第 2 章 常 用 运 算

CHAPTER 2

第 1 章介绍的内容基本是 MATLAB 内置函数。本章开始介绍常用运算的 MATLAB 实现方法，包括多项式运算、矩阵求逆、求行列式的值等，虽然这些运算在 MATLAB 中基本都有对应的函数实现，但是，本章依然编写了这些算法的 MATLAB 函数，便于读者在解决实际问题时调用修改。

2.1 多项式运算

1. 多项式求值

计算多项式 $p(x)$ 在指定点 x 处的函数值。

$$p(x)=a_{n-1}x^{n-1}+a_{n-2}x^{n-2}+\cdots+a_1x+a_0$$

将多项式 $p(x)$ 表述为如下嵌套形式。

$$p(x)=(\cdots((a_{n-1}x+a_{n-2})x+a_{n-3})x+\cdots+a_1)x+a_0$$

从内向外一层一层地进行计算，递推计算公式为

$$u=a_{n-1}$$
$$u=u\cdot x+a_i,\quad i=n-2,\cdots,1,0$$

最后得到的 u 即为多项式值。

在 MATLAB 中编写 poly_value()函数，用于实系数多项式求值。

```
function u=poly_value(x,p)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 多项式求值
% 输入:
%       x: 参数值
%       p: 多项式系数值
% 输出:
%       u:多项式数值
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N=length(p)-1;                              %多项式次数
u=p(N+1);
for k=N:-1:1
    u=u*x+p(k);                             %递推计算公式，从里往外一层一层计算
end
end
```

2. 多项式相乘

求两个多项式 $P(x)$ 和 $Q(x)$ 的乘积多项式 $S(x)$。

$$P(x)=p_{m-1}x^{m-1}+p_{m-2}x^{m-2}+\cdots+p_1x+p_0$$
$$Q(x)=q_{n-1}x^{n-1}+q_{n-2}x^{n-2}+\cdots+q_1x+q_0$$
$$S(x)=P(x)Q(x)=s_{m+n-2}x^{m+n-2}+\cdots+s_1x+s_0$$

乘积多项式 $S(x)$ 中的各系数按以下公式进行计算。

$$s_k=0\ ,\quad k=0,1,\cdots,m+n-2$$
$$s_{i+j}=s_{i+j}+p_iq_j\ ,\quad i=0,1,\cdots,m-1\ ;\quad j=0,1,\cdots,n-1$$

在 MATLAB 中编写 poly_mul()函数，用于实系数多项式相乘。

```
function s=poly_mul(q,p)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 多项式相乘 s=p*q
% 输入：多项式系数 p 和 q
% 输出：多项式系数 s
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Np=length(p)-1;                                    %p 的次数
Nq=length(q)-1;                                    %q 的次数
s=zeros(Np+Nq+1,1);
for i=0:Np
    for j=0:Nq
        s(i+j+1)=s(i+j+1)+p(i+1)*q(j+1);
    end
end
end
```

3. 多项式相除

求多项式 $P(x)$ 被多项式 $Q(x)$ 除得的商多项式 $S(x)$ 和余多项式 $R(x)$。

$$P(x)=p_{m-1}x^{m-1}+p_{m-2}x^{m-2}+\cdots+p_1x+p_0$$
$$Q(x)=q_{n-1}x^{n-1}+q_{n-2}x^{n-2}+\cdots+q_1x+q_0$$

采用综合除法求商多项式 $S(x)$ 中的各系数。设商多项式 $S(x)$ 的最高次数为 $k=m-n$，则 $S(x)$ 的系数由以下递推公式进行计算。

$$s_{k-i}=p_{m-1-i}/q_{n-1}$$
$$p_j=p_j-s_{k-i}q_{j+i-k}\ ,\quad j=m-i-1,\cdots,k-i$$

其中，$i=0,1,\cdots,k$。最后的 $p_0,p_1,\cdots,p_{n-2}$ 即为余多项式的系数 $r_0,r_1,\cdots,r_{n-2}$。

在 MATLAB 中编写 poly_div()函数，用于实系数多项式相除。

```
function [s,r]=poly_div(q,p)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 多项式相除：s=p/q+r
% 输入：  多项式系数 p 和 q
% 输出：  结果多项式系数 s
%         余项多项式系数 r
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Np=length(p)-1;                                %p 的次数
Nq=length(q)-1;                                %q 的次数
Ns=Np-Nq;                                      %结果多项式 q 的次数
Nr=Nq-1;                                       %余项多项式 r 的次数
if q(Nq+1)==0
    s=0;
    r=0;
    fprintf('除数多项式系数输入错误！')
    return;
end
ll=Np;
for i=Ns+1:-1:1
    s(i)=p(ll+1)/q(Nq+1);
    mm=ll;
    for j=1:Nq
        p(mm)=p(mm)-s(i)*q(Nq-j+1);
        mm=mm-1;
    end
    ll=ll-1;
end
for i=1:Nr+1
    r(i)=p(i);
end
end
```

上述算法既适用于实系数多项式运算，也适用于复系数多项式运算。

【例 2-1】实系数多项式基本运算示例。

（1）计算多项式 $P_1(x)$ 在 $x=\pm0.9,\pm1.1,\pm1.3$ 处的函数值。其中

$$P_1(x)=2x^6-5x^5+3x^4+x^3-7x^2+7x-20$$

（2）计算多项式 $P_2(x)$ 与 $Q_2(x)$ 的乘积多项式 $S_2(x)=P_2(x)Q_2(x)$。其中

$$P_2(x)=3x^5-x^4+2x^3+5x^2-6x+4$$

$$Q_2(x)=2x^3-6x^2+3x+2$$

（3）求多项式 $P_3(x)$ 被多项式 $Q_3(x)$ 除得的商多项式 $S_3(x)$ 和余多项式 $R_3(x)$。其中

$$P_3(x)=3x^4+6x^3-3x^2-5x+8$$

$$Q_3(x)=2x^2-x+1$$

主函数如下。

```
clc, clear
% 实系数多项式运算
x=[0.9 1.1 1.3 -0.9 -1.1 -1.3];
p1=[-20 7 -7 1 3 -5 2];
p2=[4 -6 5 2 -1 3];
q2=[2 3 -6 2];
p3=[8 -5 -3 6 3];
q3=[1 -1 2];
```

```
fprintf('多项式求值:\n');
for k=1:length(x)
   fprintf('p(%f)=%f\n',x(k),poly_value(x(k),p1));
end
fprintf('乘积多项式 s2=p2*q2:\n');
s=poly_mul(q2,p2);
for k=1:length(s)
   fprintf('s2(%d)=%f\n',k,s(k));
end
fprintf('p3/q3 商多项式 s3:\n');
[s3, r3]=poly_div(q3,p3);
for k=1:length(s3)
   fprintf('s3(%d)=%f\n',k,s3(k));
end
fprintf('p3/q3 余多项式 r3:\n');
for k=1:length(r3)
   fprintf('r3(%d)=%f\n',k,r3(k));
end
```

运行程序，输出结果如下。

```
多项式求值:
p(0.900000)=-18.562268
p(1.100000)=-19.556128
p(1.300000)=-20.875732
p(-0.900000)=-26.715368
p(-1.100000)=-21.513028
p(-1.300000)=-6.340432
乘积多项式 s2=p2*q2:
s2(1)=8.000000
s2(2)=0.000000
s2(3)=-32.000000
s2(4)=63.000000
s2(5)=-38.000000
s2(6)=1.000000
s2(7)=19.000000
s2(8)=-20.000000
s2(9)=6.000000
p3/q3 商多项式 s3:
s3(1)=-0.375000
s3(2)=3.750000
s3(3)=1.500000
p3/q3 余多项式 r3:
r3(1)=8.375000
r3(2)=-9.125000
```

【例 2-2】复系数多项式基本运算示例。

（1）计算复系数多项式 $P_1(z)$ 在 $z=1+\mathrm{j}$ 时的函数值。其中

$$P_1(z)=(2+\mathrm{j}2)z^3+(1+\mathrm{j})z^2+(2+\mathrm{j})z+(2+\mathrm{j})$$

（2）求多项式 $P_2(z)$、$Q_2(z)$ 的乘积多项式 $S_2(z)$。其中

$$P_2(z)=(3+\mathrm{j}2)z^5+(-1-\mathrm{j})z^4+(2+\mathrm{j})z^3+(5-\mathrm{j}4)z^2+(-6+\mathrm{j}3)z+(4+\mathrm{j}2)$$

$$Q_2(z)=(2+\mathrm{j})z^3+(-6-\mathrm{j}4)z^2+(3+\mathrm{j}2)z+(2+\mathrm{j})$$

（3）求复系数多项式 $P_3(z)$ 被复系数多项式 $Q_3(z)$ 除得的商多项式 $S_3(z)$ 和余多项式 $R_3(z)$。其中

$$P_3(z)=(3-\mathrm{j})z^4+(6-\mathrm{j}5)z^3+(-3+\mathrm{j}4)z^2+(-5+\mathrm{j}4)z+(8+\mathrm{j}3)$$

$$Q_3(z)=(2+\mathrm{j}2)z^2+(-1-\mathrm{j}3)z+(1+\mathrm{j}2)$$

主函数如下。

```
clc, clear
% 复数多项式运算类例
x=input('输入 x=');
p1=[2+1i 2+1i 1+1i 2+2i];
p2=[4+2i -6+3i 5-4i 2+1i -1-1i 3+2i];
q2=[2+1i 3+2i -6-4i 2+1i];
p3=[8+3i -5+4i -3+4i 6-5i 3-1i];
q3=[1+2i -1-3i 2+2i];
fprintf('多项式求值:\n');
fprintf('x=(%f,%f)\n',real(x),imag(x));
u=poly_value(x, p1);
fprintf('p(x)=(%f,%f)\n',real(u),imag(u));
fprintf('乘积多项式 s2=p2*q2:\n');
s2=poly_mul(q2,p2);                          %多项式相乘 s2 = p2*q2
for k=1:length(s2)                           %输出乘积多项式 s2 的系数
   fprintf('s2(%d)=(%f,%f)\n',k,real(s2(k)),imag(s2(k)));
end
fprintf('p3/q3 商多项式 s3:\n');
[s3, r3]=poly_div(q3,p3);                    %多项式相除 s3 = p3/q3+r3
for k=1:length(s3)                           %输出商多项式 s3 的系数
   fprintf('s3(%d)=(%f, %f)\n',k,real(s3(k)),imag(s3(k)));
end
fprintf('p3/q3 余多项式 r3:\n');
for k=1:length(r3)                           %输出余多项式 r3 的系数
   fprintf('r3(%d)=(%f,%f)\n',k,real(r3(k)),imag(r3(k)));
end
```

运行程序，输出结果如下。

```
输入 x=1+1j
多项式求值:
x=(1.000000, 1.000000)
p(x)=(-7.000000, 6.000000)
乘积多项式 s2=p2*q2:
s2(1)=(6.000000, 8.000000)
s2(2)=(-7.000000, 14.000000)
s2(3)=(-26.000000, -34.000000)
s2(4)=(80.000000, 16.000000)
```

```
s2(5)=(-58.000000, 8.000000)
s2(6)=(9.000000, -15.000000)
s2(7)=(10.000000, 26.000000)
s2(8)=(-11.000000, -27.000000)
s2(9)=(4.000000, 7.000000)
p3/q3 商多项式 s3 :
s3(1)=(2.625000, -0.500000)
s3(2)=(1.250000, -3.500000)
s3(3)=(0.500000, -1.000000)
p3/q3 余多项式 r3 :
r3(1)=(4.375000, -1.750000)
r3(2)=(-9.125000, 12.375000)
```

2.2　常规矩阵求逆

用全选主元高斯–约当（Gauss–Jordan）消去法求 n 阶实（或复）矩阵 $\boldsymbol{A}$ 的逆矩阵 $\boldsymbol{A}^{-1}$。求逆过程可以用以下计算步骤表示。

对于 $k=1,2,\cdots,n$，进行如下运算。

（1）归一化计算。

$$1/a_{kk} \Rightarrow a_{kk}$$
$$a_{kj}a_{kk} \Rightarrow a_{kj}, \quad j=1,2,\cdots,n; \quad j\neq k$$

（2）消元计算。

$$a_{ij}-a_{ik}a_{kj} \Rightarrow a_{ij}, \quad i=1,2,\cdots,n; \quad i\neq k; \quad j=1,2,\cdots,n; \quad j\neq k$$
$$-a_{ik}a_{kk} \Rightarrow a_{ik}, \quad i=1,2,\cdots,n; \quad i\neq k$$

矩阵求逆的计算工作量（乘除法次数）为 $O(n^3)$。

为了数值计算的稳定性，整个求逆过程需要全选主元。全选主元的过程如下：对于矩阵求逆过程中的第 k 步，首先，在 a_{kk} 右下方（包括 a_{kk}）的 $n-k+1$ 阶子阵中选取绝对值最大的元素，并将该元素所在的行号记录在 IS(k) 中，列号记录在 JS(k) 中。然后，通过行交换与列交换将该绝对值最大的元素交换到主对角线 a_{kk} 的位置上，即执行以下两步。

（1）$\boldsymbol{A}(k,l) \Leftrightarrow \boldsymbol{A}[\mathrm{IS}(k),l]$，$l=1,2,\cdots,n$；

（2）$\boldsymbol{A}(l,k) \Leftrightarrow \boldsymbol{A}[l,\mathrm{JS}(k)]$，$l=1,2,\cdots,n$。

经过全选主元后，矩阵求逆的计算过程是稳定的，但最后需要对结果进行恢复。恢复的过程如下：对于 $k=n,n-1,\cdots,1$，分别执行以下两步。

（1）$\boldsymbol{A}(k,l) \Leftrightarrow \boldsymbol{A}[\mathrm{JS}(k),l]$，$l=1,2,\cdots,n$；

（2）$\boldsymbol{A}(l,k) \Leftrightarrow \boldsymbol{A}[l,\mathrm{IS}(k)]$，$l=1,2,\cdots,n$。

在 MATLAB 中编写 inv_d()函数，实现常规矩阵求逆。在 MATLAB 中，也可以直接使用自带的矩阵求逆函数 inv()实现矩阵求逆，inv(A)等效于 A^(−1)。

```
function [flag, a]=inv_d(a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 求逆矩阵
```

```
% 输入:
%       a:原矩阵
% 输出:
%       flag:若矩阵奇异，则返回标志值 0，否则返回非 0
%       a:逆矩阵
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n=length(a);
a=reshape(a',1,n^2);
for k=0:n-1
    d=0;
    for i=k:n-1                                          %选主元
        for j=k:n-1
            l=i*n+j;
            q=abs(a(l+1));                               %计算元素绝对值
            if q>d
                d=q;
                is(k+1)=i;
                js(k+1)=j;
            end
        end
    end
    if d==0
        disp('矩阵奇异');
        flag=0;
    end
    if is(k+1)~=k
        for j=0:n-1                                      %行交换
            u=k*n+j;
            v=is(k+1)*n+j;
            p=a(u+1);
            a(u+1)=a(v+1);
            a(v+1)=p;
        end
    end
    if js(k+1)~=k
        for i=0:n-1                                      %列交换
            u=i*n+k;
            v=i*n+js(k+1);
            p=a(u+1);
            a(u+1)=a(v+1);
            a(v+1)=p;
        end
    end
    l=k*n+k;
    a(l+1)=1/a(l+1);                                     %计算 1/a(l+1)
    for j=0:n-1                                          %归一化
```

```
        if j~=k
            u=k*n+j;
            a(u+1)=a(u+1)*a(l+1);
        end
    end
    for i=0:n-1                                          %消元计算
        if i~=k
            for j=0:n-1
                if j~=k
                    u=i*n+j;
                    a(u+1)=a(u+1)-a(i*n+k+1)*a(k*n+j+1);
                end
            end
        end
    end
    for i=0:n-1                                          %恢复行列交换
        if i~=k
            u=i*n+k;
            a(u+1)=-a(u+1)*a(l+1);
        end
    end
end
for k=n-1:-1:0
    if js(k+1)~=k
        for j=0:n-1
            u=k*n+j;
            v=js(k+1)*n+j;
            p=a(u+1);
            a(u+1)=a(v+1);
            a(v+1)=p;
        end
    end
    if is(k+1)~=k
        for i=0:n-1
            u=i*n+k;
            v=i*n+is(k+1);
            p=a(u+1);
            a(u+1)=a(v+1);
            a(v+1)=p;
        end
    end
end
a=reshape(a,n,n)';
flag=1;
end
```

【例 2-3】求下列实矩阵的逆矩阵。

$$\boldsymbol{A}=\begin{bmatrix}0.2368 & 0.2471 & 0.2568 & 1.2671\\ 1.1161 & 0.1254 & 0.1397 & 0.1490\\ 0.1582 & 1.1675 & 0.1768 & 0.1871\\ 0.1968 & 0.2071 & 1.2168 & 0.2271\end{bmatrix}$$

在编辑器中编写如下程序。

```
clc, clear
% 实矩阵求逆示例
A=[0.2368 0.2471 0.2568 1.2671;
   1.1161 0.1254 0.1397 0.1490;
   0.1582 1.1675 0.1768 0.1871;
   0.1968 0.2071 1.2168 0.2271];
if det(A)~=0
    fprintf('实矩阵 A:\n');disp(A);
    fprintf('内置函数求逆矩阵 A-:\n');
    disp(inv(A));
    fprintf('自编函数求逆矩阵 A-:\n');
    [flag,AA]=inv_d(A);disp(AA);
end
```

运行程序，输出结果如下。

```
实矩阵 A:
    0.2368    0.2471    0.2568    1.2671
    1.1161    0.1254    0.1397    0.1490
    0.1582    1.1675    0.1768    0.1871
    0.1968    0.2071    1.2168    0.2271
内置函数求逆矩阵 A-:
   -0.0859    0.9379   -0.0684   -0.0796
   -0.1056   -0.0885    0.9060   -0.0992
   -0.1271   -0.1114   -0.1170    0.8784
    0.8516   -0.1355   -0.1402   -0.1438
自编函数求逆矩阵 A-:
   -0.0859    0.9379   -0.0684   -0.0796
   -0.1056   -0.0885    0.9060   -0.0992
   -0.1271   -0.1114   -0.1170    0.8784
    0.8516   -0.1355   -0.1402   -0.1438
```

【例 2-4】求复矩阵 $\boldsymbol{A}$ 的逆矩阵，其中 $\boldsymbol{A}=\boldsymbol{R}+\mathrm{j}\boldsymbol{I}$ 。

$$\boldsymbol{R}=\begin{bmatrix}0.2368 & 0.2471 & 0.2568 & 1.2671\\ 1.1161 & 0.1254 & 0.1397 & 0.1490\\ 0.1582 & 1.1675 & 0.1768 & 0.1871\\ 0.1968 & 0.2071 & 1.2168 & 0.2271\end{bmatrix}$$

$$\boldsymbol{I}=\begin{bmatrix} 0.1345 & 0.1678 & 0.1875 & 1.1161 \\ 1.2671 & 0.2017 & 0.7024 & 0.2721 \\ -0.2836 & -1.1967 & 0.3556 & -0.2078 \\ 0.3576 & -1.2345 & 2.1185 & 0.4773 \end{bmatrix}$$

在编辑器中编写如下程序。

```
clc, clear
% 复矩阵求逆示例
A=[0.2368+0.1345i 0.2471+0.1678i 0.2568+0.1875i 1.2671+1.1161i;
   1.1161+1.2671i 0.1254+0.2017i 0.1397+0.7024i 0.1490+0.2721i;
   0.1582-0.2836i 1.1675-1.1967i 0.1768+0.3556i 0.1871-0.2078i;
   0.1968+0.3576i 0.2071-1.2345i 1.2168+2.1185i 0.2271+0.4773i];
if det(A)~=0
    fprintf('复矩阵 A: \n');disp(A);
    fprintf('内置函数求逆矩阵 A-: \n');
    disp(inv(A));
    fprintf('自编函数求逆矩阵 A-: \n');
    [flag, AA]=inv_d(A);disp(AA);
end
```

运行程序，输出结果如下。

```
复矩阵 A:
   0.2368 + 0.1345i   0.2471 + 0.1678i   0.2568 + 0.1875i   1.2671 + 1.1161i
   1.1161 + 1.2671i   0.1254 + 0.2017i   0.1397 + 0.7024i   0.1490 + 0.2721i
   0.1582 - 0.2836i   1.1675 - 1.1967i   0.1768 + 0.3556i   0.1871 - 0.2078i
   0.1968 + 0.3576i   0.2071 - 1.2345i   1.2168 + 2.1185i   0.2271 + 0.4773i
内置函数求逆矩阵 A-:
  -0.0057 + 0.0451i   0.4851 - 0.4817i   0.0217 - 0.2382i  -0.1874 + 0.1212i
  -0.0700 + 0.1162i  -0.0471 + 0.1487i   0.5546 + 0.5124i  -0.0558 - 0.1430i
  -0.1764 + 0.1032i  -0.1421 + 0.1142i   0.0737 + 0.4515i   0.2620 - 0.4689i
   0.4848 - 0.4431i  -0.0311 + 0.0410i  -0.1258 - 0.1227i  -0.0025 + 0.0910i
自编函数求逆矩阵 A-:
  -0.0057 + 0.0451i   0.4851 - 0.4817i   0.0217 - 0.2382i  -0.1874 + 0.1212i
  -0.0700 + 0.1162i  -0.0471 + 0.1487i   0.5546 + 0.5124i  -0.0558 - 0.1430i
  -0.1764 + 0.1032i  -0.1421 + 0.1142i   0.0737 + 0.4515i   0.2620 - 0.4689i
   0.4848 - 0.4431i  -0.0311 + 0.0410i  -0.1258 - 0.1227i  -0.0025 + 0.0910i
```

2.3　对称正定矩阵求逆

求 n 阶对称正定矩阵 $\boldsymbol{A}$ 的逆矩阵 $\boldsymbol{A}^{-1}$，采用变量循环重新编号法，其计算公式为

$$a'_{n-1,n-1}=1/a_{00}$$

$$a'_{n-1,j-1}=-a_{0j}/a_{00},\quad j=1,2,\cdots,n-1$$

$$a'_{i-1,n-1}=a_{i0}/a_{00},\quad i=1,2,\cdots,n-1$$

$$a'_{i-1,j-1}=a_{ij}-a_{i0}a_{0j}/a_{00}\ ,\quad i,j=1,2,\cdots,n-1$$

当 $\boldsymbol{A}$ 为对称正定矩阵时，其逆矩阵 $\boldsymbol{A}^{-1}$ 也是对称正定矩阵。

在 MATLAB 中编写 ssgj()函数，用于实现对称正定矩阵的求逆。

```
function [a,flag]=ssgj(a,n)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 对称正定矩阵求逆
% 输入:
%       a(n,n):存放对称正定矩阵
% 输出:
%       a(n,n):返回逆矩阵
%       flag:函数返回标志值，等于 0 表示失败，大于 0 表示成功
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
a=reshape(a.',numel(a),1);
for k=0:n-1
    w=a(1);
    if abs(w)==0
        flag=0;
        return;
    end
    m=n-k-1;
    for i=1:n-1
        g=a(i*n+1);
        b(i+1)=g/w;
        if i<=m
            b(i+1)=-b(i+1);
        end
        for j=1:i
            a((i-1)*n+j)=a(i*n+j+1)+g*b(j+1);
        end
    end
    a(n^2)=1/w;
    for i=1:n-1
        a((n-1)*n+i)=b(i+1);
    end
end
for i=0:n-2
    for j=i+1:n-1
        a(i*n+j+1)=a(j*n+i+1);
    end
end
flag=1;
a=reshape(a,n,n).';
end
```

【例 2-5】求以下 4 阶对称正定矩阵 $\boldsymbol{A}$ 的逆矩阵 $\boldsymbol{A}^{-1}$，并计算 $\boldsymbol{A}\boldsymbol{A}^{-1}$ 以检验结果的正确性。

$$\boldsymbol{A}=\begin{bmatrix}5 & 7 & 6 & 5\\ 7 & 10 & 8 & 7\\ 6 & 8 & 10 & 9\\ 5 & 7 & 9 & 10\end{bmatrix}$$

在编辑器中编写如下程序。

```
clc, clear
% 对称正定矩阵求逆
A=[5 7 6 5;
   7 10 8 7;
   6 8 10 9;
   5 7 9 10];
n=4;
[b,flag]=ssgj(A,n);
if flag>0
   fprintf('复矩阵 A:\n');disp(A);
   fprintf('逆矩阵 A:\n');disp(b);
   fprintf('检验矩阵 AA-:\n');disp(A*b);
end
```

运行程序，输出结果如下。

```
复矩阵 A:
     5     7     6     5
     7    10     8     7
     6     8    10     9
     5     7     9    10
逆矩阵 A:
   68.0000  -41.0000  -17.0000   10.0000
  -41.0000   25.0000   10.0000   -6.0000
  -17.0000   10.0000    5.0000   -3.0000
   10.0000   -6.0000   -3.0000    2.0000
检验矩阵 AA-:
    1.0000   -0.0000   -0.0000    0.0000
    0.0000    1.0000   -0.0000    0.0000
    0.0000   -0.0000    1.0000    0.0000
    0.0000   -0.0000   -0.0000    1.0000
```

2.4　托普利兹矩阵求逆

求托普利兹（Toeplitz）矩阵的逆矩阵，可以采用特兰持（Trench）法。设 n 阶托普利兹矩阵为

$$\boldsymbol{T}^{(n)}=\begin{bmatrix} t_0 & t_1 & t_2 & \cdots & t_{n-1} \\ \tau_1 & t_0 & t_1 & \cdots & t_{n-2} \\ \tau_2 & \tau_1 & t_0 & \cdots & t_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \tau_{n-1} & \tau_{n-2} & \tau_{n-3} & \cdots & t_0 \end{bmatrix}$$

该矩阵简称 T 型矩阵，其求逆过程如下。

（1）首先取初值 $\alpha_0=t_0$，$c_1^{(0)}=\tau_1/t_0$，$r_1^{(0)}=t_1/t_0$。

（2）对于 k =0,1,⋯, $n-3$ 执行以下运算。

$$c_i^{(k+1)}=c_i^{(k)}+\frac{r_{k+2-i}^{(k)}}{\alpha_k}\left(\sum_{j=1}^{k+1}c_{k+2-j}^{(k)}\tau_j-\tau_{k+2}\right),\quad i=0,1,\cdots,k$$

$$c_{k+2}^{(k+1)}=\frac{1}{\alpha_k}\left(\tau_{k+2}-\sum_{j=1}^{k+1}c_{k+2-j}^{(k)}\tau_j\right)$$

$$r_i^{(k+1)}=r_i^{(k)}+\frac{c_{k+2-i}^{(k)}}{\alpha_k}\left(\sum_{j=1}^{k+1}r_{k+2-j}^{(k)}t_j-t_{k+2}\right),\quad i=0,1,\cdots,k$$

$$r_{k+2}^{(k+1)}=\frac{1}{\alpha_k}\left(t_{k+2}-\sum_{j=1}^{k+1}r_{k+2-j}^{(k)}t_j\right)$$

$$\alpha_{k+1}=t_0-\sum_{j=1}^{k+2}t_jc_j^{(k+1)}$$

计算出 α_{n-2} 以及 $c_i^{(n-2)}$、$r_i^{(n-2)}(i=0,1,\cdots,n-2)$。该步骤的计算工作量为 $O(n^2)$。

（3）计算逆矩阵 $\boldsymbol{B}^{(n)}$ 中的各元素。

$$b_{00}^{(n)}=\frac{1}{\alpha_{n-2}}$$

$$b_{0,j+1}^{(n)}=-\frac{1}{\alpha_{n-2}}r_j^{(n-2)},\quad j=0,1,\cdots,n-2$$

$$b_{i+1,0}^{(n)}=-\frac{1}{\alpha_{n-2}}c_i^{(n-2)},\quad i=0,1,\cdots,n-2$$

$$b_{i+1,j+1}^{(n)}=b_{ij}^{(n)}+\frac{1}{\alpha_{n-2}}\left[c_i^{(n-2)}r_j^{(n-2)}-r_{n-2-i}^{(n-2)}c_{n-2-j}^{(n-2)}\right],\quad i,j=0,1,\cdots,n-2$$

该步骤的计算工作量也为 $O(n^2)$。

因此，特兰持法的总工作量为 $O(n^2)$，比通常的求逆方法（工作量为 $O(n^3)$）低一阶。

在 MATLAB 中编写 trch()函数，采用特兰持法求解托普利兹矩阵的逆矩阵。

```
function [b,flag]=trch(t,tt)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 托普利兹矩阵求逆
% 输入:
%       t: 存放 T 型矩阵中的元素 t(1)-t(n)
%       tt: 后 n-1 个元素存放 T 型矩阵中的元素 tt(2)-tt(n)
% 输出:
%       b: 返回 T 型矩阵的逆矩阵
%       flag: 函数返回标志值，等于 0 表示失败，大于 0 表示成功
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
flag=0;
n=length(t);
c=zeros(n,1);
r=zeros(n,1);
p=zeros(n,1);
if t(1)==0
    fprintf('fail\n');
    return;
end
a=t(1);
c(1)=tt(2)/t(1);
r(1)=t(2)/t(1);
for k=0:n-3                                  %对于 k=0,1,…,n-3 执行以下运算
    s=0;
    for j=1:k+1
        s=s+c(k+2-j)*tt(j+1);
    end
    s=(s-tt(k+3))/a;
    for i=0:k
        p(i+1)=c(i+1)+s*r(k-i+1);
    end
    c(k+2)=-s;
    s=0;
    for j=1:k+1
        s=s+r(k+2-j)*t(j+1);
    end
    s=(s-t(k+3))/a;
    for i=0:k
        r(i+1)=r(i+1)+s*c(k-i+1);
        c(k-i+1)=p(k-i+1);
    end
    r(k+2)=-s;
    a=0;
    for j=1:k+2
        a=a+t(j+1)*c(j);
    end
    a=t(1)-a;
    if abs(a)==0
        fprintf('fail\n');
        return;
    end
end
b(1)=1/a;
for i=0:n-2                                  %计算逆矩阵 B 中的各元素
    k=i+1;
```

```
    j=(i+1)*n;
    b(k+1)=-r(i+1)/a;
    b(j+1)=-c(i+1)/a;
end
for i=0:n-2
      for j=0:n-2
         k=(i+1)*n+j+1;
         b(k+1)=b(i*n+j+1)-c(i+1)*b(j+2);
         b(k+1)=b(k+1)+c(n-j-1)*b(n-i);
      end
end
b=reshape(b,n,n)';
flag=1;
end
```

【例 2-6】求以下 6 阶 T 型矩阵 $\boldsymbol{T}^{(6)}$ 的逆矩阵 $\boldsymbol{B}$，并计算 $\boldsymbol{A}=\boldsymbol{T}^{(6)}\boldsymbol{B}$ 检验结果的正确性。其中，$\boldsymbol{t}=(10,5,4,3,2,1)$，$\mathbf{tt}=(0,-1,-2,-3,-4,-5)$，$n=6$。

$$\boldsymbol{T}^{(6)}=\begin{bmatrix}10 & 5 & 4 & 3 & 2 & 1\\ -1 & 10 & 5 & 4 & 3 & 2\\ -2 & -1 & 10 & 5 & 4 & 3\\ -3 & -2 & -1 & 10 & 5 & 4\\ -4 & -3 & -2 & -1 & 10 & 5\\ -5 & -4 & -3 & -2 & -1 & 10\end{bmatrix}$$

在编辑器中编写如下程序。

```
clc, clear
% 特兰持法求托普利兹矩阵的逆
t=[10 5 4 3 2 1];
tt=[0 -1 -2 -3 -4 -5];
n=length(t);
T=zeros(n);                                          %T 型矩阵
for i=1:n
    T(i,i:end)=t(1:n-i+1);
    T(i+1:end,i)=tt(2:n-i+1);
end
[B,flag]=trch(t,tt);
fprintf('B=inv(T):\n');disp(B);
if flag==1
    fprintf('A=T*B:\n');disp(T*B);
end
```

运行程序，输出结果如下。

```
B=inv(T):
    0.0947   -0.0470   -0.0137   -0.0018    0.0019    0.0038
   -0.0043    0.0949   -0.0469   -0.0137   -0.0018    0.0019
   -0.0010   -0.0047    0.0948   -0.0469   -0.0137   -0.0018
```

```
    0.0018   -0.0010   -0.0047    0.0948   -0.0469   -0.0137
    0.0131    0.0021   -0.0010   -0.0047    0.0949   -0.0470
    0.0470    0.0131    0.0018   -0.0010   -0.0043    0.0947
A=T*B:
    1.0000    0.0000   -0.0069   -0.0000   -0.0000   -0.0000
   -0.0021    1.0010    0.0003   -0.0069   -0.0000   -0.0001
    0.0071   -0.0056    1.0000    0.0002   -0.0068    0.0002
   -0.0035    0.0087   -0.0051    1.0001    0.0001   -0.0069
   -0.0001   -0.0000    0.0097   -0.0050    1.0000   -0.0002
    0.0000         0    0.0000    0.0097   -0.0050    1.0000
```

2.5　求一般行列式的值

计算 n 阶方阵 $\boldsymbol{A}$ 所对应的行列式值，采用全选主元高斯消去法。计算时，采用高斯消去法对方阵 $\boldsymbol{A}$ 进行一系列变换，使之成为上三角矩阵，其对角线上的各元素乘积即为行列式值。变换过程如下。

对于 $k=0,1,\cdots,n-2$，进行如下变换。

$$a_{ij}-a_{ik}a_{kj}/a_{kk}\Rightarrow a_{ij}\ ,\quad i,j=k+1,\cdots,n-1$$

为保证数值计算的稳定性，在实际变换过程中采用全选主元。

在 MATLAB 中编写 sdet()函数，实现求一般行列式的值。在 MATLAB 中，也可以直接使用自带的函数 det()实现求矩阵行列式的值。

```
function detnum=sdet(a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 行列式求值
% 输入:
%       a: 行列式
% 输出:
%       detnum: 行列式的值
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n=length(a);                                                    %阶数
a=reshape(a',1,n^2);
f=1;
detnum=1;
for k=0:n-2
    q=0;
    for i=k:n-1
        for j=k:n-1
            l=i*n+j;
            d=abs(a(l+1));
        end
    end
    if d>q
        q=d;
        is=i;
        js=j;
```

```
    end
    if q==0
        detnum=0;
        return;
    end
    if is~=k
        f=-f;
        for j=k:n-1
            u=k*n+j;
            v=is*n+j;
            d=a(u+1);
            a(u+1)=a(v+1);
            a(v+1)=d;
        end
    end
    if js~=k
        f=-f;
        for i=k:n-1
            u=i*n+js;
            v=i*n+k;
            d=a(u+1);
            a(u+1)=a(v+1);
            a(v+1)=d;
        end
    end
    l=k*n+k;
    detnum=detnum*a(l+1);
    for i=k+1:n-1
        d=a(i*n+k+1)/a(l+1);
        for j=k+1:n-1
            u=i*n+j;
            a(u+1)=a(u+1)-d*a(k*n+j+1);
        end
    end
end
detnum=f*detnum*a(n^2);
end
```

【例 2-7】求方阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的行列式值 $\det(\boldsymbol{A})$ 和 $\det(\boldsymbol{B})$。

$$\boldsymbol{A}=\begin{bmatrix}1&2&3&4\\5&6&7&8\\9&10&11&12\\13&14&15&16\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}3&-3&-2&4\\5&-5&1&8\\11&8&5&-7\\5&-1&-3&-1\end{bmatrix}$$

在编辑器中编写如下程序。

```
clc, clear
% 行列式求值
```

```
A=[1 2 3 4;
   5 6 7 8;
   9 10 11 12;
   13 14 15 16];
B=[3 -3 -2 4;
   5 -5 1 8;
   11 8 5 -7;
   5 -1 -3 -1];
fprintf('内置函数求解: \n');
fprintf('det(A)=%f\n',det(A));
fprintf('det(B)=%f\n',det(B));
fprintf('自编函数求解: \n');
fprintf('det(A)=%f\n',sdet(A));
fprintf('det(B)=%f\n',sdet(B));
```

运行程序，输出结果如下。

```
内置函数求解:
det(A)=-0.000000
det(B)=595.000000
自编函数求解:
det(A)=0.000000
det(B)=595.000000
```

2.6　产生随机数

1. 均匀分布随机数

用于产生 0 ~ 1 均匀分布的一个随机数。设 $m=2^{16}$，产生 0 ~ 1 均匀分布随机数的计算式为

$$r_i=\mathrm{mod}(2053r_{i-1}+13849,m)\text{，}\quad p_i=r_i/m\text{，}\quad i=1,2,\cdots$$

其中，p_i 为第 i 个随机数；$r_0\geqslant 1$（r_0 为随机数种子）。

在 MATLAB 中编写 rnd1()函数，用于产生 0 ~ 1 均匀分布的一个随机数。

```
function [p,R]=rnd1(R)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 产生 0 ~ 1 均匀分布的一个随机数
% 输入:
%       R:随机数种子
% 输出:
%       p:随机数
%       R:更新后的随机数种子
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
s=65536;
u=2053;
v=13849;
m=fix(R/s);
R=R-m*s;
R=u*R+v;
```

```
m=fix(R/s);
R=fix(R-m*s);
p=R/s;
end
```

2. 均匀分布随机整数

产生给定区间$[a,b]$内均匀分布的一个随机整数。首先，产生在区间$[0,s]$内均匀分布的随机整数（$s=b-a+1$），计算式为

$$r_i=\mathrm{mod}(5r_{i-1},4m)\ ,\quad p_i=\mathrm{int}(r_i/4)$$

其中，初值（随机数种子）r_0为大于或等于 1 的奇数；$m=2^k$，$k=\lfloor \mathrm{lb}\,s \rfloor+1$。

然后，将每个随机数加上a，即得到实际需要的随机整数。

在 MATLAB 中编写 rndab()函数，用于产生给定区间内均匀分布的一个随机整数。

```
function [p,R]=rndab(a,b,R)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 产生给定区间[a,b]内均匀分布的一个随机整数
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
k=b-a+1;
j=2;
while j<k
    j=2*j;
end
m=4*j;
k=R;
i=1;
while i<=1
    k=5*k;
    k=mod(k,m);
    j=round(k/4)+a;
    if j<=b
        p=j;
        i=i+1;
    end
end
R=k;
end
```

3. 正态分布随机数

产生均值为μ，方差为σ^2的正态分布随机数y，计算式为

$$y=\mu+\sigma\frac{\left(\sum_{i=0}^{n-1}\mathrm{rnd}_i\right)-\frac{n}{2}}{\sqrt{n/12}}$$

其中，rnd_i为 0～1 均匀分布的随机数。n足够大，通常取$n=12$时，其近似程度已经相当好了，此时有

$$y=\mu+\sigma\left(\sum_{i=0}^{11}\mathrm{rnd}_i-6\right)$$

在 MATLAB 中编写 rndg()函数，用于产生给定均值与方差的正态分布随机数。

```
function [t,R]=rndg(u,g,R)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 产生给定均值 u 与方差 g^2 的正态分布随机数
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
s=65536;
w=2053;
v=13849;
t=0;
for i=1:12
    R=R*w+v;
    m=fix(R/s);
    R=R-m*s;
    t=t+R/s;
end
t=u+g*(t-6);
end
```

【例 2-8】分别产生以下随机数序列。

（1）50 个 0 ~ 1 均匀分布的随机数序列，取随机数种子（即初值）$r=5$；

（2）50 个 101 ~ 200 的随机整数序列，取随机数种子（即初值）$r=1$；

（3）50 个均值为 1.0，方差为1.5^2的正态分布随机数序列，取随机数种子（即初值）$r=3$。

在编辑器中编写如下程序。

```
clc, clear
fprintf('产生 50 个 0 ~ 1 的随机数如下：\n')
%设置随机数种子
R=5;
for i=1:10
    for j=1:5
        [p,R]=rnd1(R);
        fprintf('%f  ',p);
    end
    fprintf('\n')
end

fprintf('产生 50 个 101 ~ 200 的随机数如下：\n')
%设置随机数种子
R=1;
a=101;
b=200;
for i=1:10
    for j=1:5
        [p,R]=rndab(a,b,R);
        fprintf('%f  ',p);
    end
    fprintf('\n')
```

```
end

fprintf('产生 50 个均值为 1，方差为 1.5^2 的正态分布的随机数如下：\n')
%设置随机数种子
R=3;
u=1;
g=1.5;
for i=1:10
    for j=1:5
        [t,R]=rndg(u,g,R);
        fprintf('%f  ',t);
    end
    fprintf('\n')
end
```

运行程序，输出结果如下。

```
产生 50 个 0～1 的随机数如下：
0.367950  0.613571  0.872925  0.325943  0.372284
0.510239  0.731262  0.492630  0.580719  0.427414
0.692139  0.172012  0.352631  0.161972  0.739929
0.285965  0.297394  0.760788  0.109009  0.006363
0.274384  0.520737  0.283752  0.755081  0.392975
0.988693  0.998535  0.203995  0.012543  0.961533
0.237732  0.274979  0.742462  0.486130  0.235718
0.139908  0.442108  0.859360  0.476868  0.220657
0.220856  0.628098  0.695557  0.189102  0.438080
0.589218  0.876160  0.967117  0.703156  0.789597
产生 50 个 101～200 的随机数如下：
102.000000  107.000000  132.000000  129.000000  114.000000
167.000000  176.000000  190.000000  163.000000  156.000000
121.000000  200.000000  150.000000  180.000000  113.000000
162.000000  151.000000  110.000000  147.000000  105.000000
122.000000  120.000000  197.000000  198.000000  135.000000
144.000000  189.000000  158.000000  131.000000  124.000000
170.000000  191.000000  168.000000  181.000000  118.000000
187.000000  148.000000  130.000000  119.000000  192.000000
173.000000  115.000000  172.000000  175.000000  165.000000
166.000000  171.000000  196.000000  193.000000  178.000000
产生 50 个均值为 1，方差为 1.5^2 的正态分布的随机数如下：
-0.412430   0.367233  -0.042557   3.701950  1.944504
 1.528854  -1.201248   2.097946  -2.729813  0.159225
-0.391190   2.462692   0.564621  -0.741653  2.387619
 1.796188   1.327805   0.326218   2.635178  1.598434
 1.059738   3.362839  -1.148514   3.369431  1.260422
 2.368210   1.536545   1.109177   0.429855  3.342331
 0.190353   2.317673   1.568039   1.785202  0.812912
 0.994919   0.174973   2.196823   1.904221  1.640915
 0.750656   3.077194   0.464279   2.755661   0.295090
-1.573685   0.993088   2.839157   1.808273   3.244186
```