APPENDIX-2

The simplex search technique of Nedler and Mead consists of an iterative method of parameter estimation through optimisation. Convergence of the program is by the minimisation of residual sum of squares of an objective function viz. reaction rate in case of differential analysis and space time in the case of integral analysis. The method requires supply of initial guess of parameter values (based on linear regression estimates from linearised rate equations or other sources) after which it creates (P+1) sets (where P is the number of parameters) wherein one parameter is modified in each set. Residual sum of squares of each set is compared and parameter values are modified depending on this feed back. A listing of the program is attached below. It consists of a main program SIM:FTN which modifies the parameters and decides the convergence. A subroutine FMIN:FTN calculates the function based on current values passed on from the main program and also calculates and feeds back the residual sum of squares to the main program.

```
      SIM:FTN
c     Simplex program for multiple parameter estimation
      implicit real *8(a-h,o-z)
      dimension xx(10),f(10),xb(10),xs(10),xm(10),xe(10),
     1 xr(10),xk(10),s(21),x(11,10),zf(10),rep(10)
      common y(25),pp(25),po(25),ph(25),Pi(25),fcal(25),ycal(25),
     1 con(25),eq,pp0,t0,ta,modeln,modeld
      open(unit=3,file='sim:dat',status='old')
      open(unit=4,file='prg:prt',status='unknown',access='print')
      read(3,*)nobs,npar,eq,pp0,p0,t0,modeln,modeld
      write(4,*)nobs,npar,eq,pp0,p0,t0,modeln,modeld
         do 200 n=1,nobs
         read(3,*)y(n),con(n),zf(n)
c         read(3,*)y(n),con(n),pp(n),po(n),ph(n),pi(n)
c       con(n)=n*o.2
c     y(n)=200.0 +150.0*dtanh(3.00*(dlog(con(n))-1.00))
 200    continue
```

```
        write(4,*)(y(n),con(n),zf(n),n=1,nobs)
c       write(4,*)(y(n),con(n),pp(n),po(n),ph(n),n=1,nobs)
       if (modeln .eq. 9) goto 701
      do 100 n=1,nobs
c     den=pp0*(con(n)+7)-zf(n)
       pp(n)=(1-con(n))/(7+con(n))
c      pp(n)=(pp0*(1-con(n))-zf(n)*(1-con(n)))/den
c      po(n)=(pp0*con(n)-zf(n)*con(n))/den
c      ph(n)=(pp0*con(n)-zf(n)*con(n)+6*pp0)/den
     po(n)=con(n)/(7+con(n))
     ph(n)=(6+con(n))/(7+con(n))
     if(modeln .eq. 0) goto 100
     if(modeln .eq. 1) goto 4
     if(modeln .eq. 2) goto 4
     if(modeln .eq. 3) goto 2
     if(modeln .eq. 4) goto 5
     if(modeln .eq. 5) goto 4
     if(modeln .eq. 6) goto 4
     if(modeln .eq. 7) goto 2
     if(modeln .eq. 8) goto 5
 4    y(n)=pp(n)-(po(n)*ph(n)/eq))/y(n)
    if(modeln .eq. 2) goto 3
      goto 100
 3     y(n)=dsqrt(y(n))
     goto 100
 2    y(n)=pp(n)/ph(n)-po(n)/eq)/y(n)
     goto 100
 5     y(n)=(pp(n)/po(n)-ph(n)/eq)/y(n)
100    continue
c     ************************************************
 701     nn=npar+1
      n=npar
      read(3,*) (x(1,I),i=1,npar)
         read(3,*) sa
c       sa=1.0d-07
            do 201 j=2,nn
            do 201 I=1,n
            if(j-I-1) 202,203,202
  203       x(j,i)=1.1*x(1,i)
            goto 201
  202       x(j,i)=x(1,i)
  201       continue
       write(4,300)
  300  format(1x,'initial parameters'/1x,18('-')//5x,'xx(1)',10x,
     1 'xx(2)',10x,'xx83)',10x,'xx(4)'/80('-'))
            do 204 j=1,nn
           write(4,400)(x(j,i),i=1,n)
  204      continue
 1001  do 1 j=1,nn
       do 21 i=1,n
  21   xx(i)=x(j,i)
       CALL FMIN(npar,nobs,xx,ff)
       F(j)=FF
       nf=nn
       alfa=1.0d+00
```

```
      beta=0.5d+00
      gamma=2.0d+00
      iter=0
      jpr=0
400   format(5(d12.5,3x))
      goto 27
 25   fb=f(1)
           do 98 i=1,n
 98    xb(i)=x(1,i)
       jb=1
       do 31  J=2,nn
       if(fb-f(j))31,31,108
108    fb=f(j)
       jb=j
        do 41 i=1,n
 41        xb(i)=x(j,i)
 31        continue
      fs=f(1)
      do 51 i=1,n
 51   xs(i)=x(1,i)
      js=1
          do 61 j=2,nn
          if(fs-f(j)) 111,61,61
111       fs=f(j)
          js=j
      do 71 i=1,n
 71   xs(i)=x(j,i)
 61           continue
     do 81 i=1,n
 81   xm(i)=-xs(i)
             do 9 j=1,nn
                  do 122 i=1,n
122               xm(i)=xm(i)+x(j,i)
  9             continue
       do 121 i=1,n
 121   xm(i)=xm(i)/float(n)
       do 131 i=1,n
      xr(i)=xm(i)+alfa*(xm(i)-xs(i)
 131   continue
       CALL FMIN(npar,nobs,xr,fr)
       nf=nf+1
       if(fr-fb) 141,151,151
  141           do 161 i=1,n
         xe(i)=xm(i)+gamma*(xr(i)-xm(i))
 161     continue
       CALL FMIN(npar,nobs,xe,fe)
       nf=nf+1
       if(fe-fb)  17,18,18
  17    do 19 i=1,n
        x(js,i)=xe(i)
  19    xs(i)=xe(i)
        f(js)=fe
  27    fm=0.0d+00
          do 20 j=1,nn
  20    fm=fm+f(j)
```

```
      fm=fm/float(nn)
      frms=0.0d+00
      do 22 j=1,nn
 22   frms=(f(j)-fm)**2+frms
      rms=dsqrt(frms/float(n))
      iter=iter+1
      jpr=jpr+1
      if(iter-500)  500,500,23
500   continue
      if(jpr-1)  902,902,903
903   continue
      if(jpr-6)  901,904,904
904   jpr=1
902   continue
      write(4,107) iter,nf
107   format(1x,'iteration',i4,5x,'No.of calls for subroutine
     1 fmin',i5/1x,9('-'),9x,30('-'))
      write(4,109)
109   format(1x,'parameters'/1x,10('-'))
      write(4,400) (x(js,i),i=1,n)
      write(4,106) f(js),rms
106   format(1x,'fmin=',d15.7,5x,'sd=',d15.7/1x,4('-'),23x,2('-'))
901   continue
      if (rms-sa) 23,23,25
 18       do 26 i=1,n
      x(js,i)=xr(i)
 26   xs(i)=xr(i)
      f(js)=fr
      fs=fr
      go to 27
151    do 30 j=1,nn
      if(j-js)   28,30,28
 28   if(fr-f(j))  18,18,30
 30    continue
      if(fr-fs) 91,91,32
 91       do  33 i=1,n
          x(js,i)=xr(i)
 33   xs(i)=xr(i)
      f(js)=fr
      fs=fr
 32    do  34 i=1,n
      xk(i)=xm(i)+beta*(xs(i)-xm(i))
 34   continue
      CALL FMIN(npar,nobs,xk,fk)
      nf=nf+1
      if(fk-fs) 35,35,36
 35       do  37 i=1,n
         x(js,i)=xk(i)
 37      xs(i)=xk(i)
      f(js)=fk
      fs=fk
      go to 27
 36   do  38 j=1,nn
         do 39 i=1,n
 39   x(j,i)=(x(j,i)+xb(i))/2.0
```

```
 38   continue
      go to 1001
 23   write(4,905) iter
905   format(/5x,'convergence obtained at iteration',i4/1x
   1  ,'final parameters'/1x,16('-')/5x,'xx(1)',12x,'xx(2)',12x,
   2  'xx(3)',14x,'xx(4)'/80('-'))
      write(4,400)  (x(js,i),i=1,n)
      write(4,106) f(js),rms
      do 101 i=1,npar
      xx(i)=x(js,i)
101   continue
      if(rms  .gt.  sa)  go to 402
      go to 405
402   write(4,403)
403   format(/5x,'convergence not obtained after Max. iterations'/)
405   write(4,207)
207   format(//80('-')/2x,'Sr.No.',3x,'X(n)',8x,'Y(n)',6x,'Ycal(n)
   1  ',8x,'DEV.',8x,'Pp(n)',6x,'Po(n)',6x,'Ph(n)',6x,'Pi(n)'/80
   2  ('-')/)
      CALL FMIN(npar,nobs,xx,ff)
      do 222 n=1,nobs
 write(4,96)n,con(n),y(n),ycal(n),fcal(n),Pp(n),Po(n),Ph(n),Pi(n)
 96   format(3x,i3,8e12.4)
222   continue
     write(4,315)
315  format(1x,'rate const.',6x,'C12H26 ads. const',4x,'C12H24 ads.
   1  const.',3x,'H2  ads.const.'/)
     write(4,400) (x(js,i),i=1,npar)
c     calculation of Lack of Fit,read No.rep runs for pue & expt.
      read(3,*) nrep,krep
      ybar=0.0
      Read values of rates/conversions of replicated runs
      read(3,*) (rep(n),n=1,nrep)
      do 444 n=1,nrep
      ybar=ybar+rep(n)
444   continue
      ybar=ybar/nrep
      pue=0.0
      do 555 n=1,nrep
      pue=pue+(rep(n)-ybar)**2
555   continue
      dftot=nobs
      dfreg=npar
      dfres=nobs-npar
      dflf=dfres-erep+1
      dfpue=nrep-1
      tot=0.0
      reg=0.0
      res=0.0
         do 333 n=1,nobs
         tot=tot+(y(n))**2
         reg=reg+(ycal(n))**2
         res=res+y(n)-ycal(n))**2
333   continue
      Alf=res-pue
```

```
      Fc=(alf/dflf)/(pue/dfpue)
      write(4,313) tot, reg, res, pue, alf
313   format(/3x,'F-TEST FOR MODEL ADEQUACY'/3x,25('-')/3x,'TOTAL
     1 SUM OF SQUARES =',D15.4/3x,20('-')/3x,'REGRESSION SUM OF
     2 SQUARES=',d15.4/3x,25('-')/3x,'RESIDUAL SUM OF SQUARES=',
     3 d15.4/3x,23('-')/3x,'PURE ERROR SUM OF SQUARES=',d15.4/3x,'
     4 LACK OF FIT SUM OF SQUARES=',d15.4/3x,26('-'))
      write(4,314)  Fc
314   format(//3x,'F-CALCULATED =',d15.4,7x,'F-TABLE=',/3x,12('-'),
     1  25x,7('-')/80('*'))
      close (unit=3)
      close (unit=4)
      stop
      end


      SUB ROUTINE FMIN(NPAR,NOBS,XX,FF)
c     For DMKALR  equations
      implicit real*8(a-h,o-z)
      dimension xx(10),fcaln(25),fcald(25)
      common y(25),Pp(25),Po(25),Ph(25),Pi(25),fcal(25),ycal(25),
     1 con(25),eq,pp0,t0,ta,modeln,modeld
      ff=0.0d+00
      r=1.987d+00
           do 21 n=1,nobs
           ek=xx(1)
           ekp=xx(2)
           eko=xx(3)
           ekh=xx(4)
           eki=xx(5)
        if (modeln. eq. 0) go to 12
        if (modeln. eq. 1) go to 1
        if (modeln. eq. 2) go to 1
        if (modeln. eq. 3) go to 2
        if (modeln. eq. 4) go to 3
        if (modeln. eq. 5) go to 1
        if (modeln. eq. 6) go to 1
        if (modeln. eq. 7) go to 2
        if (modeln. eq. 8) go to 3
        if (modeln. eq. 9) go to 13
 13   a=7.0
      b= -6-6/eq
      c= -1-1/eq
      alpha=(7+ekp+6*ekh)**2
      gamma=(1.0-ekp+eko+ekh)**2
      beta=2*dsqrt(alpha*gamma)
      qu=a+b*con(n)+c*con(n)**2
      bi=alpha+beta*con(n)
      dta=b**2-4*a*c
      to=dsqrt(dta)
      dath=dlog(dabs(((b+2*c*con(n)-to)*(b+to))/((b+2*c*con(n)+to)
     1 *(b-to))))
      dxr=(1/to)*dath
      fcal(n)= (1/(ek*ekp))*(alpha*dxr+(beta)*((0.5/c)*dlog(qu/a)-
     1 (0.5*b/c)*dxr)+(gamma)*(con(n)/c-(0.5*b/c**2)*dlog(qu/a)+
```

```
      2  ((b**2-2*a*c)/(2*c**2))*dxr))
c        term1=49*(-1-1/(con(n)-1))
c        term2=14*(dlog(-(con(n)-1))-(1/(con(n)-1))-1)
c        term3=(con(n)-1-(1/(con(n)-1))+2*dlog(-(con(n)-1)))
c        fcal(n)=(1/ek)*(term1+term2+term3)
c        term0=dabs((1/dsqrt(6*6))*atan(con(n)*dsqrt(6*6)/6))
c        term1=49*term0
c        term2=-14*((1/(2*6))*dlog((6+6*con(n)**2)/6))
c        term3=1*term0
c        term3= (con(n)/6)-term3
c        fcal(n)=(1/ek)*(term1+term2+term3)
c        fcal(n)=(1/ek)*(-con(n)-8*dlog(1-con(n)))
c        fcal(n)=ek+ekp+dtanh(eko*(dlog(con(n))-ekh))
       if (modeld .eq. 9) go to 20
    12   fcal(n)=ek*(Pp(n)**ekp-(Ph(n)**ekh*Po(n)**eko*1/eq))
       if (modeld .eq. 0) go to 20
c         go to 22
    1    fcaln(n)=ek*(Pp(n)-Po(n)*Ph(n)*1/eq)
    1    if (modeld .eq. 1) go to 4
         if (modeld .eq. 2) go to 5
         if (modeld .eq. 5) go to 8
         if (modeld .eq. 6) go to 9
           fcaln(n)=ek*(Pp(n)/Ph(n)-(Po(n)*1/eq))
         if (modeld .eq. 3) go to 6
         if (modeld .eq. 7) go to 10
c   3    fcaln(n)=ek*(Pp(n)/Po(n)-(Ph(n)*1/eq))
    3    if (modeld .eq. 4) go to 7
         if (modeld .eq. 8) go to 11
       DUAL SITE ADSORPTION CONTROL
c   4    fcald(n)=(1.0+ekp*Po(n)*Ph(n)+eko*Po(n)+ekh*Ph(n))
    4           fcal(n)=ek+ekp*Po(n)*Ph(n)+eko*Po(n)+ekh*Ph(n)
         go to 20
c      DUAL SITE SURFACE REACTION CONTROL
c   5    fcald(n)=(1.0+ekp*Pp(n)+eko*Po(n)+ekh*Ph(n))**2
    5    ETM=1/DSQRT(ek*ekp)
       fcal(n)=etm+ETM*ekp*Pp(n)+ETM*eko*Po(n)+ETM*ekH*PH(n)
        go to 20
c      DUAL SITE OLEFIN DESORPTION CONTROL
c   6  fcald(n)=(1.0+ekp*Pp(n)+eko*Pp(n)/Ph(n)+ekh*Ph(n))
    6  facl(n)=ek+ekp*Pp(n)+eko*Pp(n)/Ph(n)+ekh*Ph(n)
        go to 20
c      DUAL SITE HYDROGEN DESORPTION CONTROL
c   7  fcald(n)=(1.0+ekp*Pp(n)+eko*Po(n)+ekh*Pp(n)/Po(n))
    7   fcal(n)=ek+ekp*Pp(n)+eko*Po(n)+ekh*Pp(n)/Po(n)
        go to 20
c      SINGLE SITE ADSORPTION CONTROL
c   8  fcald(n)=(1.0+ekp*Po(n)*Ph(n)+eko*Po(n))
    8      fcal(n)=ek+ekp*Po(n)*Ph(n)+eko*Po(n)
         go to 20
c      SINGLE SITE SURFACE REACTION CONTROL
c   9  fcald(n)=(1.0+ekp*Pp(n)+eko*Po(n))
    9    fcal(n)=ek+ekp*Pp(n)+eko*Po(n)
         go to 20
c      SINGLE SITE OLEFIN DESORPTION CONTROL
c  10  fcald(n)=(1.0+ekp*Pp(n)+eko*Pp(n)/Ph(n))
```

```
   10      fcal(n)=ek+ekp*Pp(n)+eko*Pp(n)/Ph(n)
c       SINGLE SITE HYDROGEN DESORPTION CONTROL
c   11   fcald(n)=(1.0+ekp*Pp(n)+eko*Pp(n)/Po(n))
   11      fcal(n)=ek+ekp*Pp(n)+eko*Pp(n)/Po(n)
           go to 20
c         fcal(n)=(1+ek*con(n))** -ekp
c   20     fcal(n)=fcaln(n)/fcald(n)
   20     fcal(n)=y(n)-fcal(n)
         ycal(n)=fcal(n)+y(n)
         ff=ff+(fcal(n))**2
   21    continue
         return
         end

PROGRAM USED FOR POLYNOMIAL FITTING

         PRG:FTN
c        POLYNOMIAL FITTING
c        load program PRGSTAT:FTN also (subroutine STATS)
         IMPLICIT REAL*8 (A-H,O-Z)
         DIMENSION VEC(30),SIG(41),A(5,5),XI(5,5),XX(20),
     1   YY(20),Z(20),E(20),BE(20),CE(20)

c        open files PRG:DAT & PRG:PRT for storage of data and
         output respt.
         OPEN(UNIT=3,FILE='prg:DAT',STATUS='OLD')
         OPEN(UNIT=4,FILE='prg:PRT',STATUS='UNKNOWN',ACCESS='PRINT')
c        reads the order of the polynomial and the no. of data points
         READ(3,*)N,M,Temp
  801    FORMAT(2I4)
         write(4,*) 'POLYNOMIAL REGRESSION'
         write(4,*) 'IInd order fit of conversion Vs. space time'
         write(4,*) 'reaction temperature, K:',Temp
         write(4,*) 'order of polynomial, No. of data sets'
         WRITE(4,801)N,M
c        Reads the independant XX(I),and dependant YY(I) variables
         READ(3,*)(XX(I),I=1,M)
c           do 711 i=1,m
c        xx(i)=1/xx(i)
c 711      continue
c 802    FORMAT(5F6.3)
         READ(3,*)(YY(I),I=1,M)
c           do 222 j=1,m
c        yy(j)=log(yy(j))
c 222     continue
c 803    FORMAT(5F7.4)
         Write(4,*) 'Independant Variable XX(I) Space time'
         WRITE(4,804)(XX(I),I=1,M)
 804     FORMAT(5F10.5)
         Write(4,*) 'Dependant variable YY(I), Conversion'
         WRITE(4,804)(YY(I),I=1,M)
         write(4,*) 'Parametric constants of the equation are'
         N1=N*2+1
         NP1=N+1
         DO 1 I=1,NP1
```

```
 1      VEC(I)=0
        DO 3 I=1,N1
 3      SIG(I)=0
        DO 13 J=1,M
        X=XX(J)
        Y=YY(J)
        C=1.0
        DO 11 I=1,N1
        IF(NP1-I)9,7,7
 7      VEC(I)=VEC(I)+C*Y
 9      SIG(I)=SIG(I)+C
11      C=C*X
13      CONTINUE
        DO 19 I=1,NP1
        J=NP1+I
17      J=J-1
        K=NP1+I-J
        A(I,K)=SIG(J)
        IF(K-NP1) 17,19,17
19      CONTINUE
        DO 25 I=1,NP1
        DO 25 J=1,NP1
        IF(I-J) 23,21,23
21      XI(I,J)=1
        GO TO 25
23      XI(I,J)=0
25      CONTINUE
        DO 57 IP=1,NP1
        IMAX=IP
        ELMAX=ABS(A(IP,1))
        DO 29 I=IP,NP1
        DO 29 J=1,NP1
        IF(ELMAX-ABS(A(I,J)))27,29,29
27      IMAX=I
        ELMAX=ABS(A(I,J))
29      CONTINUE
        IF(IMAX-IP)31,35,31
31      DO 33 J=1,NP1
        X=A(IP,J)
        A(IP,J)=A(IMAX,J)
        A(IMAX,J)=X
        X=XI(IP,J)
        XI(IP,J)=XI(IMAX,J)
33      XI(IMAX,J)=X
35      K=0
37      IF(A(IP,IP)-0.0)49,39,49
39      K=K+1
        DO 41 J=1,NP1
        A(IP,J)=A(IP,J)+A(K,J)
41      XI(IP,J)=XI(IP,J)+XI(K,J)
        IF(K-NP1)37,43,43
43      IF(A(IP,IP)-0.0)49,45,49
45      WRITE(4,47)
47      FORMAT(2X,'FORMAT IS SINGULAR',/)
        GO TO 673
```

```
   49     ELMAX=1./A(IP,IP)
          DO 51 J=1,NP1
          A(IP,J)=A(IP,J)*ELMAX
   51     XI(IP,J)=XI(IP,J)*ELMAX
          DO 57 I=1,NP1
          IF(I-IP)53,57,53
   53     AIP=A(I,IP)
          DO 55 J=1,NP1
          A(I,J)=A(I,J)-AIP*A(IP,J)
   55     XI(I,J)=XI(I,J)-AIP*XI(IP,J)
   57     CONTINUE
          DO 59 K2=1,NP1
          SIG(K2)=0
          DO 59 J=1,NP1
   59     SIG(K2)=SIG(K2)+XI(K2,J)*VEC(J)
c         Reverts the order of the storage array for parameters VEC(IJ)
          IJ=N
          DO 61 J=1,NP1
          VEC(IJ)=SIG(J)
          IJ=IJ-1
   61     IF(IJ.LT.0) GO TO 62
   62     write(4,*) (vec(ij),ij=0,n)
c         computes the predicted values of the dependant variable
          DO 599 I=1,M
          BE(I)=0.0
          DO 599 J=0,N
          E(I)=VEC(J)*XX(I)**J
  599     BE(I)=BE(I)+E(I)
          write(4,*) 'independant variable space time'
          write(4,*) (XX(i),i=1,m)
          write(4,*) 'Predicted dep. variable conversion'
          write(4,*) (BE(i),i=1,m)
          write(4,*) 'Derivative of the dep. var. wrt the indep. var.'
          write(4,*) 'Rate of reaction, g mol/h/g cat'
c         computes derivative of polynomial equation
          do 600 ia=1,m
          CE(IA)=0.0
          do 610 JA=0,N
          IF(JA-1)101,111,111
  111     Z(IA)=JA*VEC(JA)*XX(IA)**(JA-1)
          GO TO 611
  101     Z(IA)=0.0
          CE(IA)=CE(IA)+Z(IA)
  610     continue
  600     continue
          write(4,*) (CE(ia),ia=1,m)
c 602     FORMAT(2X,I4,5X,E15.8,5X,E15.8,5X,E15.8,/)

c         Calls SUBROUTINE STATS to test for significance
          call stats(xx,yy,n,m,vec)
  673     CLOSE(UNIT=3)
          CLOSE(UNIT=4)
          STOP
          END
```

```
      SUBROUTINE STATS(X,Y,N,M,VEC)
c     This subroutine calculates the  total sum of  squares SSTOT
c     regression sum of squares SSR, residual sum of squares RSS,
c     pure error sum of squares SSPE, lack of fit sum of  squares
c     SSLF and the multiple coefficient of determination RSQ. The
c     F-Test for significance and the F-value are computed.  This
c     value of F needs to be compared with the statistical F-table
c     value of corresponding to degrees of freedom relating to any
c     any specific problem and desired level of confidence

c     X(I)  value of the independant variable at observation i
c     Y(I)  value of the dependant variable at observation i
c     N     order of the polynomial
c     M     Number of observations/experiments
c     VEC(I) value of the parameter beginning from parameter no.2
      implicit real*8(a-h,o-z)
      dimension x(50),y(50),syx(5),sumx(5),sy(5),vec(5)

c     computes summation of Y(I) and of Y(I)**2
            sumy=0.0
            sumy2=0.0
      do 1 i=1,m
      sumy=sumy+y(I)
      sumy2=sumy2+y(I)*Y(I)
    1 continue
      SSTOT =sumy2
      write(4,*)  'Total sum of squares =',  sstot
c     computes square of the summation of Y(I)
      sumys2=sumy**2

c     do loops 2, 3, & 4 compute the variability of Y about
      the trend lines of X and its higher orders, SY(ma)
      do 2 j=1,n
      syx(j)=0.0
      sy(J)=0.0
      sumx(j)=0.0
    2 continue
        do 3 k=1,n
        do 3 l=1,m
      syx(k)=syx(k)+x(l)**k*y(l)
      sumx(k)=sumx(k)+x(l)**k
    3 continue
       do 4 ma=1,n
      sy(ma)=syx(ma)-sumx(ma)*sumy/m
    4 continue

c     computes the variability of Y about Ybar(Y-average), SYY
      syy=sumys2-sumys2/m
c     computes the summation of the products of the respective
      parameters and the corresponding variabilities of Y(I)
      about X(I) and its higher orders, SRR
      ssr=0.0
      do 5 ix=1,n
      ssr=ssr+vec(ix)*sy(ix)
    5 continue
```

```
      write(4,*)'Regression sum of squares =', SSR
c     computes the residual sum of squares RSS via the short
      formula rss=syy-ssr
      write(4,*)'Residual sum of squares =',RSS
c     computes the degrees of freedom for Total, Regression,
      Residual, Pure error and Lack of fit sum of squares
      respectively.
      dftot=m
      dfreg=n
      dfres=m-n-1
c      dfpe=Nr-1
c      dflf=m-(n+1)-(Nr-1)
c     Reads Nr, no. of replicate experiments, Yr(I) dep. variable
      of these replicate experiments & computes pure error  sum
      ofsquares
c     read(*,*)Nr
c     read(*,*) (Yr(I),I=1,Nr)
c     Yrbar=0.0
c     do 6 j=1,Nr
c     Yrbar=Yrbar+Yr(j)
c  6  continue
c     yrbar=Yrbar/Nr
c     sspe=0.0
c     do 7 k=1,Nr
c     sspe=sspe+(Yr(K)-Yrbar)
c  7  continue
c     sslf=RSS-sspe
c     computes F value and the multiple coefficient of
      determination RSQ
      Fnum=SSR/dfreg
      Fden=rss/dfres
      F=Fnum/Fden
      RSQ=SSR/Syy
      write(4,*) 'F =', F,  'Coeff. of Det.=', RSQ
      return
      end
```