VB常用算法大全

何光渝 主编

- 近百种计算方法
- VB过程 随您调用
- 光碟内容 超值奉送
- 方便实用 事半功倍

西安电子科技大学出版社
http://www.xduph.com

VB常用算法大全

主编　何光渝
参编　张元冲　任世安　李　瑛

西安电子科技大学出版社
2001

内 容 简 介

本书共有科学计算中常用的 VB 过程 171 个，内容包括：解线性代数方程组，插值，数值积分，特殊函数，函数逼近，特征值问题，数据拟合，方程求根和非线性方程组求解，函数的极值和最优化，数据的统计描述，傅里叶变换谱方法，解常微分方程组和解偏微分方程组。每一个过程都包括功能、方法、使用说明、子过程和例子 5 部分。本书的所有过程都在 VB 5.0 版本上进行了验证。为方便读者学习，本书配有光盘一张，其中包含了书中全部子过程及验证这些子过程的全部工程。

本书可供大专院校师生和科研院所、工矿企业的工程技术人员使用。

图书在版编目(CIP)数据

VB 常用算法大全/何光渝主编. —西安：西安电子科技大学出版社，2001.1
ISBN 7-5606-0967-8
Ⅰ.V... Ⅱ.何... Ⅲ.BASIC 语言-算法设计 Ⅳ.TP312
中国版本图书馆 CIP 数据核字(2000)第 57189 号

责任编辑 马乐惠
出版发行 西安电子科技大学出版社(西安市太白南路 2 号)
电 话 (029)8227828 邮 编 710071
http://www.xduph.com E-mail:xdupfxb@pub.xaonline.com
经 销 新华书店
印 刷 西安兰翔印刷厂
版 次 2001 年 1 月第 1 版 2001 年 1 月第 1 次印刷
开 本 787 毫米×1092 毫米 1/16 印张 30.375
字 数 726 千字
印 数 1～4 000 册
定 价 45.00 元(含光盘)
ISBN 7-5606-0967-8/TP·0874

* * * 如有印装问题可调换 * * *

前　言

近几年来，随着 Miscorsoft 公司 VB(Visual BASIC)的推出，广大工程技术人员和电脑爱好者能利用 VB 事件驱动的编程机制和新颖、易用的可视化工具，做出自己所需要的各种各样功能强大的 Windows 应用软件。本书所介绍的常用算法，可以使人们在制作应用程序中减少不必要的重复劳动，大大提高计算机的使用效率。

本书收集了 171 个常用算法的 VB 过程，内容包括：解线性代数方程组，插值，数值积分，特殊函数，函数逼近，特征值问题，数据拟合，方程求根和非线性方程组求解，函数的极值和最优化，傅里叶变换谱方法，数据的统计描述，解常微分方程组和解偏微分方程组。每种算法都给出了例子和验证程序，帮助读者了解如何调用子过程，怎样输入数据，得到计算结果。验证程序建在窗体 FORM1 中命令钮的程序代码中。全书所有算法过程都在 VB5.0 版本下通过(随书发行光盘)。为了方便读者，在本书末附有过程调用索引表，可以很方便地查到每个子过程所需调用的子过程。

作者特别感谢所在单位西安石油学院以及西安交通大学、长安大学、中原石油勘探局对出版本书给予的关心和支持。由于编者水平有限，缺点错误在所难免，恳请广大读者批评指正。

编　者

2000.9

——目　录——

第 1 章　线性代数方程组的解法

本章包括线性代数方程组的求解、矩阵求逆、行列式计算、奇异值分解和线性最小二乘问题等的算法和程序，所给算法具有广泛的适用性和很强的通用性.

1. 一般实矩阵

高斯-约当全主元消去法(见 1.1 节)具有数值稳定的特点，所给过程在得到解的同时还得到系数矩阵的逆，但计算量大，对于方程组阶数不高而要求精度较高时，可采用此方法；*LU* 分解法采用隐式的部分选主元方法，数值稳定性好，存储量小，特别对于要解系数矩阵相同的多个方程组时最为适用，它还可用于求矩阵的逆和行列式. *LU* 分解法的计算量大约是 $n^3/3$，与列主元消去法相当，而高斯-约当消去法的计算量大约是它们的 3 倍，即大约是 n^3. 对于对称矩阵，特别是正定矩阵宜采用乔累斯基分解法(见 1.10 节)，它的程序简单，计算量小. *QR* 分解法即正交三角分解法(见 1.11 节)，由于其数值稳定性非常好，因此现在已越来越多地应用于各种数值求解中，现常用 *QR* 分解代替 *LU* 分解. 缺点是计算量和存储量均较大，计算速度亦较慢.

2. 病态矩阵

病态矩阵即条件数很大的矩阵. 对于病态矩阵，高斯消去法和 *LU* 分解法都不能给出满意的结果，*QR* 方法有时也同样不能给出满意的解，通常采用以下的处理办法：

(1) 增加计算的有效位数，如采用双精度(双倍字长)计算，这是一个比较有效的措施. 但这样做会使计算时间增加，且所需存储单元也会增到近两倍.

(2) 采用迭代改善的办法(见 1.5 节)，它是成功地改进解的精度的办法之一. 该方法的基本思想是在消去法的基础上利用迭代逐步改善方程组的解(关键在于在迭代过程中有些运算必须用双精度).

(3) 采用奇异值分解(*SVD*)法或共轭斜量法(见 1.8、1.9 节). 实验表明，共轭斜量法对病态矩阵常常是一种有效的方法.

3. 特殊形式的矩阵

这里包括三对角矩阵(见 1.3 节)和五对角矩阵(见 1.4 节)的追赶法、范德蒙矩阵的 G. Rybicki 方法和陶普立兹矩阵的 Rybicki 推广的 Levinson 方法(见 1.6、1.7 节). 对于以这些特殊矩阵为系数矩阵的方程组，若用一般矩阵的方法，效率太低，时间和空间的浪费也很大，因此对它们有专门有效的方法.

4. 稀疏矩阵

对于大稀疏矩阵的方程组，常用迭代法求解，这里我们给出两种迭代法：共轭斜量法和松弛迭代法(见 1.9、1.12 节). 它们均不要求矩阵具有任何特殊结构，因此可用于一般稀疏矩阵方程组的求解. 其中松弛迭代法当取松弛因子为 1.0 时，即为高斯 - 塞德尔迭代

法. 当然要注意迭代可能不收敛. 具体应用可参考第 13 章.

5. 奇异值分解(*SVD*)和最小二乘问题

SVD 对于奇异矩阵或数值上很接近奇异的矩阵是一个非常有效的方法，它可以精确地诊断问题. 在某些情形，*SVD* 将不仅诊断问题，而且也解决问题.

对于最小二乘问题，*SVD* 也是一个常选用的方法.

对于解方程组，*LU* 分解法和 *SVD* 都是先对系数矩阵作分解，然后再用分解矩阵求解. 它们的重大差别是用 *SVD* 解方程组之前即调用子过程 SVBKSB 之前要对奇异值进行剪辑，请参考 SVBKSB 的验证程序 DIR8. *SVD* 的用法细节可参考第 7 章.

1.1 全主元高斯 - 约当(Gauss-Jordan)消去法

1. 功能

用高斯 - 约当消去法求解 $\boldsymbol{A}[\boldsymbol{X}\ \boldsymbol{Y}]=[\boldsymbol{B}\ \boldsymbol{I}]$，其中 $\boldsymbol{A}$ 为 $n\times n$ 非奇异矩阵，$\boldsymbol{B}$ 为 $n\times m$ 矩阵，均已知；$\boldsymbol{X}_{n\times m}$，$\boldsymbol{Y}_{n\times n}$未知. 由于消去过程是在全矩阵中选主元(绝对值最大的元素)来进行的，故可使舍入误差对结果的影响减到最小.

2. 方法

(1) 施行初等变换把 $\boldsymbol{A}$ 变为单位矩阵，则

$$\boldsymbol{X}=\boldsymbol{A}^{-1}\boldsymbol{B},\ \boldsymbol{Y}=\boldsymbol{A}^{-1}$$

(2) 算法. 记

$$\boldsymbol{A}^{(0)}=(a_{ij}^{(0)})=\boldsymbol{A}=(a_{ij}),\ \boldsymbol{b}^{(0)}=\boldsymbol{b}=(b_{ij}^{(0)})$$

第 k 步的矩阵为 $\boldsymbol{A}^{(k)}=(a_{ij}^{(k)})_{n\times n}$，$\boldsymbol{b}=(b_{ij}^{(k)})_{n\times m}$，$(k=1,\cdots,n)$.

第 k 步的计算为

① 选主元，设为 $a_{i_0j_0}^{(k-1)}$.

② 若 $i_0=j_0$，则转③，否则交换矩阵$[A^{(k-1)}B^{(k-1)}]$的第 i_0 行与第 j_0 行，则 $a_{i_0j_0}^{(k-1)}$移至矩阵 $A^{(k-1)}$的对角线上，得到的矩阵仍记为$[A^{(k-1)}B^{(k-1)}]=[(a_{ij}^{(k-1)})\ (b_{ij}^{(k-1)})]$，主元为 $a_{j_0j_0}^{(k-1)}$.

③ 消元过程计算公式：

$$\begin{aligned}
&p_k=1/a_{j_0i_0}^{(k-1)}\\
&a_{j_0j}^{(k)}=a_{j_0j}^{(k-1)}\cdot p_k, && j=1,\cdots,n\\
&b_{j_0l}^{(k)}=b_{j_0l}^{(k-1)}\cdot p_k, && l=1,\cdots,m\\
&a_{ij}^{(k)}=a_{ij}^{(k-1)}-a_{j_0j}^{(k)}\cdot a_{ij_0}^{(k-1)}, && j=1,\cdots,n\\
&b_{il}^{(k)}=b_{ij}^{(k-1)}-b_{j_0l}^{(k)}\cdot a_{ij_0}^{(k-1)}, && l=1,\cdots,m\\
&i=1,\cdots,n && i\neq j_0
\end{aligned}$$

3. 使用说明

GAUSSJ (A(), N, B())

N　整型变量，输入参数，矩阵 $\boldsymbol{A}$ 的阶数；

A()　实型数组，输入、输出参数，输入时按列存放实方阵 $\boldsymbol{A}$，计算结束时输出逆矩阵 $\boldsymbol{A}^{-1}$；

B()　实型数组，输入、输出参数，输入时按列存放实方阵 $\boldsymbol{B}$，计算结束时输出解 $\boldsymbol{A}^{-1}\boldsymbol{B}$.

4. 过程

子过程 GAUSSJ.

```
Sub GAUSSJ(A(), N, B())
    Dim IPIV(50), INDXR(50), INDXC(50)
    For J = 1 To N
        IPIV(J) = 0
    Next J
    For I = 1 To N
        BIG = 0#
        For J = 1 To N
            If IPIV(J) <> 1 Then
                For K = 1 To N
                If IPIV(K) = 0 Then
                    If Abs(A(J, K)) >= BIG Then
                        BIG = Abs(A(J, K))
                        IROW = J
                        ICOL = K
                    End If
                ElseIf IPIV(K) > 1 Then
                    Print "Singular matrix"
                End If
                Next K
            End If
        Next J
        IPIV(ICOL) = IPIV(ICOL) + 1
        If IROW <> ICOL Then
            For L = 1 To N
                DUM = A(IROW, L)
                A(IROW, L) = A(ICOL, L)
                A(ICOL, L) = DUM
            Next L
            DUM = B(IROW)
            B(IROW) = B(ICOL)
            B(ICOL) = DUM
        End If
        INDXR(I) = IROW
        INDXC(I) = ICOL
        If A(ICOL, ICOL) = 0# Then Print "Singular matrix."
        PIVINV = 1# / A(ICOL, ICOL)
        A(ICOL, ICOL) = 1#
```

```
        For L = 1 To N
            A(ICOL, L) = A(ICOL, L) * PIVINV
        Next L
        B(ICOL) = B(ICOL) * PIVINV
        For LL = 1 To N
            If LL <> ICOL Then
                DUM = A(LL, ICOL)
                A(LL, ICOL) = 0#
                For L = 1 To N
                    A(LL, L) = A(LL, L) - A(ICOL, L) * DUM
                Next L
                B(LL) = B(LL) - B(ICOL) * DUM
            End If
        Next LL
    Next I
    For L = N To 1 Step -1
        If INDXR(L) <> INDXC(L) Then
            For K = 1 To N
                DUM = A(K, INDXR(L))
                A(K, INDXR(L)) = A(K, INDXC(L))
                A(K, INDXC(L)) = DUM
            Next K
        End If
    Next L
End Sub
```

5. 例子

验证程序 D1R1 调用子过程 GAUSSJ 可以对例子中的每一个矩阵求出其逆矩阵，并通过它们相乘看是否成为单位矩阵，最后将解乘以已知系数矩阵检查是否和方程右端的向量相等. 验证程序 D1R1 如下：

```
Private Sub Command1_Click()
    'PROGRAM D1R1
    'Driver program for routine GAUSSJ
    N = 3
    Dim A(3, 3), B(3), A1(3, 3), B1(3)
    '输入已知的方程组的系数矩阵
    A(1, 1) = 2: A(1, 2) = 1: A(1, 3) = 2
    A(2, 1) = 5: A(2, 2) = -1: A(2, 3) = 1
    A(3, 1) = 1: A(3, 2) = -3: A(3, 3) = -4
    '输入已知的方程组的右端向量 B
    B(1) = 5
    B(2) = 8
    B(3) = -4
```

```
    Print
    Print Tab(5); "已知的方程组的右端向量"
    Print Tab(14); Format $ (B(1), "# #. # #")
    Print Tab(14); Format $ (B(2), "# #. # #")
    Print Tab(14); Format $ (B(3), "# #. # #")
    For I = 1 To N
        For J = 1 To 3
        A1(I, J) = A(I, J)
        Next J
    Next I
    Call GAUSSJ(A(), N, B())
    Print
    Print Tab(5); "计算出的方程组的解"
    Print Tab(14); Format $ (B(1), "# #. # #")
    Print Tab(14); Format $ (B(2), "# #. # #")
    Print Tab(14); Format $ (B(3), "# #. # #")
    '将计算出的解 B 乘以系数矩阵,以验证计算结果正确
    For L = 1 To N
        B1(L) = 0#
        For J = 1 To N
            B1(L) = B1(L) + A1(L, J) * B(J)
        Next J
    Next L
    Print
    Print Tab(5); "计算出的解乘以系数矩阵的结果"
    Print Tab(14); Format $ (B1(1), "# #. # #")
    Print Tab(14); Format $ (B1(2), "# #. # #")
    Print Tab(14); Format $ (B1(3), "# #. # #")
End Sub
```

计算结果如下：

```
已知的方程组的右端向量
      5.
      8.
     -4.
计算出的方程组的解
      1.
     -1.
      2.
计算出的解乘以系数矩阵的结果
      5.
      8.
     -4.
```

1.2 *LU* 分解法

1. 功能

求解系数矩阵为非奇异的线性代数方程组 $\boldsymbol{A}\cdot\boldsymbol{X}=\boldsymbol{b}$，它能串联式地逐次解 $\boldsymbol{A}$ 相同 $\boldsymbol{b}$ 不同的方程组. 本方法也叫杜利图(Doolittle)方法，它将高斯主元消去法中的中间结果的记录次数从约 $n^3/3$ 次减少到约 n^2 次. 子过程 LUDCMP 将系数矩阵 $\boldsymbol{A}$ 分解为上三角矩阵和下三角矩阵. 子过程 LUBKSB 利用 LUDCMP 的分解结果求得线性方程组 $\boldsymbol{A}\cdot\boldsymbol{x}=\boldsymbol{b}$ 的解.

2. 方法

(1) 采用隐式部分选主元的杜利图方法.

(2) 首先作 $\boldsymbol{A}$ 的 $\boldsymbol{LU}$ 分解：

$$\boldsymbol{A}=\boldsymbol{L}\cdot\boldsymbol{U}=\begin{bmatrix}1 & & & & \\ l_{21} & 1 & & & \\ l_{31} & l_{32} & 1 & & \\ & \cdots & & \cdots & \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1\end{bmatrix}\begin{bmatrix}u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ & u_{22} & u_{23} & \cdots & u_{2n} \\ & & u_{33} & \cdots & u_{3n} \\ & \cdots & & \cdots & \\ & & & & u_{nn}\end{bmatrix}$$

考虑到数值稳定性，其中 l_{ij}，u_{ij} 计算如下：

① 选取 $\boldsymbol{A}$ 中每行中的主元(绝对值最大元)$\{a_{ij_i}\}$，$i=1,\cdots,n$.

② 取

$$|a_{i_1 j}/a_{1j_1}|=\max_{1\leqslant i\leqslant n}|a_{i_1}/a_{1j_1}|$$

若 $i_1\neq 1$，则交换第 1 行与第 i_1 行得 $\boldsymbol{A}=(a_{ij})$.

$$u_{1j}=a_{1j},\qquad j=1,2,\cdots,n$$
$$l_{i1}=a_{i1}/u_{11},\qquad i=2,\cdots,n$$

③ 一般地，令

$$S_i=a_{ik}-\sum_{j=1}^{k-1}l_{ij}u_{jk},\quad i=k,\cdots,n$$
$$|S_{i_k}/a_{i_k j_{ik}}|=\max_{k\leqslant i\leqslant n}|S_i/a_{ij_i}|$$

由于 $\boldsymbol{A}$ 非奇异，所以 $S_{i_k}\neq 0$. 若 $i_k\neq k$，则交换 $\boldsymbol{A}$ 与所得 $\boldsymbol{L}$ 的第 k 行与第 i_k 行，于是 $U_{kk}=S_{i_k}\neq 0$，且 $|l_{ij}|\leqslant 1(i>k)$.

$$u_{kj}=a_{kj}-\sum_{m=1}^{k-1}l_{km}u_{mj},\quad k=2,\cdots,n;\ j=k,\cdots,n$$

$$l_{ik}=\frac{a_{ik}-\sum_{j=1}^{k-1}l_{ij}iu_{jk}}{u_{kk}},\quad k=2,\cdots,n-1;\ i=k+1,\cdots,n$$

$$U_{kj}=0,\quad k>j,\ L_{ik}=0,\ i<k$$

(3) 解 $\boldsymbol{A}\cdot\boldsymbol{x}=\boldsymbol{b}$ 等价于解

$$\boldsymbol{P}_n\cdots\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{x}=\boldsymbol{P}_n\cdots\boldsymbol{P}_1\boldsymbol{b}$$

即

$$\boldsymbol{LUx}=\tilde{\boldsymbol{b}}=\boldsymbol{P}_n\cdots\boldsymbol{P}_1\boldsymbol{b}$$

此式等价于求解 $\boldsymbol{Ly}=\tilde{\boldsymbol{b}}$, $\boldsymbol{Ux}=\boldsymbol{y}$. 计算公式为

$$y_1 = \tilde{b}_1,\ y_i = \tilde{b}_i - \sum_{j=1}^{i-1} l_{ij}y_j,\quad i = 2, \cdots, n$$

$$x_n = y_n/u_{nn}$$

$$x_i = \frac{y_i - \sum_{j=i+1}^{n} u_{1j}x_j}{u_{ii}},\quad i = n-1, \cdots, 1$$

(4) 一般地说，优先推荐解线性方程组 $\boldsymbol{A}\cdot\boldsymbol{x}=\boldsymbol{b}$ 的方法是：

```
CALL LUDCMP (A(), N, INDX(), D)
CALL LUBKSB (A(), N, INDX(), B())
```

解 $\boldsymbol{X}$ 将存储在 $\boldsymbol{B}$ 中. 原始矩阵 $\boldsymbol{A}$ 已经被存储.

如果要连续解具有相同 $\boldsymbol{A}$ 而不同 $\boldsymbol{b}$ 的线性方程组时，只需重复

```
CALL LUBKSB (A(), N, INDX(), B())
```

因为 LUBKSB 所需要的输入 $\boldsymbol{A}$ 和 INDX()可以从 LUDCMP 中得到.

3. 使用说明

LUDCMP (A(), N, INDX(), D)

LUBKSB (A(), N, INDX(), B())

N　　整型变量，输入参数，$\boldsymbol{A}$ 的阶数；

A()　实型数组，输入、输出参数. 在 LUDCMP 中，输入时按列存放实方阵 $\boldsymbol{A}$，输出时，对角线以下部分存放单位下三角矩阵 $\boldsymbol{L}$，对角线及其以上部分存放上三角矩阵 $\boldsymbol{U}$. 在 LUBKSB 中，将 LUDCMP 中输出结果 $\boldsymbol{A}$ 作为输入；

INDX() 整型数组，在子过程 LUDCMP 中为输出参数，用于记录置换矩阵，称为置换向量，在子过程 LUBKSB 中为输入参数，输入子过程 LUDCMP 的输出结果；

D　　±1，输出参数，依赖于行交换次数为偶(+1)还是奇(−1)；

B()　实型数组，输入、输出参数，输入实向量 $\boldsymbol{b}$. 输出时，方程组的解 X 存储在数组 B()中.

4. 过程

(1) 子过程 LUDCMP.

```
Sub LUDCMP(A(), N, INDX(), D)
    NMAX = 100
    TINY = 1E-20
    Dim VV(100)
    D = 1#
    For I = 1 To N
        AAMAX = 0#
        For J = 1 To N
            If Abs(A(I, J)) > AAMAX Then AAMAX = Abs(A(I, J))
        Next J
        If AAMAX = 0# Then Print "Singular matrix."
```

```
        VV(I) = 1# / AAMAX
    Next I
    For J = 1 To N
        If J > 1 Then
            For I = 1 To J - 1
                Sum = A(I, J)
                If I > 1 Then
                    For K = 1 To I - 1
                        Sum = Sum - A(I, K) * A(K, J)
                    Next K
                    A(I, J) = Sum
                End If
            Next I
        End If
        AAMAX = 0#
        For I = J To N
            Sum = A(I, J)
            If J > 1 Then
                For K = 1 To J - 1
                    Sum = Sum - A(I, K) * A(K, J)
                Next K
                A(I, J) = Sum
            End If
            DUM = VV(I) * Abs(Sum)
            If DUM >= AAMAX Then
                IMAX = I
                AAMAX = DUM
            End If
        Next I
        If J <> IMAX Then
            For K = 1 To N
                DUM = A(IMAX, K)
                A(IMAX, K) = A(J, K)
                A(J, K) = DUM
            Next K
            D = -D
            VV(IMAX) = VV(J)
        End If
        INDX(J) = IMAX
        If J <> N Then
            If A(J, J) = 0# Then A(J, J) = TINY
            DUM = 1# / A(J, J)
            For I = J + 1 To N
```

```
                A(I, J) = A(I, J) * DUM
            Next I
        End If
    Next J
    If A(N, N) = 0# Then A(N, N) = TINY
End Sub
```

(2) 子过程 LUBKSB.

```
Sub LUBKSB(A(), N, INDX(), B())
    II = 0
    For I = 1 To N
        LL = INDX(I)
        Sum = B(LL)
        B(LL) = B(I)
        If II <> 0 Then
            For J = II To I - 1
                Sum = Sum - A(I, J) * B(J)
            Next J
        ElseIf Sum <> 0# Then
            II = I
        End If
        B(I) = Sum
    Next I
    For I = N To 1 Step -1
        Sum = B(I)
        If I < N Then
            For J = I + 1 To N
                Sum = Sum - A(I, J) * B(J)
            Next J
        End If
        B(I) = Sum / A(I, I)
    Next I
End Sub
```

在验证程序 D1R2 中，为了解线性方程组，还需调用 LUDCMP. 为了验证程序的正确性，将解与原系数矩阵相乘，以便与给定的右端向量相比较. LUBKSB 不能单独使用，必须和 LUDCMP 联合使用. 验证程序 D1R2 如下：

```
Private Sub Command1_Click()
    'PROGRAM D1R2
    'Driver program for routine LUBKSB,LUDCMP
    N = 3
    Dim A(3, 3), B(3), A1(3, 3), INDX(3), X(3)
    '输入已知的方程组的系数矩阵
    A(1, 1) = 1#: A(1, 2) = 2#: A(1, 3) = 3#
```

```
    A(2, 1) = 2#: A(2, 2) = 2#: A(2, 3) = 3#
    A(3, 1) = 3#: A(3, 2) = 3#: A(3, 3) = 3#
    '输入已知的方程组的右端向量
    B(1) = 1#
    B(2) = 2#
    B(3) = 3#
    Print
    Print Tab(5); "已知的方程组的右端向量"
    Print Tab(14); Format$(B(1), "##.##")
    Print Tab(14); Format$(B(2), "##.##")
    Print Tab(14); Format$(B(3), "##.##")
    For I = 1 To N
        For J = 1 To N
            A1(I, J) = A(I, J)
        Next J
    Next I
    Call LUDCMP(A1(), N, INDX(), P)
    For K = 1 To N
        For L = 1 To N
            X(L) = B(L)
        Next L
    Next K
    Call LUBKSB(A1(), N, INDX(), X())
    Print
    Print Tab(5); "计算出的方程组的解"
    Print Tab(14); Format$(X(1), "#.####E+00")
    Print Tab(14); Format$(X(2), "#.####E+00")
    Print Tab(14); Format$(X(3), "#.####E+00")
    '将计算出的B乘以系数矩阵,以验证计算结果正确
    For L = 1 To N
        B(L) = 0#
        For J = 1 To N
            B(L) = B(L) + A(L, J) * X(J)
        Next J
    Next L
    Print
    Print Tab(5); "计算出的解乘以系数矩阵的结果"
    Print Tab(14); Format$(B(1), "##.##")
    Print Tab(14); Format$(B(2), "##.##")
    Print Tab(14); Format$(B(3), "##.##")
End Sub
```