ภาคผนวก ก

การประเมินสมรรถนะทางเศรษฐศาสตร์ของเครื่องอบแห้งพลังงานแสงอาทิตย์แบบอุโมงค์ลม

ในการประเมินสมรรถนะทางเศรษฐศาสตร์ของเครื่องอบแห้งพลังงานแสงอาทิตย์แบบอุโมงค์ลมที่ใช้กระจกปิดด้านบนนี้ ผู้วิจัยได้ทำการประเมินค่าวัสดุที่ใช้ในการก่อสร้างดังรายละเอียดต่อไปนี้

- ค่าโครงสร้าง ถาดวาง อุปกรณ์ต่างๆ	60,000 บาท
- ค่าแผ่นกระจก	7,500 บาท
- ค่าแผงโซลาร์เซลล์และพัดลม	8,000 บาท
- ค่าแรง	8,400 บาท
รวม	83,900 บาท

ดังนั้นจะได้ค่าใช้จ่ายในการสร้างเครื่องอบแห้งดังกล่าวเป็นจำนวนเงิน 83,900 บาท ในการประเมินทางเศรษฐศาสตร์ ผู้วิจัยจะใช้กล้วยน้ำว้าเป็นผลิตภัณฑ์ในการประเมิน โดยใน 1 ปีจะประมาณว่าสามารถอบกล้วยได้ 40 ครั้ง โดยแต่ละครั้งจะใช้กล้วยน้ำว้าสุกที่ปอกเปลือกแล้ว 100 กิโลกรัม เมื่อทำการอบแห้งแล้วจะได้กล้วยอบแห้ง 35 กิโลกรัม ซึ่งสามารถนำไปบรรจุกล่องได้ประมาณ 100 กล่อง ซึ่งราคาขายตามท้องตลาดอยู่ที่ 25 บาทต่อกล่อง ดังนั้นในการอบแห้งกล้วย 1 ครั้งจะมีรายได้ 2,500 บาท ในระยะเวลา 1 ปี จะมีรายได้ 100,000 บาท โดยที่ต้นทุนของกล้วยน้ำว้าสดและค่าบรรจุภัณฑ์รวมกันเป็นจำนวนเงิน 1,400 บาท ในระยะเวลา 1 ปี คิดเป็นเงิน 56,000 บาท ถ้าคิดค่าบำรุงรักษาเป็น 1 % ต่อปีของต้นทุนในการสร้างเครื่องอบแห้งนี้ จะคิดเป็นเงิน 839 บาท เมื่อรวมแล้วใน 1 ปีต้องเสียค่าใช้จ่ายในการทำกล้วยอบแห้งเป็นจำนวนเงิน 56,839 บาท ดังนั้นใน 1 ปีจะมีกำไรจากการทำกล้วยอบแห้งทั้งสิ้น 43,161 บาทต่อปี จากผลกำไรดังกล่าว การลงทุนสร้างเครื่องอบแห้งนี้จะคุ้มทุนภายในระยะเวลาประมาณ 2 ปี

ภาคผนวก ข

ค่าประสิทธิภาพของแผงรับรังสีดวงอาทิตย์

ค่าประสิทธิภาพของแผงรับรังสีดวงอาทิตย์

ในการคำนวณหาค่าประสิทธิภาพของแผงรับรังสีดวงอาทิตย์สามารถหาได้จากสมการ

$$\eta = \frac{\dot{m} C_p \Delta T}{I_T A} \text{ x } 100$$

เมื่อ η = ค่าประสิทธิภาพของแผงรับรังสีดวงอาทิตย์ [%]

$\dot{m}$ = อัตราการไหลของมวลอากาศ [kg/s]

C_p = ค่าความร้อนจำเพาะของอากาศ [J/kg-K]

ΔT = ผลต่างของอุณหภูมิที่ไหลออกจากแผงรับรังสีดวงอาทิตย์กับอุณหภูมิที่ไหลเข้าแผงรับรังสีดวงอาทิตย์ [K]

I_T = ค่าความเข้มรังสีดวงอาทิตย์ที่ตกกระทบบนแผงรับรังสีดวงอาทิตย์ [W/m^2]

A = ขนาดพื้นที่รับแสงของแผงรับรังสีวงอาทิตย์ [m^2]

ตารางที่ 1 แสดงค่าประสิทธิภาพของแผงรับรังสีดวงอาทิตย์

การทดลองอบแห้งกล้วย	ค่าประสิทธิภาพของแผงรับรังสีดวงอาทิตย์ , %
ครั้งที่ 1	64.7
ครั้งที่ 2	69.8
ครั้งที่ 3	61.4
ครั้งที่ 4	68.3
ครั้งที่ 5	58.0

ภาคผนวก ค

โปรแกรมคอมพิวเตอร์

1. โปรแกรมสำหรับ simulation ระบบ

```
          PROGRAM MAIN
          USE DFLIB
          PARAMETER (LDA=180)
      COMMON /TEMP/T
      DIMENSION T(200,2),TIM(300),IT(300),TA(300),TFI(300),WIND(300),
     1     TFO(300),FLOW(300),QU(300),MC(300),TDI(300),TOUT(300)
      REAL IT,LENGTH,MC,HA,GBR,FLOW,MFLOW,MASSFLOW
      DATA CPA/1002./
C    ALPHAP..Aborsorption coefficient of the absorber (for incoming solar radiation),[decimal]
C    D.......Average high of air gab in the collector,[m]
C    DELT....Time step in finit difference with implicit approach,[s]
C    DELX....Length of each descrete section in finit diferrence method,[m]
C    DENS....Density of fluid,[kg/m3]
C    EC......Emissivity of the cover,[decimal]
C    EO......Eccentricity correction factor of earth,[radiance]
C    EJ .....Emissivity of the banana,[decimal]
C    EP......Emissivity of the plate,[decimal]
C    FLOW....Total mass flow rate of the collector,[m3/s]
C    G.......Mass flow rate per unit area perpendicular to the flow, [kg/s-m2]
C    GBR.....Gobal radiation [W/m2]
C    HFF.....Heat transfer coefficient for forced convection in the collector , [W/K-m2]
C    IT......Incident solar radiation on the collector,[J/m2]
C    LENGTH..Length of the collector,[m]
C   LENGTHD..Length of the dryer,[m]
C   LL......Local longitude,[degree]
C   MASSFLOW. Total mass flow rate of the collector,[m3/s]
C   NSEC....Total number of sections of the collector for the finite difference
C   RHOC....Reflectance of cover glass,[decimal]
C   TA......Ambient air temperature,['C]
C   TAM.....Ambient air temperature,[K]
C   TFI.....Inlet air temperature of the collector,[K]
C   TFO.....Outlet air temperature of the collector,[K]
C   TIM.....Time [s]
C   TOUT....Outlet air temperature of the dryer,[K]
```

```
C    THICKI..Thickness of back insulator,[m]
C    TRANC...Transmition coefficient of the cover (for thermal radiation), [decimal]
C    WIDTH...Width of the collector,[m]
C    W.......Wight,[kg]
C    XMC... .Actual moisture content of  banana, [kg/kg,db]
C    XMO... .Initial moisture content of banana,[kg/kg,db]
C    XME... .Equilibrium moisture content of banana, [kg/kg,db]
C
         NSEC    = 60
         Istart  =  1
         Istop   =  55
         LENGTH  =  4.00
         LENGTHD =  8.00
         DELT    =  600.0
         WIDTH   =  1.160
         THICKI  =  0.030
         D           = 0.16
         EC          = 0.84
         EP          = 0.90
         EJ  = 0.90
         TRANC  = 0.88
         ALPHAP = 0.98
         ALPHAJ = 0.80
         RHOC    = 0.08
C
C        *** Define parameter of product for any experiment ***
C
         W = 100.0
         XMO = 2.860
         XMC = 2.000
C
C        *** Read all MEASURMENTAL data ***
C
         OPEN(1,FILE='exam.DAT',STATUS='OLD')
         DO 11 ITIME = 1,ISTOP
```

```
      READ(1,10) TIM(ITIME),IT(ITIME),TA(ITIME),
    1 TFO(ITIME),TOUT(ITIME),FLOW(ITIME)
C      WRITE(*,10)TIM(ITIME),IT(ITIME),TA(ITIME),
C   1 TFO(ITIME),TOUT(ITIME),FLOW(ITIME)
   11 CONTINUE
   10 FORMAT(F6.3,X,F7.3,X,F4.1,X,F5.2,X,F5.2,X,F4.2)
      CLOSE(1)
C
      TSTART = ISTART*1.0
      ITEND=INT(600.0/DELT)
      DELX=LENGTH/NSEC

C
C *** Calculate the temperature of the collector for each hour ***
C
      OPEN(4,FILE='OUT.DAT',STATUS='OLD')
C
C *** Initialize the temperature of all element of the collector ***
C
C     DO 70 I=1,150
C     T(I,1)=TFI(ISTART)+273.0
C  70 CONTINUE
C
      DO 90 ITIME=ISTART,ISTOP
C
C      *** Choose air flow rate between experimental and simulated value ***
C

      TAM = TA(ITIME)+273.15
      GBR = IT(ITIME)*1.0
C     WRITE(*,*) TAM,GBR
C
C     MASSFLOW = MFLOW(TAM,GBR)
C      FLOW(ITIME) = MASSFLOW
C      FLOW(ITIME) = 0.05
```

```
C           WRITE(*,*) MASSFLOW*3600
C
           G = FLOW(ITIME)/(D*WIDTH)
           HFF = HCF(G)
           TFI(ITIME) = TA(ITIME)
           TI=ITIME*1.0
     CALL COL1(TFC,NSEC,TI,IT(ITIME),TFI(ITIME)+273.,TA(ITIME)+273.,
     1          WIND(ITIME),FLOW(ITIME),DELX,DELT,WIDTH,THICKI,D,EC,EP,
     2          TRANC,ALPHAP,RHOC,HFF)
C           DO 87 N=1,NSEC
C           WRITE(*,86) (T(J+3*(N-1))-273.0,J=1,3)
C  86       FORMAT(3(1X,F8.2))
C  87       CONTINUE
C
C           QU(ITIME)=CPA*FLOW(ITIME)*(TFO(ITIME)-TFI(ITIME))/60.
            QU(ITIME)=CPA*FLOW(ITIME)*(TFC-273.-TFI(ITIME))/60.
C
C     *** Write the values of outlet temperater of the collector ***
C
C           WRITE(*,85)TIM(ITIME),TFI(ITIME),T(2+3*(NSEC-1),1)-273.,
C    1      TFO(ITIME), IT(ITIME),TA(ITIME),QU(ITIME),FLOW(ITIME)
C  85       FORMAT('t=',F6.2,X,'TFI=',F6.2,X,'TFOcal=',F7.2,X,'TFout=',F8.2,X,
C    1      'IT=',F9.2,2X,' TA=',F6.2,2X,' Qu=',F8.4,2X,' Hcf=',F6.3)
C
                  TIN=TFC
                  RHA=0.55
                  TDI(ITIME)=TIN
                  HA=HADBRH(TA(ITIME)+273.,RHA)
                  HIN=HA
                  MC(1) = XMC
                  RHOPS = W/(1.0+XMO)/LENGTHD/WIDTH
C          WRITE(*,*) XMC,RHOPS
C
     CALL TUN(TDO,XM,NSEC,TSTART,ITEND,IT(ITIME),TDI(ITIME),
     1      TA(ITIME)+273.,TI,WIND(ITIME),FLOW(ITIME),HIN,DELX,DELT,WIDTH,
```

```
  2       THICKI,D,EC,EJ,TRANC,ALPHAJ,RHOC,RHOPS,HFF,XMO,XMC)
              TIME=ITIME*1.0
              MC (ITIME)=XM
              XMC = MC(ITIME)
        WRITE(4,96) TIM(ITIME),TFI(ITIME),TFO(ITIME),TDI(ITIME)-273.,
  1       TOUT(ITIME),TDO-273.,FLOW(ITIME),MC(ITIME)
 96       FORMAT(F6.3,X,F7.2,X,F7.2,X,F8.2,X,
            1     F6.2,2X,F8.2,2X,F9.3,2X,F10.4)
 90 CONTINUE
  CLOSE(4)
  STOP
  END
```

2. โปรแกรมสำหรับคำนวณค่าอุณหภูมิอากาศตรงทางออกของแผงรับรังสี

```
      SUBROUTINE COL1 (TFO,NSEC,IT,TFI,TA,FLOW,WIND,DELX,DELT,
                    WIDTH,THICKI,D,EC,EP,TRANC,ALPHAP,HFF)
      PARAMETER (LDA=180)
      COMMON /TEMP/ T
      DIMENSION A(LDA,LDA),B(LDA),WORK(LDA),T(LDA)
      INTEGER IWORK(LDA),I,J,N,NT,ITASK,IND
      REAL IT,LENGTH,RCOND,FLOW,TFO,L
      DATA SIGMA/5.6697E-8/,CONDI/0.03/,DENA/1.000/,
               CPA/1002./,DENC/200./,DENP/6000./,CPC/500./,CPP/400./
C
C     Important parameter using in this program
C
C  CONDI....Conductivity of  insulating material, [W/m]
C   CPA...Specific heat of  air, [kJ/kg-°C]
C   CPC...Specific heat of  cover material, [kJ/kg-°C]
C   CPP...Specific heat of  absorber material , [kJ/kg-°C]
C   DENA...Density of  air, [kg/m3]
C   DENC...Density of  cover material, [kg/m3]
C   DENP...Density of  absorber material, [kg/m3]
C   HCPF, HCCF..Force convective heat transfer coefficient of the fluid [W/m2-K]
C   HRCS...Radiative heat transfer coefficient of the cover to sky, [W/m2-K]
C   HRPC...Radiative heat transfer coefficient of the absorber plate to cover, [W/m2-K]
C   HW.......Convective heat transfer coefficient of the cover due to wind [W/m2-K]
C   SIXMA...Stefan-Boltzmann constant
C   TS......Sky temperature, [°C]
C   UB.......Conductive heat transfer coefficient of the back insulator [W/m2-K]
C
      ALPHAC=1.-TRANC-RHOC
      ITASK=1
      NT=3*NSEC
C
C     Calculation of parameters
C
      LENGTH=DELX*NSEC
```

```
      AREA=LENGTH*WIDTH
      G=FLOW/(D*WIDTH)
      TS=0.0552*TA**1.5
      HW=5.7+3.8*WIND
      UB=CONDI/THICKI
      CONC=1.3*CPC/DELT
      CONP=0.001*CPP*DENP/DELT
      CONX=0.25*D*G*CPA/DELX
      CONT=DENA*D*CPA/DELT
      HCCF = HFF
      HCPF = HFF
C
C     Increase time to the next time step, dt, after calculating temperature of all nodes.
C
      DO 15 I=1,NT
      T(I)=TFI
   15 CONTINUE
C
C     Initialization of the matrix elements
C
      DO 20 I=1,NT
      B(I)=0.0
      DO 20 J=1,NT
      A(I,J)=0.0
   20 CONTINUE
C
C     Begin the calculation of the temperature of all nodes at t+dt,
C     starting with the computation of the matrix element
C
      DO 90 N=1,NSEC
C
C     Calculate radiative heat transfer coefficient (depend on temperature)
C
       HRCS=EC*SIGMA*(TS*TS+T(1+3*(N-1))*T(1+3*(N-1)))*(TS+T(1+3*(N-1)))
```

```
      HRPC=SIGMA*(T(1+3*(N-1))*T(1+3*(N-1))+T(3+3*(N-1))*T(3+3*(N-1)))
           *(T(1+3*(N-1))+T(3+3*(N-1)))/(1./EC+1./EP-1)
C
C     Calculate the matrix elements
C
C     The matrix elements of Tc
C
      A(1+3*(N-1),1+3*(N-1))= CONC+HCCF+HW+HRCS+HRPC
      A(2+3*(N-1),1+3*(N-1))=-HCCF
      A(3+3*(N-1),1+3*(N-1))=-HRPC
C
C     The matrix element of Tf (normal elements)
C
      A(1+3*(N-1),2+3*(N-1))=-HCCF
      A(3+3*(N-1),2+3*(N-1))=-HCPF
C
C   The matrix element of Tp
C
      A(1+3*(N-1),3+3*(N-1))=-HRPC
      A(2+3*(N-1),3+3*(N-1))=-HCPF
      A(3+3*(N-1),3+3*(N-1))= CONP+HRPC+HCPF+UB
C
C     Right hand side vector B, except for the term from Tf eq.
C
      B(1+3*(N-1))=CONC*T(1+3*(N-1))+HW*TA+HRCS*TS+QSC
      B(3+3*(N-1))=CONP*T(3+3*(N-1))+UB*TA+QSP
C
C     Matrix elements of Tfx-dx,Tf,Tfx+dx and B
C
      IF(N.EQ.1) THEN
            A(2+3*(N-1),5+3*(N-1))=CONX
            A(2+3*(N-1),2+3*(N-1))=HCPF+HCCF+CONT
            B(2+3*(N-1))=CONX*TFI -CONX*T(5+3*(N-1))+CONX*TFI+CONT*T(2+3*(N-1))
      ELSE
      IF(N.GT.1.AND.N.LT.NSEC) THEN
```

```
                A(2+3*(N-1),3*(N-1)-1)=-CONX
                A(2+3*(N-1),2+3*(N-1))= HCPF+HCCF+CONT
                A(2+3*(N-1),5+3*(N-1))= CONX
                B(2+3*(N-1))=CONX*T(3*(N-1)-1)-CONX*T(5+3*(N-1)) +CONT*T(2+3*(N-1))
        ELSE
        IF(N.EQ.NSEC) THEN
                A(2+3*(N-1),2+3*(N-1))= 2.*CONX+HCPF+HCCF+CONT
                A(2+3*(N-1),3*(N-1)-1)=-2.*CONX
                B(2+3*(N-1))=-2.*CONX*T(2+3*(N-1))+2.*CONX*T(3*(N-1)-1)+CONT*T(2+3*(N-1))
        ELSE
        WRITE(*,30)
   30   FORMAT(' Some calculation of matrix elements are missing')
        ENDIF
        ENDIF
        ENDIF
   90 CONTINUE
C
C         End of the computation of the matrix element
C
C       Use a subroutine to solve the matrix equation [A].T=B
C       where NT in the argument, original is N
C
        CALL SGEFS(A,LDA,NT,B,ITASK,IND,WORK,IWORK,RCOND)
C
C       Transfer the solution of matrix equation to the temperature
C       values for use to calculate the heat transfer coeff. for the next time step.
C
        DO 110 I=1,NT
                T(I)=B(I)
   110 CONTINUE
        TFO   = T(2+3*(NSEC-1))
        RETURN
        END SUBROUTINE
```

3. โปรแกรมสำหรับคำนวณค่าความชื้นของผลิตภัณฑ์ในส่วนอบแห้ง

```
      SUBROUTINE TUN(TDO,XM,NSEC,TSTART,ITEND,IT,TDI,TA,TI,FLOW,HIN,DELX,DELT,
                 WIDTH,THICKI,D,EC,EP, TRANC,ALPHAP,RHOC,RHOPS,HFF,XMO,XMC)
      PARAMETER (LDA=240)
      COMMON /TEMP/ T
      DIMENSION A(LDA,LDA),B(LDA),WORK(LDA),TF(LDA),RH(LDA),T(LDA),HM(LDA),MC(LDA)
      INTEGER IWORK(LDA),I,J,N,NT,ITASK,IND
      REAL IT,LENGTH,RCOND,FLOW,MC,SUMXM,XM,TF,RH,HM,HIN,TDO,L
      DATA SIGMA/5.6697E-8/,CONDI/0.03/,DENA/1.000/, CPA/1002./,DENC/200./,DENB/6000./,
         CPC/500./,CPB/400./,CW/4186./,CV/1890./
C
C     Important parameter using in this program
C
C   CPB…Specific heat of  absorber material , [kJ/kg-oC]
C   CPP…Specific heat of  product , [kJ/kg-oC]
C   CV…Specific heat of water vapor , [kJ/kg-oC]
C   CW…Specific heat of liquid water , [kJ/kg-oC]
C   DENB..Density of absorber material, [kg/m3]
C   HCPF, HCCF,HCBF...Force convective heat transfer coefficient of the fluid [W/m2-K]
C   HFG    Latent heat for evaporate liquid water, [kJ]
C   HRPC...Radiative heat transfer coefficient of the products to cover, [W/m2-K]
C   HM…Humidity of  drying fluid, [decimal]
C   MC…Moisture content of drying product, [decimal, db]
C   POROS...Porosity of  drying products, [decimal]
C   RH…Relative humidity of drying fluid, [decimal]
C   TF….Temperature of drying fluid, [K]
C   XMT...Moisture content of  product at next time step, [decimal]
C
C     Optical properties of the collector
C
      ALPHAC=1.-TRANC-RHOC
      ITASK=1
      NT=4*NSEC
C
C     Calculation of parameters which are independent of time.
```

```
C
      LENGTHD=DELX*NSEC
      AREA=LENGTHD*WIDTH
      G=FLOW/(D*WIDTH)
      TS=0.0552*TA**1.5
      HW=5.7+3.8*WIND
      UB=CONDI/THICKI
      CONC=1.3*CPC/DELT
      CONB=0.001*CPB*DENB/DELT
      CONH=RHOPS*DELX/(G*D)/DELT
      HCPF = HFF
      HCCF = HFF
      HCBF = HFF
      POROS=0.01
C
C     Increase time to the next time step, dt, after calculating temperature of all nodes.
      DO 15 I=1,NT
      T(I)=TDI
   15 CONTINUE
      XMS = 0.0
C
C     Initialization of the matrix elements
C
      DO 20 I=1,NT
      B(I)=0.0
      DO 20 J=1,NT
      A(I,J)=0.0
   20 CONTINUE
C
C     Begin the calculation of the temperature of all nodes at t+dt,
C     starting with the computation of the matrix element
C
      DO 90 N=1,NSEC
C
C      Calculate initial and boundary conditions
```

```
C
            HM(1)= HIN
            IF (TI.EQ.TSTART) THEN
                MC(N) = XMC
                TF(N) = TDI
                RH(N) = RHDBHA(TDI,HIN)
            ELSE
                IF (N .EQ. 1) THEN
                TF(1) =   TDI
                ELSE
                TF(N) = T(2+4*(N-1))
                RH(N) = RHDBHA(TF(N),HM(N))
                ENDIF
                XMC = MC(N)
            ENDIF
                TFC=TF(N)-273.
                RHP=RH(N)*100.
C
C    Calculate moisture content and humidity at any distant with subroutine layer
C
        CALL LAYER (XMT,XMC,TFC,RHP,DELT,XMO)
            HM(N+1) = HM(N)+CONH*(XMC-XMT)
            RH(N) = RHDBHA(TF(N),HM(N))
            MC(N) = XMT
            XMS = XMS+XMT
            XM  = XMS/N
C
C    Calculate variable term for calculating metrix element
C
            CPP  = 2890.-30.3*(T(3+4*(N-1))-273.)
            CONP = RHOPS*(CPP+CW*XMC)/DELT
            DELM = RHOPS*(XMT-XMC)/DELT
            CONX = 0.25*D*G*(CPA+CV*HM(N))/DELX
            CONT = 0.0
C
```

```
C        Calculate radiative heat transfer coefficient (depend on temperature)
C
         HRCS=EC*SIGMA*(TS*TS+T(1+4*(N-1))*T(1+4*(N-1)))*(TS+T(1+4*(N-1)))
         HRPC=SIGMA*(T(1+4*(N-1))*T(1+4*(N-1))+T(3+4*(N-1))*T(3+4*(N-1)))
     *(T(1+4*(N-1))+T(3+4*(N-1)))/(1./EC+1./EP-1.)
         HFG = 2610.0
C
C        Calculate the matrix elements
C
C        The matrix elements of Tc
C
         A(1+4*(N-1),1+4*(N-1))= (CONC+HCCF+HRPC+HRCS+HW)
         A(2+4*(N-1),1+4*(N-1))=-(HCCF)
         A(3+4*(N-1),1+4*(N-1))=-(HRPC)
         A(4+4*(N-1),1+4*(N-1))=  0.0
C
C        The matrix element of Tf (normal elements)
C
         A(1+4*(N-1),2+4*(N-1))=-(HCCF)
         A(3+4*(N-1),2+4*(N-1))= (CV*DELM-HCPF)
         A(4+4*(N-1),2+4*(N-1))=-(HCBF)
C
C        The matrix element of Tp
C
         A(1+4*(N-1),3+4*(N-1))=-(HRPC)
         A(2+4*(N-1),3+4*(N-1))= (CV*DELM-HCPF)
         A(3+4*(N-1),3+4*(N-1))= (CONP+(CV-CW)*DELM+HRPC+HCPF)
         A(4+4*(N-1),3+4*(N-1))=  0.0
C
C        The matrix element of Tb
C
         A(1+4*(N-1),4+4*(N-1))=  0.0
         A(2+4*(N-1),4+4*(N-1))= -(HCBF)
         A(3+4*(N-1),4+4*(N-1))=  0.0
         A(4+4*(N-1),4+4*(N-1))=  (CONB+HCBF+UB)
```

```
C
C      Right hand side vector B, except for the term from Tf eq.
C
       B(1+4*(N-1))=CONC*T(1+4*(N-1))+HW*TA+HRCS*TS+QSC
       B(3+4*(N-1))=CONP*T(3+4*(N-1))-HFG*DELM+QSP
       B(4+4*(N-1))=CONB*T(4+4*(N-1))+UB*TA
C
C      Matrix elements of Tfx-dx,Tf,Tfx+dx and B
C
       IF(N.EQ.1) THEN
            A(2+4*(N-1),6+4*(N-1))=CONX
            A(2+4*(N-1),2+4*(N-1))=HCPF+HCCF+HCBF+CONT-DELM*CV
            B(2+4*(N-1))=CONX*(TDI-T(6+4*(N-1))+TDI)+CONT*T(2+4*(N-1))
       ELSE

       IF(N.GT.1.AND.N.LT.NSEC) THEN
            A(2+4*(N-1),4*(N-1)-2)=-CONX
            A(2+4*(N-1),6+4*(N-1))= CONX
            B(2+4*(N-1))=CONX*(T(4*(N-1)-2)-T(6+4*(N-1)))+CONT*T(2+4*(N-1))
            A(2+4*(N-1),2+4*(N-1))= HCPF+HCCF+HCBF+CONT-(DELM*CV)
       ELSE
       IF(N.EQ.NSEC) THEN
            A(2+4*(N-1),2+4*(N-1))= 2.*CONX+HCPF+HCPF+HCBF+CONT-DELM*CV
            A(2+4*(N-1),4*(N-1)-2)=-2.*CONX
            B(2+4*(N-1))=-2.*CONX*(T(2+4*(N-1))-T(4*(N-1)-2))+CONT*T(2+4*(N-1))
       ELSE
       WRITE(*,30)
    30 FORMAT(' Some calculation of matrix elements are missing')
       ENDIF
       ENDIF
       ENDIF
    90 CONTINUE
C
C      End of the computation of the matrix element
C
```

```
C           Use a subroutine to solve the matrix equation [A].T=B
C           where NT in the argument, original is N
C
            CALL SGEFS(A,LDA,NT,B,ITASK,IND,WORK,IWORK,RCOND)
C
C            Transfer the solution of matrix equation to the temperature
C           values for use to calculate the heat transfer coeff. for the next time step.
            DO 110 I=1,NT
            T(I)=B(I)
        110 CONTINUE
            TDO   = T(2+4*(NSEC-1))
            RETURN
            END SUBROUTINE
```

4. โปรแกรมสำหรับคำนวณค่าความชื้นของผลิตภัณฑ์จากสมการอบแห้งชั้นบาง

```
            SUBROUTINE LAYER(XMT,XMC,TFC,RHP,DELT,XMO)
            C    PARAMETERS USED :
            C      TP.....Temperatue of bananas ,K
            C      XMC....Actual moisture content of bananas , kg/kg,db
            C      XMO....Initial moisture content of bananas, kg/kg,db
            C      XME....Equilibrium moisture content of bananas, kg/kg,db
            C      TI.....Equivalent time,time at which the thin layer equation
            C            gives the value of bananas moistue to be equal to that
            C            calculate using actual value of RH,T and XMR , min
            C      DELT...Time step of for calculation of the moisture content, s
            C      DELM...Difference between the initial and equilibrium moisture
            C            content, kg/kg, db
            C      RH.....Relative humidity value transfered from main program ,decimal
            C      Compute equilibrium moisture content (RH and TH come from both t
            C       t+delt
            C        write(*,12)
            C   12  format(' Beginning of the LAYER ')
            C
                  AW  = RHP/100.
                  EMC = 74.66023-(1.144253*TFC)+(37.07224*AW)
```

```
 1        +(0.001166*TFC**2.)+(51.55674*AW**2.)
      XME = EMC/100.0
C           write(*,*) EMC
C
C      Set initial moisture content for thin layer calculation.
C
      IF ((XMO-XMC).LT.0.) THEN
      DELM = XMC-XME
      ELSE
      DELM = XMO-XME
      ENDIF
C           WRITE(*,*) XMC
C
C      Compute moist ratio at time t.
C
        IF(DELM.NE.0.0) THEN
        XMR = (XMC-XME)/DELM
        ELSE
        write(*,16)
  16    format(' XMR becomes infinity')
      ENDIF

C     write(*,17) TFC,RHP,EMC
  17 format(' TF= ',F8.2,' RH= ',F8.2,' EMC= ',f8.3)
C
C             Check desorption or absorption
C
            IF (XMC.GT.XME) THEN
C           Use semiempirical model for continued drying
C
C           A = 1.503574-0.013267*TFC-0.505455*AW +0.000094*TFC**2-2.141736*AW**2
C           B = 0.1814-0.006347*TFC+0.193*AW+0.000081*TFC**2-0.797778*AW**2
            A = 1.503574-0.013267*TFC-0.505455*AW +0.000094*TFC**2
            B = 0.1814-0.006347*TFC+0.193*AW+0.000081*TFC**2
```

```
C         A = 1.503574-0.013267*TFC-0.505455*AW +0.000094*TFC**2-2.141736*AW**2
          B = 0.1814-0.006347*TFC+0.193*AW+0.000081*TFC**2-0.797778*AW**2
      IF(XMR.GT.0.0.AND.B.GT.0.0) THEN
      TI=(-ALOG(XMR/A)/B)+DELT/3600.
      ELSE
      ENDIF
C         WRITE(*,*) A,B
          XMT=DELM*A*EXP(-B*TI)+XME
C
C   Assume water absorption of product isn't occured
C           although Me is great than M
C
          ELSE
          XMT=XMC
      ENDIF
C
C    write(*,20) XMC,XME,XMT,XMO
C  20 format('XMC =',F10.4,' XME =',F10.4,' XMT =',F10.4,' XMO =',F10.4)

      END
```


5. โปรแกรมสำหรับคำนวณค่าอัตราการไหลเชิงมวลของอากาศภายในเครื่องอบแห้ง

```
      FUNCTION MFLOW(TA,IT)
      REAL IPH,IPHR,TPV,USC,ITR,IT,N,NR,FLOWR,MFLOW
      REAL A,B,C,D,E,X,ISC,IC,IO,KB,KM,K,VC,VM,IM,IS,VS,RS,RP,TORQE
C
C     Important parameters using in this program
C
C  ALPHA...Absorption coefficient of solar cell, [decimal]
C  C....Constant of silicon material, [A/K3]
C  EFFR....Reference efficiency of solar cell at reference temperature, [decimal]
C  EGO...Energy gap of silicon at 0 K, [eV]
C  IC...Solar cell current, [A]
C  IM...PV Module current,[A]
C  IO...Reverse saturation current of solar cell, [A]
```

```
C  IPH...Photo current of solar cell, [A]
C  IPHR...Reference photo current of solar cell at reference irradiance, [A]
C  ITR...Reference irradiance, [W/m2]
C  K...Constant of fan, [Nm/rpm2]
C  KB...Constant of DC motor, [Nm/A]
C  MFLOW...Mass flow rate obtained from the PV ventilated system, [kg/s]
C  MFLOWR...Reference mass flow rate obtained from the PV ventilated system, [kg/s]
C  N...Rotational speed of blade of fan, [rpm]
C  NP...Number of parallel cell in the module
C  NR...Reference rotational speed of blade of fan, [rpm]
C  NS...Number of series cell in the module
C  RP...Series resistance of solar cell, [omh]
C  RS...Parallel resistance of solar cell, [omh]
C  TAU...Transmittance coefficient of solar cell, [decimal]
C  TORQUE...Torque of blade of fan, [Nm]
C  TPV...Temperature of  PV cell, [K]
C  TPVR...Reference temperature of PV cell, [K]
C  UEFF...Efficient decreasing coefficient of PV cell, [decimal/K]
C  ULPV...Total heat loss coefficient of the module, [W/m2]
C  USC...Short circuit current increasing coefficient of PV cell, [A/K]
C  VC...Solar cell voltage, [V]
C  VM...PV Module voltage,[V]
C
C        PV part
C        First, calculate the temperature of photovoltaic cell depended on IT and TA
C
         ULPV  = 28.8
         TAU   = 0.8
         ALPHA = 0.8
         EFFR  = 0.10
         TPVR  = 298.15
         UEFF  = 0.000125
         TPV   = (TA*ULPV+IT*(TAU*ALPHA-EFFR-UEFF*TPVR))/(ULPV-UEFF*IT)
C
C        Calculate photo current while weather condition at IT and TV
```

```
C
      ITR   = 1000.0
      USC   = 0.00035
      IPHR  = 0.818
      IPH   = (IT/ITR)*(IPHR+USC*(TPV-TPVR))
C
C     Calculate saturation dark current only depended on TPV
C
      C   = 5.0E3
      EGO   = 1.16*1.61E-19
      KB    = 1.38E-23
      IO    = C*(TPV)**3*EXP (-EGO/(KB*TPV))
C
C     Motor part
C     find the I-V characteristic curves of PV panel and fan
C
      NS = 40.0
      RS = 0.012
      RP = 140.0
      KB = 1.38E-23
      KM = 0.0432
      MN = -113601
      E  = 1.61E-19
      K  = 6E-8
      T = TPV
      ISC = IPH+IO
      DO 1  I = 1,1000
C
C     We assume that ISC=IPH
C
       IC = ISC-0.001*ISC*I
       A  = LOG((ISC-IC)/IO)
       VC = (KB*T/E)*A*NS
      C  = (KM*IC/K)**(0.5)*(3.14*KM/30.0)
      D  = (KM*IC*MN)*(3.14*KM/30.0)
```

```
      VM = (C-D)
       X  = VC-VM
              IF (ABS(X).LT.1.0E-3) THEN
                     IM=IC
                     VM=VC
                     VS = VC/40.
                     IS = ISC-IO*EXP(E*(VS+IM*RS)/(KB*T))-(VS-IM*RS)/RP
                     VM = ((KM*IC/K)**(0.5)-(KM*IC*MN))*(3.14*KM/30.0)
                     VS = VM/40.
                     IC = ISC-IO*EXP(E*(VS+IM*RS)/(KB*T))-(VS-IM*RS)/RP
                     VM = ((KM*IC/K)**(0.5)-(KM*IC*MN))*(3.14*KM/30.0)
                     GOTO 10
              ELSE
                     VM = ((KM*ISC/K)**(0.5)-(KM*ISC*MN))*(3.14*KM/30.0)
              END IF
    1 CONTINUE
C
C     Blade part
C     Calculate angular velocity of blade to determine mass flow rate of blower
C
   10 PI = 3.141593
      FLOWR = 480.0/3600.
      NR = 4000.0
      IM = IC
      TORQE = KM*IM
      NO = 30.0*VM/PI/KM
      N = NO+MN*TORQE/VM
      MFLOW = N*FLOWR/NR
      END FUNCTION
```

6. โปรกรมสำหรับคำนวณค่าสัมประสิทธิ์การพาความร้อนแบบบังคับ

```
      FUNCTION HCF(V)
C
C     Calculate characteristic diameter, DE
C
      W=1.15
      D=0.12
      DE = 4*W*D/(2*(W+D))
C
C     Calculate Reynolds number, RE
C
      DENS = 1.05
      VISC = 1.88E-05
       RE = DE*V*DENS/VISC
C
C     Calculate Nusselt number, NU
C
      L=6.0
      NU=0.0158*(Re**0.8)*(1+(De/L)**0.7)
C
C     Calculate the force convective heat transfer coefficient, HCF
C
      K = 0.029
      HCF = NU*K/DE
      RETURN
      END FUNCTION
```


7. โปรแกรมสำหรับคำนวณค่าสมบัติต่างๆ ของอากาศชื้น

```
FUNCTION RHDBHA(DB,HA)
PATM = 101.325E03
PS = PSDB(DB)
PV = PVHA(HA)
IF(PV.LT.PS) THEN
RHDBHA = PV/PS
ELSE
```

```
      RHDBHA=.99
      ENDIF
      RETURN
      END
C
C     PURPOSE : To compute vapor pressure from humidity ratio of moist air
C     PVHA...Vapor pressure, Pa
C     HA.....Humidity ratio, kg vapor/kg dry air
C     PATM...Atmospheric pressure, Pa
C
      FUNCTION PVHA(HA)
      PATM = 101.325E03
      PVHA = HA*PATM/(0.6219+HA)
      RETURN
      END
C     PURPOSE : To compute saturated vapor pressure from dry bulk temperature
C     PSDB...Saturate vapor pressure, Pa
C     DB.....Dry bulk temperature, K
C
      FUNCTION PSDB(DB)
      R =  0.3206182232E04
      A = -0.274055258361426E05
      B =  0.541896076328951E02
      C = -0.451370384112655E-1
      D =  0.215321191636354E-4
      E = -0.462026656819982E-8
      F =  0.2416127209874E01
      G =  0.121546516706055E-2
      Q =  6.894757E03
      IF ((DB-273.16).LT.0.) THEN
      IF(DB.GT.0.0) THEN
      PSDB = Q*(EXP(23.3924-11286.6489/(1.8*DB)-0.46057*ALOG(1.8*DB)))
      ELSE
      write(*,10)
   10 format(' Argument of LOG ,DB, in PSDB is zero or negative' )
```

```
      ENDIF
      ELSE
      PSDB = Q*(R*EXP((A+1.8*DB*(B+1.8*DB*(C+1.8*DB*
     &      (D+1.8*DB*E))))/(1.8*DB*(F-G*1.8*DB))))
      ENDIF
      RETURN
      END
C  ------------------------------------------------------
C     PURPOSE : To compute specific volume of moist air
C     VSDBHA...Specific volume of moist air, m3 moist air/kg dry air
C     DB.......Dry bulk temperature, K
C     HA.......Humidity ratio, kg vapor/kg dry air
C     PATM.....Atmospheric pressure, Pa
C
      FUNCTION VSDBHA (DB,HA)
      PATM = 101.325E03
      VSDBHA =.06243386*(53.35*1.8*DB*(0.6219+HA)/144.0/0.6219/
     &       (PATM/6.894575E03))
      RETURN
      END
C  ------------------------------------------------------
C     PURPOSE : To compute humidity ratio from dry bulk temperature
C              and relative humidity
C     FUNCTIONS USED : 1) PSDB
C                      2) HAPV
C     HADBRH..Humidity ratio, kg vapor/kg dry air
C     PSDB..Saturated vapor pressure, Pa
C     RH...Relative humidity, decimal
C     DB...Dry bulk temperture,K
C     PV...Vapor pressure, Pa
C
      FUNCTION HADBRH(DB,RH)
      PV = RH*PSDB(DB)
      HADBRH = HAPV(PV)
      RETURN
```

```
      END
C
C     PURPOSE : To compute humidity ratio from vapor pressure
C     HAPV...Humidity ratio, kg vapor/kg dry air
C     PV.....Vapor pressure, Pa
C     ATM....Atmospheric pressure, Pa
C
      FUNCTION HAPV(PV)
      PATM = 101.325E03
      HAPV = 0.6219*PV/(PATM-PV)
      RETURN
      END
C
C   PSDB(DB) is aldeady given previosly.
C  -------------------------------------------------------
C     PROPOSE: To compute wet bulk temperature of moist air from
C           dry bulk temperature and humidity ratio.
C     NOTE :  The value of the wet bulk temperature need to be
C           computed by using iterative method. The geuss value
C           of wet bulk temperature G1 and G2 are choosed to
C           be G1=DB and G2=DB-35.
C     FUNCTION USED : 1) PSDB(DB)
C                 2) PVHA(HA)
C                 3) HLDB(DB)
C                 4) WBL(TWB), objective function which the zero
C                   value is searched.
C                 5) ZEROIN , subroutine used for searching the value
C                   of independent  valiable which make the objective
C                   function become zero.
C     PARAMETERS :
C      G1..Beginning value of the guess interval in which wet bulk temperature
C         is probably exists, K
C      G2..End value of the guess inteval in which wet bulk temperatue is
C         probably exists, K  (G1=DB, G2=DB-40)
C      EPS Aceptable error range in iterative process
```

```
C       DB..Dry bulk temperatur, K
C       HA..Humidity ratio, kg vapor/kg dry air
C       PV..Vapor pressure, Pa
C       A,B,TB .. Intermidiate parameters
C       WBL..Objective function which zero value is searched. WBL is
C           is written as FUNCTION  its value is transfered to the
C           subroutine ZEROIN by  the subroutine argument.
C           EXTERNAL statement must de used for declaring this process.
C
      FUNCTION WBDBHA(DB,HA,G1,G2,EPS)
      EXTERNAL WBL
      COMMON /SPEC/PV,TB,XTRA
C
      A = G1
      B = G2
      TB = DB
      PV = PVHA(HA)
      CALL ZEROIN(A,B,EPS,WBL)
      WBDBHA = (A+B)/2.0
      RETURN
      END
C
C      PURPOSE : copute the value of the objective function, WBL.
C      PARAMETERS :
C       WBL..Objective function, K
C       TWB..Wet bulk temperature, K
C       PV...Vapor pressure. Pa
C       DB...Dry bulk temperature, K
C       PATM.. Amospheric pressure,Pa
C       PWB...Saturated vapor pressure, Pa
C
      FUNCTION WBL(TWB)
      COMMON /SPEC/PV,DB,XTRA
      PATM = 101.325E03
      PWB = PSDB(TWB)
```

```
      WBL = TWB-DB-((PWB-PV)/(0.2405*(PWB-PATM)*
     &     (1.0+0.15577*PV/PATM))*
     &     0.62194*HLDB(TWB)/2325.8377)/1.8
      RETURN
      END
C
C     PURPOSE : To copute latent heat of evaporization, J/kg
C     PARAMETERS:
C       HLDB.. Latent heat of evaporization, J/kg
C       DB.....Dry bulk temperature, K
C       R......Convertion factor for unit change
C
      FUNCTION HLDB(DB)
C
      R=2325.8377
      IF ((DB-273.16).LT.0.) THEN
      HLDB =R*(1220.884-0.05077*(1.8*DB-459.69))
      ELSE
      IF ((DB-338.72).LT.0.) THEN
      HLDB = R*(1075.8965-0.569835*(1.8*DB-459.69))
      ELSE
C     Original value is 0.9125275587  ,
      A=1354673.214-0.09125275587*1.8*DB*1.8*DB
      IF(A.GE.0.0) THEN
      HLDB = R*(SQRT(A))
      ELSE
      write(*,9) DB
    9 format(' DB= ',E15.7,/)
C
      write(*,10)
   10 format(' Argument of SQRT in HLDB is negative')
      ENDIF
      ENDIF
      ENDIF
      RETURN
```

```
      END
C
C   PVHA(HA) and PSBD(DB) are given previously.
C
C    PURPOSE : To search the  smallest interval (A,B) in which the
C             value of FUNC become nearly zero or zero. This
C             subroutine is used for searching the value of
C             the wet bulk temperature of the function WBL.
C
      SUBROUTINE ZEROIN(A,B,EPS,FUNC)
      EXTERNAL FUNC
      REAL I,M
      FA = FUNC(A)
      FB = FUNC(B)
      FC = FA
      C = A
      IF (SIGN(1.0,FB).NE.SIGN(1.0,FC)) GO TO 1
    1 IF (ABS(FC)-ABS(FB)) 2,3,3
    2 C=B
      B=A
      A=C
      FC=FB
      FB=FA
      FA=FC
    3 IF (ABS(C-B)-2.0*EPS) 12,12,4
C
C    Replace LEGVAR statement, the statement used for preventing
C    zero division of CDC 6500 computer, with IF THEN statement.
C
    4 IF((FB-FA).GT.0.0) THEN
      I=(B-A)*FB/(FB-FA)
      J=0
      ELSE
      J=1
      ENDIF
```

```
      M=(C+B)/2.
      IF(J-0) 7,5,7
    5 I = -I+B
      CHINT = (B-I)*(M-I)
      IF (CHINT) 8,8,7
    7 I = M
    8 IF (ABS(B-I)-EPS) 9,10,10
    9 I = SIGN(1.0,(C-B))*EPS+B
   10 A = B
      B = I
      FA = FB
      FB = FUNC(B)
      IF (SIGN(1.0,FB)-SIGN(1.0,FC)) 1,11,1
   11 C = A
      FC = FA
      GO TO 1
   12 A = (C+B)/2.0
      FA = FUNC(A)
      IF (SIGN(1.0,FA).EQ.SIGN(1.0,FB)) B = C
      RETURN
      END
```


8. โปรแกรมสำหรับแก้สมการของระบบสมการขนาด N x N

```
SUBROUTINE SGEFS(A,LDA,N,V,ITASK,IND,WORK,IWORK,RCOND)
C***DATE WRITTEN   800317   (YYMMDD)
C***REVISION DATE  870916   (YYMMDD)
C***CATEGORY NO.  D2A1
C***KEYWORDS  GENERAL SYSTEM OF LINEAR EQUATIONS,LINEAR EQUATIONS
C***AUTHOR  VOORHEES, E., (LOS ALAMOS NATIONAL LABORATORY)
C***PURPOSE  SGEFS solves a GENERAL single precision real NXN system of linear equations.
C***DESCRIPTION
C    From the book "Numerical Methods and Software"
C       by  D. Kahaner, C. Moler, S. Nash
C          Prentice Hall 1988
C    Subroutine SGEFS solves a general NxN system of single precision linear equations using LINPACK
```

```
C    subroutines SGECO and SGESL.  That is, if A is an NxN real matrix and if X and B are real N-vectors,
C    then SGEFS solves the equation
C                                                  A*X=B.
C    The matrix A is first factored into upper and lower triangular matrices U and L using partial pivoting.  These
C    factors and the pivoting information are used to find the solution vector X.  An approximate condition number is
C    calculated to provide a rough estimate of the number of digits of accuracy in the computed solution.
C    If the equation A*X=B is to be solved for more than one vector B, the factoring of A does not need to be
C    performed again and the option to only solve (ITASK .EQ. 2) will be faster for the succeeding solutions.  In this
C    case, the contents of A,LDA, N and IWORK must not have been altered by the user following factorization
C     (ITASK=1).  IND will not be changed by SGEFS in this case.  Other settings of ITASK are used to solve linear
C    systems involving the transpose of A.
C
C  Argument Description ***
C
C    A      REAL(LDA,N)
C    on entry, the doubly subscripted array with dimension (LDA,N) which contains the coefficient matrix.
C    on return, an upper triangular matrix U and the multipliers necessary to construct a matrix L so  that A=L*U.
C    LDA    INTEGER
C     the leading dimension of the array A. LDA must be greater than or equal to N. (terminal error message IND=-1)
C    N      INTEGER
C    the order of the matrix A.  The first N elements of the array A are the elements of the first column of
C    the  matrix A.  N must be greater than or equal to 1. (terminal error message IND=-2)
C    V      REAL(N)
C     on entry, the singly subscripted array(vector) of dimension N which contains the right hand side B of a
C      system of simultaneous linear equations A*X=B. on return, V contains the solution vector, X .
C    ITASK  INTEGER
C     If ITASK=1, the matrix A is factored and then the linear equation is solved.
C     If ITASK=2, the equation is solved using the existing factored matrix A and IWORK.
C     If ITASK=3, the matrix is factored and A'x=b is solved
C     If ITASK=4, the transposed equation is solved using the existing factored matrix A and IWORK.
C     ITASK .LT. 1 or ITASK .GT. 4, then the terminal error message IND=-3 is printed.
C    IND    INTEGER
C    GT. 0  IND is a rough estimate of the number of digits of accuracy in the solution, X.
C    LT. 0  see error message corresponding to IND below.
C    WORK   REAL(N)
```

```
C    a singly subscripted array of dimension at least N.
C    IWORK  INTEGER(N)
C    a singly subscripted array of dimension at least N.
C    RCOND  REAL
C    estimate of 1.0/cond(A)
C  Error Messages Printed ***
C    IND=-1  fatal   N is greater than LDA.
C    IND=-2  fatal   N is less than 1.
C    IND=-3  fatal   ITASK is less than 1 or greater than 4.
C    IND=-4  fatal   The matrix A is computationally singular.
C                    A solution has not been computed.
C    IND=-10 warning    The solution has no apparent significance.
C                    The solution may be inaccurate or the matrix
C                    A may be poorly scaled.
C***REFERENCES  SUBROUTINE SGEFS WAS DEVELOPED BY GROUP C-3, LOS ALAMOS
C               SCIENTIFIC LABORATORY, LOS ALAMOS, NM 87545.
C               THE LINPACK SUBROUTINES USED BY SGEFS ARE DESCRIBED IN
C               DETAIL IN THE *LINPACK USERS GUIDE* PUBLISHED BY
C               THE SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS
C               (SIAM) DATED 1979.
C***ROUTINES CALLED  R1MACH,SGECO,SGESL,XERROR
C***END PROLOGUE  SGEFS
      INTEGER LDA,N,ITASK,IND,IWORK(*)
      REAL A(LDA,*),V(*),WORK(*),R1MACH
      REAL RCOND
      CHARACTER MSG*54
C***FIRST EXECUTABLE STATEMENT  SGEFS
      IF (LDA.LT.N)  GO TO 101
      IF (N.LE.0)  GO TO 102
      IF (ITASK.LT.1) GO TO 103
      IF (ITASK.GT.4) GO TO 103
      IF (ITASK.EQ.2 .OR. ITASK.GT.3) GO TO 20
C
C     FACTOR MATRIX A INTO LU
      CALL SGECO(A,LDA,N,IWORK,RCOND,WORK)
```

```
C
C    CHECK FOR COMPUTATIONALLY SINGULAR MATRIX
     IF (RCOND.EQ.0.0)  GO TO 104
C
C    COMPUTE IND (ESTIMATE OF NO. OF SIGNIFICANT DIGITS)
     IND=-INT(ALOG10(R1MACH(4)/RCOND))
C
C    CHECK FOR IND GREATER THAN ZERO
     IF (IND.GT.0)  GO TO 20
     IND=-10
     CALL XERROR( 'SGEFS ERROR (IND=-10) -- SOLUTION MAY HAVE NO SIGNIF
    1ICANCE',58,-10,0)
C
C    SOLVE AFTER FACTORING
  20 JOB=0
     IF (ITASK.GT.2) JOB=1
     CALL SGESL(A,LDA,N,IWORK,V,JOB)
     RETURN
C
C    IF LDA.LT.N, IND=-1, FATAL XERROR MESSAGE
 101 IND=-1
     WRITE(MSG, '(
    * "SGEFS ERROR (IND=-1) -- LDA=", I5, " IS LESS THAN N=",
    *    I5     )'  ) LDA, N
     CALL XERROR(MSG(1:54), 54, -1, 0)
     RETURN
C
C    IF N.LT.1, IND=-2, FATAL XERROR MESSAGE
 102 IND=-2
     WRITE(MSG, '(
    * "SGEFS ERROR (IND=-2) -- N=", I5, " IS LESS THAN 1.") ')N
     CALL XERROR(MSG(1:47), 47, -2, 0)
     RETURN
C
C    IF ITASK.LT.1, IND=-3, FATAL XERROR MESSAGE
```

```
 103 IND=-3
     WRITE(MSG, '(
    * "SGEFS ERROR (IND=-3) -- ITASK=", I5, " IS LT 1 OR GT 4.")
    *            ') ITASK
     CALL XERROR(MSG(1:52), 52, -3, 0)
     RETURN
C
C    IF SINGULAR MATRIX, IND=-4, FATAL XERROR MESSAGE
 104 IND=-4
     CALL XERROR( 'SGEFS ERROR (IND=-4) -- SINGULAR MATRIX A - NO SOLUT
    1ION',55,-4,0)
     RETURN
C
     END
     SUBROUTINE SGECO(A,LDA,N,IPVT,RCOND,Z)
C***BEGIN PROLOGUE  SGECO
C    THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C    FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***ROUTINES CALLED  SASUM,SAXPY,SDOT,SGEFA,SSCAL
C***END PROLOGUE  SGECO
     INTEGER LDA,N,IPVT(*)
     REAL A(LDA,*),Z(*)
     REAL RCOND
C
     REAL SDOT,EK,T,WK,WKM
     REAL ANORM,S,SASUM,SM,YNORM
     INTEGER INFO,J,K,KB,KP1,L
C
C    COMPUTE 1-NORM OF A
C
C***FIRST EXECUTABLE STATEMENT  SGECO
     ANORM = 0.0E0
     DO 10 J = 1, N
       ANORM = AMAX1(ANORM,SASUM(N,A(1,J),1))
```

```
   10 CONTINUE
C
C     FACTOR
C
      CALL SGEFA(A,LDA,N,IPVT,INFO)
C
C     RCOND = 1/(NORM(A)*(ESTIMATE OF NORM(INVERSE(A)))) .
C     ESTIMATE = NORM(Z)/NORM(Y) WHERE  A*Z = Y  AND  TRANS(A)*Y = E .
C     TRANS(A)  IS THE TRANSPOSE OF A .  THE COMPONENTS OF  E  ARE
C     CHOSEN TO CAUSE MAXIMUM LOCAL GROWTH IN THE ELEMENTS OF W  WHERE
C     TRANS(U)*W = E .  THE VECTORS ARE FREQUENTLY RESCALED TO AVOID
C     OVERFLOW.
C
C     SOLVE TRANS(U)*W = E
C
      EK = 1.0E0
      DO 20 J = 1, N
        Z(J) = 0.0E0
   20 CONTINUE
      DO 100 K = 1, N
        IF (Z(K) .NE. 0.0E0) EK = SIGN(EK,-Z(K))
        IF (ABS(EK-Z(K)) .LE. ABS(A(K,K))) GO TO 30
          S = ABS(A(K,K))/ABS(EK-Z(K))
          CALL SSCAL(N,S,Z,1)
          EK = S*EK
   30   CONTINUE
        WK = EK - Z(K)
        WKM = -EK - Z(K)
        S = ABS(WK)
        SM = ABS(WKM)
        IF (A(K,K) .EQ. 0.0E0) GO TO 40
          WK = WK/A(K,K)
          WKM = WKM/A(K,K)
        GO TO 50
   40   CONTINUE
```

```
      WK = 1.0E0
      WKM = 1.0E0
 50   CONTINUE
     KP1 = K + 1
     IF (KP1 .GT. N) GO TO 90
      DO 60 J = KP1, N
        SM = SM + ABS(Z(J)+WKM*A(K,J))
        Z(J) = Z(J) + WK*A(K,J)
        S = S + ABS(Z(J))
 60     CONTINUE
      IF (S .GE. SM) GO TO 80
        T = WKM - WK
        WK = WKM
        DO 70 J = KP1, N
          Z(J) = Z(J) + T*A(K,J)
 70       CONTINUE
 80     CONTINUE
 90   CONTINUE
     Z(K) = WK
100 CONTINUE
   S = 1.0E0/SASUM(N,Z,1)
   CALL SSCAL(N,S,Z,1)
C
C    SOLVE TRANS(L)*Y = W
C
   DO 120 KB = 1, N
     K = N + 1 - KB
     IF (K .LT. N) Z(K) = Z(K) + SDOT(N-K,A(K+1,K),1,Z(K+1),1)
     IF (ABS(Z(K)) .LE. 1.0E0) GO TO 110
       S = 1.0E0/ABS(Z(K))
       CALL SSCAL(N,S,Z,1)
110   CONTINUE
     L = IPVT(K)
     T = Z(L)
     Z(L) = Z(K)
```

```
      Z(K) = T
 120 CONTINUE
     S = 1.0E0/SASUM(N,Z,1)
     CALL SSCAL(N,S,Z,1)
C
     YNORM = 1.0E0
C
C    SOLVE L*V = Y
C
     DO 140 K = 1, N
       L = IPVT(K)
       T = Z(L)
       Z(L) = Z(K)
       Z(K) = T
       IF (K .LT. N) CALL SAXPY(N-K,T,A(K+1,K),1,Z(K+1),1)
       IF (ABS(Z(K)) .LE. 1.0E0) GO TO 130
         S = 1.0E0/ABS(Z(K))
         CALL SSCAL(N,S,Z,1)
         YNORM = S*YNORM
 130   CONTINUE
 140 CONTINUE
     S = 1.0E0/SASUM(N,Z,1)
     CALL SSCAL(N,S,Z,1)
     YNORM = S*YNORM
C
C    SOLVE  U*Z = V
C
     DO 160 KB = 1, N
       K = N + 1 - KB
       IF (ABS(Z(K)) .LE. ABS(A(K,K))) GO TO 150
         S = ABS(A(K,K))/ABS(Z(K))
         CALL SSCAL(N,S,Z,1)
         YNORM = S*YNORM
 150   CONTINUE
       IF (A(K,K) .NE. 0.0E0) Z(K) = Z(K)/A(K,K)
```

```
         IF (A(K,K) .EQ. 0.0E0) Z(K) = 1.0E0
         T = -Z(K)
         CALL SAXPY(K-1,T,A(1,K),1,Z(1),1)
  160 CONTINUE
C     MAKE ZNORM = 1.0
      S = 1.0E0/SASUM(N,Z,1)
      CALL SSCAL(N,S,Z,1)
      YNORM = S*YNORM
C
      IF (ANORM .NE. 0.0E0) RCOND = YNORM/ANORM
      IF (ANORM .EQ. 0.0E0) RCOND = 0.0E0
      RETURN
      END
      SUBROUTINE SGEFA(A,LDA,N,IPVT,INFO)
C***BEGIN PROLOGUE  SGEFA
C     THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C     FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C     From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SGEFA
      INTEGER LDA,N,IPVT(*),INFO
      REAL A(LDA,*)
C
      REAL T
      INTEGER ISAMAX,J,K,KP1,L,NM1
C
C     GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING
C
C***FIRST EXECUTABLE STATEMENT  SGEFA
      INFO = 0
      NM1 = N - 1
      IF (NM1 .LT. 1) GO TO 70
      DO 60 K = 1, NM1
         KP1 = K + 1
C
C        FIND L = PIVOT INDEX
```

```
C
      L = ISAMAX(N-K+1,A(K,K),1) + K - 1
      IPVT(K) = L
C
C       ZERO PIVOT IMPLIES THIS COLUMN ALREADY TRIANGULARIZED
C
      IF (A(L,K) .EQ. 0.0E0) GO TO 40
C
C          INTERCHANGE IF NECESSARY
C
         IF (L .EQ. K) GO TO 10
            T = A(L,K)
            A(L,K) = A(K,K)
            A(K,K) = T
   10    CONTINUE
C
C          COMPUTE MULTIPLIERS
C
         T = -1.0E0/A(K,K)
         CALL SSCAL(N-K,T,A(K+1,K),1)
C
C          ROW ELIMINATION WITH COLUMN INDEXING
C
         DO 30 J = KP1, N
            T = A(L,J)
            IF (L .EQ. K) GO TO 20
               A(L,J) = A(K,J)
               A(K,J) = T
   20       CONTINUE
            CALL SAXPY(N-K,T,A(K+1,K),1,A(K+1,J),1)
   30    CONTINUE
      GO TO 50
   40 CONTINUE
         INFO = K
   50 CONTINUE
```

```
  60 CONTINUE
  70 CONTINUE
     IPVT(N) = N
     IF (A(N,N) .EQ. 0.0E0) INFO = N
     RETURN
     END
     SUBROUTINE SGESL(A,LDA,N,IPVT,B,JOB)
C***BEGIN PROLOGUE  SGESL
C    THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C    FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash,  Prentice Hall 1988
C***END PROLOGUE  SGESL
     INTEGER LDA,N,IPVT(*),JOB
     REAL A(LDA,*),B(*)
C
     REAL SDOT,T
     INTEGER K,KB,L,NM1
C***FIRST EXECUTABLE STATEMENT  SGESL
     NM1 = N - 1
     IF (JOB .NE. 0) GO TO 50
C
C       JOB = 0 , SOLVE  A * X = B
C       FIRST SOLVE  L*Y = B
C
        IF (NM1 .LT. 1) GO TO 30
        DO 20 K = 1, NM1
           L = IPVT(K)
           T = B(L)
           IF (L .EQ. K) GO TO 10
              B(L) = B(K)
              B(K) = T
  10       CONTINUE
           CALL SAXPY(N-K,T,A(K+1,K),1,B(K+1),1)
  20    CONTINUE
  30    CONTINUE
```

```
C
C     NOW SOLVE  U*X = Y
C
      DO 40 KB = 1, N
        K = N + 1 - KB
        B(K) = B(K)/A(K,K)
        T = -B(K)
        CALL SAXPY(K-1,T,A(1,K),1,B(1),1)
   40 CONTINUE
      GO TO 100
   50 CONTINUE
C
C     JOB = NONZERO, SOLVE  TRANS(A) * X = B
C     FIRST SOLVE  TRANS(U)*Y = B
C
      DO 60 K = 1, N
        T = SDOT(K-1,A(1,K),1,B(1),1)
        B(K) = (B(K) - T)/A(K,K)
   60 CONTINUE
C
C     NOW SOLVE TRANS(L)*X = Y
C
      IF (NM1 .LT. 1) GO TO 90
      DO 80 KB = 1, NM1
        K = N - KB
        B(K) = B(K) + SDOT(N-K,A(K+1,K),1,B(K+1),1)
        L = IPVT(K)
        IF (L .EQ. K) GO TO 70
          T = B(L)
          B(L) = B(K)
          B(K) = T
   70   CONTINUE
   80 CONTINUE
   90 CONTINUE
  100 CONTINUE
```

```
      RETURN
      END
      INTEGER FUNCTION ISAMAX(N,SX,INCX)
C***BEGIN PROLOGUE  ISAMAX
C    THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C    FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  ISAMAX
C
      REAL SX(*),SMAX,XMAG
C***FIRST EXECUTABLE STATEMENT  ISAMAX
      ISAMAX = 0
      IF(N.LE.0) RETURN
      ISAMAX = 1
      IF(N.LE.1)RETURN
      IF(INCX.EQ.1)GOTO 20
C
C      CODE FOR INCREMENTS NOT EQUAL TO 1.
C
      SMAX = ABS(SX(1))
      NS = N*INCX
      II = 1
        DO 10 I=1,NS,INCX
        XMAG = ABS(SX(I))
        IF(XMAG.LE.SMAX) GO TO 5
        ISAMAX = II
        SMAX = XMAG
    5   II = II + 1
   10   CONTINUE
      RETURN
C
C      CODE FOR INCREMENTS EQUAL TO 1.
C
   20 SMAX = ABS(SX(1))
      DO 30 I = 2,N
```

```
      XMAG = ABS(SX(I))
      IF(XMAG.LE.SMAX) GO TO 30
      ISAMAX = I
      SMAX = XMAG
  30 CONTINUE
     RETURN
     END
     REAL FUNCTION SASUM(N,SX,INCX)
C***BEGIN PROLOGUE  SASUM
C     THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C     FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C     From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SASUM
C
     REAL SX(*)
C***FIRST EXECUTABLE STATEMENT  SASUM
     SASUM = 0.0E0
     IF(N.LE.0)RETURN
     IF(INCX.EQ.1)GOTO 20
C
C       CODE FOR INCREMENTS NOT EQUAL TO 1.
C
     NS = N*INCX
       DO 10 I=1,NS,INCX
       SASUM = SASUM + ABS(SX(I))
  10    CONTINUE
     RETURN
C
C       CODE FOR INCREMENTS EQUAL TO 1.
C
C       CLEAN-UP LOOP SO REMAINING VECTOR LENGTH IS A MULTIPLE OF 6.
C
  20 M = MOD(N,6)
     IF( M .EQ. 0 ) GO TO 40
     DO 30 I = 1,M
```

```
      SASUM = SASUM + ABS(SX(I))
   30 CONTINUE
      IF( N .LT. 6 ) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,6
        SASUM = SASUM + ABS(SX(I)) + ABS(SX(I + 1)) + ABS(SX(I + 2))
     1  + ABS(SX(I + 3)) + ABS(SX(I + 4)) + ABS(SX(I + 5))
   50 CONTINUE
      RETURN
      END
      SUBROUTINE SAXPY(N,SA,SX,INCX,SY,INCY)
C***BEGIN PROLOGUE  SAXPY
C     THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C     FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C     From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SAXPY
C
      REAL SX(*),SY(*),SA
C***FIRST EXECUTABLE STATEMENT  SAXPY
      IF(N.LE.0.OR.SA.EQ.0.E0) RETURN
      IF(INCX.EQ.INCY) IF(INCX-1) 5,20,60
    5 CONTINUE
C
C        CODE FOR NONEQUAL OR NONPOSITIVE INCREMENTS.
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        SY(IY) = SY(IY) + SA*SX(IX)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
```

```
C
C     CODE FOR BOTH INCREMENTS EQUAL TO 1
C
C     CLEAN-UP LOOP SO REMAINING VECTOR LENGTH IS A MULTIPLE OF 4.
C
   20 M = MOD(N,4)
      IF( M .EQ. 0 ) GO TO 40
      DO 30 I = 1,M
        SY(I) = SY(I) + SA*SX(I)
   30 CONTINUE
      IF( N .LT. 4 ) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,4
        SY(I) = SY(I) + SA*SX(I)
        SY(I + 1) = SY(I + 1) + SA*SX(I + 1)
        SY(I + 2) = SY(I + 2) + SA*SX(I + 2)
        SY(I + 3) = SY(I + 3) + SA*SX(I + 3)
   50 CONTINUE
      RETURN
C
C     CODE FOR EQUAL, POSITIVE, NONUNIT INCREMENTS.
C
   60 CONTINUE
      NS = N*INCX
          DO 70 I=1,NS,INCX
          SY(I) = SA*SX(I) + SY(I)
   70     CONTINUE
      RETURN
      END
      SUBROUTINE SCOPY(N,SX,INCX,SY,INCY)
C***BEGIN PROLOGUE  SCOPY
C    THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C    FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SCOPY
```

```
C
      REAL SX(*),SY(*)
C***FIRST EXECUTABLE STATEMENT  SCOPY
      IF(N.LE.0)RETURN
      IF(INCX.EQ.INCY) IF(INCX-1) 5,20,60
    5 CONTINUE
C
C        CODE FOR UNEQUAL OR NONPOSITIVE INCREMENTS.
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        SY(IY) = SX(IX)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
C
C        CODE FOR BOTH INCREMENTS EQUAL TO 1
C
C
C        CLEAN-UP LOOP SO REMAINING VECTOR LENGTH IS A MULTIPLE OF 7.
C
   20 M = MOD(N,7)
      IF( M .EQ. 0 ) GO TO 40
      DO 30 I = 1,M
        SY(I) = SX(I)
   30 CONTINUE
      IF( N .LT. 7 ) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,7
        SY(I) = SX(I)
        SY(I + 1) = SX(I + 1)
```

```
      SY(I + 2) = SX(I + 2)
      SY(I + 3) = SX(I + 3)
      SY(I + 4) = SX(I + 4)
      SY(I + 5) = SX(I + 5)
      SY(I + 6) = SX(I + 6)
   50 CONTINUE
      RETURN
C
C       CODE FOR EQUAL, POSITIVE, NONUNIT INCREMENTS.
C
   60 CONTINUE
      NS = N*INCX
         DO 70 I=1,NS,INCX
         SY(I) = SX(I)
   70    CONTINUE
      RETURN
      END
      REAL FUNCTION SDOT(N,SX,INCX,SY,INCY)
C***BEGIN PROLOGUE  SDOT
C     THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C     FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C     From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SDOT
C
      REAL SX(*),SY(*)
C***FIRST EXECUTABLE STATEMENT  SDOT
      SDOT = 0.0E0
      IF(N.LE.0)RETURN
      IF(INCX.EQ.INCY) IF(INCX-1)5,20,60
    5 CONTINUE
C
C       CODE FOR UNEQUAL INCREMENTS OR NONPOSITIVE INCREMENTS.
C
      IX = 1
      IY = 1
```

```
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        SDOT = SDOT + SX(IX)*SY(IY)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
C
C        CODE FOR BOTH INCREMENTS EQUAL TO 1
C
C        CLEAN-UP LOOP SO REMAINING VECTOR LENGTH IS A MULTIPLE OF 5.
C
   20 M = MOD(N,5)
      IF( M .EQ. 0 ) GO TO 40
      DO 30 I = 1,M
        SDOT = SDOT + SX(I)*SY(I)
   30 CONTINUE
      IF( N .LT. 5 ) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,5
        SDOT = SDOT + SX(I)*SY(I) + SX(I + 1)*SY(I + 1) +
     1   SX(I + 2)*SY(I + 2) + SX(I + 3)*SY(I + 3) + SX(I + 4)*SY(I + 4)
   50 CONTINUE
      RETURN
C
C        CODE FOR POSITIVE EQUAL INCREMENTS .NE.1.
C
   60 CONTINUE
      NS=N*INCX
      DO 70 I=1,NS,INCX
        SDOT = SDOT + SX(I)*SY(I)
   70   CONTINUE
      RETURN
      END
```

```
      REAL FUNCTION SNRM2(N,SX,INCX)
C***BEGIN PROLOGUE  SNRM2
C     THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C     FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C     From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SNRM2
      INTEGER         NEXT
      REAL   SX(*),  CUTLO, CUTHI, HITEST, SUM, XMAX, ZERO, ONE
      DATA   ZERO, ONE /0.0E0, 1.0E0/
C
      DATA CUTLO, CUTHI / 4.441E-16,  1.304E19 /
C***FIRST EXECUTABLE STATEMENT  SNRM2
      IF(N .GT. 0) GO TO 10
         SNRM2  = ZERO
         GO TO 300
C
   10 ASSIGN 30 TO NEXT
      SUM = ZERO
      NN = N * INCX
C                                 BEGIN MAIN LOOP
      I = 1
   20    GO TO NEXT,(30, 50, 70, 110)
   30 IF( ABS(SX(I)) .GT. CUTLO) GO TO 85
      ASSIGN 50 TO NEXT
      XMAX = ZERO
C
C                       PHASE 1.  SUM IS ZERO
C
   50 IF( SX(I) .EQ. ZERO) GO TO 200
      IF( ABS(SX(I)) .GT. CUTLO) GO TO 85
C
C                           PREPARE FOR PHASE 2.
      ASSIGN 70 TO NEXT
      GO TO 105
C
```

```
C                       PREPARE FOR PHASE 4.
C
  100 I = J
      ASSIGN 110 TO NEXT
      SUM = (SUM / SX(I)) / SX(I)
  105 XMAX = ABS(SX(I))
      GO TO 115
C
C              PHASE 2.  SUM IS SMALL.
C                     SCALE TO AVOID DESTRUCTIVE UNDERFLOW.
C
   70 IF( ABS(SX(I)) .GT. CUTLO ) GO TO 75
C
C               COMMON CODE FOR PHASES 2 AND 4.
C               IN PHASE 4 SUM IS LARGE.  SCALE TO AVOID OVERFLOW.
C
  110 IF( ABS(SX(I)) .LE. XMAX ) GO TO 115
        SUM = ONE + SUM * (XMAX / SX(I))**2
        XMAX = ABS(SX(I))
        GO TO 200
C
  115 SUM = SUM + (SX(I)/XMAX)**2
      GO TO 200
C
C             PREPARE FOR PHASE 3.
C
   75 SUM = (SUM * XMAX) * XMAX
C
C    FOR REAL OR D.P. SET HITEST = CUTHI/N
C    FOR COMPLEX      SET HITEST = CUTHI/(2*N)
C
   85 HITEST = CUTHI/FLOAT( N )
C
C              PHASE 3.  SUM IS MID-RANGE.  NO SCALING.
C
```

```
      DO 95 J =I,NN,INCX
      IF(ABS(SX(J)) .GE. HITEST) GO TO 100
   95    SUM = SUM + SX(J)**2
      SNRM2 = SQRT( SUM )
      GO TO 300
C
  200 CONTINUE
      I = I + INCX
      IF ( I .LE. NN ) GO TO 20
C
C            END OF MAIN LOOP.
C
C            COMPUTE SQUARE ROOT AND ADJUST FOR SCALING.
C
      SNRM2 = XMAX * SQRT(SUM)
  300 CONTINUE
      RETURN
      END
      SUBROUTINE SSCAL(N,SA,SX,INCX)
C***BEGIN PROLOGUE  SSCAL
C    THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C    FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash,  Prentice Hall 1988
C***END PROLOGUE  SSCAL
C
      REAL SA,SX(*)
C***FIRST EXECUTABLE STATEMENT  SSCAL
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
C       CODE FOR INCREMENTS NOT EQUAL TO 1.
C
      NS = N*INCX
        DO 10 I = 1,NS,INCX
        SX(I) = SA*SX(I)
```

```
  10    CONTINUE
        RETURN
C
C       CODE FOR INCREMENTS EQUAL TO 1.
C
C       CLEAN-UP LOOP SO REMAINING VECTOR LENGTH IS A MULTIPLE OF 5.
C
  20 M = MOD(N,5)
     IF( M .EQ. 0 ) GO TO 40
     DO 30 I = 1,M
       SX(I) = SA*SX(I)
  30 CONTINUE
     IF( N .LT. 5 ) RETURN
  40 MP1 = M + 1
     DO 50 I = MP1,N,5
       SX(I) = SA*SX(I)
       SX(I + 1) = SA*SX(I + 1)
       SX(I + 2) = SA*SX(I + 2)
       SX(I + 3) = SA*SX(I + 3)
       SX(I + 4) = SA*SX(I + 4)
  50 CONTINUE
     RETURN
     END
     SUBROUTINE SSWAP(N,SX,INCX,SY,INCY)
C***BEGIN PROLOGUE  SSWAP
C    THIS PROLOGUE HAS BEEN REMOVED FOR REASONS OF SPACE
C    FOR A COMPLETE COPY OF THIS ROUTINE CONTACT THE AUTHORS
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash, Prentice Hall 1988
C***END PROLOGUE  SSWAP
C
     REAL SX(*),SY(*),STEMP1,STEMP2,STEMP3
C***FIRST EXECUTABLE STATEMENT  SSWAP
     IF(N.LE.0)RETURN
     IF(INCX.EQ.INCY) IF(INCX-1) 5,20,60
   5 CONTINUE
```

```
C
C     CODE FOR UNEQUAL OR NONPOSITIVE INCREMENTS.
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        STEMP1 = SX(IX)
        SX(IX) = SY(IY)
        SY(IY) = STEMP1
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
C
C     CODE FOR BOTH INCREMENTS EQUAL TO 1
C
C     CLEAN-UP LOOP SO REMAINING VECTOR LENGTH IS A MULTIPLE OF 3.
C
   20 M = MOD(N,3)
      IF( M .EQ. 0 ) GO TO 40
      DO 30 I = 1,M
        STEMP1 = SX(I)
        SX(I) = SY(I)
        SY(I) = STEMP1
   30 CONTINUE
      IF( N .LT. 3 ) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,3
        STEMP1 = SX(I)
        STEMP2 = SX(I+1)
        STEMP3 = SX(I+2)
        SX(I) = SY(I)
        SX(I+1) = SY(I+1)
```

```
      SX(I+2) = SY(I+2)
      SY(I) = STEMP1
      SY(I+1) = STEMP2
      SY(I+2) = STEMP3
   50 CONTINUE
      RETURN
   60 CONTINUE
C
C     CODE FOR EQUAL, POSITIVE, NONUNIT INCREMENTS.
C
      NS = N*INCX
      DO 70 I=1,NS,INCX
      STEMP1 = SX(I)
      SX(I) = SY(I)
      SY(I) = STEMP1
   70 CONTINUE
      RETURN
      END
      REAL FUNCTION R1MACH(I)
C***BEGIN PROLOGUE  R1MACH
C***DATE WRITTEN   790101   (YYMMDD)
C***REVISION DATE  831014   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns single precision machine dependent constants
C***DESCRIPTION
C     From the book, "Numerical Methods and Software" ,  Kahaner, C. Moler, S. Nash,  Prentice Hall, 1988
C
C     R1MACH can be used to obtain machine-dependent parameters
C     for the local machine environment.  It is a function
C     subroutine with one (input) argument, and can be called
C     as follows, for example
```

```
C
C        A = R1MACH(I)
C
C    where I=1,...,5.  The (output) value of A above is
C    determined by the (input) value of I.  The results for
C    various values of I are discussed below.
C
C  Single-Precision Machine Constants
C  R1MACH(1) = B**(EMIN-1), the smallest positive magnitude.
C  R1MACH(2) = B**EMAX*(1 - B**(-T)), the largest magnitude.
C  R1MACH(3) = B**(-T), the smallest relative spacing.
C  R1MACH(4) = B**(1-T), the largest relative spacing.
C  R1MACH(5) = LOG10(B)
C***REFERENCES  FOX, P.A., HALL, A.D., SCHRYER, N.L, *FRAMEWORK FOR
C              A PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHE-
C              MATICAL SOFTWARE, VOL. 4, NO. 2, JUNE 1978,
C              PP. 177-188.
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  R1MACH
C
      INTEGER SMALL(2)
      INTEGER LARGE(2)
      INTEGER RIGHT(2)
      INTEGER DIVER(2)
      INTEGER LOG10(2)
C
      REAL RMACH(5)
C
      EQUIVALENCE (RMACH(1),SMALL(1))
      EQUIVALENCE (RMACH(2),LARGE(1))
      EQUIVALENCE (RMACH(3),RIGHT(1))
      EQUIVALENCE (RMACH(4),DIVER(1))
      EQUIVALENCE (RMACH(5),LOG10(1))
C
C
```

```
C    MACHINE CONSTANTS FOR THE CDC CYBER 170 SERIES (FTN5).
C
C     DATA RMACH(1) / O"00014000000000000000" /
C     DATA RMACH(2) / O"37767777777777777777" /
C     DATA RMACH(3) / O"16404000000000000000" /
C     DATA RMACH(4) / O"16414000000000000000" /
C     DATA RMACH(5) / O"17164642023241175720" /
C
C    MACHINE CONSTANTS FOR THE CDC CYBER 200 SERIES
C
C    DATA RMACH(1) / X'9000400000000000' /
C    DATA RMACH(2) / X'6FFF7FFFFFFFFFFF' /
C    DATA RMACH(3) / X'FFA3400000000000' /
C    DATA RMACH(4) / X'FFA4400000000000' /
C    DATA RMACH(5) / X'FFD04D104D427DE8' /
C
C    MACHINE CONSTANTS FOR THE CDC 6000/7000 SERIES.
C
C    DATA RMACH(1) / 00564000000000000000B /
C    DATA RMACH(2) / 37767777777777777776B /
C    DATA RMACH(3) / 16414000000000000000B /
C    DATA RMACH(4) / 16424000000000000000B /
C    DATA RMACH(5) / 17164642023241175720B /
C
C    MACHINE CONSTANTS FOR THE CRAY 1
C
C    DATA RMACH(1) / 200034000000000000000B /
C    DATA RMACH(2) / 577767777777777777776B /
C    DATA RMACH(3) / 377224000000000000000B /
C    DATA RMACH(4) / 377234000000000000000B /
C    DATA RMACH(5) / 377774642023241175720B /
C
C
C    MACHINE CONSTANTS FOR THE IBM 360/370 SERIES,
C    THE XEROX SIGMA 5/7/9, THE SEL SYSTEMS 85/86  AND
```

```
C    THE PERKIN ELMER (INTERDATA) 7/32.
C
C    DATA RMACH(1) / Z00100000 /
C    DATA RMACH(2) / Z7FFFFFFF /
C    DATA RMACH(3) / Z3B100000 /
C    DATA RMACH(4) / Z3C100000 /
C    DATA RMACH(5) / Z41134413 /
C
C    MACHINE CONSTANTS FOR THE IBM PC FAMILY (D. KAHANER NBS)
C
     DATA RMACH/1.18E-38,3.40E+38,0.595E-07,1.19E-07,0.30102999566/
C
C    MACHINE CONSTANTS FOR THE PDP-10 (KA OR KI PROCESSOR).
C
C    DATA RMACH(1) / "000400000000 /
C    DATA RMACH(2) / "377777777777 /
C    DATA RMACH(3) / "146400000000 /
C    DATA RMACH(4) / "147400000000 /
C    DATA RMACH(5) / "177464202324 /
C
C    MACHINE CONSTANTS FOR THE SUN-3 (INCLUDES THOSE WITH 68881 CHIP,
C      OR WITH FPA BOARD. ALSO INCLUDES SUN-2 WITH SKY BOARD. MAY ALSO
C      WORK WITH SOFTWARE FLOATING POINT ON EITHER SYSTEM.)
C
C    DATA SMALL(1) / X'00800000' /
C    DATA LARGE(1) / X'7F7FFFFF' /
C    DATA RIGHT(1) / X'33800000' /
C    DATA DIVER(1) / X'34000000' /
C    DATA LOG10(1) / X'3E9A209B' /
C
C    MACHINE CONSTANTS FOR THE VAX 11/780
C   (EXPRESSED IN INTEGER AND HEXADECIMAL)
C  *** THE INTEGER FORMAT SHOULD BE OK FOR UNIX SYSTEMS***
C
C    DATA SMALL(1) /       128 /
```

```
C    DATA LARGE(1) /    -32769 /
C    DATA RIGHT(1) /     13440 /
C    DATA DIVER(1) /     13568 /
C    DATA LOG10(1) / 547045274 /
C
C  ***THE HEX FORMAT BELOW MAY NOT BE SUITABLE FOR UNIX SYSTEMS***
C    DATA SMALL(1) / Z00000080 /
C    DATA LARGE(1) / ZFFFF7FFF /
C    DATA RIGHT(1) / Z00003480 /
C    DATA DIVER(1) / Z00003500 /
C    DATA LOG10(1) / Z209B3F9A /
C
C***FIRST EXECUTABLE STATEMENT  R1MACH
      IF (I .LT. 1  .OR.  I .GT. 5)
     1   CALL XERROR ( 'R1MACH -- I OUT OF BOUNDS',25,1,2)
C
      R1MACH = RMACH(I)
      RETURN
C
      END
      DOUBLE PRECISION FUNCTION D1MACH(I)
C***BEGIN PROLOGUE  D1MACH
C***DATE WRITTEN   750101   (YYMMDD)
C***REVISION DATE  831014   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns double precision machine dependent constants
C***DESCRIPTION
C    From the book, "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash,  Prentice Hall, 1988
C
C    D1MACH can be used to obtain machine-dependent parameters
C    for the local machine environment.  It is a function
```

```
C    subprogram with one (input) argument, and can be called
C    as follows, for example
C
C        D = D1MACH(I)
C
C    where I=1,...,5.  The (output) value of D above is
C    determined by the (input) value of I.  The results for
C    various values of I are discussed below.
C
C  Double-precision machine constants
C  D1MACH( 1) = B**(EMIN-1), the smallest positive magnitude.
C  D1MACH( 2) = B**EMAX*(1 - B**(-T)), the largest magnitude.
C  D1MACH( 3) = B**(-T), the smallest relative spacing.
C  D1MACH( 4) = B**(1-T), the largest relative spacing.
C  D1MACH( 5) = LOG10(B)
C***REFERENCES  FOX P.A., HALL A.D., SCHRYER N.L.,*FRAMEWORK FOR A
C               PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHEMATICAL
C               SOFTWARE, VOL. 4, NO. 2, JUNE 1978, PP. 177-188.
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  D1MACH
C
      INTEGER SMALL(4)
      INTEGER LARGE(4)
      INTEGER RIGHT(4)
      INTEGER DIVER(4)
      INTEGER LOG10(4)
C
      DOUBLE PRECISION DMACH(5)
C
      EQUIVALENCE (DMACH(1),SMALL(1))
      EQUIVALENCE (DMACH(2),LARGE(1))
      EQUIVALENCE (DMACH(3),RIGHT(1))
      EQUIVALENCE (DMACH(4),DIVER(1))
      EQUIVALENCE (DMACH(5),LOG10(1))
C
```

```
C     MACHINE CONSTANTS FOR THE CDC CYBER 170 SERIES (FTN5).
C
C      DATA SMALL(1) / O"00604000000000000000" /
C      DATA SMALL(2) / O"00000000000000000000" /
C
C      DATA LARGE(1) / O"37767777777777777777" /
C      DATA LARGE(2) / O"37167777777777777777" /
C
C      DATA RIGHT(1) / O"15604000000000000000" /
C      DATA RIGHT(2) / O"15000000000000000000" /
C
C      DATA DIVER(1) / O"15614000000000000000" /
C      DATA DIVER(2) / O"15010000000000000000" /
C
C      DATA LOG10(1) / O"17164642023241175717" /
C      DATA LOG10(2) / O"16367571421742254654" /
C
C     MACHINE CONSTANTS FOR THE CDC CYBER 200 SERIES
C
C     DATA SMALL(1) / X'9000400000000000' /
C     DATA SMALL(2) / X'8FD1000000000000' /
C
C     DATA LARGE(1) / X'6FFF7FFFFFFFFFFF' /
C     DATA LARGE(2) / X'6FD07FFFFFFFFFFF' /
C
C     DATA RIGHT(1) / X'FF74400000000000' /
C     DATA RIGHT(2) / X'FF45000000000000' /
C
C     DATA DIVER(1) / X'FF75400000000000' /
C     DATA DIVER(2) / X'FF46000000000000' /
C
C     DATA LOG10(1) / X'FFD04D104D427DE7' /
C     DATA LOG10(2) / X'FFA17DE623E2566A' /
C
C
```

```
C    MACHINE CONSTANTS FOR THE CDC 6000/7000 SERIES.
C
C    DATA SMALL(1) / 00564000000000000000B /
C    DATA SMALL(2) / 00000000000000000000B /
C
C    DATA LARGE(1) / 37757777777777777777B /
C    DATA LARGE(2) / 37157777777777777777B /
C
C    DATA RIGHT(1) / 15624000000000000000B /
C    DATA RIGHT(2) / 00000000000000000000B /
C
C    DATA DIVER(1) / 15634000000000000000B /
C    DATA DIVER(2) / 00000000000000000000B /
C
C    DATA LOG10(1) / 17164642023241175717B /
C    DATA LOG10(2) / 16367571421742254654B /
C
C    MACHINE CONSTANTS FOR THE CRAY 1
C
C    DATA SMALL(1) / 201354000000000000000B /
C    DATA SMALL(2) / 000000000000000000000B /
C
C    DATA LARGE(1) / 577767777777777777777B /
C    DATA LARGE(2) / 000007777777777777774B /
C
C    DATA RIGHT(1) / 376434000000000000000B /
C    DATA RIGHT(2) / 000000000000000000000B /
C
C    DATA DIVER(1) / 376444000000000000000B /
C    DATA DIVER(2) / 000000000000000000000B /
C
C    DATA LOG10(1) / 377774642023241175717B /
C    DATA LOG10(2) / 000007571421742254654B /
C
C    MACHINE CONSTANTS FOR THE IBM 360/370 SERIES,
```

```
C    THE XEROX SIGMA 5/7/9, THE SEL SYSTEMS 85/86, AND
C    THE PERKIN ELMER (INTERDATA) 7/32.
C
C    DATA SMALL(1),SMALL(2) / Z00100000, Z00000000 /
C    DATA LARGE(1),LARGE(2) / Z7FFFFFFF, ZFFFFFFFF /
C    DATA RIGHT(1),RIGHT(2) / Z33100000, Z00000000 /
C    DATA DIVER(1),DIVER(2) / Z34100000, Z00000000 /
C    DATA LOG10(1),LOG10(2) / Z41134413, Z509F79FF /
C
C    MACHINE CONSTATNS FOR THE IBM PC FAMILY (D. KAHANER NBS)
C
      DATA DMACH/2.23D-308,1.79D+308,1.11D-16,2.22D-16,
     *  0.30102999566398119 5D0/
C
C    MACHINE CONSTANTS FOR THE PDP-10 (KA PROCESSOR).
C
C    DATA SMALL(1),SMALL(2) / "033400000000, "000000000000 /
C    DATA LARGE(1),LARGE(2) / "377777777777, "344777777777 /
C    DATA RIGHT(1),RIGHT(2) / "113400000000, "000000000000 /
C    DATA DIVER(1),DIVER(2) / "114400000000, "000000000000 /
C    DATA LOG10(1),LOG10(2) / "177464202324, "144117571776 /
C
C    MACHINE CONSTANTS FOR THE PDP-10 (KI PROCESSOR).
C
C    DATA SMALL(1),SMALL(2) / "000400000000, "000000000000 /
C    DATA LARGE(1),LARGE(2) / "377777777777, "377777777777 /
C    DATA RIGHT(1),RIGHT(2) / "103400000000, "000000000000 /
C    DATA DIVER(1),DIVER(2) / "104400000000, "000000000000 /
C    DATA LOG10(1),LOG10(2) / "177464202324, "476747767461 /
C
C    MACHINE CONSTANTS FOR THE SUN-3 (INCLUDES THOSE WITH 68881 CHIP,
C     OR WITH FPA BOARD. ALSO INCLUDES SUN-2 WITH SKY BOARD. MAY ALSO
C     WORK WITH SOFTWARE FLOATING POINT ON EITHER SYSTEM.)
C
C     DATA SMALL(1),SMALL(2) / X'00100000', X'00000000' /
```

```
C     DATA LARGE(1),LARGE(2) / X'7FEFFFFF', X'FFFFFFFF' /
C     DATA RIGHT(1),RIGHT(2) / X'3CA00000', X'00000000' /
C     DATA DIVER(1),DIVER(2) / X'3CB00000', X'00000000' /
C     DATA LOG10(1),LOG10(2) / X'3FD34413', X'509F79FF' /
C
C    MACHINE CONSTANTS FOR VAX 11/780
C    (EXPRESSED IN INTEGER AND HEXADECIMAL)
C   *** THE INTEGER FORMAT SHOULD BE OK FOR UNIX SYSTEMS***
C
C    DATA SMALL(1), SMALL(2) /        128,           0 /
C    DATA LARGE(1), LARGE(2) /     -32769,          -1 /
C    DATA RIGHT(1), RIGHT(2) /       9344,           0 /
C    DATA DIVER(1), DIVER(2) /       9472,           0 /
C    DATA LOG10(1), LOG10(2) /  546979738,  -805796613 /
C
C   ***THE HEX FORMAT BELOW MAY NOT BE SUITABLE FOR UNIX SYSYEMS***
C    DATA SMALL(1), SMALL(2) / Z00000080, Z00000000 /
C    DATA LARGE(1), LARGE(2) / ZFFFF7FFF, ZFFFFFFFF /
C    DATA RIGHT(1), RIGHT(2) / Z00002480, Z00000000 /
C    DATA DIVER(1), DIVER(2) / Z00002500, Z00000000 /
C    DATA LOG10(1), LOG10(2) / Z209A3F9A, ZCFF884FB /
C
C   MACHINE CONSTANTS FOR VAX 11/780 (G-FLOATING)
C    (EXPRESSED IN INTEGER AND HEXADECIMAL)
C   *** THE INTEGER FORMAT SHOULD BE OK FOR UNIX SYSTEMS***
C
C    DATA SMALL(1), SMALL(2) /         16,           0 /
C    DATA LARGE(1), LARGE(2) /     -32769,          -1 /
C    DATA RIGHT(1), RIGHT(2) /      15552,           0 /
C    DATA DIVER(1), DIVER(2) /      15568,           0 /
C    DATA LOG10(1), LOG10(2) / 1142112243, 2046775455 /
C
C   ***THE HEX FORMAT BELOW MAY NOT BE SUITABLE FOR UNIX SYSYEMS***
C    DATA SMALL(1), SMALL(2) / Z00000010, Z00000000 /
C    DATA LARGE(1), LARGE(2) / ZFFFF7FFF, ZFFFFFFFF /
```

```
C    DATA RIGHT(1), RIGHT(2) / Z00003CC0, Z00000000 /
C    DATA DIVER(1), DIVER(2) / Z00003CD0, Z00000000 /
C    DATA LOG10(1), LOG10(2) / Z44133FF3, Z79FF509F /
C
C***FIRST EXECUTABLE STATEMENT  D1MACH
     IF (I .LT. 1  .OR.  I .GT. 5)
    1   CALL XERROR( 'D1MACH -- I OUT OF BOUNDS',25,1,2)
C
     D1MACH = DMACH(I)
     RETURN
C
     END
     INTEGER FUNCTION I1MACH(I)
C***BEGIN PROLOGUE  I1MACH
C***DATE WRITTEN   750101   (YYMMDD)
C***REVISION DATE  840405   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C          HALL, A. D., (BELL LABS)
C          SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns integer machine dependent constants
C***DESCRIPTION
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C   These machine constant routines must be activated for
C   a particular environment.
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C    I1MACH can be used to obtain machine-dependent parameters
C    for the local machine environment.  It is a function
C    subroutine with one (input) argument, and can be called
C    as follows, for example
C
C         K = I1MACH(I)
C
C    where I=1,...,16.  The (output) value of K above is
```

```
C    determined by the (input) value of I.  The results for
C    various values of I are discussed below.
C
C  I/O unit numbers.
C   I1MACH( 1) = the standard input unit.
C   I1MACH( 2) = the standard output unit.
C   I1MACH( 3) = the standard punch unit.
C   I1MACH( 4) = the standard error message unit.
C
C  Words.
C   I1MACH( 5) = the number of bits per integer storage unit.
C   I1MACH( 6) = the number of characters per integer storage unit.
C
C  Integers.
C   assume integers are represented in the S-digit, base-A form
C
C            sign ( X(S-1)*A**(S-1) + ... + X(1)*A + X(0) )
C
C            where 0 .LE. X(I) .LT. A for I=0,...,S-1.
C   I1MACH( 7) = A, the base.
C   I1MACH( 8) = S, the number of base-A digits.
C   I1MACH( 9) = A**S - 1, the largest magnitude.
C
C  Floating-Point Numbers.
C   Assume floating-point numbers are represented in the T-digit,
C   base-B form
C            sign (B**E)*( (X(1)/B) + ... + (X(T)/B**T) )
C
C            where 0 .LE. X(I) .LT. B for I=1,...,T,
C            0 .LT. X(1), and EMIN .LE. E .LE. EMAX.
C   I1MACH(10) = B, the base.
C
C  Single-Precision
C   I1MACH(11) = T, the number of base-B digits.
C   I1MACH(12) = EMIN, the smallest exponent E.
```

```
C   I1MACH(13) = EMAX, the largest exponent E.
C
C  Double-Precision
C   I1MACH(14) = T, the number of base-B digits.
C   I1MACH(15) = EMIN, the smallest exponent E.
C   I1MACH(16) = EMAX, the largest exponent E.
C
C  To alter this function for a particular environment,
C  the desired set of DATA statements should be activated by
C  removing the C from column 1.  Also, the values of
C  I1MACH(1) - I1MACH(4) should be checked for consistency
C  with the local operating system.
C***REFERENCES  FOX P.A., HALL A.D., SCHRYER N.L.,*FRAMEWORK FOR A
C               PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHEMATICAL
C               SOFTWARE, VOL. 4, NO. 2, JUNE 1978, PP. 177-188.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  I1MACH
C
      INTEGER IMACH(16),OUTPUT
      EQUIVALENCE (IMACH(4),OUTPUT)
C
C
C     MACHINE CONSTANTS FOR THE CDC CYBER 170 SERIES (FTN5).
C
C      DATA IMACH( 1) /    5 /
C      DATA IMACH( 2) /    6 /
C      DATA IMACH( 3) /    7 /
C      DATA IMACH( 4) /    6 /
C      DATA IMACH( 5) /   60 /
C      DATA IMACH( 6) /   10 /
C      DATA IMACH( 7) /    2 /
C      DATA IMACH( 8) /   48 /
C      DATA IMACH( 9) / O"00007777777777777777" /
C      DATA IMACH(10) /    2 /
C      DATA IMACH(11) /   48 /
```

```
C     DATA IMACH(12) / -974 /
C     DATA IMACH(13) / 1070 /
C     DATA IMACH(14) /   96 /
C     DATA IMACH(15) / -927 /
C     DATA IMACH(16) / 1070 /
C
C    MACHINE CONSTANTS FOR THE CDC CYBER 200 SERIES
C
C    DATA IMACH( 1) /      5 /
C    DATA IMACH( 2) /      6 /
C    DATA IMACH( 3) /      7 /
C    DATA IMACH( 4) /      6 /
C    DATA IMACH( 5) /     64 /
C    DATA IMACH( 6) /      8 /
C    DATA IMACH( 7) /      2 /
C    DATA IMACH( 8) /     47 /
C    DATA IMACH( 9) / X'00007FFFFFFFFFFF' /
C    DATA IMACH(10) /      2 /
C    DATA IMACH(11) /     47 /
C    DATA IMACH(12) / -28625 /
C    DATA IMACH(13) /  28718 /
C    DATA IMACH(14) /     94 /
C    DATA IMACH(15) / -28625 /
C    DATA IMACH(16) /  28718 /
C
C
C    MACHINE CONSTANTS FOR THE CDC 6000/7000 SERIES.
C
C    DATA IMACH( 1) /    5 /
C    DATA IMACH( 2) /    6 /
C    DATA IMACH( 3) /    7 /
C    DATA IMACH( 4) /6LOUTPUT/
C    DATA IMACH( 5) /   60 /
C    DATA IMACH( 6) /   10 /
C    DATA IMACH( 7) /    2 /
```

```
C     DATA IMACH( 8) /   48 /
C     DATA IMACH( 9) / 00007777777777777777B /
C     DATA IMACH(10) /    2 /
C     DATA IMACH(11) /   47 /
C     DATA IMACH(12) / -929 /
C     DATA IMACH(13) / 1070 /
C     DATA IMACH(14) /   94 /
C     DATA IMACH(15) / -929 /
C     DATA IMACH(16) / 1069 /
C
C     MACHINE CONSTANTS FOR THE CRAY 1
C
C     DATA IMACH( 1) /   100 /
C     DATA IMACH( 2) /   101 /
C     DATA IMACH( 3) /   102 /
C     DATA IMACH( 4) /   101 /
C     DATA IMACH( 5) /    64 /
C     DATA IMACH( 6) /     8 /
C     DATA IMACH( 7) /     2 /
C     DATA IMACH( 8) /    63 /
C     DATA IMACH( 9) /  777777777777777777777B /
C     DATA IMACH(10) /     2 /
C     DATA IMACH(11) /    47 /
C     DATA IMACH(12) / -8189 /
C     DATA IMACH(13) /  8190 /
C     DATA IMACH(14) /    94 /
C     DATA IMACH(15) / -8099 /
C     DATA IMACH(16) /  8190 /
C
C
C     MACHINE CONSTANTS FOR THE IBM 360/370 SERIES,
C     THE XEROX SIGMA 5/7/9, THE SEL SYSTEMS 85/86, AND
C     THE PERKIN ELMER (INTERDATA) 7/32.
C
C     DATA IMACH( 1) /   5 /
```

```
C     DATA IMACH( 2) /    6 /
C     DATA IMACH( 3) /    7 /
C     DATA IMACH( 4) /    6 /
C     DATA IMACH( 5) /   32 /
C     DATA IMACH( 6) /    4 /
C     DATA IMACH( 7) /   16 /
C     DATA IMACH( 8) /   31 /
C     DATA IMACH( 9) / Z7FFFFFFF /
C     DATA IMACH(10) /   16 /
C     DATA IMACH(11) /    6 /
C     DATA IMACH(12) /  -64 /
C     DATA IMACH(13) /   63 /
C     DATA IMACH(14) /   14 /
C     DATA IMACH(15) /  -64 /
C     DATA IMACH(16) /   63 /
C
C     MACHINE CONSTANTS FOR THE IBM PC FAMILY (D. KAHANER NBS)
C
      DATA IMACH/5,6,0,6,32,4,2,31,2147483647,2,24,
     * -125,127,53,-1021,1023/
C           NOTE! I1MACH(3) IS NOT WELL DEFINED AND IS SET TO ZERO.
C
C
C     MACHINE CONSTANTS FOR THE PDP-10 (KA PROCESSOR).
C
C     DATA IMACH( 1) /    5 /
C     DATA IMACH( 2) /    6 /
C     DATA IMACH( 3) /    5 /
C     DATA IMACH( 4) /    6 /
C     DATA IMACH( 5) /   36 /
C     DATA IMACH( 6) /    5 /
C     DATA IMACH( 7) /    2 /
C     DATA IMACH( 8) /   35 /
C     DATA IMACH( 9) / "377777777777 /
C     DATA IMACH(10) /    2 /
```

```
C     DATA IMACH(11) /   27 /
C     DATA IMACH(12) / -128 /
C     DATA IMACH(13) /  127 /
C     DATA IMACH(14) /   54 /
C     DATA IMACH(15) / -101 /
C     DATA IMACH(16) /  127 /
C
C     MACHINE CONSTANTS FOR THE PDP-10 (KI PROCESSOR).
C
C     DATA IMACH( 1) /    5 /
C     DATA IMACH( 2) /    6 /
C     DATA IMACH( 3) /    5 /
C     DATA IMACH( 4) /    6 /
C     DATA IMACH( 5) /   36 /
C     DATA IMACH( 6) /    5 /
C     DATA IMACH( 7) /    2 /
C     DATA IMACH( 8) /   35 /
C     DATA IMACH( 9) / "377777777777 /
C     DATA IMACH(10) /    2 /
C     DATA IMACH(11) /   27 /
C     DATA IMACH(12) / -128 /
C     DATA IMACH(13) /  127 /
C     DATA IMACH(14) /   62 /
C     DATA IMACH(15) / -128 /
C     DATA IMACH(16) /  127 /
C
C     MACHINE CONSTANTS FOR THE SUN-3 (INCLUDES THOSE WITH 68881 CHIP,
C       OR WITH FPA BOARD. ALSO INCLUDES SUN-2 WITH SKY BOARD. MAY ALSO
C       WORK WITH SOFTWARE FLOATING POINT ON EITHER SYSTEM.)
C
C      DATA IMACH( 1) /    5 /
C      DATA IMACH( 2) /    6 /
C      DATA IMACH( 3) /    6 /
C      DATA IMACH( 4) /    0 /
C      DATA IMACH( 5) /   32 /
```

```
C     DATA IMACH( 6) /    4 /
C     DATA IMACH( 7) /    2 /
C     DATA IMACH( 8) /   31 /
C     DATA IMACH( 9) / 2147483647 /
C     DATA IMACH(10) /    2 /
C     DATA IMACH(11) /   24 /
C     DATA IMACH(12) / -125 /
C     DATA IMACH(13) /  128 /
C     DATA IMACH(14) /   53 /
C     DATA IMACH(15) / -1021 /
C     DATA IMACH(16) /  1024 /
C
C    MACHINE CONSTANTS FOR THE VAX 11/780
C
C    DATA IMACH(1) /    5 /
C    DATA IMACH(2) /    6 /
C    DATA IMACH(3) /    5 /
C    DATA IMACH(4) /    6 /
C    DATA IMACH(5) /   32 /
C    DATA IMACH(6) /    4 /
C    DATA IMACH(7) /    2 /
C    DATA IMACH(8) /   31 /
C    DATA IMACH(9) /2147483647 /
C    DATA IMACH(10)/    2 /
C    DATA IMACH(11)/   24 /
C    DATA IMACH(12)/ -127 /
C    DATA IMACH(13)/  127 /
C    DATA IMACH(14)/   56 /
C    DATA IMACH(15)/ -127 /
C    DATA IMACH(16)/  127 /
C
C***FIRST EXECUTABLE STATEMENT  I1MACH
     IF (I .LT. 1  .OR.  I .GT. 16)
    1   CALL XERROR ( 'I1MACH -- I OUT OF BOUNDS',25,1,2)
C
```

```
      I1MACH=IMACH(I)
      RETURN
C
      END
      SUBROUTINE XERROR(MESSG,NMESSG,NERR,LEVEL)
C***BEGIN PROLOGUE  XERROR
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  870930   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Processes an error (diagnostic) message.
C***DESCRIPTION
C    From the book "Numerical Methods and Software", D. Kahaner, C. Moler, S. Nash,  Prentice Hall 1988
C     Abstract
C        XERROR processes a diagnostic message. It is a stub routine
C        written for the book above. Actually, XERROR is a sophisticated
C        error handling package with many options, and is described
C        in the reference below. Our version has the same calling sequence
C        but only prints an error message and either returns (if the
C        input value of ABS(LEVEL) is less than 2) or stops (if the
C        input value of ABS(LEVEL) equals 2).
C
C     Description of Parameters
C      --Input--
C        MESSG - the Hollerith message to be processed.
C        NMESSG- the actual number of characters in MESSG.
C                (this is ignored in this stub routine)
C        NERR  - the error number associated with this message.
C                NERR must not be zero.
C                (this is ignored in this stub routine)
C        LEVEL - error category.
C                =2 means this is an unconditionally fatal error.
C                =1 means this is a recoverable error.  (I.e., it is
C                   non-fatal if XSETF has been appropriately called.)
```

```
C            =0 means this is a warning message only.
C            =-1 means this is a warning message which is to be
C              printed at most once, regardless of how many
C              times this call is executed.
C             (in this stub routine
C                 LEVEL=2 causes a message to be printed and then a
C                          stop.
C                 LEVEL<2 causes a message to be printed and then a
C                          return.
C
C   Examples
C     CALL XERROR('SMOOTH -- NUM WAS ZERO.',23,1,2)
C     CALL XERROR('INTEG  -- LESS THAN FULL ACCURACY ACHIEVED.',
C             43,2,1)
C     CALL XERROR('ROOTER -- ACTUAL ZERO OF F FOUND BEFORE INTERVAL F
C    1ULLY COLLAPSED.',65,3,0)
C     CALL XERROR('EXP    -- UNDERFLOWS BEING SET TO ZERO.',39,1,-1)
C
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C             HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C             1982.
C***ROUTINES CALLED  XERRWV
C***END PROLOGUE  XERROR
      CHARACTER*(*) MESSG
C***FIRST EXECUTABLE STATEMENT  XERROR
      CALL XERRWV(MESSG,NMESSG,NERR,LEVEL,0,0,0,0,0.,0.)
      RETURN
      END
      SUBROUTINE XERRWV(MESSG,NMESSG,NERR,LEVEL,NI,I1,I2,NR,R1,R2)
C***BEGIN PROLOGUE  XERRWV
C***DATE WRITTEN   800319   (YYMMDD)
C***REVISION DATE  870930   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
```

```
C***PURPOSE  Processes error message allowing 2 integer and two real
C          values to be included in the message.
C***DESCRIPTION
C   From the book "Numerical Methods and Software"
C      by  D. Kahaner, C. Moler, S. Nash
C         Prentice Hall 1988
C    Abstract
C       XERRWV prints a diagnostic error message.
C       In addition, up to two integer values and two real
C       values may be printed along with the message.
C       A stub routine for the book above. The actual XERRWV is described
C       in the reference below and contains many other options.
C
C    Description of Parameters
C     --Input--
C       MESSG - the Hollerith message to be processed.
C       NMESSG- the actual number of characters in MESSG.
C             (ignored in this stub)
C       NERR  - the error number associated with this message.
C             NERR must not be zero.
C             (ignored in this stub)
C       LEVEL - error category.
C             =2 means this is an unconditionally fatal error.
C             =1 means this is a recoverable error.  (I.e., it is
C               non-fatal if XSETF has been appropriately called.)
C             =0 means this is a warning message only.
C             =-1 means this is a warning message which is to be
C               printed at most once, regardless of how many
C               times this call is executed.
C              (in this stub LEVEL=2 causes an error message to be
C                              printed followed by a stop,
C                        LEVEL<2 causes an error message to be
C                              printed followed by a return.)
C       NI    - number of integer values to be printed. (0 to 2)
C       I1    - first integer value.
```

```
C       I2    - second integer value.
C       NR    - number of real values to be printed. (0 to 2)
C       R1    - first real value.
C       R2    - second real value.
C
C     Examples
C       CALL XERRWV('SMOOTH -- NUM (=I1) WAS ZERO.',29,1,2,
C    1   1,NUM,0,0,0.,0.)
C       CALL XERRWV('QUADXY -- REQUESTED ERROR (R1) LESS THAN MINIMUM (
C    1R2).,54,77,1,0,0,0,2,ERRREQ,ERRMIN)
C
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  XERRWV
      CHARACTER*(*) MESSG
C***FIRST EXECUTABLE STATEMENT  XERRWV
      WRITE(*,*) MESSG
      IF(NI.EQ.2)THEN
        WRITE(*,*) I1,I2
      ELSEIF(NI.EQ.1) THEN
        WRITE(*,*) I1
      ENDIF
      IF(NR.EQ.2) THEN
        WRITE(*,*) R1,R2
      ELSEIF(NR.EQ.1) THEN
        WRITE(*,*) R1
      ENDIF
      IF(ABS(LEVEL).LT.2)RETURN
      STOP
      END
```

ประวัติผู้วิจัย

ชื่อ	นายยุทธศักดิ์ บุญรอด
ที่อยู่	44 หมู่ 7 ตำบลหนองปากโลง อำเภอเมือง จังหวัดนครปฐม 73000
ประวัติการศึกษา	
พ.ศ.2542	จบมัธยมศึกษาปีที่ 6 โรงเรียนศรีวิชัยวิทยา อำเภอเมือง จังหวัดนครปฐม
พ.ศ.2546	จบการศึกษาระดับปริญญาบัณฑิต สาขาฟิสิกส์ มหาวิทยาลัยศิลปากร จังหวัดนครปฐม
พ.ศ.2547	ศึกษาต่อระดับปริญญาวิทยาศาสตรมหาบัณฑิต สาขาฟิสิกส์ มหาวิทยาลัยศิลปากร จังหวัดนครปฐม