第 7 章　在科技及工程中的应用实例

7.1　由拉压杆组成的桁架结构

由 13 根拉压杆件组成的桁架结构，如图 7-1 所示，13 个平衡方程已给出，它们来自 6 个中间节点，每个节点有 x,y 两个方向的平衡方程，还有一个整体结构的 y 方向平衡方程。现求其各杆所受的力。

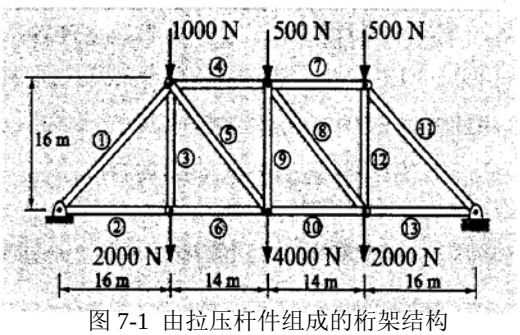


图 7-1 由拉压杆件组成的桁架结构

解：按照题给方程组改写成矩阵形式，令 $k_1=\cos\theta_1=14/\sqrt{16\wedge 2+14\wedge 2}=0.6585$ $k_2=\cos\theta_2=16/\sqrt{16\wedge 2+16\wedge 2}=0.7071$ 列 $k_1=\sin\theta_1=16/\sqrt{16\wedge 2+14\wedge 2}=0.7526$ 方程时假设各杆的受力均为拉力，其相应的方程组及化为矩阵后的形式为：

$$\left.\begin{array}{l}
F_2+k_2F_1=0\\
-F_2+F_6=0\\
F_3=2000\\
F_4+k_1F_5-k_2F_1=0\\
k_2F_1+F_3+k_3F_5=-1000\\
F_7+k_1F_8-F_4=0\\
k_3F_8+F_9=-500\\
F_{10}-k_1F_5-F_6=0\\
F_9+k_3F_5=4000\\
k_2F_{11}-F_7=0,\\
k_2F_{11}+F_{12}=-500\\
F_{12}+k_3F_8=2000\\
F_{13}+k_2F_{11}=0
\end{array}\right\}\Rightarrow
\begin{bmatrix}
k_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
-k_2 & 0 & 0 & 1 & k_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
k_2 & 0 & 1 & 0 & k_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & -1 & 0 & 0 & 1 & k_1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & k_3 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & -k_1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & k_3 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & k_2 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_2 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & k_3 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_2 & 0 & 1
\end{bmatrix}
\begin{bmatrix}F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\\F_8\\F_9\\F_{10}\\F_{11}\\F_{12}\\F_{13}\end{bmatrix}=
\begin{bmatrix}0\\0\\2000\\0\\-1000\\0\\-500\\0\\4000\\0\\-500\\2000\\0\end{bmatrix}
\qquad (7.1.1)$$

将它看作 **A*F=B，**编成的程序为 pla701，核心语句为给 **A,B** 赋值，再求 **F=A\B**，结果为：

F=[-7236; 5117; 2000; -6969; 2812; 5117; -4883; -3167; 1883; 6969; -6906; 4383; 4883]

其中负号表示杆受的是压力。

7.2　格型梯形滤波器系统函数的推导

使用计算机解题后，用矩阵模型几乎是最简便的数学方法了。这将给后续课的建模

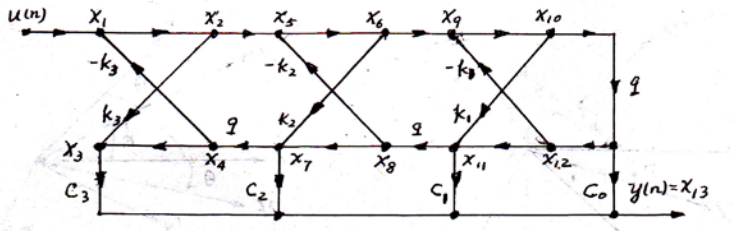


图 7-2　三阶格型梯形滤波器的结构图

和计算带来革命性的好处。例如要求出图 7-2 所示的滤波器的系统函数：

先列出方程，令 $q=z^{-1}$，得到

$x_1= u- k_3x_4$;　$x_2=x_1$;　$x_3=k_3x_2+x_4$;　$x_4=qx_7$; $x_5= x_2- k_2x_8$;　$x_6=x_5$;

$x_7=k_2x_6+x_8$;　$x_8=qx_{11}$; $x_9= x_6- k_1x_{12}$;　$x_{10}=x_9$;　$x_{11}=k_1x_{10}+x_{12}$; $x_{12}=qx_{10}$;

$x_{13}=y= C_0x_{12}+ C_1x_{11}+ C_2x_7+ C_3x_3$

这是一组含有 13 个变量的 13 个联立方程，用过去的手工方法一个一个消元，理论上是可行的，但它运算极其繁琐，可以预期，95%以上的师生恐怕一个小时也解不出来，而且做对的概率极低。

用矩阵的思路和方法来解就完全不同，它不是通过消元来减少变量，而是想办法补上所有的零元素，把方程扩充为完整的矩阵形式：

$$\left.\begin{array}{l} x_1 = u - k_3x_4 \\ x_2 = x_1 \\ x_3 = k_3x_2 + x_4 \\ x_4 = qx_7 \\ x_5 = x_2 - k_2x_8 \\ x_6 = x_5 \\ x_7 = k_2x_6 + x_8 \\ x_8 = qx_{11} \\ x_9 = x_6 - k_1x_{12} \\ x_{10} = x_9 \\ x_{11} = k_1x_{10} + x_{12} \\ x_{12} = qx_{10} \\ x_{13} = \cdots\cdots \end{array}\right\} \Rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \end{bmatrix} = \begin{bmatrix} 0, & 0, & 0, & -k_3, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\ 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\ 0, & k_3, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0, & q, & 0, & 0, & 0, & 0, & 0, & 0 \\ 0, & 1, & 0, & 0, & 0, & 0, & 0, & -k_2, & 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & k_2, & 0, & 1, & 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & q, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & -k_1, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & k_1, & 0, & 1, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & q, & 0, & 0, & 0 \\ 0, & 0, & C_3, & 0, & 0, & 0, & C_2, & 0, & 0, & 0, & C_1, & C_0, & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} u$$

$$\Rightarrow \mathbf{X = QX + PU}, \Rightarrow \mathbf{X / U = inv(I - Q) * P}$$

看似把模型搞复杂了，其实计算却非常容易。程序 pla703 先对 **P，Q** 矩阵赋值，键入 **W = inv(I - Q) * P**，马上就得出了系统函数。

编程时要注意，本例虽然是数值计算，但计算的内容中带有 z 变换算子 $q=z^{-1}$，所以 **P，Q** 矩阵仍然必须用符号属性，对 **P，Q** 赋值时第一个元素必须取含 q 的算式。熟练后不必列出 **Q** 和 **P** 的矩阵形式，可以按其下标规律直接进行元素赋值。

用以下参数：$k_0=1$, $k_1=1/4$, $k_2=1/2$, $k_3=1/3$, $C_0=-0.2$, $C_1=0.8$, $C_2=1.5$, $C_3=1$，编成了程序 `pla703`。运行此程序就得到：

$$W(13) = \frac{x(13)}{u} = \frac{24q^3 + 49q^2 + 42.9q + 30.8}{8q^3 + 15q^2 + 13q + 24} = \frac{24z^{-3} + 49z^{-2} + 42.9z^{-1} + 30.8}{8z^{-3} + 15z^{-2} + 13z^{-1} + 24}$$

用矩阵模型解信号流图的最大优点是一步到位，依靠计算机，既快速，又极易查错。

7.3　计算频谱用的 DFT 矩阵

有限长序列 $x(n)$ $(0 \leqslant n \leqslant N-1)$ 有 N 个样本值。它的傅里叶变换 $X(k)$ 在频率区间（$0 \leqslant \omega < 2\pi$）的 N 个等间隔分布的点 $\omega_k=2\pi k/N$ $(0 \leqslant k \leqslant N-1)$ 上也有 N 个样本值。这两组有限长的序列之间可以用简单的关系联系起来：

$$\mathbf{X}(k)=\sum_{n}^{N-1}x(n)W_N^{kn}\qquad 0\le k\le N-1$$

其中$W_N=\exp\left(2\pi j/N\right)$是一个相角$2\pi/N$为的单位向量，也称为旋转因子。$X(k)$就称为 $x(n)$的离散傅里叶变换(DFT)，也就是 x(n)的频谱。用矩阵来表示,可写成:

$$\mathbf{X}=\begin{bmatrix}X(0)\\X(1)\\\vdots\\X(N-1)\end{bmatrix}=\begin{bmatrix}1&1&\cdots&1\\1&W_N^{1}&\cdots&W_N^{N-1}\\1&W_N^{2}&\cdots&W_N^{2(N-1)}\\\vdots&\vdots&\ddots&\vdots\\1&W_N^{N-1}&\cdots&W_N^{(N-1)(N-1)}\end{bmatrix}\begin{bmatrix}x(0)\\x(1)\\\vdots\\x(N-1)\end{bmatrix}=\mathbf{W}\cdot\mathbf{x}$$

所以信号频谱的计算，可以简单地用一个矩阵乘法来完成。信号是 N×1 维数组，矩阵 **W** 称为 DFT 矩阵，它是 N×N 维的复数阵。把矩阵乘法看作一个变换，我们就可以把频谱计算看作信号从时域到频域的线性变换。

这个矩阵运算可用下列几条 MATLAB 语句来实现，程序名为 pla704。

```
xn=input('xn=(长度为N的数组) '), N=length(xn);  % 输入数据
n = [0:1:N-1]; k = [0:1:N-1];      % 设定n和k的行向量
WN = exp(-j*2*pi/N);               % 设定Wn因子
nk = n'*k;                         % 产生一个含nk值的N乘N维的整数矩阵
WNnk = WN .^ nk;                   % 求出W矩阵
Xk = xn * WNnk                     % 求出离散傅里叶级数系数
plot(k,Xk)                         % 画出幅频特性
```

前两条语句无须解释。第三条语句把 n 转为列向量 n'，再与行向量 k 做矩阵乘，它产生的是一个整数矩阵：

$$\text{n'*k}=\begin{bmatrix}0\\1\\2\\\vdots\\N-1\end{bmatrix}\begin{bmatrix}0,1,2,\cdots,N-1\end{bmatrix}=\begin{bmatrix}0&0&\cdots&0\\0&1&\cdots&N-1\\0&2&\cdots&2\times(N-1)\\\vdots&\vdots&\ddots&\vdots\\0&N-1&\cdots&(N-1)\times(N-1)\end{bmatrix}$$

这个整数方阵恰好是算式 $\mathbf{X}=\mathbf{W}\cdot\mathbf{x}$ 中 **W** 矩阵的指数部分。第四条语句就产生了 **W** 矩阵，而最后一条语句就完成了 DFS 的全部运算。由于实际上离散傅里叶级数并不太用到。它已被下节将介绍的离散傅里叶变换取代，而且两者的计算程序是完全相同的。所以如果写成 MATLAB 子程序，它可以命名为 DFT。

在这 5 行 DFT 子程序前面，加上输入语句：

```
xn=input('xn= '); N=length(xn);
```

在子程序后面，加上绘图语句：

```
subplot(2,1,1),stem(xn,'.');
subplot(2,1,2),stem(abs(Xk),'.')
```

就得到本题的程序 pla604。

运行 pla604，并按提示输入 xn，即可在子图 1 上得到 xn 序列的波形，并在子图 2 上得到此信号的幅频特性。例如输入序列为 xn=[ones(1,64),zeros(1,192)]，则得到的时域波形及其频谱图如图 7-3。

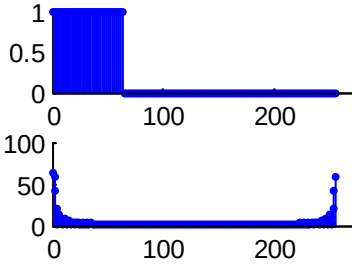


图 7-3　序列 x(n)及其频谱

计算每一个 DFT 分量 $X(i)$，需要做 N次复数乘法和 N-1 次复数加法。而要完成整个 DFT，求出 X，则需要做 $N\times N$次复数乘法和 $N(N$-1)次复数加法。另外，它占的数据存储量为 $N\times N$。在频谱计算中，N 的典型值是 1024。所以参与运算的数据达百万，需要的存储量

超过 16M， 乘法运算的次数超过数百万。因此，在低档的微机上，当 N较大时，MATLAB 会拒绝执行这个程序，并警告内存不足。实际上，MATLAB 内置的是加快计算并节省内存快速傅里叶变换(FFT)程序。

从这里也可以看出，矩阵给我们提供了一个组织海量数据进行运算的最好方法，对上百万数据进行赋值、运算和绘图，只用了几行程序。线性代数的重要性之所以在近 20 年来可与微积分相妣美，这是一个重要原因。如果学线性代数而不涉及工程计算实践，那就无法真正体会线性代数的精髓所在。

7.4　显示器色彩制式转换问题

彩色显示器使用红(R)、绿(G)和蓝(B)光的叠成效应生成颜色。 显示器屏幕的内表面由微粒象素组成, 每个微粒包括三个荧光点: 红、绿、蓝。 电子枪位于屏幕的后方, 向屏幕上每个点发射电子束。计算机从图形应用程序或扫描仪发出数字信号到电子枪, 这些信号控制电子枪设置的电压强度. 不同 RGB 的强度组合将产生不同的颜色. 电子枪由偏转电磁场帮助瞄准以确保快速精确地屏幕刷新。

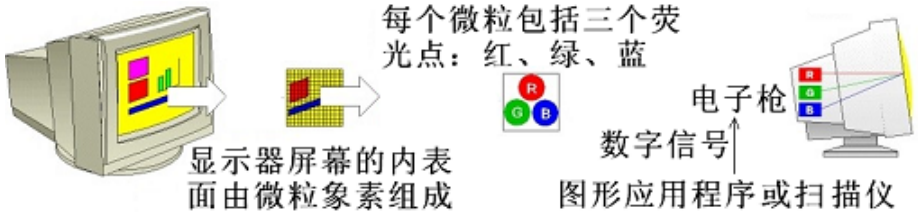


图 7-4　彩色显示器的工作原理

颜色模型规定一些属性或原色, 将颜色分解成不同属性的数字化组合. 这就是色彩制式的转换问题。

观察者在屏幕上实际看到的色彩要受色彩制式和屏幕上荧光点数量的影响。. 因此每家计算机屏幕制造商都必须在(*R*, *G*, *B*)数据和国际通行的 CIE 色彩标准之间进行转换, CIE 标准使用三原色, 分别称为 *X*, *Y* 和 *Z*。 针对短余辉荧光点的一类典型转换是

$$\begin{pmatrix} 0.61 & 0.29 & 0.150 \\ 0.35 & 0.59 & 0.063 \\ 0.04 & 0.12 & 0.787 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \Rightarrow \mathbf{A\alpha=\beta} \quad .$$

计算机程序把用 CIE 数据(*X*, *Y*, *Z*)表示的色彩信息流发送到屏幕，求屏幕上的电子枪将这些数据转换成(*R*, *G*, *B*)数据的方程。现在要根据 CIE 数据(*X*, *Y*, *Z*)计算对应的(*R*, *G*, *B*)数据, 就是等式 $\boldsymbol{A\alpha=\beta}$ 中 $\boldsymbol{A}$ 和$\boldsymbol{\beta}$ 已知, 求$\boldsymbol{\alpha}$。如果 $\boldsymbol{A}$ 是可逆矩阵, 则由 $\boldsymbol{A\alpha=\beta}$可得$\boldsymbol{\alpha=A^{-1}\beta}$.

解：在 Matlab 命令窗口输入以下命令

```
A = [0.61,0.29,0.15;0.35,0.59,0.063;0.04,0.12,0.787]; %
B = inv(A),
```

程序执行的结果为：

```
B =    2.2586   -1.0395   -0.3473
      -1.3495    2.3441    0.0696
       0.0910   -0.3046    1.2777
```

可见将 CIE 数据(*X*, *Y*, *Z*)转换成(*R*, *G*, *B*)数据的方程为$\boldsymbol{\alpha=B\beta}$

在彩色电视技术中，还有许多地方会用到线性变换.比如民用电视信号的发送并不直接采用

(*R, G, B*)数据，而是 使用向量(*Y, I, Q*)来描述每种颜色.其中 Y 称为亮度信号，I 和 Q 为色差信号，这样的制式可以达到彩色和黑白的兼容.如果屏幕是黑白的, 则只用到了 *Y*，这比 CIE 数据能提供更好的单色图象。YIQ 与“标准”RGB 色彩之间的对应如下

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

它的逆变换矩阵留给读者自行计算。

7.5　人员流动问题

某试验性生产线每年一月份进行熟练工与非熟练工的人数统计, 然后将$1/6$熟练工支援其他生产部门, 其缺额由招收新的非熟练工补齐。新、老非熟练工经过培训及实践至年终考核有$2/5$成为熟练工，假设第一年一月份统计的熟练工和非熟练工各占一半，求以后每年一月份统计的熟练工和非熟练工所占百分比。

设第 n 年一月份统计的熟练工和非熟练工所占百分比分别为 x_n 和 y_n, 记成向量 $\mathbf{Z_n} = \begin{pmatrix} x_n \\ y_n \end{pmatrix}$.

因为第一年统计的熟练工和非熟练工各占一半, 所以 $\mathbf{Z_1} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$。为了求以后每年的熟练工和非熟练工所占百分比, 先求 $\mathbf{Z_{n+1}}$ 与 $\mathbf{Z_n}$ 的关系式。根据已知条件可得:

$$x_{n+1} = (1-\frac{1}{6})x_n + \frac{2}{5}(\frac{1}{6}x_n + y_n) = \frac{9}{10}x_n + \frac{2}{5}y_n,$$

$$y_{n+1} = (1-\frac{2}{5})(\frac{1}{6}x_n + y_n) = \frac{1}{10}x_n + \frac{3}{5}y_n,$$

即

$$\mathbf{Z_{n+1}} = \begin{pmatrix} 9/10 & 2/5 \\ 1/10 & 3/5 \end{pmatrix} \mathbf{Z_n} = \mathbf{A}\mathbf{Z_n}$$

$$\mathbf{Z_{n+1}} = \boldsymbol{A}\,\mathbf{Z_n} = \boldsymbol{A}^2\,\mathbf{Z_{n-1}} = \ldots = \boldsymbol{A}^n\,\mathbf{Z_1}.$$

将 $\mathbf{A}$ 对角化成为 $\mathbf{A} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^{-1}$ 可得到简明易算的结果：$\mathbf{Z_{n+1}} = \left(\mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^{-1}\right)^n \mathbf{Z_1} = \mathbf{P}\,\boldsymbol{\Lambda}^n\,\mathbf{P}^{-1}\mathbf{Z_1}$。

键入命令[P,D] = eig(A)，得 $\mathbf{P} = \begin{bmatrix} 0.9701 & -0.7071 \\ 0.2425 & 0.7071 \end{bmatrix}$，$\mathbf{D} = \boldsymbol{\Lambda} = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix}$，于是便有：

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{bmatrix} 0.9701 & -0.7071 \\ 0.2425 & 0.7071 \end{bmatrix} \begin{bmatrix} 1^n & 0 \\ 0 & 0.5^n \end{bmatrix} \begin{bmatrix} 0.9701 & -0.7071 \\ 0.2425 & 0.7071 \end{bmatrix}^{-1} \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} = \begin{bmatrix} 0.8 - 0.3/(2^n) \\ 0.2 + 0.3/(2^n) \end{bmatrix},$$

当 n 不断增加时，x_{n+1}，y_{n+1} 分别趋向于 0.8 和 0.2。

7.6　二氧化碳分子结构的振动频率

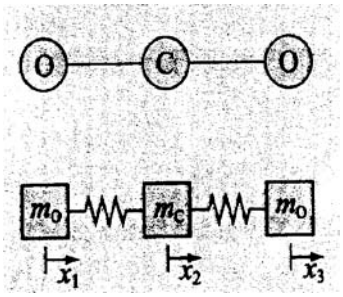
二氧化碳分子可看成中间一个碳原子，左右分别以弹簧（化学键）联接两个氧原子，构成一个三质量振动系统。按牛顿定律，其动力学方程为：

$$m_o x_1'' = k(x_2 - x_1) = -kx_1 + kx_2$$

$$m_c x_2'' = k(x_3 - x_2) - k(x_2 - x_1) = kx_1 - 2kx_2 + kx_3$$

$$m_o x_3'' = -k(x_3 - x_2) = kx_2 - kx_3$$

其中：$x_1'' = \frac{d^2 x_1}{dt^2}, x_2'' = \frac{d^2 x_2}{dt^2}, x_3'' = \frac{d^2 x_3}{dt^2}$



题 7-8 图　二氧化碳分子结构

k=14.2e2 为化学键的弹性常数，mo=16*1.6605e-27, mc=12*1.6605e-27 分别为氧原子和碳原子的质量，x_1,x_2,x_3 为各原子的轴向位移。今要求其分子振动的频率。

解：令 $y_1 = x_1', y_1' = x_1'', y_2 = x_2', y_2' = x_2'', y_3 = x_3', y_3' = x_3'',$

设 h=k/mo=5.3448e+028，则 k/mc=h*mo/mc=h*4/3，将方程组写成矩阵形式，便可得到：

$$\left.\begin{array}{l} m_o y_1' = -kx_1 + kx_2 \\ x_1' = y_1 \\ m_c y_2' = -2kx_2 + kx_1 + kx_3 \\ x_2' = y_2 \\ m_o y_3' = kx_2 - kx_3 \\ x_3' = y_3 \end{array}\right\} \Rightarrow \begin{bmatrix} y_1' \\ x_1' \\ y_2' \\ x_2' \\ y_3' \\ x_3' \end{bmatrix} = \begin{bmatrix} 0 & -h & 0 & h & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4h/3 & 0 & -8h/3 & 0 & 4h/3 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & h & 0 & -h \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ x_1 \\ y_2 \\ x_2 \\ y_3 \\ x_3 \end{bmatrix} \Rightarrow \frac{d\mathbf{X}}{dt} = \mathbf{AX}$$

这是标准的一阶线性微分方程组的矩阵形式。它在初始条件 $X=X_0$ 下的解为：

$$\mathbf{X} = \mathbf{X_0}\exp(\mathbf{At}) = \mathbf{X_0}\mathbf{P}\exp(\mathbf{\Lambda t})\mathbf{P}^{-1}$$

其中 **P**，**Λ** 分别为 A 的特征向量和特征值。因为题目只要求求振动频率而不是全部过程，所以也不需要求特征向量，这里就采用分步的算法，先用 f=poly(A)求特征方程的系数多项式，再用 L=roots(f) 求它的特征根。

于是可编制 MATLAB 程序 pla709，核心语句如下：

```
k=14.2e2, mo=16*1.6605e-27, mc=12*1.6605e-27, h=k/mo
A=[0,-h,0,h,0,0;1,0,0,0,0,0;0,h*4/3,0,-h*8/3,0,h*4/3;0,0,1,0,0,0;0,0,0,h,0,-h;0,0,0,0,1,0]
f=poly(A), L=roots(f)
```

程序运行的结果是：

```
A =   1.0e+029 *[ 0      -0.5345         0       0.5345         0         0
                  0            0         0            0         0         0
                  0       0.7126         0      -1.4253         0    0.7126
                  0            0    0.0000            0         0         0
                  0            0         0       0.5345         0   -0.5345
                  0            0         0            0    0.0000         0 ]
f = 1.0e+071*[ 0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    2.5711]
L=   1.0e+014 *[ -0.0000 ± 4.4269i；-0.0000 ± 2.3119i；0.0000 ± 0.0000i ]
```

L 是人为缩减显示格式后的形式，f 是特征多项式的系数向量，由于各元素值差别太大而无法用一个矩阵显示，只好单个显示，依次键入 f(1),…,f(7)，得知

f(1)=1, f(2)=-0.0119, f(3)=2.4942e+029, f(4) = -6.0409e+027, f(5) = 1.0474e+058, f(6) = -2.8876e+056, f(7) = 2.5711e+071

L 是矩阵 A 的特征值，矩阵是六阶的，共有六个特征值，除去两个零后，剩下两对共轭虚

数 1.0e+014 * [±4.4269i，±2.3119i]，分别表示了两种振动角频率【弧度/秒】。

将 L 除以 2π 后可得到赫兹为单位的两种振动频率 70.456 和 36.795 THz【太赫兹】，用这个频率除以光速 3*10^8【米/秒】将得到两种波长 4.258 和 8.533μm【微米】。

7.7　二自由度机械振动

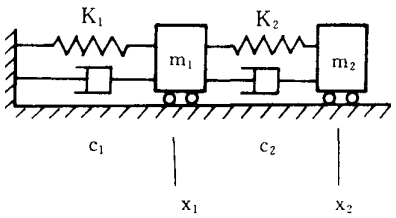


图 7-8　两自由度机械振动模型

图 7-9 表示了一个由两个质量、两个弹簧和两个阻尼器构成的二自由度振动系统，今要求在给定初始位置和初始速度下的运动。

解：设 x1 和 x2 分别表示两个质量关于它们平衡位置的偏移，则此二自由度振动系统的一般方程为：

$$m_1\ddot{x}_1+(c_1+c_2)\dot{x}_1-c_2\dot{x}_2+(k_1+k_2)x_1-k_2x_2=0$$
$$m_2\ddot{x}_2+c_2(\dot{x}_2-\dot{x}_1)+k_2(x_2-x_1)=0 \qquad (7.10.1)$$

可写成矩阵形式：

$$\mathbf{M}\ddot{\mathbf{X}}+\mathbf{C}\dot{\mathbf{X}}+\mathbf{K}\mathbf{X}=\mathbf{0} \qquad (7.10.2)$$

其中

$$\mathbf{M}=\begin{bmatrix}m_1 & 0\\ 0 & m_1\end{bmatrix},\ \mathbf{C}=\begin{bmatrix}c_1+c_2 & -c_2\\ -c_2 & c_2\end{bmatrix},\ \mathbf{K}=\begin{bmatrix}k_1+k_2 & -k_2\\ -k_2 & k_2\end{bmatrix},\ \mathbf{X}=\begin{bmatrix}x_1\\ x_2\end{bmatrix} \qquad (7.10.4)$$

这是一个四阶的常系数齐次微分方程组，给定其初始位置 X_0 和初始速度 X_{d0}：

$$\mathbf{X}_0=\begin{bmatrix}x_{10}\\ x_{20}\end{bmatrix},\quad \dot{\mathbf{X}}_0=\begin{bmatrix}\dot{x}_{10}\\ \dot{x}_{20}\end{bmatrix}=\mathbf{X}_{d0}=\begin{bmatrix}x_{1d0}\\ x_{2d0}\end{bmatrix} \qquad (7.10.4)$$

引进符号 $\mathbf{x_d}$ 是为了在 MATLAB 编程中给导数以变量名的需要。用解析法求这个方程的解非常麻烦，只有当 **C=0**，即无阻尼时，系统可解耦为两种独立的振动模态，通常书上只给出解耦简化后的解。

有了 MATLAB 工具，根本无需设 **C＝0**，也无需解耦，就可以用很简洁的程序求出其数值解。其基本思路是把原始非常化成典型的四个一阶方程构成的状态方程组：

$$\dot{\mathbf{Y}}=\mathbf{A}\mathbf{Y} \qquad (7.10.5)$$

这个方程在初始条件 $\mathbf{Y=Y_0}$ 下的解为 $\mathbf{Y}=\mathbf{Y_0}e^{\mathbf{A}t}$。用 MATLAB 表示为 **Y=Y0*expm(A*t)**。其中 expm 表示把(A*t)看成矩阵来求其指数。已经知道标量 x 求指数的方法，求矩阵指数虽然要麻烦一些，但概念是相仿的。我们不必都弄清细节。MATLAB 提供了 expm, expm1, …等多种函数供用户调用。所以我们只要把 **Y,Y0** 和 **A** 找到就行。

先把方程（2）写成两个一阶矩阵方程：

$$\left.\begin{aligned}&\dot{\mathbf{X}}=\mathbf{X_d}\\ &\dot{\mathbf{X}}_\mathbf{d}=\ddot{\mathbf{X}}=\mathbf{-M\backslash CX_d-M\backslash KX}\end{aligned}\right\}\Rightarrow\begin{bmatrix}\dot{\mathbf{X}}\\ \dot{\mathbf{X}}_\mathbf{d}\end{bmatrix}=\begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{-M\backslash K} & \mathbf{-M\backslash C}\end{bmatrix}\begin{bmatrix}\mathbf{X}\\ \mathbf{X_d}\end{bmatrix} \qquad (7.10.6)$$

于是

$$\mathbf{Y}=\begin{bmatrix}\mathbf{X}\\ \mathbf{X_d}\end{bmatrix},\ \mathbf{Y_0}=\begin{bmatrix}\mathbf{X_0}\\ \mathbf{X_{d0}}\end{bmatrix},\ \mathbf{A}=\begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{-M\backslash K} & \mathbf{-M\backslash C}\end{bmatrix} \qquad (7.10.7)$$

对于本题的二自由度系统，$\mathbf{X}=\begin{bmatrix}x_1\\ x_2\end{bmatrix}$ 和 $\mathbf{X_d}=\begin{bmatrix}x_{d1}\\ x_{d2}\end{bmatrix}$，所以 **Y** 和 **Y0** 都是 4×1 的单列变量；

由于 **A** 中的四个元素都是 2×2 方阵，**A** 是 4×4 方阵。对于更多自由度的系统，公式(7)都是正确的。

下面给出二自由度系统的一个数值例，设 m1=1; m2=9; k1 = 4; k2=2; c1 和 c2 可由用户输入。求在初始条件 `x0 = [1;0]`和 `xd0 = [0;-1]`下，系统的输出 `x1,x2` 曲线。

根据上面的模型可以写出程序 pla710，运行此程序，输入 c1=0.2,c2=0.5 所得的结果见图 7-9。从中可清楚地看到振动的两种模态。特别 是 x1 的运动反映了两种模态的迭合。给出不同的初始条件，各模态的幅度也会变化。输入 c1=0,c2=0 所得的结果见图 7-10，这时两种振动模态是可

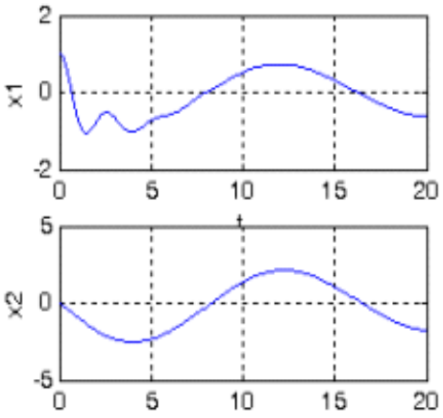


图 7-9　二自由度振动的输出波形

c1=0.2，c2=0.8

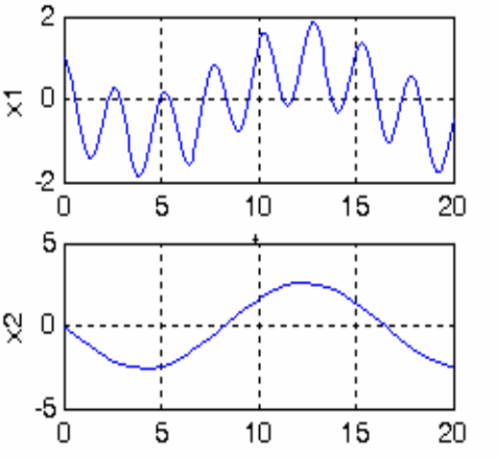


图 7-10　二自由度振动波形 c1=c2=0

解耦的。可以看出，用计算机解题时，问题和数据的复杂性对解题的过程并无根本影响。通常我们选择比较简单，且有解析解的数据作为检验程序之用。一旦确定程序正确，它就可用在很复杂的情况。例如作者就曾把 b 本例题的核心语句用在一个五自由度的系统上，解一个十阶的线性方程组，同样得出满意的结果。

7.8　FIR 数字滤波器最优化设计[12]

设给定滤波器的预期幅频特性 $D(\omega)$在 $\omega=\omega_i(i=1,2,\ldots,K)$的 K 个频点上的值，若实际滤波器的脉冲响应的长度为 N=2L+1，则其幅特性与预期特性相拟合的方程为：

$$A(\omega_i)=\sum_{n=0}^{L} d(n)\cos n\omega_i = D(\omega_i)=D_i \qquad (i=1,2,\ldots,K)$$

这 K 个方程联立。由于 $A(\omega_i)$ 是 $L+1$ 个谐波的线性组合，这方程组中的待定参数 d(n) 有 $L+1$ 个。

如果 $K=L+1$，即方程数与未知数的数目相等，则这个线性方程组是适定的，恰好可以解了这些系数。

如果 $K>L+1$，即方程数比未知数的数目多，形成了所谓超定方程组。那就不可能找到精确满足这些方程组的系数 $d(n)$，只能找到最近似地满足这些方程的最小二乘解。

如果 $K<L+1$，则形成了不定方程组，那会有无穷个解，工程上是没有意义的。所以只考虑 $K\geq L+1$ 的情况。

$$\mathbf{e=D-Pd}$$

其中：**e** 为单列 K 元误差向量，**D** 为预期幅特性在样本点列上的 K 元单列向量，**d** 为 L+1 元待定系数单列向量，而 P 则是由 cos（$n\,\omega_i$）组成的 $K\times(L+1)$的系数矩阵。

$$\mathbf{e}=\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_K \end{bmatrix},\ \mathbf{D}=\begin{bmatrix} D_1 \\ D_2 \\ \vdots \\ D_K \end{bmatrix},\ \mathbf{P}=\begin{bmatrix} 1 & \cos(\omega_1) & \cdots & \cos(L\omega_1) \\ 1 & \cos(\omega_2) & \cdots & \cos(L\omega_2) \\ & \vdots & & \\ 1 & \cos(\omega_K) & \cdots & \cos(L\omega_K) \end{bmatrix},\ \mathbf{d}=\begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_L \end{bmatrix}$$

前面已经得到它的最小二乘解的公式：

$$\mathbf{d}=\left(\mathbf{P}^{\mathrm{T}}\mathbf{P}\right)^{-1}\mathbf{P}^{\mathrm{T}}\mathbf{D}$$

用 MATLAB 计算此式特别方便，可以表示为 **d=inv（P'*P）*P'*D**，也可以更简便地用左除运算 d=P\D 来完成。

例 7.8　举一个数字例，要求设计长度为 N=9(有 5 个待定系数)的 FIR 滤波器，要使它在 0～π 之间的八个频点 ω 上逼近预期的低通幅特性 D：

ω=[0, 0.33, 0.67, 1, 1.5, 2, 2.5, 3.14]; D=[1,1,1,1,0,0,0,0];

解：列出方程组的完整形式：

$$\begin{bmatrix} 1 & \cos\omega_1 & \cos 2\omega_1 & \cos 3\omega_1 & \cos 4\omega_1 \\ 1 & \cos\omega_2 & \cos 2\omega_2 & \cos 3\omega_2 & \cos 4\omega_2 \\ 1 & \cos\omega_3 & \cos 2\omega_3 & \cos 3\omega_3 & \cos 4\omega_3 \\ 1 & \cos\omega_4 & \cos 2\omega_4 & \cos 3\omega_4 & \cos 4\omega_4 \\ 1 & \cos\omega_5 & \cos 2\omega_5 & \cos 3\omega_5 & \cos 4\omega_5 \\ 1 & \cos\omega_6 & \cos 2\omega_6 & \cos 3\omega_6 & \cos 4\omega_6 \\ 1 & \cos\omega_7 & \cos 2\omega_7 & \cos 3\omega_7 & \cos 4\omega_7 \\ 1 & \cos\omega_8 & \cos 2\omega_8 & \cos 3\omega_8 & \cos 4\omega_8 \end{bmatrix}\begin{bmatrix} d(0) \\ d(1) \\ d(2) \\ d(3) \\ d(4) \end{bmatrix}=\begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \\ D_6 \\ D_7 \\ D_8 \end{bmatrix}\ \Rightarrow\ \mathbf{P}*\mathbf{d}=\mathbf{D}$$

看来系数矩阵 **P** 的赋值比较麻烦，其实，回想求离散傅里叶变换时的做法，可以利用列矩阵 **w′**=[w_1; w_2; …; w_8]乘行矩阵[0:4]形成一个 8×5 矩阵，然后求余弦得到 **P**。这可以用 MATLAB 语句 P=cos(**w**'*[0:4])轻而易举地列出。因此用下列几行 MATLAB 程序 ag1010 就可以方便地完成最小二乘最优滤波器的设计。

```
N=9; L=floor((N-1)/2);          % 列出待求系数序数
w=[0,0.33,0.67,1,1.5,2,2.5,3.14]; % 列出频率向量
D=[1,1,1,1,0,0,0,0];            % 列出预期幅特性向量
P=cos(w'*[0:L]);              % 列出 P 矩阵
d=inv(P'*P)*P'*D'             % 求出待求系数
```

程序运行的结果为

```
d =  0.3981
   0.6039
   0.2137
  -0.1205
   -0.1506
```

知道 $d(n)$以后，就可以计算滤波器的频率特性进行校核。得到的幅频特性见图 7-11。因为 L=4，最高的谐波次数为 4，在 0～π 之间幅特性摆动最多两个周期，达到峰值的次数应为四次，这从图上也看得很清楚。

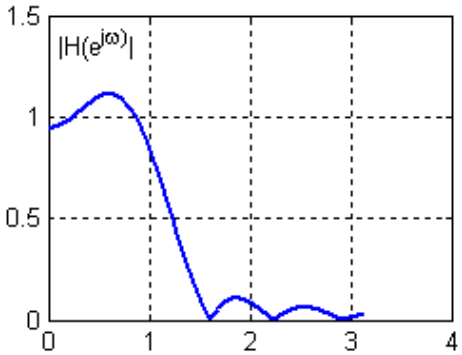
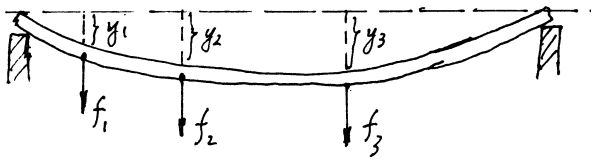


图 7-11　设计出的最优化滤波器的频率特性

7.9　弹性梁的柔度矩阵

设简支梁如图 7-12 所示，在梁的三个位置分别施加力 f_1，f_2 和 f_3 后，在该处产生的综合变形为图示的 y_1，y_2 和 y_3，通常称为挠度。根据虎克定律，在材料未失去弹性的范围内，力与它引起的变形呈线性关系，可以写出完全的矩阵形式为：



$$\mathbf{y = D*f} \quad \Rightarrow \quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}$$

不难从这个矩阵乘法关系中得知矩阵 D 的物理意义。假如只施加一个力 f_1，其余两个力 f_2 和 f_3 为零，则引起的挠度为：$y_1=d_{11}*f_1$，$y_2=d_{21}*f_1$，$y_3=d_{31}*f_1$，如果加的力为一个单位，则在 1，2，3 三个位置引起的挠度分别为 d_{11}，d_{21} 和 d_{31}。可以用同样的方法来理解其他的 d，利用这个概念，可以用实测的方法来得到矩阵 **D** 中的各个元素。这些挠度元素愈大，表明这个梁愈柔软，所以矩阵 **D** 被称作柔度矩阵。

柔度矩阵的逆就是刚度矩阵 **K**，$\mathbf{K=D^{-1}}$，在本例中它也是一个 3×3 矩阵。

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

其物理意义为造成单位挠度所需要的力。这些力愈大，表明这个梁愈刚劲。

在图 7-3 所示的情况下，柔度矩阵的 9 个元素中，没有一个为零。这说明它们之间相互耦合。在位置 1 施加力不但会引起该处的变形，同时也必然引起位置 2 和 3 处的变形；在其他位置加力亦然。如果矩阵变为对角矩阵，那也就意味着互相独立。位置 1 加的力只引起位置 1 处的挠度，常识告诉我们，在图 7-3 所示的简支梁上是不可能出现这种情况的。

现在举一个数字的例子，设柔度矩阵 $\mathbf{D}=\begin{bmatrix} .005 & .002 & .001 \\ .002 & .004 & .003 \\ .001 & .003 & .006 \end{bmatrix}$，单位为【公分/牛顿】。

（1）设在位置 1，2，3 处施加的力为 30，50 和 20【牛顿】，试求出其挠度。

（2）设要在位置 3 处产生 0.4【公分】的挠度，其他两处挠度为零，试求应施加的力。

解：列出解此题的 Matlab 程序 pla712

```
D=0.001*[5,2,1;2,4,3;1,3,6]      % 输入柔度矩阵
f=[30;50;20], y=D*f              % 解题（1）,给定力求挠度
y1=[0;0;0.4]                     % 解题（2）给定挠度
K=inv(D), f1=K*y1                % 求刚度矩阵，求力
```

程序运行的结果如下：

题（1）的三个挠度为 $y=\begin{bmatrix} 0.2700 \\ 0.3200 \\ 0.3000 \end{bmatrix}$

刚度矩阵为 **K = inv(D)** $=\begin{bmatrix} 254.2373 & -152.5424 & 33.8983 \\ -152.5424 & 491.5254 & -220.3390 \\ 33.8983 & -220.3390 & 271.1864 \end{bmatrix}$

题（2）的三个力为 $f1=\begin{bmatrix} 13.5593 \\ -88.1356 \\ 108.4746 \end{bmatrix}$

最后的力中有负数是合理的，因为题目要求有两个位置上的挠度为零，如果三个力都同向，造成的变形也同向，那就没有一处的挠度会等于零，必须有反向加的力才行。

可以看到，在刚度矩阵中也出现了负值元素，这是什么原因呢？可从其物理意义去想。对照柔度矩阵的物理意义，刚度矩阵中的第一列元素应该是：使挠度 y_1 达到一个单位，而 y_2 和 y_3 都等于零时，在三个位置上应施加的力。即同时在位置 1 施加 k_{11} 的力，在位置 2 施加 k_{21} 的力，而在位置 3 施加 k_{31} 的力。正如上面的分析，要使两个位置上的挠度为零，这三个力必须有正有负。所以表现为刚度矩阵中每列元素中必定有负值元素。只在一个位置加力（意味着其他两个力为零），在三个位置引起挠度是非常直观，符合常识的，所以读者容易接受柔度矩阵的物理概念；只在一个位置加“挠度”（意味着其他两个挠度为零），在三个位置需要加多大力却需要想像，在常识中大概很少有人经历这种状况，所以刚度矩阵的物理概念就要难理解一些。

7.10　用二次样条函数插值 5 个点

x	8	11	15	18	22
y	5	9	10	8	7

设给出以下五点坐标数据：

(a)　求通过这些点的二次样条函数；

(b)　求在 x=12.7 处的 y 的插值值；

(c)　画出插值多项式和这些数据点

解：由于有 5 个点(n=5)，需要 4 个样条函数(i=1,...,4)。样条函数的方程为：

$$f_i(x)=a_i x^2+b_i x+c_i \qquad (i=1,2,3,4)$$

四个多项式，每个多项式有三个系数，待求的总共为 12 个系数。按二次样条函数的规定，设 $a_1=0$，其余 11 个系数由 11 个线性方程组成的方程组决定。

其中 8 个方程由各区段的多项式通过数据点决定，即

$i=1,\quad f_1(x_1)=a_1x_1^2+b_1x_1+c_1=b_18+c_1=5$

$\quad\quad f_1(x_2)=a_1x_2^2+b_1x_2+c_2=b_111+c_1=9$

$i=2,\quad f_2(x_2)=a_2x_2^2+b_2x_2+c_2=a_211^2+b_211+c_2=9$

$\quad\quad f_2(x_3)=a_2x_3^2+b_2x_3+c_2=a_215^2+b_215+c_2=10$

$i=3,\quad f_3(x_3)=a_3x_3^2+b_3x_3+c_3=a_315^2+b_315+c_3=10$

$\quad\quad f_3(x_4)=a_3x_4^2+b_3x_4+c_3=a_318^2+b_318+c_3=8$

$i=4,\quad f_4(x_4)=a_4x_4^2+b_4x_4+c_4=a_418^2+b_418+c_4=8$

$\quad\quad f_4(x_5)=a_4x_5^2+b_4x_5+c_4=a_422^2+b_422+c_4=7$

3 个方程由相邻多项式在共同节点处的斜率相等的条件决定。即

$i=2\quad 2a_1x_2+b_1=2a_2x_2+b_2\rightarrow b_1=2a_211+b_2$ 或 $b_1-2a_211-b_2=0$

$i=3\quad 2a_2x_3+b_2=2a_3x_3+b_3\rightarrow 2a_215+b_2=2a_315+b_3$ 或 $2a_215+b_2-2a_315-b_3=0$

$i=4\quad 2a_3x_4+b_3=2a_4x_3+b_4\rightarrow 2a_318+b_3=2a_418+b_4$ 或 $2a_318+b_3-2a_418-b_4=0$

把 11 个联立方程写成矩阵形式，有：

$$\begin{bmatrix} 8 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1^2 & 11 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1^2 & 51 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1^2 & 51 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1^2 & 81 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1^2 & 81 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2^2 & 22 & 1 \\ 1 & 0 & -2 & 21 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 01 & 0 & -3 & 01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 61 & 0 & -3 & 61 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \\ a_3 \\ b_3 \\ c_3 \\ a_4 \\ b_4 \\ c_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 9 \\ 9 \\ 10 \\ 10 \\ 8 \\ 8 \\ 7 \\ 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \text{用解得 } \mathbf{A} \backslash \mathbf{B} \quad \begin{bmatrix} b_1 = 1.3333 \\ c_1 = -5.6667 \\ a_2 = -0.2708 \\ b_2 = 7.2917 \\ c_2 = -38.4375 \\ a_3 = 0.0556 \\ b_3 = -2.5000 \\ c_3 = 35.0000 \\ a_4 = 0.0625 \\ b_4 = -2.7500 \\ c_4 = 37.2500 \end{bmatrix}$$

程序 pla710 如下，在输入大的数据矩阵 A 时，采用了一些块矩阵技巧：

```
A=[8,1,zeros(1,9);11,1, zeros(1,9);0,0,11^2,11,1,zeros(1,6);0,0,15^2,15,1,zeros(1,6);...
zeros(2,5),[15^2,15,1;18^2,18,1],zeros(2,3);zeros(2,8),[18^2,18,1;22^2,22,1];...
1,0,-22,-1,zeros(1,7);0,0,30,1,0,-30,-1,0,0,0,0;zeros(1,5),36,1,0,-36,-1,0]
B=[5;9;9;10;10;8;8;7;0;0;0], X=A\B
```

其中 X=[b1;c1;a2;b2;c2;a3;b3;c3;a4;b4;c4]。

解出这 11 个系数后，写出这几段多项式方程为：

$$f_1(x) = 1.3333x - 5.6667 \qquad 8 \le x \le 11$$

$$f_2(x) = -0.27x^2 + 7.29x - 38.44 \qquad 11 \le x \le 15$$

$$f_3(x) = 0.0556x^2 - 2.5x + 35 \qquad 15 \le x \le 18$$

$$f_4(x) = 0.0625x^2 - 2.75x + 37.25 \qquad 18 \le x \le 22$$

(b) 在 x=12.7 处的 y 值为：

$$f_2(x) = -0.27 * 12.7^2 + 7.29 * 12.7 - 38.44 = 10.49$$

(c) 曲线也要分段画， 画出的曲线如右图，可以看出其第一段是一根直线，对应于 $a_1 = 0$。

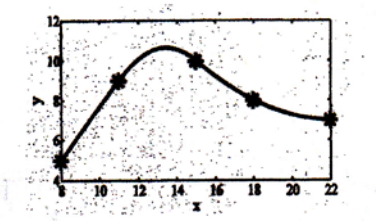


图 7-13　二次样条函数插值结果

7.11　飞行器三维空间运动的矩阵描述

这里主要举例说明线性代数在刚体空间运动学中的应用。

机械手、飞行器和机器人都是在空间运动的刚体或多刚体组。对它的运动的描述非常复杂，因为一个刚体要由三个转动和三个平移，即用 6 个自由度才能定量地决定它的位置。

飞行器在空中可以围绕三个轴旋转。假如它在向北飞行时，机头正对北方，则它围绕铅垂轴的旋转角称为偏航角（Yaw），它描述了飞

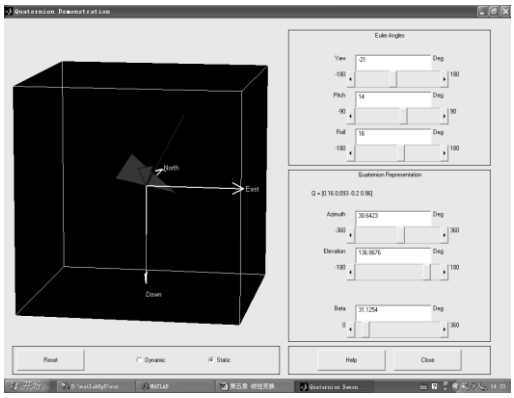


图 7-14　飞行器姿态角演示

机左右的偏转，用 u 表示；围绕翼展轴的旋转角称为倾斜角（Pitch），它描述了飞机俯仰姿态，用 v 表示；围绕机身轴的旋转角称为滚动角（Roll），用 w 表示；u,v 和 w 三个变量统称为欧拉角，它们完全描述了飞机的姿态。

MATLAB 中有一个演示程序 quatdemo.m，专门演示这几个姿态角所造成的飞机状态。键入:

```
quatdemo
```

屏幕上将出现如图 6-7 所示的画面。左方为飞行器在三维空间中的模型，其中红色的是飞行器；右上方为三个姿态角 u,v,w 的设定标尺和显示窗，右下方为在地面坐标系中的另外三个姿态角: 方位角、俯仰角和倾侧角。左下方还有【静态】和【动态】两个复选钮，我们只介绍【静态】，读者可自行试用【动态】进行演示。

用键入参数或移动标尺的方法分别给 u,v,w 赋值并回车后，就可以得出相应的飞行器姿态，同时出现一根蓝色的线表示合成旋转的转轴。

本例用数值来讨论这个程序的实现方法。先把飞行器的三维图像用 N 个顶点的三维坐标描述，写成一个 $3\times N$ 的数据矩阵 $\boldsymbol{G}$。其顶点次序要适当安排，使得用 plot3 命令时按顶点连线能绘制出飞行器的外观。例如，以下的程序（pla607 前半部分）
即可画出一个最简单的飞行器立体图，如图 6-8 所示。

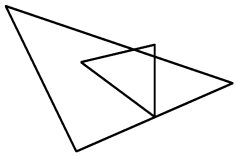


图 7-15　用 G 画出的飞行器

```
Gw=[-4, -3, 0; 4, -3, 0; 0, 7, 0;-4, -3, 0]';
% 主翼的顶点坐标
Gt=[0, -3, 0; 0, -3, 3; 0, 2, 0; 0, -3, 0]';
 尾翼的顶点坐标
G=[Gw,Gt]          %  整个飞行器外形的数据集
plot3(Gw(1,:),Gw(2,:),Gw(3,:),'r'),hold on
plot3(Gt(1,:),Gt(2,:),Gt(3,:),'g'),
axis equal
```

运行此程序得出整个飞行器外形的数据集为

$$\boldsymbol{G}=\begin{bmatrix}-4 & 4 & 0 & -4 & 0 & 0 & 0 & 0\\ -3 & -3 & 7 & -3 & -3 & -3 & 2 & -3\\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0\end{bmatrix} \tag{7.11.1}$$

飞行器围绕各个轴旋转的结果，表现为各个顶点坐标发生变化，也就是 $\boldsymbol{G}$ 的变化。只要把三种姿态的变换矩阵 $\boldsymbol{Y}$, $\boldsymbol{P}$ 和 $\boldsymbol{R}$ 乘以图形数据矩阵 $\boldsymbol{G}$ 即可。其中

$$\boldsymbol{Y}=\begin{bmatrix}\cos u & \sin u & 0\\ -\sin u & \cos u & 0\\ 0 & 0 & 1\end{bmatrix},\ \boldsymbol{R}=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos w & -\sin w\\ 0 & \sin w & \cos w\end{bmatrix},\ \boldsymbol{P}=\begin{bmatrix}\cos v & 0 & -\sin v\\ 0 & 1 & 0\\ \sin v & 0 & \cos v\end{bmatrix} \tag{7.11.2}$$

如果三个姿态角都变化，转动的次序为: 先滚动 $\boldsymbol{R}$，再倾斜 $\boldsymbol{P}$，最后偏航 $\boldsymbol{Y}$，则最后的图像数据成为 $\boldsymbol{G}_f=\boldsymbol{Y}\boldsymbol{G}_2=\boldsymbol{Y}\boldsymbol{P}\boldsymbol{G}_1=\boldsymbol{Y}\boldsymbol{P}\boldsymbol{R}\boldsymbol{G}=\boldsymbol{Q}\boldsymbol{G}$。最后变换矩阵成为

$$\boldsymbol{Q}=\begin{bmatrix}\cos u\cos v & \sin u\cos w-\cos u\sin v\sin w & -\sin u\sin w-\cos u\sin v\cos w\\ -\sin u\cos v & \cos u\cos w+\sin u\sin v\sin w & -\cos u\sin w+\sin u\sin v\cos w\\ \sin v & \cos v\sin w & \cos v\cos w\end{bmatrix} \tag{7.11.3}$$

由于矩阵乘法不服从交换律，转动次序不同时结果也不同。这个过程可以用手工计算，也可以用 MATLAB 程序来实现。程序 pla714 半部分如下:

```
syms u w v
Y=[cos(u),sin(u),0;sin(u), cos(u),0;0,0,1)]
R=[1,0,0;0,cos(w),sin(w);0,sin(w),cos(w)]
P=[cos(v),0,sin(v);0,1,0;sin(v),0,cos(v)]
```

```
Q=Y*P*R
```

这里采用了符号运算工具箱以得到普遍的公式。当设定了 u,v,w 的具体数值后，例如，$u=\pi/4$, $v=0$, $w=\pi/3$，要求出 Q 矩阵的数字结果时，要用 subs（代换）命令:

```
A=subs(Q,{u,v,w},{pi/4,0,pi/3})
```

运行结果为

$$\boldsymbol{A}=\begin{bmatrix} 0.7071 & 0.3536 & -0.6124 \\ -0.7071 & 0.3536 & -0.6124 \\ 0 & 0.8660 & 0.5000 \end{bmatrix}$$

我们知道，二维变换矩阵的行列式代表的是两个向量组成的平行四边形的面积，三维变换矩阵的行列式代表的是三个向量组成的平行六面体的体积。不难算出，$\det(\boldsymbol{Y})=1$，$\det(\boldsymbol{R})=1$，$\det(\boldsymbol{P})=1$，$\det(\boldsymbol{Q})=1$。当然也有 $\det(\boldsymbol{A})=1$。说明这几个变换都不会改变图形对象的体积。这是描述刚体运动的一个必须遵守的原则。不仅不允许改变体积，而且不允许改变形状，后者要求其变换的特征值必须等于 1。

飞行器在空中的运动应该由六个自由度表征，其中三个是转动，三个是平移，要完整地描述这些运动，三维空间和 3×3 的变换矩阵是肯定不够的。所以就需要研究三维以上的线性空间和线性变换问题。就本题来看，研究的对象不是整个飞行器，而是飞行器外形上的 N 个顶点。这些顶点可用三维空间中的三个坐标来描述，也就是 $3\times N$ 数据集 $\boldsymbol{G}_3$。考虑了平移运动后，如同二维情况那样，也必须要用齐次坐标系，即把三维空间扩展一维，在四维空间来建立数据组和 4×4 变换矩阵。即数据集 $\boldsymbol{G}_4$ 成为 $4\times N$ 矩阵，其最下面一行元素的取值都为 1。

$$\boldsymbol{G}_4=\begin{bmatrix} \boldsymbol{G}_3 \\ \mathbf{ones}(1,\boldsymbol{N}) \end{bmatrix} \tag{7.11.4}$$

而平移变换矩阵 $\boldsymbol{M}$ 和旋转变换矩阵 $\boldsymbol{Y}$, $\boldsymbol{R}$, $\boldsymbol{P}$ 成为

$$\boldsymbol{M}_4=\begin{bmatrix} 1 & 0 & 0 & c_1 \\ 0 & 1 & 0 & c_2 \\ 0 & 0 & 1 & c_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.11.5}$$

$$\boldsymbol{Y}_4(\boldsymbol{R}_4,\boldsymbol{P}_4)=\begin{bmatrix} & \boldsymbol{Y}_3(\boldsymbol{R}_3,\boldsymbol{P}_3) & & \begin{matrix}0\\0\\0\end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.11.6}$$

其中 c_1,c_2,c_3 为在三个轴 x_1,x_2,x_3 方向上的平移距离。这种方法也适用于机器人运动学的研究。

7.12　金融公司支付基金的流动

金融机构为保证现金充分支付，设立一笔总额 5400 万的基金，分开放置在位于 A 城和 B 城的两家公司，基金在平时可以使用，但每周末结算时必须确保总额仍然为 5400 万．经过相当长的一段时期的现金流动，发现每过一周，A 城公司有 10%支付基金流动到 B 城公司，B 城公司则有 12%支付基金流动到 A 城公司．起初 A 城公司基金为 2600 万，B 城公司基金为 2800 万．按此规律，两公司支付基金数额变化趋势如何？如果金融专家认为每个公司的支付基金不能少于 2200 万，那么是否需要在必要时调动基金?

设第 k+1 周末结算时，A 城公司 B 城公司的支付基金数分别为 a_{k+1}，b_{k+1} (单位:万元)，则有 a_0=2600，b_0=2800，

$$\begin{cases} a_{k+1} = 0.9a_k + 0.12b_k \\ b_{k+1} = 0.1a_k + 0.88b_k \end{cases}.$$

原问题转化为:

(1) 把 a_{k+1}, b_{k+1} 表示成 k 的函数, 并确定 $\lim\limits_{k\to+\infty} a_k$ 和 $\lim\limits_{k\to+\infty} b_k$.

(2) 看 $\lim\limits_{k\to+\infty} a_k$ 和 $\lim\limits_{k\to+\infty} b_k$ 是否小于 2200.

解：令 $\mathbf{X_k} = \begin{bmatrix} a_k \\ b_k \end{bmatrix}$, $\mathbf{A} = \begin{bmatrix} 0.9 & 0.12 \\ 0.1 & 0.88 \end{bmatrix}$, $\mathbf{X_0} = \begin{bmatrix} 2600 \\ 2800 \end{bmatrix}$

由此可得：$\mathbf{X_{k+1}} = \mathbf{AX_k} = \mathbf{A^2X_{k-1}} = \cdots = \mathbf{A^{k+1}X_0}$

为了计算 $\boldsymbol{A}^{k+1}\begin{pmatrix} 2600 \\ 2800 \end{pmatrix}$,固然可以用矩阵乘法硬算，但不是高明的办法。较好的方法是将 A 对角化。输入 MATLAB 命令:[P,D] = eig([0.9,0.12;0.1,0.88])，得到

$$\mathbf{P} = \begin{bmatrix} 0.7682 & -0.7071 \\ 0.6402 & 0.7071 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 1.00 & 0 \\ 0 & 0.78 \end{bmatrix}$$

因此　　$\mathbf{X_{k+1}} = \begin{bmatrix} a_{k+1} \\ b_{k+1} \end{bmatrix} = \mathbf{P} * \begin{bmatrix} 1 & 0 \\ 0 & 0.78^{k+1} \end{bmatrix} * inv(\mathbf{P}) * \begin{bmatrix} 2600 \\ 2800 \end{bmatrix} =$

键入以下语句即可完成这一计算：Xkp1=P*[D(1,1)^k,0;0,D(2,2)^k]*inv(P)*X0

得到：　　$Xkp1 = \begin{bmatrix} a_{k+1} \\ b_{k+1} \end{bmatrix} = \begin{pmatrix} 32400/11 - (0.78)^{k+1} 3800/11 \\ 27000/11 + (0.78)^{k+1} 3800/11 \end{pmatrix},$

可见 a_k 单调递增, b_k 单调递减, 且 $\lim\limits_{k\to+\infty} a_k = \dfrac{32400}{11} = 2945$, $\lim\limits_{k\to+\infty} b_k = \dfrac{27000}{11} = 2454$.

两者都大于 2200, 所以不需要调动基金。

7.13　质谱图实验结果分析

某一样本的质谱图给出了一系列的尖峰，

它们表示了样本中所含的各种离子成分的质量。各尖峰的高度 Ij 取决于不同成分的质量。

$$\sum_{i=1}^{N} C_{ij} n_j$$

其中，Cij 是物质 i 的离子对尖峰 j 所作贡献，nj 是物质 j 的离子总数，每个尖峰的系数 Cij 如右表。

峰点	物质的 Cij				
	CH_4	C_2H_4	C_2H_6	C_3H_6	C_3H_8
1	0.2	0.2	0.3	0.2	0.2
2	28	1	0	0	0.1
3		18	12	2.4	16
4			10	0	1
5				10	2
6					18

设某样本产生的质谱图的尖峰为：

I_1=3.4, I_2=20.5, I_3=170, I_4=49, I_5=39.8, I_6=96.3, 试求出样本中各种物质的含量。

解：设五种物质的含量分别为 x1,x2,x3,x4,x5，则可列出下列方程组：

0.2*x1+0.2*x2+0.3*x3+0.2*x4+0.2*x5=3.4

28*x1+x2+0.1*x5=20.5

18*x2+12*x3+2.4*x4+16*x5=170

10*x3+x5=49

10*x4+2*x5=39.8

18*x5=96.3

这里有六个方程和五个未知数，是一个超定问题，可用左除求解。写成矩阵形式为：

$$\begin{bmatrix} 0.2 & 0.2 & 0.3 & 0.2 & 0.2 \\ 28 & 1 & 0 & 0 & 0.1 \\ 0 & 18 & 12 & 2.4 & 16 \\ 0 & 0 & 10 & 0 & 1 \\ 0 & 0 & 0 & 10 & 2 \\ 0 & 0 & 0 & 0 & 18 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 3.4 \\ 20.5 \\ 170 \\ 49 \\ 39.8 \\ 96.3 \end{bmatrix}$$

解题的程序 pla702 如下：

```
A=[0.2,0.2,0.3,0.2,0.2;28,1,0,0,0.1;0,18,12,2.4,16;0,0,10,0,1;0,0,0,10,2;0,0,0,0,18]
B=[3.4;20.5;170;49;39.8;96.3], X=A\B
```

解得 X 的五个分量为：x_1=0.6634，x_2=1.3909　x_3=4.365　x_4=2.91　x_5=5.35。

7.14　用特征方程解 Fibonacci 数列问题

Fibonacci 数列于公元 1200 年左右被发现，在现代物理、准晶体结构、化学等领域都有直接的应用，它把 0 和 1 作为初始项 F_0 和 F_1，以后的每一项为其前两项的和，$F_{k+2}=F_{k+1}+F_k$，于是得到数列 1,1,2,3,5,8,13,21,…。数列递推的输入是两项，输出是一项，所以需要用两维矩阵建模。为了求出 k 无限增大时的 F_k，相当于研究稳态的趋势，就需要应用特征值和特征向量。这是一个研究矩阵和特征值应用的极好例子。解的过程如下。

一、建立矩阵模型：

先把 Fibonacci 规则写成联立方程 $\begin{cases} F_{k+2}=F_{k+1}+F_k \\ F_{k+1}=F_{k+1} \end{cases}$，设变量为两维向量 $\mathbf{u_k}=\begin{bmatrix} F_{k+1} \\ F_k \end{bmatrix}$，可写成矩阵方程：

$$\mathbf{u_{k+1}}=\begin{bmatrix} F_{k+2} \\ F_{k+1} \end{bmatrix}=\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} F_{k+1} \\ F_k \end{bmatrix}=\mathbf{A}\mathbf{u_k}$$

其中 $\mathbf{A}=\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$，初始条件 $\mathbf{u_0}=\begin{bmatrix} F_1 \\ F_0 \end{bmatrix}=\begin{bmatrix} 1 \\ 0 \end{bmatrix}$。

如果要求出第 k 项的 Fibonacci 数，就可以用矩阵连乘的方法，而写成

$$\mathbf{u_k}=\underbrace{\mathbf{A}\mathbf{A}\cdots\mathbf{A}}_{\mathbf{k}}\mathbf{u_0}=\mathbf{A}^{\mathbf{k}}\mathbf{u_0}$$

矩阵连乘运算也很麻烦，注意 A 是对称实矩阵，把它对角化，可以大大简化计算。令 $\mathbf{A}=\mathbf{P}\Lambda\mathbf{P}^{-1}$，则上式可写成 $\mathbf{u_k}=\mathbf{P}\Lambda^{\mathbf{k}}\mathbf{P}^{-1}\mathbf{u_0}=\mathbf{A}^{\mathbf{k}}\mathbf{u_0}$，因为对角矩阵乘方就等于主对角线各元素自行乘方：$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}=\begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix}$，以此类推，所以其计算就方便得多。

二、将系数矩阵对角化

① 建立特征方程并求根。为了将 **A** 对角化，首先求 **A** 的特征方程和特征值：

令　$\det(\mathbf{A}-\lambda\mathbf{I})=\begin{vmatrix}1-\lambda & 1\\ 1 & -\lambda\end{vmatrix}=\lambda^2-\lambda-1=0$

用二项式定理，得到两个特征根为 $\lambda_1=\dfrac{1+\sqrt{5}}{2}\approx 1.618,\quad \lambda_2=\dfrac{1-\sqrt{5}}{2}\approx -0.618,$

② 求特征向量，将 λ_1 代入特征矩阵的系数，进行消元，得出行最简型

（注意 $\lambda_1+\lambda_2=1,$ 及因而 $\lambda_1\lambda_2=-1,\qquad 1/(1-\lambda_1)=\lambda_1$ ）

$$\begin{bmatrix}1-\lambda_1 & 1\\ 1 & -\lambda_1\end{bmatrix}\Rightarrow\begin{bmatrix}1-\lambda_1 & 1\\ 0 & -\lambda_1-1/(1-\lambda_1)\end{bmatrix}\Rightarrow\begin{bmatrix}1 & 1/(1-\lambda_1)\\ 0 & 0\end{bmatrix}\Rightarrow\begin{bmatrix}1 & \lambda_1\\ 0 & 0\end{bmatrix}$$

这是秩 r=1，n=2 的欠定方程，它的一个基础解即特征向量为 $\boldsymbol{p}_1=\begin{bmatrix}\lambda_1\\ 1\end{bmatrix}$。同理得对应于 λ_2 的特征向量为 $\boldsymbol{p}_2=\begin{bmatrix}\lambda_2\\ 1\end{bmatrix}$，因而 $\mathbf{P}=[\boldsymbol{p}_1,\boldsymbol{p}_2]=\begin{bmatrix}\lambda_1 & \lambda_2\\ 1 & 1\end{bmatrix}$，其逆 $\mathbf{P}^{-1}=\dfrac{1}{\lambda_1-\lambda_2}\begin{bmatrix}1 & -\lambda_2\\ -1 & \lambda_1\end{bmatrix}$，**P** 没有规范化并不要紧，因为 $\mathbf{P}^{-1}$ 中的系数 $\dfrac{1}{\lambda_1-\lambda_2}$ 可以补偿。于是得出矩阵形式公式：

$$\mathbf{u}^k=\mathbf{A}^k\mathbf{P}\begin{bmatrix}1\\ 0\end{bmatrix}=\frac{1}{\lambda_1-\lambda_2}\begin{bmatrix}\lambda_1 & \lambda_2\\ 1 & 1\end{bmatrix}\begin{bmatrix}\lambda_1^k & 0\\ 0 & \lambda_2^k\end{bmatrix}\begin{bmatrix}1 & -\lambda_2\\ -1 & \lambda_1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}\qquad (a)$$

也可把矩阵展开，取其第二行，得出标量公式

$$\mathbf{u}^k=\begin{bmatrix}F_{k+1}\\ F_k\end{bmatrix}=\frac{1}{\lambda_1-\lambda_2}\begin{bmatrix}\lambda_1^{k+1}-\lambda_2^{k+1}\\ \lambda_1^{k}-\lambda_2^{k}\end{bmatrix},\quad \text{第二行为}\quad F_k=\frac{\lambda_1^k-\lambda_2^k}{\lambda_1-\lambda_2}\qquad (b)$$

三、程序编写及验证

用下列程序 pla614 可以很快求出任意 k 阶的 Fibonacci 数。

```
A=[1,1;1,0], [p,lamda]=eig(A)
for k=[1,3,5,10,20,50,100]
    F=p*lamda^k*inv(p)*[1;0];  % 用矩阵公式(a)计算
    % 下一语句是用标量公式(b)计算
    F1=(lamda(1,1)^k-lamda(2,2)^k)/(lamda(1,1)-lamda(2,2));
    [k,F(2,:),F1]                    % 显示计算结果
end
```

算出的结果见下表，`F(1,:)`和 `F1` 当然是一样的。

k	3	5	10	20	50	100
F	2	5	55	6765	12586269025	3.542248481792631e+020

k=73 以上时，F 的有效位数已超过了 MATLAB 的精度（15 位 ，靠 MATLAB 已经无法算得精确到个位的 Fibonacci 数了。

有趣的是：这样一个完全是自然数的数列，通项公式却是用无理数来表达的。 而且当 k 趋向于无穷大时，后一项与前一项的比值越来越逼近黄金分割 1.618（其倒数是 0.618）。因 为 λ_2

小于 1，当 k 无限增大时，公式(b)中的后一项趋于零，F_k 的变化趋势仅取决于 λ_1。$F_{k+1}//F_k$ 的值随 k=3 到 9 变化如下：1.5，1.667，1.6，1.625，1.6154，1.6190，1.6176，…，愈来愈趋近 1.618。

例 7.15　简单线性规划问题

线性规划是一门研究如何使用最少的人力、物力和财力去最优地完成科学研究、工业设计、经济管理中实际问题的专门学科. 主要在以下两类问题中得到应用：一是在人力、物力、财务等资源一定的条件下，如何使用它们来完成最多的任务；二是给一项任务，如何合理安排和规划，能以最少的人力、物力、资金等资源来完成该项任务。它比线性代数多了两个概念：第一、在欠定方程组的基础上，又引进了不等式（包括解必须大于零），从而建立了解的可行域的概念；第二、在可行域内求极大值的概念。最简单的线性规划问题在中学数学中就介绍了。

某工厂用 A、B 两种配件生产甲、乙两种产品，每生产一件甲产品使用 4 个 A 配件并耗 100 工时，每生产一件乙产品使用 4 个 B 配件并耗时 200 工时，该厂每天最多可从配件厂获得 16 个 A 配件和 12 个 B 配件，按每天工作 800 工时计算，该厂所有可能的日生产安排是什么？若生产一件甲产品获利 2 万元，生产一件乙产品获利 3 万元，采用哪种生产安排获得的利润最大？

解：设甲、乙两种产品的日生产分别为 x,y 件时，工厂获得的利润为 z 万元，则要求

最大化目标函数 $z=2x+3y$

而 x,y 应满足约束条件为

$$\begin{cases} 100x+200y\le 800 \\ 4x\le 16 \\ 4y\le 12 \\ x,\ y\ge 0 \end{cases},$$

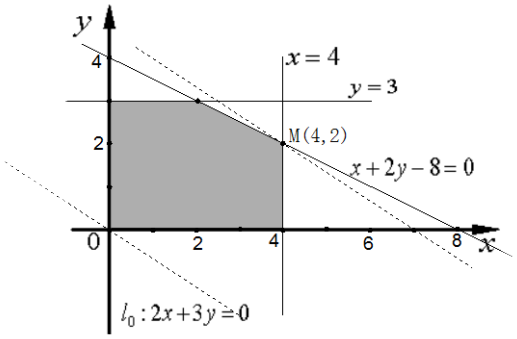


图 7-16　例 7.15 的可行域绘制和目标函数的最大化

在线性规划的问题中，约束条件通常由一组欠定线性方程组和一些不等式构成。在本例中，第一个方程 $x+2y\le 8$ 如果取等式，它就是 r=1，n=2 的二元欠定方程（组），其解集就是一根斜线 $x+2y-8=0$。由于是不等式，其解集（即可行域）是该斜线左下方的半平面。其他三个不等式的约束条件比较简单，作出约束条件所表示的可行域，如图 7-16 所示。

设目标函数为任意常数 c，其等值线 $2x+3y=c$ 是一系列斜率为-2/3 的平行线。当此直线平移经过可行域时，在点 M（4,2）处达到 z 的最大值. $z_{\max}=2x+3y=14$。即每天安排生产 4 件甲产品，2 件乙产品时，工厂获利最大为 14 万元。

规模较大的线性规划问题就需要用到线性代数的理论。它的一般数学模型如下

$$\max(\min) z=c_1x_1+c_2x_2+\cdots+c_jx_j+\cdots+c_nx_n$$

$$s.t.\begin{cases} a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n\le(=,\ge)b_1 \\ \cdots\quad\cdots\quad\cdots\quad\cdots \\ a_{m1}x_1+a_{m2}x_2+\cdots+a_{mn}x_n\le(=,\ge)b_m \\ x_1,x_2,\cdots,x_n\ge 0 \end{cases}\Rightarrow \mathbf{AX}\le(=,\ge)\mathbf{B}$$

max(min) z 表示使目标函数极大（极小）化，*s.t.*(subject to 受约束于)…。可以看出，约束条件的主体部分就是一个线性方程组，去掉其中的不等式，余下的线性方程组必定是欠定的，这样它才会有无穷多个解并组成一个子空间。加上其中的不等式以后，可行域就会向半平面扩大，所以在学线性代数时需要弄清欠定线性方程组的解的特性。约束条件中最后的 $x_1, x_2, \cdots, x_n \geq 0$，则是对自变量取值的限制，通常是默认的。

MATLAB 中有解线性规划问题的专用子程序 LINPROG(f,A,B)，它用来解决如下的问题：

最小化　　z=f*x

约束条件:　**Ax <= b** (x 的所有分量为正的条件是默认的，不必列入)

对于例 6.14，如果要套用这个函数，就要做一些修改。首先是把求极大改为求极小，那只要把 z 的系数 f 号的正负号反转即可。

$$z = \boldsymbol{f} * \boldsymbol{X} = [-2, -3]\begin{bmatrix} x \\ y \end{bmatrix} = -2x - 3y$$

约束条件应写成一个完全的矩阵，要把大于改成小于，也可用改变系数符号的方法：

$$\mathbf{AX} = \begin{bmatrix} 100 & 200 \\ 4 & 0 \\ 0 & 4 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} 800 \\ 16 \\ 12 \end{bmatrix} = \mathbf{B}$$

于是相应的 MATLAB 程序 pla715 核心语句可写成：

```
f=[-2,-3], A=[100,200;4,0;0,4], B=[800;16;12],
X=linprog(f,A,B), zmax= -f*X
```

程序运行的结果为：$\mathbf{X} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$，$\mathrm{zmax} = 14$