Ю.С.Кушликис

ВЫЧИСЛЕНИЕ СЕРИЙ ИНДЕКСОВ ГОДИЧНЫХ КОЛЕЦ И ОТОБРАЖЕНИЕ ИХ В ВИДЕ ГРАФИКОВ ПРИ ПОМОЩИ АЦПУ ИЛИ ГРАФОПОСТРОИТЕЛЯ

В дендрохронологических исследованиях вместо серий годичных колец часто используются серии индексов годичных колец [1] , вычисляемые по формуле

$$r_u^* = \frac{r_u}{s_u} \quad , u = 1, 2, \dots, n,$$

где $R = (r_1, r_2, \dots, r_n)$ – серия годичных колец,

r_u – ширина кольца,

$S = (s_1, s_2, \dots, s_n)$ – кривая роста, получаемая путем аппроксимации низкочастотной составляющей серии R ,

$R^* = (r_1^*, r_2^*, \dots, r_n^*)$ – серия индексов годичных колец.

Кривую роста можно аппроксимировать различными функциями. Наиболее просто аппроксимация производится прямой или отрицательной экспоненциальной функцией. В описываемых программах аппроксимация производится ортогональными многочленами [2] , позволяющими аппроксимировать более сложные кривые. В этом случае кривая роста выражается формулой

$$S_u^* = \sum_{c=1}^{m+1} \varepsilon_c \cdot (u + 10)^{c-1}$$

где m – степень многочлена,

ε_c – коэффициенты, полученные из серии R .

Ниже описывается комплекс программ, вычисляющих индексы годичных колец и отображающих серии индексов в виде графиков при помощи графопостроителя или АЦПУ. Подпрограммы составлены на алгоритмических языках ФОРТРАН [3] и АЛГОЛ [4] (подпрограммы ФОРТРАН'а могут обращаться к независимо транслированным процедурам АЛГОЛ'а и наоборот). Обращение к графопостроителю производится при помощи комплекса графических программ ГРАФОР [5] . Серии годичных колец должны быть записаны в базу данных, находящуюся на магнитной ленте МЛ 4-0. Каждое дерево, характеризуемое инвентарным номером I_i , $i = 1, 2, 3, \dots$, в базе данных представляется в виде двух серий годичных колец

$$R_j^{(i)} = \left(r_{j,1}^{(i)}, r_{j,2}^{(i)}, \dots, r_{j,n^{(i)}}^{(i)}\right) \quad , j = 1, 2,$$

где $R_1^{(i)}$ – серия колец ранней древесины,

$R_2^{(i)}$ – серия колец поздней древесины,

$r_{j,n}^{(i)}$ – ширина кольца,

$n^{(i)}$ – количество колец в серии.

Обращение к основной подпрограмме комплекса имеет вид

CALL DRATRE (NAME, YEAR, NN, KZO, KZMAX, KQ, NRMAX, XPAG, YPAG, XREG, YREG, KPLOT, KQ1, KQ2, KERR)

где NN – количество серий годичных колец,

NAME (NN) – инвентарные номера серий, т.е. $NAME(i) = I_i$,

YEAR (NN) – календарные годы первых колец серий, т.е. $YEAR(i)$ – календарный год первого кольца серии I_i,

KZO – номер первой зоны базы данных на МЦ 4–0,

KZMAX – номер последней зоны базы данных,

KQ – коэффициент, указывающий, какая древесина нужна:

$$KQ = \begin{cases} 0 - \text{ранняя древесина,} \\ 1 - \text{поздняя древесина,} \\ 2 - \text{общая древесина (сумма толщины колец ранней и поздней древесины),} \end{cases}$$

$NRMAX \leq 300$ – максимально допустимая длина серии,

XPAG – ширина страницы графопостроителя [см],

YPAG – высота страницы графопостроителя [см],

XREG – ширина графика (см для графопостроителя и позициями для АЦПУ)

$$KPLOT = \begin{cases} 0 - \text{графики печатаются на АЦПУ,} \\ 1 - \text{графики вычерчиваются на графопостроителе,} \end{cases}$$

$$KQ1 = \begin{cases} 0 - \text{графики серий годичных колец не выводятся,} \\ 1 - \text{графики серий годичных колей выводятся,} \end{cases}$$

$$KQ2 = \begin{cases} 0 - \text{графики серий индексов годичных колец не выводятся,} \\ 1 - \text{графики серий индексов годичных колец выводятся,} \end{cases}$$

KERR – признак ошибки

$$KERR = \begin{cases} 0 - \text{нет ошибки,} \\ 1 - \text{ошибка.} \end{cases}$$

При выводе графиков на графопостроитель должны выполняться следующие условия:

$$XREG < XPAG, \quad YREG < YPAG.$$

Графики на странице графопостроителя располагаются сверху вниз (с промежутком между графиками 2 см) и слева направо (с промежутком между графиками 1 см). Таким образом, все графики на странице разместятся при условии

$$entier\left(\frac{YPAG-4}{YREG+2}\right) \cdot entier\left(\frac{XPAG-4}{XREG+1}\right) > NN \cdot (KQ1 + KQ2)$$

При выводе графиков на АЦПУ ($KPLOT = 0$) значения XPAG, YPAG – произвольные. Высота графика не ограничивается. Ширина графика XREG должна быть не больше 110 (т.е. $XREG \leq 110$). Если длина (количество колец), серии больше XREG, то

график автоматически подразделяется на интервалы длиной $XREG$, которые печатаются один под другим; высота каждого - $YREG$ строк (см.рис.I).

В подпрограмме имеется общий блок

$$COMMON\ /BDT/\ RMNO, RMXO, IMNO, IMXO, RMN1, RMX1, IMN1, IMX1$$

Перед обращением к $DRATRE$ в этот общий блок засылаются следующие данные:

$RMNO$ - минимальное значение ширины кольца,

$RMXO$ - максимальное значение ширины кольца,

$IMNO$ - минимальное значение индекса,

$IMXO$ - максимальное значение индекса.

При выполнении программы с базы данных последовательно считываются серии годичных колец

$$R_j^{(i)},\ i = 1, 2, \dots, NN,$$

с инвентарными номерами

$$I_i = NAME(i) \quad \text{и} \quad j = KQ.$$

Для каждой серии

$$R_j^{(i)} = (r_{j,1}^{(i)}, r_{j,2}^{(i)}, \dots, r_{j,n^{(i)}}^{(i)})$$

выполняется:

I. при $KQ = 1$ серия $R_j^{(i)}$ выводится в виде графика:

а) при $KPLOT = 0$ - на АЦПУ,

б) при $KPLOT = 1$ - на графопостроитель.

Математическая область функции (т.е. значения нижнего и верхнего края графика) равна:

$$[RMNO, RMXO] - \text{при } RMNO < RMXO,$$
$$[RMN^*, RMX^*] - \text{при } RMNO \geqslant RMXO.$$

Тут

$$RMN^* = \min_u r_{j,u}^{(i)}, \quad RMX^* = \max_u r_{j,u}^{(i)},$$

2. вычисляется серия индексов годичных колец

$$R_j^{*(i)} = (r_{j,1}^{*(i)}, r_{j,2}^{*(i)}, \dots, r_{j,n^{(i)}}^{*(i)})$$

3. при $KQ2 = 1$ серия индексов годичных колец выводится в виде графика:

а) при $KPLOT = 0$ - на АЦПУ,

б) при $KPLOT = 1$ - на графопостроитель.

Математическая область функции (т.е. значения нижнего и верхнего края графика) равна:

$$[IMNO, IMXO] - \text{при } IMNO < IMXO,$$
$$[IMN^*, IMX^*] - \text{при } IMNO \geqslant IMXO.$$

Тут

$$IMN^* = \min_u r_{j,u}^{*(i)}, \quad IMX^* = \max_u r_{j,u}^{*(i)}.$$

После завершения подпрограммы *DRATRE* в общем блоке *BDT* остаются следующие результаты:

RMN1 – минимальное значение ширины кольца по всем сериям

$R_{kq}^{(i)}$, $i = 1, 2, \ldots, NN$,

$RMN1 = \min\limits_{i,\mu} r_{kq,\mu}^{(i)}$,

RMX1 – максимальное значение ширины кольца по всем сериям

$R_{kq}^{(i)}$, $i = 1, 2, \ldots, NN$,

$RMX1 = \max\limits_{i,\mu} r_{kq,\mu}^{(i)}$)

IMN1 – минимальное значение индекса по всем сериям

$R_{kq}^{*(i)}$, $i = 1, 2, \ldots, NN$,

$IMN1 = \min\limits_{i,\mu} r_{kq,\mu}^{*(i)}$,

IMX1 – максимальное значение индекса по всем сериям

$R_{kq}^{*(i)}$, $i = 1, 2, \ldots, NN$,

$IMX1 = \max\limits_{i,\mu} r_{kq,\mu}^{*(i)}$.

Литература

1. Т.Т.Битвинскас, Дендроклиматические исследования, Гидрометеоиздат, 1974.
2. James B. Campbell, Manual for using basic chronology programs, Laboratory of Tree-Ring Research, University of Arizona, 1973.
3. ФОРТРАН для БЭСМ-6, Вильнюс, 1974.
4. Р.Хирр, Р.Штробель, АЛГОЛ в мониторной системе ДУБНА, ИПМ АН СССР, 1973.
5. Ю.М.Баяковский, ГРАФОР: комплекс графических программ на ФОРТРАН'е, ИПМ АН СССР, 1972.

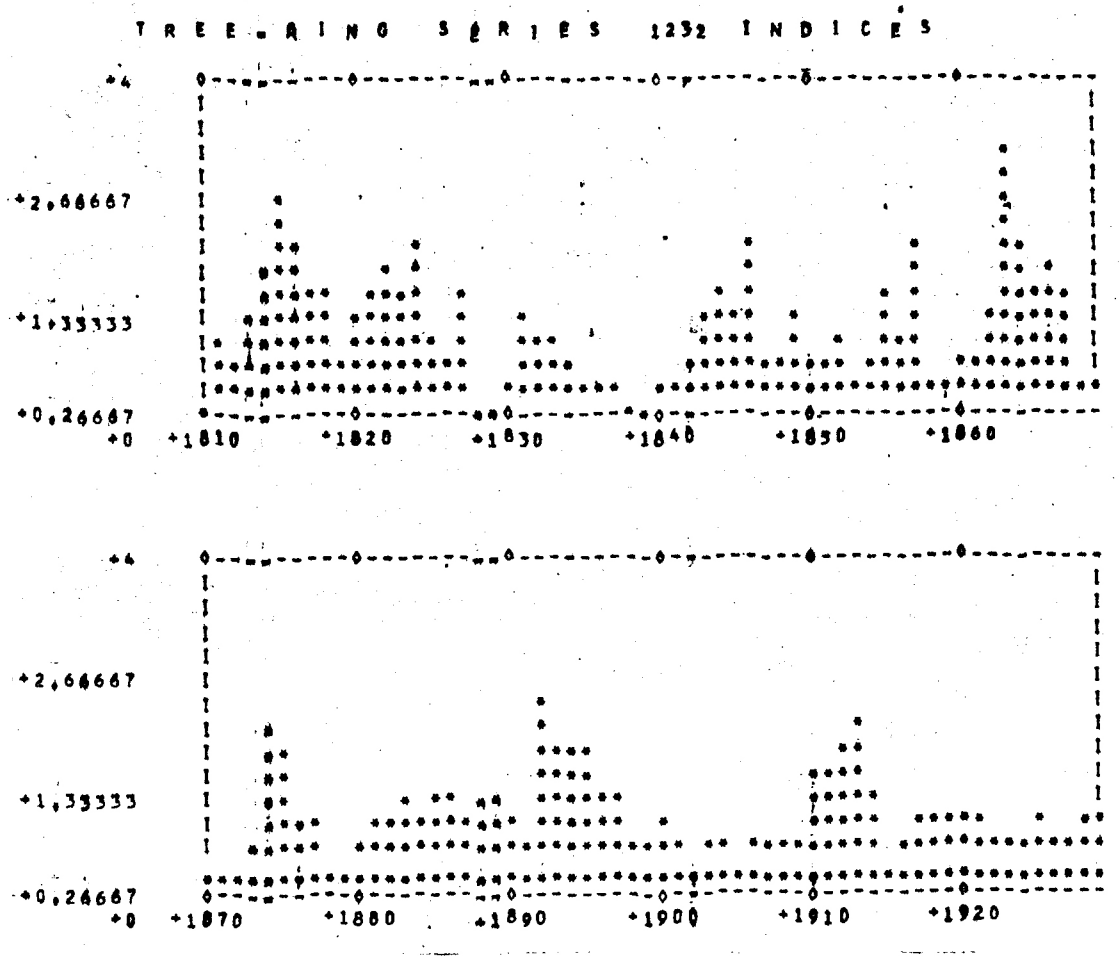
TREE-RING SERIES 1232 INDICES
+4
+2,66667
+1,33333
+0,26667
+0
+1810
+1820
+1830
+1840
+1850
+1860
+4
+2,66667
+1,33333
+0,26667
+0
+1870
+1880
+1890
+1900
+1910
+1920

Рис. 1

```
      SUBROUTINE DRATRE(NAME,YEAR,NN,KZ0,KZMAX,KQ,NRMAX,XPAG,YPAG,
     *                  XREG,YREG,KPLOT,KQ1,KQ2,KERR)
      DIMENSION NAME(NN),YEAR(NN),RW(6000),GROW(320),
     *          TEMP(320,5),COEF(31)
      COMMON /BDT/ RMN0,RMX0,IMN0,IMX0,RMN1,RMX1,IMN1,IMX1
      REAL IMN0,IMX0,IMN1,IMX1,IMN2,IMX2
      EXTERNAL RERS
C
C     SUBROUTINE   D R A T R E   IS DESIGNATED READ TREE-RING WIDTH
C     SERIES FROM MAGNETIC TAPE , COMPUTE TREE-RING INDICES AND TO
C     PLOT TREE-RING WIDTH SERIES OR INDICES BY PRINTER OR PLOTTER.
C
C     NAME(NN)    - NAMES OF TREE-RING SERIES (INTEGER NUMBERS
C                   < 1000000 )
C     YEAR(NN)    - YEARS OF FIRST RING OF SERIES
C     NN          - NUMBER OF SERIES
C     KZ0         - FIRST ZONE OF TREE-RING WIDTH SERIES ON MAGNETIC
C                   TAPE
C     KZMAX       - LAST  ZONE OF TREE-RING WIDTH SERIES ON MAGNETIC
C                   TAPE
C
C          I 0    - EARLYWOOD
C     KQ = I 1    - LATEWOOD
C          I 2    - ALLWOOD
C
C     NRMAX<=300  - MAXIMUM NUMBER OF RINGS IN SERIES
C     XPAG        - WIDTH OF PLOTTER'S PAGE (CM) , WIDTH OF PRINTER'S
C                   PAGE IS XREG=110 POSITIONS
C     YPAG        - HEIGHT OF PLOTTER'S PAGE (CM) , HEIGHT OF
C                   PRINTER'S PAGE IS UNLIMITED
C     XREG<XPAG   - WIDTH OF PLOT'S REGION ON PAGE ( CM FOR PLOTTER
C                   AND POSITIONS FOR PRINTER )
C     YREG<YPAG   - HEIGHT OF PLOT'S REGION ON PAGE ( CM FOR PLOTTER
C                   AND ROWS FOR PRINTER )
C
C          I 0    - PLOTTING BY PRINTER
C     KPLOT=I
C          I 1    - PLOTTING BY PLOTTER
C
C          I 0    - TREE-RING WIDTH SERIES NOT PLOTTED
C     KQ1 = I
C          I 1    - TREE-RING WIDTH SERIES PLOTTED
C
C          I 0    - TREE-RING INDICES NOT PLOTTED
C     KQ2 = I
C          I 1    - TREE-RING INDICES PLOTTED
C
C     KERR        - CRITERION OF ERROR ( KERR>0 - ERROR , KERR=0 -
C                   NO ERROR )
C     RMN0        - MINIMUM VALUE OF RING WIDTH
C     RMX0        - MAXIMUM VALUE OF RING WIDTH . IF RMN0<RMX0 ,
C                   LIMITS OF RING WIDTHS PLOT ARE [RMN0,RMX0] ,
C                   ELSE LIMITS ARE DETERMINED BY   D R A T R E
C     IMN0        - MINIMUM VALUE OF RING INDEX
C     IMX0        - MAXIMUM VALUE OF RING INDEX . IF IMN0<IMX0 ,
C                   LIMITS OF RING INDICES PLOT ARE [IMN0,IMX0] ,
C                   ELSE LIMITS ARE DETERMINED BY   D R A T R E
C     RMN1        - MINIMUM VALUE OF RING WIDTH ( RESULT OF
C                   D R A T P E )
C     RMX1        - MAXIMUM VALUE OF RING WIDTH ( RESULT OF
C                   D R A T P E )
C     IMN1        - MINIMUM VALUE OF RING INDEX ( RESULT OF
```

```
C                D R A T R E )
C     IMX1       = MAXIMUM VALUE OF RING INDEX ( RESULT OF
C                D R A T R E )
C
C
      XINT=1. $ YINT=2. $ NRW1=600 $ PERCEN=0.05 $ MJ=40B
      NRW=NRW1/2 $ NNV=NRW+20 $ DEL=10.**(-10.)
      IF(NRMAX-NRW) 1,1,2
    1 NRMAX1=NRMAX $ GOTO 3
    2 NRMAX1=NRW
    3 CONTINUE
C     BEGIN PAGE OF PLOTTER
      IF(KPLOT.EQ.1) CALL PAGE(XPAG,YPAG,0,0,0)
      IB=1
      DO 28 IS=1,NN
      VA=NAME(IS)
C     READ TREE-RING WIDTH SERIES VA FROM MAGNETIC TAPE
      CALL ALPROC(RERS,'VAR ',VA,'VAR ',KQ,'VAR ',MJ,'VAR ',KZ0,'VAR ',
     *            KZMAX,'VAR ',KZR,'ARRAY1',RW,NRMAX,'VAR ',NR,8)
C     TREE-RING WIDTH SERIES VA IS IN RW(NR)
      NDEL=NRW-NR
      IF(NDEL.GE.0) GOTO 4
C     TREE-RING SERIES TOO LONG
      KERR=1 $ RETURN
    4 CONTINUE
C     FIND MINIMUM AND MAXIMUM OF TREE-RING WIDTH SERIES VA
      RMN2=RW(1) $ RMX2=RMN2
      DO 5 I=2,NR
      IF(RW(I).LT.RMN2) RMN2=RW(I)
      IF(RW(I).GT.RMX2) RMX2=RW(I)
    5 CONTINUE
C     FIND ABSOLUTE MINIMUM AND MAXIMUM OF ALL TREE-RING WIDTH SERIES
      IF(IS-1) 6,6,7
    6 RMN1=RMN2 $ RMX1=RMX2 $ GOTO 8
    7 IF(RMN2.LT.RMN1) RMN1=RMN2
      IF(RMX2.GT.RMX1) RMX1=RMX2
    8 CONTINUE
      IF(KQ1.EQ.0) GOTO 15
      IF(NDEL.EQ.0) GOTO 10
      DO 9 I=1,NDEL
      I1=NR+I $ RW(I1)=0.
    9 CONTINUE
   10 IF(RMN0.GE.RMX0) GOTO 11
      RMN2=RMN0 $ RMX2=RMX0
   11 CONTINUE
      IF(KPLOT) 15,12,14
   12 PRINT 13,VA
   13 FORMAT(////10X,'T R E E - R I N G   S E R I E S',I7,
     *       2X,'W I D T H S'/)
C     DRAW TREE-RING WIDTHS BY PRINTER
      CALL DRARA(RW,NR,RMN2,RMX2,YEAR(IS),XREG,YREG,0.)
      GOTO 15
C     DRAW TREE-RING WIDTHS BY PLOTTER
   14 CALL DRARB(RW,NRW,RMN2,RMX2,YEAR(IS),NAME(IS),IB,
     *           XPAG,YPAG,XREG,YREG,XINT,YINT)
      IB=IB+1
   15 CONTINUE
C     POLYNOMIAL TREE-RING WIDTH CURVE FITTING ( RESULT GROW(NR) )
      CALL FITPOL(RW,NR,GROW,COEF,NDEG,PERCEN,TEMP,NNV)
      IF(NDEG.LT.24) GOTO 16
C     DEGREE OF POLINOMIAL TOO BIG
      KERR=2 $ RETURN
```

```
   16 CONTINUE
C     COMPUTE TREE-RING INDICES AND FIND MINIMUM AND MAXIMUM
C     INDEX FOR SERIES VA
      RW(1)=RW(1)/GROW(1) $ IMN2=RW(1) $ IMX2=IMN2
      DO 17 I=2,NR
      IF(GROW(I).EQ.0.) GROW(I)=DEL $ RW(I)=RW(I)/GROW(I)
      IF(RW(I).LT.IMN2) IMN2=RW(I)
      IF(RW(I).GT.IMX2) IMX2=RW(I)
   17 CONTINUE
C     FIND ABSOLUTE MINIMUM AND MAXIMUM FOR ALL TREE-RING INDICES
      IF(IS-1) 18,18,19
   18 IMN1=IMN2 $ IMX1=IMX2 $ GOTO 20
   19 IF(IMN2.LT.IMN1) IMN1=IMN2
      IF(IMX2.GT.IMX1) IMX1=IMX2
   20 CONTINUE
      IF(KQ2.EQ.0) GOTO 27
      DO 21 I=1,NDEL
      I1=NR+I $ RW(I1)=0.
   21 CONTINUE
      IF(IMN0.GE.IMX0) GOTO 23
      IMN2=IMN0 $ IMX2=IMX0
   23 CONTINUE
      IF(KPLOT) 27,24,26
   24 PRINT 25,VA
   25 FORMAT(///10X,'T R E E - R I N G   S E R I E S',17,
     *       2X,'I N D I C E S'/)
C     DRAW TREE-RING INDICES BY PRINTER
      CALL DRARA(RW,NR,IMN2,IMX2,VEAR(IS),XREG,YREG,0.)
      GOTO 27
C     DRAW TREE-RING INDICES BY PLOTTER
   26 CALL DRARB(RW,NRW,IMN2,IMX2,VEAR(IS),NAME(IS),IB,
     *           XPAG,YPAG,XREG,YREG,XINT,YINT)
      IB=IB+1
   27 CONTINUE
   28 CONTINUE
C     END PAGE OF PLOTTER
      IF(KPLOT.EQ.1) CALL ENDPG(0)
      RETURN
      END
```

```
'INTEGER''PROCEDURE' RMAIL(PM,N,RK);'REAL' RK;'INTEGER' N;'ARRAY' RM;
 'BEGIN''INTEGER' I,J;I:=0;
 'FOR' I:=I+1 'WHILE'((RM[I]'NE'RK)'AND'(I'LE'N))'DO' J:=I;
 'IF'(J'LT'N)'THEN' J:=J+1;RMAIL:=J;
'END'
  'EOP'


'CODE':SKZONA;
'PROCEDURE' RERS(INV,Q,MJ,ZR0,ZRMX,ZP,R,NR);
'REAL' INV;'INTEGER' Q,ZR0,ZRMX,ZR,NR;'BITS' MJ;'ARRAY' R;
  'BEGIN''REAL' INV1,R1;'ARRAY' ZZ[1:1024];
  'INTEGER' ISN,C,ZR01,ZRMX1,Z,NR1,I,I1,J;
  'IF'((ZR'LT'ZP0)'OR'(ZR'GT'ZRMX))'THEN' ZR:=ZR0;
  ISN:=-999;ZR01:=ZR;ZRMX1:=ZRMX;NR:=0;
  'FOR' C:=1,2 'DO' 'IF'(NR'EQ'0)'THEN'
    'BEGIN' Z:=ZR01-1;
    'FOR' Z:=Z+1 'WHILE'((NR'EQ'0)'AND'(Z'LE'ZRMX1))'DO'
      'BEGIN' SKZONA(ZZ,Z,MJ);
      'IF'(ZZ[1]'EQ'ISN)'THEN' ZRMX1:=Z 'ELSE'
        'BEGIN' INV1:=-1;NR1:=0;I:=1;
        'FOR' I1:=I+NR1 'WHILE'((INV1'NE'INV)'AND'
        (INV1'NE'ISN)'AND'(I'LT'1024))'DO'
        'BEGIN' INV1:=ZZ[I];NR1:=ZZ[I+1];I1:=I+1 'END';
        'IF'((INV1'EQ'INV)'AND'(I1+NR1'LT'1024))'THEN'
          'BEGIN' NR:=(NR1-2)/2;
          'FOR' J:=1 'STEP' 1 'UNTIL' NR 'DO'
            'BEGIN'
            'IF'(Q'EQ'0)'THEN' R[J]:=ZZ[I1+J];
            'IF'(Q'EQ'1)'THEN' R[J]:=ZZ[I1+J+NR];
            'IF'(Q'EQ'2)'THEN' R[J]:=ZZ[I1+J]+ZZ[I1+J+NR];
            'END';
          'END';
        'END';
      'END';
    ZP01:=ZR0;ZRMX1:=ZP-1;
    'END';
  'END'
  'EOP'
```

```
      SUBROUTINE DRAPA(R,NR,RMIN,RMAX,VEAR,XREG,VREG,CO)
      DIMENSION R(1)
C
C     SUBROUTINE   D R A P A   IS DESIGNATED TO PLOT FUNCTION R(NR) BY
C     MEANS OF PRINTER .
C
C      R(NR)   - FUNCTION TO BE PLOTTED ,
C      NR      - LENGTH OF FUNCTION R ,
C      RMIN    - MINIMUM VALUE OF FUNCTION R ,
C      RMAX    - MAXIMUM VALUE OF FUNCTION R , LIMITS OF FUNCTION ARE
C                (RMIN,RMAX) . IF RMAX<=RMIN , LIMITS ARE DEFINED BY
C                D R A R A ,
C      VEAR    - ARGUMENT OF R(1) . LIMITS OF ARGUMENT ARE
C                (VEAR,VEAR+NR-1) ,
C      XREG    - WIDTH OF FUNCTION'S REGION ON PAGE (POSITIONS) ,
C      VREG    - HEIGHT OF FUNCTION'S REGION ON PAGE (ROWS) ,
C
      DX=1. $ MPN=15 $ MDW=INT(XREG) $ NV=INT(VREG)
C     FORM ARRAY P(NR*2) , CONSISTING OF TWO FUNCTIONS R AND RR=CO
      DO 1 I=1,NR
      KR=NR+I $ P(KR)=CO
    1 CONTINUE
      CALL DRAC(R,NR,-2,VEAR,DX,RMIN,RMAX,MPN,MDW,NV,KERR)
      RETURN
      END
```

```
      SUBROUTINE DRARB(R,NP,RMIN,RMAX,VEAR,NAME,J,XPAG,VPAG,
     *                 XREG,VREG,XINT,VINT)
      DIMENSION R(NR)
C
C     SUBROUTINE   D R A R B   IS DESIGNATED TO PLOT FUNCTION R(NR)
C     BY MEANS OF PRINTER OR PLOTTER . SUBROUTINES LIMITS,XAXIS,VAXIS,
C     INCLIN ARE FROM FORTRAN PLOTTING PACKAGE   G R A F O R .
C
C     R(NR)      - FUNCTION TO BE PLOTTED
C     NR         - LENGTH OF FUNCTION R
C     RMIN       - MINIMUM VALUE OF FUNCTION
C     RMAX       - MAXIMUM VALUE OF FUNCTION , LIMITS OF FUNCTION ARE
C                  (RMIN,RMAX) . IF RMAX<=RMIN , LIMITS ARE DEFINED BY
C                  D R A R B ,
C     VEAR       - ARGUMENT OF R(1) . LIMITS OF ARGUMENT ARE
C                  (VEAR,VEAR+NR-1) ,
C     NAME       - NAME OF FUNCTION R ( INTEGER NUMBER < 1000000 ) ,
C     J>0        - NUMBER OF REGION ON PAGE ,
C     XPAG       - WIDTH OF PLOTTER'S PAGE (CM) ,
C     VPAG       - HEIGHT OF PLOTTER'S PAGE (CM) ,
C     XREG<XPAG  - WIDTH OF FUNCTION'S REGION ON PAGE (CM) ,
C     VREG<VPAG  - HEIGHT OF FUNCTION'S REGION ON PAGE (CM) ,
C     XINT,VINT  - INTERVALS AMONG REGIONS (CM) .
C
      RMIN1=RMIN $ RMAX1=RMAX $ IF(RMIN.LT.RMAX) GOTO 2
C     DEFINE LIMITS OF R
      RMIN1=R(1) $ RMAX1=R(1)
      DO 1 I=2,NR
      IF(R(I).LT.RMIN1) RMIN1=R(I) $ IF(R(I).GT.RMAX1) RMAX1=R(I)
    1 CONTINUE
    2 CONTINUE
      XINT0=1.5 $ VINT0=1.5 $ JJ=J $ VEAR1=VEAR+NR-1
      XPAD=AXIS10(NR) $ VPAD=AXIS10(RMAX1-RMIN1)
      CALL REGIO1(XPAG,VPAG,XREG,VREG,XINT,VINT,XINT0,VINT0,JJ,NAME)
      IF(JJ.LE.0) RETURN
C     THERE IS NO PLACE FOR REGION J ON PAGE
      CALL LIMITS(VEAR,VEAR1,RMIN1,RMAX1)
      CALL LOADG0(RMIN1,4HVEAR,4,XPAD,10,0,1,'XAXIS')
      CALL LOADG0(VEAR,4HRING,6,VPAD,5,0,3,'VAXIS')
      CALL LOADG0(VEAR,1,0,R,NR,0,2,'INCLIN')
      RETURN
      END
```

```
      SUBROUTINE FITPOL(V,NV,V1,COEF,NDEG,PERCEN,TEMP,NNV)
      DIMENSION V(NNV),V1(NNV),COEF(31),TEMP(NNV,5)
      CALL COPOL(V,NV,COEF,NDEG,V1,PERCEN,TEMP,NNV)
      IF(NDEG.LE.15) GOTO 3
      PRINT 1,NDEG
    1 FORMAT(/10X,'***FITPOL*** DEGREE',I3,' IS TOO HIGH')
      DO 2 IX=1,NV
    2 V1(IX)=0.
      GOTO 6
    3 XP=10.
      DO 5 IX=1,NV
      X=XP*IX $ X1=1. $ V2=COEF(1)
      DO 4 J=2,NDEG
      X1=X1*X $ V2=V2+X1*COEF(J)
    4 CONTINUE
      V1(IX)=V2
    5 CONTINUE
    6 CONTINUE
      RETURN
      END


      SUBROUTINE FITEXP(V,NV,V1,COEF,LOPT)
      DIMENSION V(NV),V1(NV),COEF(7)
      CALL COEXPL(V,NV,V1,COEF,LOPT)
      DO 1 IX=1,NV
      X=IX
      V1(IX)=COEF(1)*EXP(-COEF(2)*X)+COEF(3)*X+COEF(4)
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DRAC(V,NX,NF,X1,DX,VMIN,VMAX,MPW,MDW,NV,KERR)
      DIMENSION V(1),KS(6),KEIL(128)
      DATA(KS=1H ,1HX,1H+,1HO,1H:,1H=)
      NFMAX=5 ◊ MTARP=2 ◊ MPWMIN=15 ◊ NDX=10 ◊ NDV=5 ◊ NCODV=8
      DEL=10.**(-10.) ◊ NS=NFMAX+1
      NXA=NX ◊ NFA=IABS(NF) ◊ DXA=DX ◊ MPWA=MPW ◊ MDWA=MDW ◊ NVA=NV
      IF(NXA.GT.1) GOTO 1 ◊ KERR=1 ◊ RETURN
    1 IF((NF.GE.1).OR.(NF.EQ.-2)) GOTO 2 ◊ KERR=2 ◊ RETURN
    2 IF(NF.GT.NFMAX) NFA=NFMAX
      IF(MPWA.GE.MPWMIN) GOTO 4 ◊ KERR=3 ◊ RETURN
    4 IF(MDWA.GE.1) GOTO 5 ◊ KERR=4 ◊ RETURN
    5 IF(MDWA.LE.129-MPWA) GOTO 6 ◊ KERR=4 ◊ RETURN
    6 IF(NVA.GT.1) GOTO 7 ◊ KERR=5 ◊ RETURN
    7 NGR=((NXA-1)/MDWA)+1 ◊ MDW1=NXA-MDWA*(NGR-1)
      DXX=MDWA*DXA ◊ DXX1=DXX-DXA ◊ VMINA=VMIN ◊ VMAXA=VMAX
      IF(VMINA.LT.VMAXA) GOTO 10
      NXF=NXA*NFA ◊ VMINA=V(1) ◊ VMAXA=VMINA
      DO 9 I=2,NXF
      IF(VMINA.LT.V(I)) GOTO 8 ◊ VMINA=V(I)
    8 IF(VMAXA.GE.V(I)) GOTO 9 ◊ VMAXA=V(I)
    9 CONTINUE
      IF(VMINA.LT.VMAXA) GOTO 10 ◊ KERR=8 ◊ RETURN
   10 DVV=(VMAXA-VMINA)/NVA
      DO 27 IGR=1,NGR
      IGR1=IGR-1 ◊ IXP=IGR1*MDWA+1 ◊ XP=IGR1*DXX+X1
      IF(IGR-NGR) 11,12,12
   11 IXG=IXP+MDW-1 ◊ GOTO 13
   12 IXG=IXP+MDW1-1
   13 IDV=-1 ◊ VV2=VMAXA ◊ NFB=NF
      DO 25 IV=1,NVA
      VV1=VV2-DVV ◊ IDV=IDV+1
      IF((IV.EQ.NVA).AND.(NF.EQ.-2)) NFB=1
      IF(IDV-NDV) 15,16,16
   15 KQ=0 ◊ GOTO 17
   16 KQ=1 ◊ IDV=0
   17 CONTINUE
      IF(IV-1) 18,20,18
   18 IF(NVA-IV) 21,19,21
   19 VV1=VV1-DEL
   20 KQ=2
   21 CONTINUE
      CALL ACEIL0(MPWA,MDWA,VV2,NCODV,XP,DXA,NDX,KQ,KEIL,KERR1)
      IF(KERR1.EQ.0) GOTO 22 ◊ KERR=6 ◊ RETURN
   22 CONTINUE
      DO 23 IX=IXP,IXG
      CALL ACEIL1(V,NXA,NFB,IX,VV1,VV2,MPWA,MDWA,KS,NS,KEIL,KERR1)
      IF(KERR1.EQ.0) GOTO 23 ◊ KERR=7 ◊ RETURN
   23 CONTINUE
      PRINT 24,KEIL
   24 FORMAT(128A1)
      VV2=VV1
   25 CONTINUE
      CALL ACEIL0(MPWA,MDWA,VMINA,NCODV,XP,DXA,NDX,3,KEIL,KERR1)
      PRINT 24,KEIL
      DO 27 I=1,MTARP
      PRINT 26
   26 FORMAT(/)
   27 CONTINUE
      KERR=0
      RETURN
      END
```

```
      SUBROUTINE ACEILO(MPW,MDW,VV,NCODV,X1,DX,NDX,KQ,KEIL,KERR)
      DIMENSION KEIL(128),KCOD(5),KC(30)
      DATA(KSVM1=1H ),(KSVM2=1H1),(KSVM3=1H0),(KSVM4=1H-)
      NCODX=8 ◊ MPWA=MPW ◊ NDXA=NDX ◊ NIS=MPWA+MDW-1
      IF(MPWA.GT.1) GOTO 1 ◊ KERR=1 ◊ RETURN
    1 IF(MDW.GE.1) GOTO 2 ◊ KERR=2 ◊ RETURN
    2 IF(NIS.LE.128) GOTO 3 ◊ KERR=2 ◊ RETURN
    3 IF(NCODV.GE.5) GOTO 4 ◊ KERR=3 ◊ RETURN
    4 IF(NCODV.LE.MPWA-6) GOTO 5 ◊ KERR=3 ◊ RETURN
    5 IF(NDXA.GE.10) GOTO 6 ◊ KERR=4 ◊ RETURN
    6 IF(KQ.GE.0) GOTO 7 ◊ KERR=5 ◊ RETURN
    7 IF(KQ.LE.3) GOTO 8 ◊ KERR=5 ◊ RETURN
    8 DO 9 IS=1,128
    9 KEIL(IS)=KSVM1
      IF(KQ.GT.1) GOTO 10
      KEIL(MPWA)=KSVM2 ◊ KEIL(NIS)=KSVM2
   10 IF(KQ.NE.2) GOTO 14 ◊ IDX=NDXA-1
      DO 13 IS=MPWA,NIS
      IDX=IDX+1
      IF(IDX-NDXA) 11,12,11
   11 KEIL(IS)=KSVM4 ◊ GOTO 13
   12 KEIL(IS)=KSVM3 ◊ IDX=0
   13 CONTINUE
   14 IF(KQ.EQ.0) GOTO 17
      CALL FORUN(VV,NCODV,NCV,KCOD,KERR1)
      IF(KERR1.NE.0) GOTO 17
      DECODE(30,15,KCOD) KC
   15 FORMAT(30A1)
      IS1=MPWA-NCV-5
      DO 16 I=1,NCV
      IS=IS1+I ◊ KEIL(IS)=KC(I)
   16 CONTINUE
   17 CONTINUE
      IF(KQ.NE.3) GOTO 20
      DXX=NDXA*DX ◊ X=X1-DXX
      DO 19 IS=MPWA,NIS,NDXA
      X=X-DXX
      CALL FORUN(X,NCODX,NCX,KCOD,KERR1)
      IF(KERR1.NE.0) GOTO 19
      DECODE(30,15,KCOD) KC
      IS1=IS-NCX/2 ◊ J1=IS1-1
      DO 18 J=1,NCX
      J1=J1+1 ◊ KEIL(J1)=KC(J)
   18 CONTINUE
   19 CONTINUE
   20 KERR=0
      RETURN
      END
```

```
      SUBROUTINE ACEIL1(V,NX,NF,IX,VV1,VV2,MPW,MDW,KS,NS,KEIL,KERR)
      DIMENSION V(1),KS(NS),KEIL(128)
      IF(NS.GE.2) GOTO 1 $ KERR=1 $ RETURN
    1 IF((NF.LE.NS-1).AND.(NF.GT.0).OR.(NF.EQ.-2)) GOTO 2
      KERR=2 $ RETURN
    2 IF(VV1.LE.VV2) GOTO 3 $ KERR=3 $ RETURN
    3 K=(IX-1)/MDW $ IX1=IX-MDW*K $ IE=MPW*IX1-1
      IF(NF*2) 12,7,4
    4 IXF=IX-NX $ IS=0
      DO 6 IF=1,NF
      IXF=IXF+NX
      IF(V(IXF).GT.VV2) GOTO 6 $ IF(V(IXF).LE.VV1) GOTO 6
      IS=IS+1 $ IF(IS.EQ.1) GOTO 5
      KEIL(IE)=KS(NS) $ KERR=0 $ RETURN
    5 KEIL(IE)=KS(IF)
    6 CONTINUE
      GOTO 12
    7 A1=V(IX) $ IX1=IX+NX $ A2=V(IX1)
      IF(A1-A2) 8,8,9
    8 VV1A=A1 $ VV2A=A2 $ GOTO 10
    9 VV1A=A2 $ VV2A=A1
   10 IF((VV2.GE.VV1A).AND.(VV1.LT.VV2A)) KEIL(IE)=KS(1)
   12 KERR=0
      RETURN
      END
```

```
      SUBROUTINE FORUN(X,LM,L,KXCOD,KERR)
      DIMENSION KXCOD(5),KFF(2),KFE(2),KXCOD1(30),KXCOD2(5)
      DATA (KFF=8H(F00.00)),(KFE=8H(E00.00)),(KF1=6H(30A1))
      IF(LM.GE. 2) GOTO 1 ◊ KERR=1 ◊ RETURN
    1 IF(LM.LE.20) GOTO 2 ◊ KERR=1 ◊ RETURN
    2 X1=X ◊ LM1=LM
      IF(X1) 3,4,4
    3 ISIX=1H- ◊ GOTO 5
    4 ISIX=1H+
    5 NSA=DIGNUI(X1) ◊ NTA=DIGNUF(X1) ◊ NTA1=DIGNUL(Y1)
      IF(NSA) 6,6,7
    6 KQS=1 ◊ NSA=1 ◊ GOTO 8
    7 KQS=0
    8 LMIN=1+NSA+KQS+(NTA1+2)
      IF((LMIN.GT.LM1).AND.(X1.NE.0.)) GOTO 11
C   FORMAT OF  X  IS  (F L1.NTB)
      NTB=LM1-NSA-2
      IF((NTB.LT.0).OR.(NTB.LT.NTA1+1)) NTB=0
      IF(NTB.GT.NTA) NTB=NTA
      L1=1+NSA+NTB ◊ IF(NTB.GT.0) L1=L1+1
      CALL FORVA1(L1,NTB,KFF,X1,KXCOD2,KERR1)
      IF(KERR1.EQ.0) GOTO 10 ◊ KERR=3 ◊ RETURN
   10 LTG=L1
      DECODE(30,KF1,KXCOD2) KXCOD1
      GOTO 19
C   FORMAT OF  X  IS  (E L1.NTB)
   11 IF(LM1.GE.4) GOTO 12 ◊ KERR=2 ◊ RETURN
   12 L1=LM1+1
   13 NTB=L1-6 ◊ IF(NTB.GT.0) GOTO 14 ◊ NTB=0 ◊ L1=L1-1
   14 CALL FORVA1(L1,NTB,KFE,X1,KXCOD2,KERR1)
      IF(KERR1.EQ.0) GOTO 15 ◊ KERR=4 ◊ RETURN
   15 DECODE(30,KF1,KXCOD2) KXCOD1
C  CHECK ABS.VALUE OF EXPONENT (MORE OR LESS 10 )
      IF(KXCOD1(L1-1).EQ.1H0) GOTO 18
      IF(L1-LM1) 17,17,16
   16 L1=LM1 ◊ GOTO 13
   17 LTG=L1-3 ◊ GOTO 19
   18 L1=L1-1 ◊ LTG=L1-2 ◊ KXCOD1(L1)=KXCOD1(L1+1)
   19 CONTINUE
C  CHECK NUMBER OF TYPE  9999...9
      IF(KXCOD1(1).EQ.1H-) GOTO 23
      IF(KXCOD1(1).EQ.1H+) GOTO 23
      IF(KXCOD1(1).EQ.1H ) GOTO 22
      LTG=LTG+1 ◊ L1=L1+1 ◊ LL=L1
   20 KXCOD1(LL)=KXCOD1(LL-1)
      IF(LL-2) 21,22,21
   21 LL=LL-1 ◊ GOTO 20
   22 KXCOD1(1)=ISIX
   23 CONTINUE
      IF(NTB.LE.0) GOTO 31
C  CHECK LAST DIGITS OF FRACTION ( 0 OR 1-9 )
      LTP=LTG-NTB ◊ KTQ=0 ◊ LL=LTG
   26 IF(LL-LTP) 25,26,25
   25 IF(KXCOD1(LL).NE.1H0) GOTO 27
      LL=LL-1 ◊ GOTO 24
   26 KTQ=1
   27 NTB1=LL-LTP ◊ KSHIFT=NTB-NTB1+KTQ
      IF(KSHIFT.LE.0) GOTO 31 ◊ IF(LTG.EQ.L1) GOTO 30
      LL=LTG+1
   28 LL1=LL-KSHIFT ◊ KXCOD1(LL1)=KXCOD1(LL
      IF(LL-L1) 29,30,29
   29 LL=LL+1 ◊ GOTO 28
```

```
   30 L1=L1-KSHIFT
   31 L=L1
      L1=L1+1 $ DO 32 K=L1,30
   32 KXCOD1(K)=1H
      ENCODE(30,KF1,KXCOD) KXCOD1
      KERR=0
      RETURN
      END


      SUBROUTINE FORVA1(N,M,KF,X,KXCOD,KERR)
      DIMENSION KF(2),KXCOD(5),KF1(8),KXCOD1(30),KK(2)
      DECODE(8,20,KF) KF1
      ENCODE(2,21,N2) KF1(3),KF1(4)
      DECODE(2,22,N2) N1
      IF((N.LT.0).AND.(M.LT.0)) GOTO 11
      NN=N $ MM=M
      IF(NN.LT.0) GOTO 1
      N1=NN
      ENCODE(2,22,N2) N1
      DECODE(2,21,N2) KK
      KF1(3)=KK(1) $ KF1(4)=KK(2)
    1 CONTINUE
      IF(MM) 2,3,3
    2 ENCODE(2,21,M2) KF1(6),KF1(7)
      DECODE(2,22,M2) M1
      GOTO 4
    3 M1=MM
      ENCODE(2,22,M2) M1
      DECODE(2,21,M2) KK
      KF1(6)=KK(1) $ KF1(7)=KK(2)
    4 IF(N1.LE.30) GOTO 5 $ KERR=1 $ RETURN
    5 IF(KF1(2).NE.1HF) GOTO 6
      IF(N1.GE.M1+2) GOTO 10 $ KERR=2 $ RETURN
    6 IF(M1) 7,7,8
    7 KQ=0 $ GOTO 9
    8 KQ=1
    9 IF(N1.GE.M1+5+KQ) GOTO 10 $ KERR=3 $ RETURN
   10 ENCODE(8,20,KF) KF1
   11 ENCODE(N1,KF,KXCOD) X
      KERR=0
   20 FORMAT(8A1)
   21 FORMAT(2A1)
   22 FORMAT(I2)
      RETURN
      END


      INTEGER FUNCTION DIGNUI(X)
      DIMENSION MCOD(5),MI(30)
      IX=IABS(IN1(X)) $ KDI=0
      ENCODE(28,1,MCOD) IX
    1 FORMAT(I28)
      DECODE(28,2,MCOD) MI
    2 FORMAT(28I1)
      DO 3 I=1,28
      IF(MI(I).EQ.0) GOTO 3 $ KDI=29-I $ GOTO 4
    3 CONTINUE
    4 DIGNUI=KDI
      RETURN
      END
```

```
      INTEGER FUNCTION DIGNUF(X)
      DIMENSION MCOD(5),MF(30)
      FX=ABS(X-INT(X)) ◊ KDF=0
      ENCODE(29,1,MCOD) FX
    1 FORMAT(F29.28)
      DECODE(29,2,MCOD) MF
    2 FORMAT(1X,28I1)
      DO 3 I=1,25
      II=26-I ◊ IF(MF(II).EQ.0) GOTO 3 ◊ KDF=II ◊ GOTO 4
    3 CONTINUE
    4 DIGNUF=KDF
      RETURN
      END


      INTEGER FUNCTION DIGNUL(X)
      DIMENSION MCOD(5),ML(30)
      FX=ABS(X-INT(X)) ◊ KDL=25
      ENCODE(29,1,MCOD) FX
    1 FORMAT(F29.28)
      DECODE(29,2,MCOD) ML
    2 FORMAT(1X,28I1)
      DO 3 I=1,25
      IF(ML(I).EQ.0) GOTO 3 ◊ KDL=I-1 ◊ GOTO 4
    3 CONTINUE
    4 DIGNUL=KDL
      RETURN
      END
```

```
      SUBROUTINE REGION(XPAG,YPAG,XREG,YREG,XINT,YINT,XINTO,YINTO,
     *J,TITLE)
      IF(J.LE.0) GOTO 4
      NY=(YPAG-2.*YINTO+YINT)/(YREG+YINT)
      NX=(XPAG-2.*XINTO+XINT)/(XREG+XINT)
      IX=(J-1)/NY ◊ IF(IX.LT.NX) GOTO 1 ◊ J=-J ◊ GOTO 4
    1 IY=J-IX*NY ◊ PX=XINTO+IX*(XREG+XINT)
      PY=YPAG-YINTO+YINT-IY*(YREG+YINT)
      CALL PEGION(PX,PY,XREG,YREG,0.0,0)
      IF(ABS(TITLE).GE.1000000.) GOTO 4
      PX=PX+XREG-1.5 ◊ PY=PY+YREG+0.3
      K=TITLE*100. ◊ IF(IABS(K-100*(K/100))) 2,2,3
    2 CALL NUMBER(PX,PY,0.4,TITLE,0.0) ◊ GOTO 4
    3 CALL NUMBER(PX,PY,0.4,TITLE,2,0)
    4 CONTINUE
      RETURN
      END


      FUNCTION AXIS10(T)
      T1=T ◊ AX=1. ◊ IF(T.EQ.0.) GOTO 5 ◊ IF(T1-1.) 1,3,3
    1 AX=0.1
    2 T1=T1*10. ◊ IF(T1.GE.1.) GOTO 5 ◊ AX=AX*0.1 ◊ GOTO 2
    3 AX=1.
    4 T1=T1*0.1 ◊ IF(T1.LE.1.) GOTO 5 ◊ AX=AX*10. ◊ GOTO 4
    5 AXIS10=AX
      RETURN
      END


      SUBROUTINE INDEX(Y,NY,Y1,LOPT,NDEG,COEF,TEMP,NNY)
      DIMENSION Y(NNY),Y1(NNY),COEF(31),TEMP(NNY,5)
      NDEG=4 ◊ KK=1 ◊ IF((LOPT.LT.0).OR.(LOPT.EQ.4))GOTO 20 ◊ KK=0 ◊ PER=0.05
      IF((LOPT.GE.0).AND.(LOPT.LE.3)) CALL FITEXP(Y,NY,Y1,COEF,LOPT)
      IF(LOPT.EQ.5) CALL FITPOL(Y,NY,Y1,COEF,NDEG,PER,TEMP,NNY)
      IF(LOPT.GE.6) CALL FITMEA(Y,NY,Y1,LOPT)
      DO 1 J=1,NY
C     Y1(J)=Y(J)/Y1(J) ◊ IF(Y1(J).LT.0) KK=1
    1 CONTINUE
    2 IF(KK.EQ.1) NDEG=-NDEG
      RETURN
      END


      SUBROUTINE FITMEA(Y,NY,Y1,IVID)
      DIMENSION Y(NY),Y1(NY)
      M1=IVID/2
      DO 2 I=1,NY
      IP=I-M1 ◊ IG=IP+IVID-1 ◊ IF(IP.LE.0) IP=1 ◊ IF(IG.GT.NY) IG=NY
      M=IG-IP+1 ◊ S=0
      DO 1 J=IP,IG
    1 S=S+Y(J)
      Y1(I)=S/M
    2 CONTINUE
      RETURN
      END
```