + k(1, 1);
K(i, j) = K(i, j) + k(1, 2);
K(j, i) = K(j, i) + k(2, 1);
K(j, j) = K(j, j) + k(2, 2);
y = K;
```

```
function y = SpringElementForces(k, u)
%SpringElementForces      This function returns the element nodal force
%                          vector given the element stiffness matrix k
%                          and the element nodal displacement vector u.
y = k * u;
```

示例 2.1:

考虑图 2-2 所示的二弹簧元结构，假定 $k_1=100\text{kN/m}$ ， $k_2=200\text{kN/m}$ ， $P=15\text{kN}$ 。求：

- (1) 系统的整体刚度矩阵。
- (2) 节点 2 和节点 3 的位移。