郑维敏 主编

# 系统工程FORTRAN程序集

FORTRAN

FORTRAN

FORTRAN

FORTRAN

FORTRAN

# 系统工程FORTRAN程序集


郑维敏 主编

王永县 陈崇端 蔡旭东

王炳永 谢 平 编



清华大学出版社

## 内 容 简 介

本书根据国外有关程序和作者实际工作经验编选了一些有实用价值的系统工程方面的FORTRAN程序。书中共包含五部分程序：系统规划；控制系统的计算及设计；连续系统的仿真；系统的辨识；随机运筹等共有几十种算法程序。

书中对每种算法的基本原理作了提示性说明，对每种算法程序的使用方法都结合例题作了详细的介绍。程序结构简炼，适用性强。可作为自动化、系统工程、经济管理、计算机等方面的高等院校教师、研究生、大学生以及有关科技工作人员的参考资料和优化设计程序手册。



**系统工程FORTRAN程序集**

主编 郑维敏

编者 王永县等

★

清华大学出版社出版

（北京 清华园）

昊海公司印刷厂排版

军事科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

★

开本：787×1092 1/16 印张：31 字数：793千字

1986年8月第1版 1986年8月第1次印刷

印数：00001～10000

统一书号：15235·214 平装定价：6.50元

精装定价：7.50元

# 序　　言

近年来，系统工程已为越来越多的人所接受。系统可以是社会系统、经济系统、环境系统、生态系统、各种工程系统乃至小型的机电系统、微电子系统。社会是人造的，为了使系统产生较大的或良好的效益或效用，我们要对系统进行规划、计划、管理、设计或控制，而且要寻找较好的或满意的解答。系统的类型、性质、规模及目的可以很不一样，但是，它们都是信息——决策／控制系统，即根据信息作出较好的决策或控制（规划、计划、管理、设计或控制等）。基于这种共性，从而产生了系统优化的理论及方法。当代社会已进入了电子计算机时代，计算机为系统优化理论及方法在各个领域中的应用提供了有效的手段。然而使用计算机，必须选择算法及编制程序。为了避免重复性劳动，我们收集了国外的一些有关程序并结合我们自己的工作经验，编写了这本系统工程程序集。由于微型计算机已逐渐普及，为了扩大应用范围，全部程序采用 FORTRANⅣ，并在 IMS-8000 微处理机上调试通过。

本书包括五个部分：系统规划程序，控制系统的计算及设计程序，连续系统的仿真程序，系统辨识程序及随机运筹程序，约有程序语句 9000 余条。由于内容较多，篇幅有限，而本书的目的是强调使用，又考虑到系统工程的理论及方法在许多教材中已有详细介绍，因此，书中关于算法、程序及步骤的介绍都着眼于使用。每种程序都有例题及使用步骤的具体说明，以期不十分熟悉程序的读者，对照例题及说明就会使用。对于熟悉原理及程序语言的读者，根据算法提示及程序框图或参考有关教材，也可以看懂程序并改进程序。

本书是在郑维敏教授指导下进行编写的。其中，第一部分和第五部分由王永县同志编写，第二部分由王炳永和谢平同志编写，第三部分和第四部分由陈崇端和蔡旭东同志编写。

由于经验有限，时间仓促，缺点在所难免，欢迎指正。

本书在程序调试过程中，得到了清华大学化工系计算机室赵奎元同志及自动化系夏凯同志的支持，在此谨表谢意。

编　者

# 目　录

# 第一部分 系统规划程序

这部分属于运筹规划问题，即在实现某计划有多种方案可供选择时，如何从中选择最优方案问题。虽然都是寻优，但是，由于问题本身的差别（即系统模型不同），所以，必须采用不同的方法予以解决。本部分列举了常用的四类规划问题：线性规划，整数规划，非线性规划及动态规划。每一类又包含几种不同的算法程序。

## 第一章 线性规划程序——单纯形算法

线性规划是运筹学中最常用的方法之一，在工业上各行各业中的许多问题都可成功地列出线性规划的数学模型。线性规划求解的问题就是在满足一组线性约束条件和变量为非负数限制下，求多变量线性函数的最优值。其数学模型为：

目标函数：

$$\max(\min) z=\sum_{j=1}^{n} c_j x_j$$

约束条件：

$$\sum_{j=1}^{n} a_{ij} x_j (\geqslant, =, \leqslant) b_i, \quad \begin{pmatrix} i=1,2,\cdots,m \\ j=1,2,\cdots,n \end{pmatrix}$$

$$x_j \geqslant 0$$

求解这种形式的线性规划问题的最常用方法就是单纯形法。

线性规划的一个重要特性是具有对偶性，即以原线性规划问题的约束条件的影子价格作为新变量按一定规则可构成一个新的线性规划问题——对偶线性规划。原问题与对偶问题的定义如下：

设原问题是：

目标函数：

$$\max \ z=\sum_{j=1}^{n} c_j x_j$$

约束条件：

$$\sum_{j=1}^{n} a_{ij} x_j \leqslant b_i \quad (i=1,2,\cdots,m)$$

$$x_j \geqslant 0 \qquad (j=1,2,\cdots,n)$$

则对偶问题为：

目标函数：

$$\min \quad v=\sum_{i=1}^{m} b_i y_i$$

约束条件：

$$\sum_{i=1}^{m} a_{ij} y_i \geqslant c_j \quad (j=1,2,\cdots,n)$$

$$y_i \geqslant 0 \qquad (i=1,2,\cdots,m)$$

其中，$y_i$ 即是原问题第 i 个约束条件的影子价格。当上述原问题所有约束条件右端项 bi 为正且都是≤约束条件时，则对偶变量 $y_i$（即影子价格）必全部≥0，否则，有的 $y_i$ 值会<0(详细情况请查阅有关书籍)。另外，对偶问题还具有许多其它性质，此处一律从略。

本章列举的单纯形算法程序可以在求解原问题的同时求出对偶解。下面就把这个程序作一简要介绍。

一、功能

可求解各种线性规划问题的原问题最优解及对偶最优解。

二、算法简介

单纯形算法是比较复杂的，现只简单提一下它所考虑的思路，用户若希望了解算法的演变过程，请参阅其它书籍。

1. 首先把原始问题的线性规划模型变为单纯形法求解的等效线性规划模型。

设原始线性规划模型为：

目标函数：

$$\max \hat{z} = \sum_{j=1}^{r} \hat{c}_j x_j$$

满足约束条件：

$$\sum_{j=1}^{r} a_{ij} x_j (\geqslant, =, \leqslant) b_i \quad (b_i \geqslant 0, i=1,2,\cdots,m)$$

$$x_j \geqslant 0$$

变换为等效线性规划模型为：

目标函数：

$$\max z = \sum_{j=1}^{n} \hat{\hat{c}}_j x_j$$

满足约束条件：

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \quad (i=1,2,\cdots,m)$$

$$x_j \geqslant 0$$

上述变换的主要步骤为：

(1) 把不等式变为等式。在≤不等式左边加一个松弛变量，在≥不等式左边减一个剩余变量。

(2) 在原来的等式和≥不等式左边加一个人工变量。

(3) 设松弛变量和剩余变量的价值系数为零，人工变量价值系数为任意小的负数，例如 −T（本程序设为−1000.）。

经过这几步处理工作，等效模型的变量数便由 r 个增加到 n 个（加进了松弛、剩余及人工变量）。且松弛变量和人工变量可构成等效问题的初始基础可行解。

等效模型可改写为：

目标函数： $\max z$

满足约束条件：

$$z-\sum_{j=1}^{n}\hat{c}_j x_j=0 \tag{1-1-1}$$

$$\sum_{j=1}^{n}a_{ij} x_j=b_i \quad (i=1,2,\cdots,m) \tag{1-1-2}$$

$$x_j\geqslant 0$$

因为每个人工变量的$\hat{c}_j=-T$，必须用$-T$乘方程组（1-1-2）中含人工变量的方程，并将所得结果方程加到方程（1-1-1）中，可得到

目标函数：

$$\max: z$$

满足约束条件：

$$z-\sum_{j=1}^{n}c_j x_j=b_0 \tag{1-1-3}$$

$$\sum_{j=1}^{n}a_{ij} x_j=b_i \qquad (i=1,2,\cdots,m) \tag{1-1-4}$$

$$x_j\geqslant 0$$

其中，$b_0=-T\sum\limits_{*}b_k$，$*$表示含有人工变量的方程。这样，方程（1-1-4）中每个方程含有 1 个松弛变量或人工变量，这个变量在该方程中的系数为 1，而在方程组（1-1-4）中的其它每个方程和方程（1-1-3）中的系数为零。方程（1-1-3）又称为目标函数方程，它在单纯形算法中作为第 $m+1$ 个约束条件。

单纯形法就是在这种等效线性规划模型中不断经过初等变换求出极值的。

2. 单纯形法寻求极大值过程（极小值也一样）。

(1)单纯形法的最大特点是从一组可行解向量算起，逐步寻找最大值，即在迭代过程中，每组解都满足等效模型的约束条件，因此，一开始就以松弛变量及人工变量组成一组基础可行解，然后移到一个目标函数不递减的邻近的基础可行解。如果这样的解不存在，就说明已获得等效模型的最优解。

(2) 所得的等效模型最优解中，若人工变量等于零，这个解也就是原模型问题的最优解；否则，就说明原问题不存在最优解。

(3) 如果已达最优解，但目标函数方程中，非基础变量有零系数，则说明有无限个最优解。

(4) 如果迭代过程中，按规则找不到需换出的基变量，则说明函数无界，不存在有限值的最优解。

(5) 如果得到一组最优解，程序相应求出最优对偶解。

上面叙述的是程序求解原线性规划问题的主要步骤，用户使用时，只需将原问题的有关数据输入即可，不需化为等效模型。

三、算法框图（见图 1-1-1）

框图中的一些符号含义（与等效模型中的方程（1-1-3）及（1-1-4）的符号比较。右边是等效模型符号，左边是框图符号）：

$$a_{i,n+1}=b_i \quad (i=1,2,\cdots,m)$$

$$a_{m+1,n+1}=b_0$$

$$a_{m+1,j}=-C_j \quad (j=1,2,\cdots,n)$$

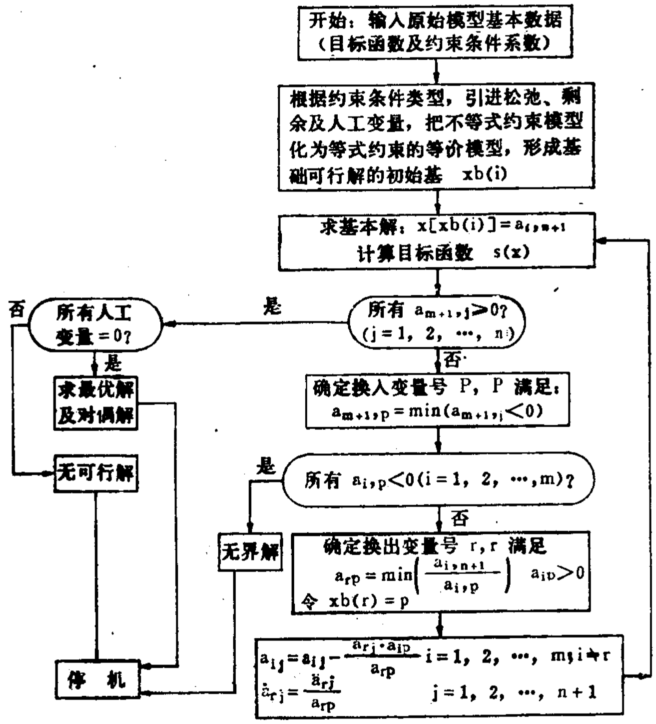


图1-1-1　线性规划的单纯形算法框图

四、举例说明程序使用方法：

程序中包括主程序和两个子程序 SSARTV，SIMPLX。

主程序提供（读入）原始线性规划有关数据，并调用子程序。

子程序 SSARTV 提供松弛变量、剩余变量及人工变量。

子程序 SIMPLX 做单纯形算法工作。

原变量价值系数绝对值必须小于 100（因人工变量价值系数取 −1000.），否则，需将所有原系数进行比例缩小。

**举例**：

设求解的线性规划模型为：

$$\min: \ z^*=-6x_1-4x_2$$

满足约束条件：

$$2x_1+3x_2\leqslant 30$$

$$3x_1+2x_2\leqslant 24$$

$$x_1+x_2 \geqslant 3$$
$$x_1\ \ x_2\ \geqslant 0$$

程序使用方法（结合上面例题）：

1. 本算法程序能解决相当于 25 个约束和 75 个变量的等效线性规划模型（对于原问题来说，应考虑松弛、剩余及人工变量数目）。若要解决大于 25 的 P 个约束和大于 75 的 Q 个变量，只需将主程序及子程序中的 DIMENSION 语句、INTEGER 语句及 COMMON语句中出现的 26 改为 P+1, 76 改为 Q+1 即可。本例题的等效模型只有 3 个约束、6 个变量（原有变量 2 个，另加 2 个松弛变量、1 个剩余变量及 1 个人工变量），故不需修改这几个语句。

2. 输入数据的含义及输入格式：

第 1 行，TITLE，标题，用字符描述问题。本例描述为“THIS IS EXAMPLE”。

第 2 行，

1—5 列，M——约束数（I5），本例为 3。

6—10 列，K —变量数（I5），本例为 2。

11—15 列，NLET——不等式（≤），约束数（I5），本例为 2。

16—20 列，NGET——不等式（≥）约束数（I5），本例为 1。

21—25 列，NET——等式（＝）约束数，（I5），本例为 0。

26—30 列，NTYPE——0 表示求最小值

1 表示求最大值（I5），本例为 0。

第 3—T 行，共 M 组，每个约束为 1 组。以第 I 组为例：

第 I 行

1—10 列，CODE (I)—0 表示该组为不等式（≤）约束。1 表示该组为不等式（≥）约束。2 表示等式约束（I10）。

11—20 列，B (I)，约束 I 右边常数项（F10.0）。

第 I+1 行，A(I,J)，约束I中变量 J 的系数（$a_{i,j}$），以 8F10.0 格式顺行输入，如果 K>8，在下行继续。

例题中，第 1 组约束条件为不等式（≤）约束，则该组输入格式为：

0，30.↙

2.，3.↙

第 T+1 行，C（J），目标函数各变量价值系数，以 8F10.0 格式顺行输入，若 K>8，下行继续。本例为 －6.，－4.↙

第 Z 行，1—5列 NOPT＝0 表示只打印最优解 NOPT＝1 表示打印寻优过程中所有解(I5)，本例题为 1。

例题详细输入格式列在程序清单后面。

3. 输出打印内容：

(1) 原问题的全部输入参数（包括题目标头）。

(2) 若 NOPT＝1，输出每次中间计算的基本解及最优解。若 NOPT＝0，只输出最优解（最优解包括原问题最优解 X(I)、对偶最优解 Y(I) 及最优目标函数值）。

(3) 若原问题无可行解，则输出标志为：

A FEASIBLE SOLUTION DOES NOT EXIST

(4) 若原问题为无界解，则输出标志为：

OBJECTIVE FUNCTION IS NOT BOUNDED BY CONSTRAINTS.

本例题在 IMS-8000 微处理机上的最后计算结果为：

目标函数最优值为 Z＝－48.00

原问题的最优解 X(1)＝8.，X(2)＝0.

对偶问题的最优解 Y(1)＝0.，Y(2)＝ 2.0，Y(3)＝0.0

详细输入和输出的中间及最后结果列在程序清单之后以 EXAMPLE 为标头的下面。其中 $X_B(I)$ 表示基础可行解。

附：程序清单及例题的输入、输出打印结果。

```
C      *      LINEAR     PROGRAMMING
C      *      SIMPLEX     METHOD
       INTEGER CODE,XB,BASICS,OPTSOL
       DIMENSION A(26,76),XB(76),TITLE(20)
       COMMON B(26),C(76),CODE(26),KP1,MP1,N,K,M,NGET,NLET,NET,NTYPE,NP1,
      *NC,NC1,INDEXG,INDEXL,INDEXE,NFLAG,BASICS,OPTSOL,SUM,NOPT,X(76)
       READ(1,10)TITLE
    10 FORMAT(20A4)
       WRITE(2,15)TITLE
    15 FORMAT(1X,20A4,//)
       READ(1,20)M,K,NLET,NGET,NET,NTYPE
    20 FORMAT(6I5)
       DO 25 I=1,M
       READ(1,30)CODE(I),B(I)
    30 FORMAT(I10,F10.0)
       READ(1,29)(A(I,J),J=1,K)
    29 FORMAT(8F10.0)
    25 CONTINUE
       READ(1,29)(C(J),J=1,K)
       WRITE(2,40)
    40 FORMAT(5X,'THE ORIGINAL COEFFICIENTS OF THE CONSTRAINTS',//15X,'CO
      *DE 0==) (OR= CONSTRAINT',/15X,'CODE 1==) )OR= CONSTRAINT',/15X,'CO
      *DE 2 ==) = CONSTRAINT',//)
       WRITE(2,55)
    55 FORMAT('   I CODE CONSTANT  A(I,1)   A(I,2)   A(I,3)   A(I,4)   A(I,5)
      * A(I,6)   A(I,7)   A(I,8)',/)
       DO 45 I=1,M
       WRITE(2,51)I,CODE(I),B(I)
    51 FORMAT(I3,I4,F9.2)
       WRITE(2,52)(A(I,J),J=1,K)
    52 FORMAT('+',15X,8F8.2,/(16X,8F8.2))
    45 CONTINUE
       IF(NTYPE.NE.0) GO TO 35
       WRITE(2,36)
    36 FORMAT(// 5X,'THE COEFFICIENTS IN THE ORIGINALOBJECTIVE FUNCTION
      *TO BE MINIMIZED ARE:',/)
       GO TO 37
    35 WRITE(2,38)
    38 FORMAT(//  5X,'THE COEFFICIENTS IN THE ORIGINAL OBJECTIVE FUNCTION
      *TO BE MAXIMIZED ARE:',/)
    37 WRITE(2,39)(C(J),J=1,K)
    39 FORMAT(16X,8F8.2/16X,8F8.2)
       READ(1,20)NOPT
C      *    *     *    *    *
       CALL  SSARTV(A,XB)
       IF(IFLAG.EQ.1) GO TO 50
       BASICS=0
       OPTSOL=0
       WRITE(2,160)
   160 FORMAT(//)
       CALL SIMPLX(A,XB)
       IF(NFLAG.EQ.1.OR.NFLAG.EQ.2) GO TO 50
       IF(NTYPE.EQ.1) GO TO 220
       SUM=-SUM
C
C
   220 WRITE(2,230)SUM
   230 FORMAT(10X,'OPTIMAL VALUE OF THE ORIGINAL OBJECTIVE FUNCTION IS',
      *F12.2)
       WRITE(2,240)
   240 FORMAT(5X,'OPTIMAL SOLUTION FOR PRIMAL X(I)')
       DO 250 I=1,K
   250 WRITE(2,260)I,X(I)
   260 FORMAT(7X,'X(',I2,')=',F12.2)
   270 WRITE(2,280)
   280 FORMAT(5X,'OPTIMAL SOLUTION FOR DUAL Y(I)'/)
```

```
      DO 290 J=NC,N
      I=J-K-NGET
  290 WRITE(2,300) I,C(J)
  300 FORMAT(7X,'Y(',I2,')=',F12.2)
   50 WRITE(2,2222)
 2222 FORMAT(/,20X,'JOB TERMINATED')
      STOP
      END

      SUBROUTINE SSARTV(A,XB)
C     SSARTV
C          A SUBROUTINE THAT SUPPLIES THE SLACK, SURPLUS, AND ARTIFICIAL
C               VARIABLES NEEDED TO PERFORM SIMPLEX METHOD.
      INTEGER CODE,XB,BASICS,OPTSOL,ARTV(52)
      DIMENSION A(26,76),XB(76)
      COMMON B(26),C(76),CODE(26),KP1,MP1,N,K,M,NGET,NLET,NET,NTYPE,NP1,
     *NC,NC1,INDEXG,INDEXL,INDEXE,NFLAG,BASICS,OPTSOL,SUM,NOPT,X(76)
C INITIALIZE VARIABLES
      IFLAG=0
      IA=1
      KP1=K+1
      MP1=M+1
      N=K+2*NGET+NLET+NET
      NP1=N+1
      NC=K+NGET+1
      NC1=NC+NLET
      INDEXG=K+1
      INDEXL=K+NGET+1
      INDEXE=K+NGET+NLET+1
      DO 69 I=1,MP1
      DO 69 J=KP1,NP1
   69 A(I,J)=0.
      DO 5  I=1,M
    5 A(I,NP1)=B(I)
      DO 4  I=1,M
      IF(CODE(I).EQ.0) GO TO 6
      IF(CODE(I).EQ.1) GO TO 8
      ARTV(IA)=I
      IA=IA+1
      XB(I)=INDEXE
      A(I,INDEXE)=1.
      INDEXE=INDEXE+1
      GO TO 4
    8 XB(I)=INDEXE
      ARTV(IA)=I
      IA=IA+1
      A(I,INDEXE)=1.
      INDEXE=INDEXE+1
      A(I,INDEXG)=-1.
      INDEXG=INDEXG+1
      GO TO 4
    6 XB(I)=INDEXL
      A(I,INDEXL)=1.
      INDEXL=INDEXL+1
    4 CONTINUE
C CHECK FOR CORRECT  DATA
      IF(INDEXG.NE.NC) GO TO 100
      IF(INDEXL.NE.NC1) GO TO 110
      IF(INDEXE.NE.NP1) GO TO 120
      GO TO 151
  100 WRITE(2,101)
  101 FORMAT(///' NUMBER OF >OR= CONSTRAINTS DOES NOT MATCH VALUE READ I
     *N')
      IFLAG=1
      RETURN
  110 WRITE(2,111)
```

```
  111 FORMAT(///' NUMBER OF <OR= CONSTRAINTS DOES NOT MATCH VALUE READ I
     *N')
      IFLAG=1
      RETURN
  120 WRITE(2,121)
  121 FORMAT(//' NUMBER OF = CONSTRAINTS DOES NOT MATCH VALUE READ IN')
      IFLAG=1
      RETURN
C CHECK FOR MAXIMIZATION
  151 CONTINUE
      IF(NTYPE.EQ.0) GO TO 12
      DO 60  J=1,K
   60 A(MP1,J)=-C(J)
      GO TO 50
   12 DO 55  J=1,K
   55 A(MP1,J)=C(J)
   50 DO 61 J=KP1,NP1
      A(MP1,J)=0.
   61 C(J)=0.
      DO 62 J=1,K
   62 C(J)=-A(MP1,J)
      DO 63 J=NC1,N
   63 C(J)=-10.E2
      IF(NGET+NET.EQ.0) RETURN
      IA=IA-1
      KPGTE=K+NGET
      DO 64 J=1,KPGTE
      SUM=0.
      DO 65  I=1,IA
      IARTVI=ARTV(I)
   65 SUM=SUM+A(IARTVI,J)
   64 A(MP1,J)=A(MP1,J)-10.E2*SUM
      SUM=0.
      DO 66 I=1,IA
      IARTVI=ARTV(I)
   66 SUM=SUM+A(IARTVI,NP1)
      A(MP1,NP1)=A(MP1,NP1)-10.E2*SUM
      RETURN
      END

      SUBROUTINE  SIMPLX(A,XB)
C     SIMPLX
C            SUBROUTINE TO PERFORM THE WORK OF THE SIMPLX METHOD.
      INTEGER CODE,XB,BASICS,OPTSOL
      DIMENSION A(26,76),XB(76)
      COMMON B(26),C(76),CODE(26),KP1,MP1,N,K,M,NGET,NLET,NET,NTYPE,NP1,
     *NC,NC1,INDEXG,INDEXL,INDEXE,NFLAG,BASICS,OPTSOL,SUM,NOPT,X(76)
      NFLAG=0
C     STEP1
C          OBTAIN BASIC FEASIBLE SOLUTION OF EQUIVALENT MODEL.
  100 BASICS=BASICS+1
      IF(NOPT.EQ.0) GO TO 200
  105 WRITE(2,104)BASICS
  104 FORMAT(5X,'BASIC SOLUTION',I4,/)
      DO 110 I=1,M
  110 WRITE(2,106)I,XB(I),A(I,NP1)
  106 FORMAT(7X,'XB(',I2,')= X(',I2,')=',F12.2)
      SUM=0.
      DO 111 I=1,M
      IXBI=XB(I)
  111 SUM=SUM+C(IXBI)*A(I,NP1)
      WRITE(2,130)SUM
  130 FORMAT( /7X,'CURRENT VALUE OF THE OBJECTIVE FUNCTION IS',E18.8//)
      IF(OPTSOL.EQ.1)GO TO 920
C     STEPS 2,3
C            CHOOSE THE NONBASIC VARIABLE WITH THE MOST NEGATIVE COEFFICIENT
```