V. DUMITRU

The International Computing Centre
United Nations — New York

A System for Interpolation and Insertion Program* Written in FORTRAN IV for I. B. M. 7044

The regression curve — linear or exponential — underlying the interpolation of statistical data depends on the specific character of the phenomenon analyzed. The choice for the most appropriate type of curve and the insertion proper, achieved by means of an electronic computer programmed in FORTRAN IV, offers an efficient and elegant solution to the problem.

A. The program provides the interpolation of data and the insertion of missing data on the National Accounts tape, using a linear, $y = a_1x + a_0$, or an exponential, $y = b_0b_1^x$, regression curve.

The program chooses between these two methods, by comparing the standard errors of the estimates.

The output includes the initial series and the interpolated and inserted series, in the order requested by the user, as well as the standard errors of estimates. An indication about the nature of new computed data is given in the first two words: "L. NEW INSERT" and "E. NEW INSERT" meaning data computed with a linear or an exponential function. "NN" and "NP" give the number of points below and above the regression line and "SYX" and "R" note the standard errors of estimates and the correlation coefficients.

The average time to compute one series on I.B.M. 7044, is 1 second.

The program uses the following I.C.C. subroutines: "I O L M T 7", "S O R 7", "S R C 7", "K D I F R 7", "C H I N 7", "C R B U F" and "B C D C R", written by Mr. H. Zigelman and given in Annex II.

---

* This project was suggested by Mr. William Mac Kay, chief system analysis and programming branch I.C.C. — United Nations, New York.

B. Selection of options and data format

I. Option cards

1. Number of reels and output option card.

This card must be the first card following ENTRY card. The number of reels must be punched in columns 1—3, right justified.

The output is a sysout, binary output on C 6 (output file 4) and on cards. If "1" is punched in columns "4", "6" or "8" it will suppress the output on cards, on sysout and the binary output, respectively.

2. Country code and table code cards.

Each table that is to be interpolated must be on this card. Up to 9 tables for each country can be computed and for a country all tables must be on the same card which will have the country code in columns 1—4.

Column 5 is a blank column and beginning with column 6 there follow the table codes, punched in a 4-column fields, right justified.

The last card of country and table code cards must be a card punched "END 1" in columns 1—4.

3. The order of words in the record and covered years card.

In order to make an easy change in the program, the order of words in the record and the number of years for which the series is to be extended are punched on this card. For the actual description of National Accounts tape, the numbers 1 till 11, in natural order, must be punched in 11 six-column fields, beginning with column 1, right justified. The last two digits of years to which the series is to be extended, must be punched in the 12th and 13th six-column fields.

II. Input and output data

1. The input data is the "National Accounts" data as processed by the United Nations — Statistical Office. The format of this data is described in Annex I.

The program assumes that the data on the tape are in the following sorted order, by:

a. country code [word 3, characters (1—4)]
b. table code [word 4, characters (1—3)]
c. first item code [word 8, characters (1—4)]
d. second item code [word 9, characters (1—4)]
e. series [word 7, character 3]
f. fiscal year [word 7, characters (5, 6)]

At the end of program samples of output are given.

C. Operating instructions.

I. Input data tape on B 5 (logical tape 1).

II. Output on:

1. Sysout
2. binary output on C 6 (output file 4)
3. cards

National Accounts — Tape Description

by

William Mac-Kay

Densitiy of Recording 800 bits per inch

Tape Contains 3 Files

1. Header label
   Tape mark
2. Data file
   Tape mark

Tape Data Sequence

Country	(word 3	character	1—4)
by Table No.	(„ 4	„	1—3)
by Base Year	(„ 5	„	1—2)
by "Word 10" Identifier	(„ 6	„	1)
by Index Year	(„ 6	„	2—3)
by First Item Code	(„ 8	„	1—4)
by Second Item Code	(„ 9	„	1—4)
by Series	(„ 7	„	3)

National Accounts. Tape Record Description. Data Records

Records are a standard = 11 words in length
Block Size = 120 records

Word No.	*Character Position*	*Format*	*Contents*
1			Zero
2	1—3	A	Source
3	1—4	A	Country Code
4	1—3	A	Table Code
5	see note 1	A	Base Year
6	1	A	"Word 10" Identifier
6	2—3	A	Index Year
6	4	A	Currency Code
6	5—6	A	Exponent
7	1—2	A	History
7	3	A	Series
7	4	I	Type of Fiscal Year
7	5—6	I	Fiscal Year
8	1—4	A	First Item Code
9	1—4	A	Last Item Code
10	1—6	F	Amount, etc.
11	1—6	A	Date

NOTE. Base Year is in the first 2 character positions of this word (e.g. 60|66| except where the Base Year is Zero, where the code is 00|00|66

Description of Record Contents

Record Content	*Location* Word-Character	*Format*	*Description*
1 "Word 1"	1—1—6		Zeros
2 Source	2—1,2,3	A	An indicator of the source of the data

Source document	1st character	2nd&3rd character
1. NAS Questionnaire	*Q*	Edition-Year of Reported Data
2. Coding sheet based on questionnaire	*S*	
3. Publication	*P*	Serial No. of Publication
4. Communication	*C*	Zeros

	2 — 4, 5, 6	A	Blanks — Octal 60's
3 Country Code	3 — 1, 2, 3, 4	A	Character 1—3 = Alphabetic Serial Number Character 4 = Suffix for Expansion (Range 201—385 plus 999)
	3 — 5, 6	A	Blanks — Octal 60's
4 Table Code	4 —1, 2, 3	A	Three digit code, Range 100—999
	4 — 4, 5, 6	A	Blanks — Octal 60's
5 Base Year	5 — 1, 2	A	Last two digits of the year at whose price or cost levels the values in the table are expressed
	5 — 3, 4, 5, 6	A	Blanks Octal 60's

NOTE. If the data are current, only characters 5,6 are blanks; characters 1,2,3,4 are zeros.

6 "Word 10" Identifier	6 — 1	A	Code used to indicate if the data in "Word 10" relate to "Amounts", "Indices" or "Percentages" *P* = Percentage with no decimal place *Q* = " " one " " *X* = Index " no " " *Y* = " " one " " *A* = Amount.

7 Index Year	6 — 2, 3	A	The "Base Year" for Index Data This code is always zeros, except where the "Word 10" identifier is *X* or *Y*
8 Currency Code	6 — 4	A	Indicator of whether national currency or US $ used to express amounts 1 = National Currency 2 = US $
9 Exponent	6 — 5,6	A	The power of 10 which must be applied to the amount in "Word 10" to express it in terms of the basic currency unit of the reporting country. Where the "Word 10" identified (see above) is other than A, i.e. where it identifies either an index or a percentage, the exponent code is zero.
10 History	7 — 1,2	A	The data may originate from the data card or, when missing, data could be generated during a checking run. The history code indicates generated or "calculated" data with the code "CO"; non-generated or original data are coded "blank, 0".
11 Series	7 — 3	A	A series is a group of fiscal years within which each item (entry on table) is comparable. The series are numbered from 1 to 9.
12 Type of Fiscal Year	7 — 4	A	This indicates the starting or ending date of the Fiscal Year 0 = Fiscal Year. Western Calendar Year 1 = Fiscal Year starting March 21 2 = Fiscal Year starting April 1 3 = Fiscal Year starting July 1 4 = Fiscal Year ending September 30 5 = Fiscal Year ending March 31
13 Fiscal Year	7 — 5,6	I	Year to which the data relate

14 First Item Code	8 — 1, 2, 3, 4	A	The code identifying each entry,or aggregate of entries, in a Table.
	8 — 5, 6	A	Blanks — Octal 60's
15 Last Item Code	9 — 1, 2, 3, 4	A	This code is normally zeros, and the presence of any other code in this word means that no separate value is available for the first item code; it is therefore given as part of a total for all item numbers between the first and second item code
16 Amount, etc.	1 — 6	F	This word is used for the value of entries either as Amounts, as Percentages, or as Indices. The content being identified by the "Word 10" identifier.
17 Date	1 — 6	A	Day, Month, Year

Exponent Codes Used with Percentages and Indices

Graphic	*Card Code*	*Meaning*			
blank	blank	Percentage — no decimal			
M	11—4	Percentage — one decimal			
=	8—3	Index = 100	1948	No	decimal
.	8—4	" "	1949	"	"
+	12	" "	1950	"	"
A	12—1	" "	1951	"	"
B	12—2	" "	1952	"	"
C	12—3	" "	1953	"	"
D	12—4	" "	1954	"	"
E	12—5	" "	1955	"	"
F	12—6	" "	1956	"	"
G	12—7	" "	1957	"	"
H	12—8	" "	1958	"	"
I	12—9	" "	1959	"	"
·	12—8—3	" "	1960	"	"
)	12—8—4	" "	1961	"	"
—	11	" "	1962	"	"
J	11—1	" "	1963	"	"
K	11—2	" "	1964	"	"
L	11—3	" "	1965	"	"
N	11—5	" "	1948	One	"
∅	11—6	" "	1949	"	"
P	11—7	" "	1950	"	"
Q	11—8	" "	1951	"	"
R	11—9	" "	1952	"	"
$	11—8—3	" "	1953	"	"
*	11—8—4	" "	1954	"	"
/	0—1	" "	1955	"	"
S	0—2	" "	1956	"	"
T	0—3	" "	1957	"	"
U	0—4	" "	1958	"	"
V	0—5	" "	1959	"	"
W	0—6	" "	1960	"	"
X	0—7	" "	1961	"	"
Y	0—8	" "	1962	"	"
Z	0—9	" "	1963	"	"
;	0—8—3	" "	1964	"	"
(	0—8—4	" "	1965	"	"

FORTRAN SOURCE LIST

```
INT. AND INS. SYST.
     SOURCE STATEMENT

BFTC IIS-VD LIST, REF
          ** A SYSTEM FOR INTERPOLATION AND INSERTION
                PROGRAM **
          WRITTEN IN FORTRAN IV FOR IBM 7044
          ** THE INTERNATIONAL COMPUTING CENTRE, UNITED
                NATIONS-NEW YORK**.
**  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **
      DIMENSION CORE (1100), ARRAY (11), SKIP (10, 100), CREC (11)
      EQUIVALENCE (IX, X)
    DATA QMASK1/0777777770000/
    DATA QMASK2/0777777000000/
    DATA QMASK3/0777700000000/
    DATA QMASK4/0000077000000/
    DATA QMASK5/0000000007777/
    DATA QMASK6/0000000770000/
    DATA QMASK7/0770000000000/
    DATA ANEWL/6HL.NEW/
    DATA ANEWE/6HE.NEW/
    DATA AINS/6HINSERT/
    DATA A/6HA00000/
    DATA C/6HC00000/
    DATA ENDI/4H9999/
    DATA SYEAR/0000000000074/
    NEND = 0
    NRD = 0
**  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **
      READ THE CONTROL CARDS
      READ (5, 5) NREEL, NC, NS, NT.
    5 FORMAT (4I2)
      IS = 0
   10 IS = I S + 1
      READ (5, 11) (SKIP (J, IS), J = 1, 10)
   11 FORMAT (A4,1X, 9(1X, A3))
      IF(SKIP(1, IS). EQ. END1) GO TO 20
      GO TO 10
   20 IS = IS - 1
      READ (5, 25) NO1, NO2, NO3, NO4, NO5, NO6, NO7, NO8, NO9,
      NO10, NO11, IA, LA
   25 FORMAT (13(3X, I3))
      FCTRY = SKIP (1, IS)
   50 CALL CSORT (SKIP (1,1) IS, 10, 1)
      CALL SET1 (1, 2, 1)
```

```
      IF(NT.EQ.1) GO  TO 60
      CALL SET1 (4, 4,—1)
   60 CONTINUE
      WRITE (6, 85) NO1, NO2, NO3, NO4, NO5, NO6, NO7, NO8, NO9,
      NO10, NO11, IA, LA
   85 FORMAT (1HO, 21HNRI (I.EQ.1 TO 11) ARE, 13 (3X, I3))
      GO TO 2000
C ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
  100 CALL RRECI (ARRAY, 1, NEND, NREEL)
      TEMP = AND (QMASK7, ARRAY (NO6))
      IF (TEMP.NE.A) GO TO 100
      TEMP = AND (QMASK7, ARRAY (NO7))
      IF(TEMP.EQ.C) GO TO 100
      TEMP = AND (QMASK3, ARRAY(NO5))
      IF(TEMP. NE.0.0) GO TO 100
      NRD = NRD + 1
      IF(NRD.EQ.1) GO TO 200
      IF(NEND.EQ.1) GO TO 1000
      IF(KDIFR(TCTRY, FCTRY)) 101, 101, 1000
  101 IF(ARRAY(NO3).EQ.STCTRY) GO TO 100
  200 TCTRY = ARRAY(NO3)
      TTABL = ARRAY(NO4)
      TBAYR = AND (QMASK3, ARRAY(N05))
      PITEM = ARRAY(NO8)
      UITEM = ARRAY(NO9)
C     T = AND(ARRAY(NO7), QMASK5)                    FOR TEST
C     IF(T.EQ.SYEAR) GO TO 100                       FOR TEST
        IF((N1.GE.22).AND.((TCTRY.NE.WTCTRY).OR.(TTABL.NE.
                    WTTABL).OR.
     1(PI TEM.NE.WPITEM).OR.(UITEM.NE.WUITEM).OR.(TBAYR.NE.
                    WTBAYR)) .
     2GO  TO  1000
      IF(TC TRY.EQ.WTCTRY.AND.TTABL.EQ.WTTABL) GO TO 250
      IF(TC TRY. EQ. WTCTRY.AND.TTABL. NE.WTTABL) GO TO 225.
  201 CALL SRCH (TCTRY, SKIP (1,1), IS, 10, ITX)
      IF(ITX) 220, 210, 220
  210 STCTRY = TCTRY
      GO TO 100
  220 WTCTRY = TCTRY
  225 CONTINUE
```

```
      DO 230 J = 2, 10
      IF(SKIP(J,I T X). EQ. TTABL) GO TO 240
  230 CONTINUE
      GO TO 100
  240 CONTINUE
      WTTABL = TTABL
  250 CONTINUE
      IF((PITEM.EQ.WPITEM).AND.(UITEM.EQ.WUITEM)) GO TO 300
      WPITEM = PITEM
      WUITEM = UITEM
      N1 = 0
  300 IF(AND(QMASK1, ARRAY(NO7)).EQ.SYEAR) GO TO 100
      DO 350 J = 1,11
      N1 = N1 + 1
      CORE (N1) = ARRAY(J)
  350 CONTINUE
      GO TO 100
C ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
 1000 N = 0
      N2 = N1/11
      WRITE (6, 1001)
 1001 FORMAT (1H1)
      IF (NS.EQ.1) GO TO 1100
      DO 1100 J1 = 1, N2
      J2 = J1 — 1
      K = 11*J2
      KPNO1 = K + NO1
      KPNO2 = K + NO2
      KPNO3 = K + NO3
      KPNO4 = K + NO4
      KPNO5 = K + NO5
      KPNO6 = K + NO6
      KPNO7 = K + NO7
      KPNO8 = K + NO8
      KPNO9 = K + NO9
      KPNO10 = K + NO10
      KPNO11 = K + NO11
      CREC(1) = CORE(KPNO1)
      CREC(2) = CORE(KPNO2)
      CREC(3) = CORE(KPNO3)
      CREC(4) = CORE(KPNO4)
      CREC(5) = CORE(KPNO5)
```

```
      CREC(6) = CORE(KPNO6)
      CREC(7) = AND(CORE(KPNO7), QMASK1)
      X = AND(CORE(KPNO7), QMASK5)
      X = IX
      NY = X
      CREC(8) = CORE(KPNO8)
      CREC(9) = CORE(KPN09)
      CREC(10) = CORE(KPNO10)
      CREC(11) = CORE(KPNO11)
      WRITE(6, 1050) (CREC(J), J = 1,7), NY, (CREC(J), J = 8, 11)
1050  FORMAT(1H, 6(1X,A6)1X, A4,I3,2(1X,A6), 1X, E14.7, 1X, A6)
1100  CONTINUE
      DO 1300 J1 = 1, N2
      J2 = J1 — 1
      K = 11*J2
      KPNO10 = K + NO10
      KPNO7 = K + NO7
        IF(CORE(KPNO10).LT.0.0001) GO TO 1300
      N = N + 1
      X = AND(CORE(KPNO7),QMASK5)
      X = IX
      Y = CORE(KPNO10)
      XY = X*Y
      X2 = X*X
      SX = SX + X
      SY = SY + Y
      SXY = SXY + XY
      SX2 = SX2 + X2
1300  CONTINUE
      VN = N
      XM = SX/VN
      SXT2 = SX*SX
      YM = SY/VN
      DNUM = VN*SX2 — SXT2
      IF((ABS(DNUM)).LT.0.0001) GO TO 1850
      AO = (SY*SX2 — SX*SXY)/DNUM
      A1 = (VN*SXY — SX*SY)/DNUM
      DO 1500 J1 = 1, N2
      J2 = J1 — 1
      K = 11*J2
      KPNO7 = K + NO7
      KPNO10 = K + NO10
      X = AND(CORE(KPNO7), QMASK5)
      X = IX
      YE = A1*X + AO
      IF(CORE(KPNO10).LT.0.0001) GO TO 1500
      XMXM = X — XM
```

```
      XMXM2 = XMXM*XMXM
      Y = CORE(KPNO10)
      YMYE = Y — YE
      YMYE2 = YMYE*YMYE
      YEMYM = YE — YM
      YEMYM2 = YEMYM*YEMYM
      YMYM = Y — YM
      YMYM2 = YMYM*YMYM
      P1 = XMXM*YMYM
      IF(YMYE) 1350, 1370, 1360
 1350 NN = NN + 1
      GO TO 1370
 1360 NP = NP + 1
 1370 CONTINUE
      SYMYE = SYMYE + YMYE2
      SYEMYM = SYEMYM + YEMYM2
      SYMYM = SYMYM + YMYM2
      SP1 = SP1 + P1
      SXMXM2 = SXMXM2 + XMXM2
 1500 CONTINUE
      SYX = SQRT(SYMYE/VN)
      PS2 = SXMXM2*SYMYM
      RPS2 = SQRT(PS2)
      R = SP1/RPS2
      WRITE (6, 1600) NN, NP, SYX, R, A1, AO
 1600 FORMAT (1HO, 1X, 5HNN IS, 1X, I4, 1X, 5HNP IS, 1X, I4, IX, 7HSYX
      IS, 1X, E14.
      17, 1X5HR IS, 1X, F9.4, 1X, 6HA1 IS, 1X, E14.7, 1X, 6HAO IS,
      1X, E14.7)
      WRITE(6, 1620)
 1620 FORMAT (1HO)
C ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
 4000 CONTINUE
      N = 0
      NN = 0
      NP = 0
      SX = 0.0
      SY = 0.0
      SX2 = 0.0
      SXY = 0.0
      SYMYE = 0.0
      SYEMYM = 0.0
      SYMYM = 0.0
```

```
      SP1 = 0.0
      SXMXM2 = 0.0
      DO 4300 J1 = 1, N2
      J2 = J1 — 1
      K = 11*J2
      KPNO10 = K + NO10
      KPNO7 = K + NO7
      IF(CORE(KPNO10).LT.0.0001) GO TO 4300
      N = N + 1
      X = AND(CORE(KPNO7),QMASK5)
      X = IX
      Y = ALOG(CORE(KPNO10)
      XY = X*Y
      X2 = X*X
      SX = SX + X
      SY + SY + Y
      SXY = SXY + XY
      SX2 = SX2 + X2
4300  CONTINUE
      VN = N
      XM = SX/VN
      SXT2 = SX*SX
      YM = SY/VN
      DNUM = VN*SX2 — SXT2
      IF((ABS(DNUM)).LT.0.0001) GO TO 1870
      BOL = (SY*SX2 — SX*SXY)/DNUM
      B1L = (VN*SXY — SX*SY)/DNUM
      BO = EXP(BOL)
      B1 = EXP(B1L)
      DO 4500 J1 = 1, N2
      J2 = J1 — 1
      K = 11*J2
      KPNO7 = K + NO7
      KPNO10 = K + NO10
      X = AND(CORE(KPNO7),QMASK5)
      X = IX
      YE = (B1**X)*BO
      IF(CORE(KPNO10).LT.0.0001) GO TO 4500
      Y = CORE(KPNO10)
      XMXM = X — XM
      XMXM2 = XMXM*XMXM
      YMYE = Y — YE
      YMYE2 = YMYE*YMYE
      YL = ALOG(CORE(KPNO10))
      YEL = (B1L*) + BOL
      YEMYM = YEL — YM
```

```
      YEMYM2 = YEMYM*YEMYM
      YMYM = YL — YM
      YMYM2 = YMYM*YMYM
      P1 = XMXM*YMYM
      IF(YMYE) 4350, 4370, 4360
4350  NN = NN + 1
      GO TO 4370
4360  NP = NP + 1
4370  CONTINUE
      SYMYE = SYMYE + YMYE2
      SYEMYM = SYEMYM + YEMYM2
      SYMYM = SYMYM + YMYM2
      SP1 = SP1 + P1
      SXMXM2 = SXMXM2 + XMXM2
4500  CONTINUE
      ZYX = SQRT(SYMYE/VN)
      PS2 = SXMXM2*SYMYM
      RPS2 = SQRT(PS2)
      RL = SP1/RPS2
      WRITE(6,4600) NN, NP, ZYX, RL, B1, BO
4600  FORMAT (1HO, 1X,5HNN IS,1X,I4, 1X,5HNP IS,1X,I4,1X,
     7HSYX IS, 1X,E14. 17,1X5HR IS, 1X,F9.4,1X,6HA1 IS, 1X,
     E14.7, 1X,6HAO IS, 1X,E14.7)
      WRITE(6,4620)
4620  FORMAT(1HO)
**  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **
5000  DO 1800 NA = IA, LA
      VNA = NA
      IF((A1.EQ.0.0).AND.(AO.EQ.O.O)) GO TO 5500
      IF((B. EQ.O.O). AND.(BC.EQ.O.O)) GO TO 5300
      IF(ZYX.LE.SYX) GO TO 5500
5300  YE = A1*VNA + AO
      CREC(1) = ANEWL
      CREC(2) = AINS
      GO TO 6000
5500  YE = (BI**VNA)*BO
      CREC(1) = ANEWE
      CREC(2) = AINS
6000  CONTINUE
      DO 1700 J1 = 1, N2
```

```
      J2 = J1 — 1
      K = 11*J2
      KPNO1 = K + NO1
      KPNO2 = K + NO2
      KPNO3 = K + NO3
      KPNO4 = K + NO4
      KPNO5 = K + NO5
      KPNO6 = K + NO6
      KPNO7 = K + NO7
      KPNO8 = K + NO8
      KPNO9 = K + NO9
      KPNO10 = K + NO10
      KPNO11 = K + NO11
      X = AND(CORE(KPNO7),QMASK5)
      X = IX
      IF(X.EQ.VNA) GO TO 1701
1700  CONTINUE
      GO TO 1750
1701  CONTINUE
      CREC(3) = CORE(KPNO3)
      CREC(4) = CORE(KPNO4)
      CREC(5) = CORE(KPNO5)
      CREC(6) = CORE(KPNO6)
      CREC(7) = AND(CORE(KPNO7),QMASK1)
      NY = X
      CREC(8) = CORE(KPNO8)
      CREC(9) = CORE(KPNO9)
      CREC(11) = CORE(KPNO11)
      IF((ABS(CORE(KPNO10))).GE.0.0001) GO TO 1720
      CREC(10) = YE
      GO TO 1725
1720  CREC(1) = CORE(KPNO1)
      CREC(2) = CORE(KPNO2)
      CREC(10) = CORE(KPNO10)
1725  IF(NS.EQ.1) GO TO 1730
      WRITE (6,1050) (CREC(J),J = 1,7), NY,(CREC(J), J = 8, 11)
1730  IF(NC.EQ.1) GO TO 1740
      WRITE(7,1051) (CREC(J), J = 1,7), NY, (CREC(J), J = 8,11)
1051  FORMAT (6(1X,A6), 1X, A4, I3, 2(1X, A6), 1X, E14. 7, 1X, A6)
1740  IF(NT.EQ.1) GO TO 1800
      CALL CRBUF (NUM)
      WRITE(99, 1745) NY
```

```
1745 Format(I2)
     CALL BCDCR (NUM)
     READ(99, 1747) ANY
1747 FORMAT(A2)
     CALL CHIN (O, ANY, 4, CREC(7)))
     CALL CHIN (1, ANY, 5, CREC(7))
     CALL WREC 1 (CREC, 4)
     GO TO 1800
1750 CONTINUE
     CREC(3) = CORE(NO3)
     CREC(4) = CORE(NO4)
     CREC(5) = CORE(NO5)
     CREC(6) = CORE(NO6)
     CREC(7) = AND(CORE(NO7), QMASK1)
     NY = NA
     CREC(8) = CORE(NO8)
     CREC(9) = CORE(NO9)
     CREC(10) = YE
     CREC(11) = CORE(NO11)
     IF(NS.EQ.1) GO TO 1760
     WRITE (6, 1050) (CREC(J), J = 1,7), NY, (CREC(J), J = 8, 11)
1760 IF(NC.EQ.1) GO TO 1770
     WRITE(7, 1051) (CREC(J), J = 1,7), NY, (CREC(J), J = 8, 11)
1770 IF(NT.EQ.1) GO TO 1800
     CALL CRBUF(NUM)
     WRITE(99, 1745) NY
     CALL BCDCR (NUM)
     READ(99, 1747) ANY
     CALL CHIN (O, ANY, 4, CREC(7))
     CALL CHIN(1, ANY, 5, CREC(7))
     CALL WREC 1 (CREC, 4)
1800 CONTINUE
     GO TO 2000
C ** ** ** ** ** ** ** ** ** ** ** ** ** **
1850 NPR = 1
     GO TO 1900
1870 NPR = 2
1900 WRITE (6, 1901) NPR
1901 FORMAT (1HO, 55HTHE ABOVE SERIES HAS THE DET. EQ.
        ZERO FOR PROGRAM NOMB 1ER, I2)
     IF(NPR.EQ.1) GO TO 4000
     IF((A1.NE.O.O).OR.(B1.NE.O.O).OR.(AO.NE.O.O).OR.(BO.NE.O.O))
     GO 1TO 5000
```

```
      C ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
 2000 DO 2001 J = 1, 11
      CREC(J) = 0.0
 2001 CONTINUE
      DO 2002 J = 1,1100
      CORE(J) = 0.0
 2002 CONTINUE
      N1 = 0.0
      A0 = 0.0
      A1 = 0.0
      B0 = 0.0
      B1 = 0.0
      R = 0.0
      NN = 0
      NP = 0
      NPR = 0
      SY = 0.0
      SX = 0.0
      SP1 = 0.0
      SX2 = 0.0
      SXY = 0.0
      SYX = 0.0
      ZYX = 0.0
      SYMYE = 0.0
      SYEMYM = 0.0
      SYMYM = 0.0
      SXMXM2 = 0.0
      IF(NEND.EQ.1) GO TO 2500
      IF(TCTRY. GT. FCTRY) GO TO 2600
      GO TO 100
 2500 WRITE (6,. 2550)
 2550 FORMAT (1HO, 9HNEND, EQ.1)
      GO TO 3000
 2600 WRITE (6, 2650)
 2650 FORMAT (1HO, 14HTCTRY,GT.FCTRY)
 3000 CALL CLFLES (1,—1,+1,+1,—1)
      IF(NT.EQ.1) GO TO 3001
      CALL CLFLES (4. 1, 1, 1, 1)
 3001 CONTINUE
      STOP
      END
```

Subroutine Library

I. *Subroutine Name:* IOLMT7

Purpose: Input and Output. Handles blocked input of any size, reading in one logical record at a time.

Subroutine Calls:

a. Set-up file for Reading or Writing
CALL SET1 (IFILE, IMODE, IREW)

b. Read one record at a time into an array
CALL RREC1 (ARRAY, IFILE, NEND, NREEL)

c. Write one record at a time, blocking output
CALL WREC 1 (ARRAY, IFILE)

d. Close any file.
CALL CLFLES (IFILE, ITAPE, IREW, IUNLD, IE⌀FP)

Variable Name	*Explanation*
IFILE	Number from 1 to 9 assigned to file card
IMODE	1 for BCD Input 2 for Binary Input 3 for BCD Output 4 for Binary Output
ARRAY	Dimensioned array where data are stored or accessed
NEND	0, if last record has not been read 1, if last record has been read
NREEL	Number of reels to be read (must not be literal)
IREW	1 Rewind —1 No rewind
ITYPE	—1 Input 1 Output
IUNLD	1 Unload —1 Do not unload

* FROM I.C.C. MANUAL OF STANDARDS

IEOFP —1 Do not execute "End of Data File" procedure
1 Execute procedure (should be set to —1 for input)

Comments:

A. End of data file procedure

1. EOF Trailer, preceded and followed by file marks produced for labelled output file.

2. A complete set of file cards must be provided as follows:

$ Pool "File 01", "File 02", ...

$ File "File 01", U01, UO1, DEFER, BLØCK =, IRL = ,... for all unused file numbers

$ File "File 07", UO6, UO6, MØUNT, BLØCK =, IRL =, RCT = , ...

For Standard labelled files

EOR = EORLAB, ADDLBL = ADLBLS

For Non Standard labelled files

EOR = EORLAB, EOF = EOFEX, NSLBL = MYLBLS

For Unlabelled files

EOR = EORUNL, EOF = EOFEX

II. *Subroutine Name:* BCDCON

Purpose: Sets up alphabetic array for re-reading data just read into storage in different format

Subroutine Call: CALL BCDCON (DATM (I, J))

Variable Name	*Explanation*
DATM (I, J)	First word of array to be re-read

Comments:

Used together with FIL 99 routine (also available on Systems Tape 9.11C). After BCDCON is called, words to be re-read are accessible by

READ (99, FMT) A, B, ...

where FMT is any FORTRAN format statement and A, B, ... is an input-output list.

III. *Subroutine Name:* SRC7

Purpose: Binary Search routine to locate a word in a table

Subroutine Call: CALL SRCH (BUFF(I,J), TABLE (K,L), NITM, NWI, INDEX)

Variable Name	*Explanation*
BUFF(I,J)	Word for which match is sought in table
TABLE (K,L)	First word in table to be inspected where K = ordinal number of word within item to be inspected L = first item of table to be inspected
NITM	Number of items in table
NWI	Number of words per item in table
INDEX	0, if no match found. If search was successful, word found at TABLE (K, INDEX) (i.e. Index equal to the ordinal number in the array where word was found

Restrictions: Table must be in sorted order prior to entry to this routine

IV. *Subroutine Name:* SOR7

Purpose: Binary Core Sort Routine

Subroutine Call: CALL CSORT (TABLE(1, J), NITM, NWI, KWD)

Variable Name	*Explanation*
TABLE (1, J)	First word of first item of table to be sorted
NITM	Number of items in table
NWI	Number of words per item
KWD	Ordinal number within item of word on which table is to be sorted

V. *Subroutine Name:* CHIN 7

Purpose: Insertion of a character in a given word at a specified character position from a particular character position in another word

Subroutine Call: CALL CHIN (CP1, WORD1, CP2, WORD2)

Variable Name	*Explanation*
CP1	Character position in origin word, 0 to 5
WORD1	Origin word
CP2	Character position in destination word, 0 to 5
WORD2	Destination word

Function Name: KDIFR7

Purpose: To set up three way FORTRAN branch based on the comparison of two variables. This routine performs a fixed point subtract regardless of the status of the variables.

Function Calls: If (KDIFR (V_1,V_2)x, y, z

Variable Names	*Explanation*
V_1	Variable one
V_2	Variable two

where x, y, z are statement numbers

Subroutine Name: CRBUF, BCDCR

Purpose: To convert data already stored within an array to another format, so that it can be written in a format other than the one in which data were originally generated.

Note: These routines operate in conjunction with F99, which sets up a file control block for the internal read and write (e.g. READ (99, FMT) or WRITE (99, FMT).

Subroutine Calls: Call CRBUF (BUFPNT)
Call BCDCR (BUFPNT)

Variable Name	*Explanation*
BUFPNT	Arbitrary name assigned to the internal buffer pointer. This is done so that when a READ (99) or WRITE (99) is given, the computer locates the proper position in the buffer into which to write or from which to read

Note: The following would then be the sequence in which these routines would be used:

CALL CRBUF (BUFPNT)
WRITE (99, FMT) list

This writes the information into the buffer according to the specified format. Then to read from the buffer into the converted format:

CALL BCDCR (BUFPNT)
READ (99, FMTI) list

UN PROGRAMME D'INTERPOLATION ET D'INSERTION DES DONNEES, ECRIT POUR LE LANGAGE FORTRAN IV

A l'aide de ce programme on réalise l'interpolation des données et l'insertion de certaines autres qui ont été omises sur la bande des données des comptes nationaux. A cette fin, on utilise, ou bien une fonction linéaire ayant la forme $y = a_1x + a_0$, ou bien une fonction exponentielle $y = b_0b_1^x$. L'option du programme va vers l'utilisation de l'une ou de l'autre de ces fonctions au moyen de la comparaison entre les erreurs standards d'approximation. A la sortie on obtient la file initiale des données, les files interpolées — ayant de nouvelles données insérées sur la demande du bénéficiaire —, ainsi que les erreurs standards d'approximation.

Le programme utilise les sous-routines suivantes, appartenant au Centre International de Calcul ONU, New York: «IOLMT7», «SOR7», «SRC7», «KDIFR7», «CHIN7», «CRBUF» et «BCDRCR».

ПРОГРАММА ИНТЕРПОЛЯЦИИ И ИНСЕРЦИИ ДАННЫХ, ЗАПИСАННЫХ НА ЯЗЫКЕ ФОРТРАН IV

Благодаря такой программе реализуется интерполяция данных и инсерция некоторых данных, отсутствующих на ленте данных национальных счетов. Для этого используется или линейная функция вида $y = a_1x + a_0$, или экспонентная функция вида $y = b_0b_1^x$. Программа применяется для использования той или иной из функций путем сравнения ошибок приблизительного стандарта. При выходе получается исходный ряд данных, по требованию заказчика —, а также ошибки приблизительного стандарта.

Программа использует подпрограмму Международного Вычислительного Центра ООН из Нью-Йорка «IOLMT7», «SOR7», «SRC7», «KDIFR7», «CHIN7», «CRBUF7» и «BCDCR».