сообщения
объединенного
института
ядерных
исследований
дубна

E10 - 12353

V.Gadjokov

PROCESSING OF DISCRETE NUCLEAR SPECTRA ON SMALL COMPUTERS.

B.Implementation of the KATØK-F Algorithm

1979

E10 - 12353

V.Gadjokov

PROCESSING OF DISCRETE NUCLEAR SPECTRA ON SMALL COMPUTERS.

B.Implementation of the KATØK-F Algorithm

Гаджоков В. **E10 - 12353**

Обработка дискретных ядерных спектров на малых ЭВМ. В. Реализация алгоритма КАТØК-Ф.

Эта работа - вторая в серии из трех, посвященных детальному описанию алгоритма программы КАТØК, которая после девятилетней эксплуатации на ЭВМ Минск-2 была усовершенствована и написана заново на ФОРТРАНе.

В работе дано описание композиционного алгоритма решения поставленной задачи и рассмотрены программные средства реализации этого алгоритма. Дан полный текст двух основных модулей программы.

Работа выполнена в Лаборатории ядерных проблем ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна 1979

Gadjokov V. **E10 - 12353**

Processing of Discrete Nuclear Spectra on Small Computers. B. Implementation of the KATØK-F Algorithm

This paper is the second in a series of three dedicated to the detailed description of the KATØK-F algorithm. This code has recently been revised and re-written in FORTRAN after being run on Minsk-2 for nine years.

The description of the composite algorithm which solves the problem posed and a discussion of programming means of its implementation are given. The full text of the two chief modules of the KATØK-F code is reported.

The investigation has been performed at the Laboratory of Nuclear Problems, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna 1979

INTRODUCTION

In a previous paper [1] we have discussed the physical and the mathematical aspects of automated processing of discrete spectra on small computers. Implementation details of the approach described are given below. To avoid duplication and to achieve continuity, through-numbering of paragraphs and formulae has been used in both previous and present reports. Formulae are referred to by pointing out their numbers in parentheses, while paragraph numbers are underlined.

4.5. Outline of the KATØK-F Algorithm

After having discussed all the essential aspects of the KATØK-F algorithm we are now in a position to explain the basis philosophy of approaching and solving problems of type (2.2). Details of this philosophy will be given for a single specific system of simultaneous equations (2.2). It is understood that the same approach applies to all the M systems constituting the stream processed.

4.5.1. Input a standard data set as given in Table 1.

4.5.2. Calculate the components of x^o .

4.5.3. Scale as in 4.3.2.

4.5.4. Attempt a solution by the method of Gauss-Newton, i.e., with $a^o = a^t = 0$. If the attempt is successful, go to 4.5.8.; else go to next step.

4.5.5. Attempt a solution by Alexandrov's method of regularized iteration process with exponentially decreasing regularizer. If the attempt is successful, go to 4.5.8.; else go to next step.

4.5.6. Attempt a solution by means of regularized iteration process with an augmented value of $a^t = \text{const.}$ If the attempt is successful, go to 4.5.8.; else go to next step.

4.5.7. Dump the problem; go to 4.5.10.

4.5.8. Check for presence of superfluous peaks according to [2] (i.e., for peaks with repeating positions or with negligible intensities). If the exit is positive, reduce problem dimensions and return to 4.5.2; else go to next step.

4.5.9. De-scale; Output solution vector $\tilde{x}$ and variances of its components.

4.5.10. Check whether more standard sets await for processing. If so, return to 4.5.1; else stop.

Practical use of KATØK-F code demonstrates that this algorithm ensures a physically meaningful solution in all cases of correct input. Hence, option 4.5.7 enters into action when some standard set contains gross mistakes, e.g., wrong dimensions, search for peaks in a "white" section, etc. Otherwise this step is never referred to.

5. IMPLEMENTATION: THE KATØK-F CODE

5.1. Technicalities

The KATØK-F code has been written in FØRTRAN-IV and tested on HP 2116 and 21MX computers. It consists of 18 modules (one main program and 17 subprograms) some of which are subdivided into smaller units for the sake of easier debugging.

The HP compilers of FØRTRAN-IV implement actually a subset of the standard version of the language, major limitations being:

5.1.1. No BLØCK DATA subprograms are admitted.

5.1.2. A single non-named CØMMØN block is only allowed.

5.1.3. Variable names appearing in CØMMØN and EXTERNAL statements (either explicit or implicit) should consists of no more than five characters.

5.1.4. The main program must have a name.

5.1.5. Alphanumeric strings in FØRMAT statements should be enclosed in quotes (") rather than in apostrophes (').

As most of these language particularities do represent limitations, it appears that the adaptation and the use of the code on other computers should not evoke any serious difficulty.

The control links between the various modules are shown on Fig.2; data links are chiefly established through a number of CØMMØN arrays as listed in Table 2. The CØMMØN

Table 2

Important CØMMØN Variables

NAME	TYPE	DIMENSION	CONTENT	NOTES
A. Status Data				
PSV	Logical	5 x 1	Regime control parameters	See section 5.2
ØØV	Logical	3 x 1	Output control parameters	See section 5.2
B. Size Data				
MM	Integer	Scalar	M	
M	Integer	Scalar	m	
K	Integer	Scalar	k	
L	Integer	Scalar	ℓ	
N	Integer	Scalar	n	
C. Input Data				
IDENTF	Integer	36 x 1	Stream Identifier	Not processed by program described
NFC	Integer	Scalar	q_{in}	
Q	Real	m x 1	Channel numbers $\{q\}$	
YQ	Real	m x 1	Channel contents $\{Y_q\}$	
D. Working Storage				
NLC	Integer	Scalar	q_{end}	
XO	Real	n x 1	x^o	
W	Real	m x 1	diag W	
FXT	Real	m x 1	$fx^t = Fx^t - Y$	
F1XT	Real	m x n	$f'(x^t)$	
YQT	Real	m x 1	$Y^t = Fx^t$	Used for output at solution point
XT	Real	n x 1	x^t	-"-
ETØT	Real	n x 1	e^{tot}	-"-
V	Real	n x n	$V(x^t)$	
DELTXT	Real	n x 1	$[V(x^t)+\alpha^t I]^{-1}\overline{f'(x^t)}Wfx^t = \Delta x^t$	Increments at t^{th} iteration
C	Real	n x 1	diag C	
BGPC	Real	6 x 1	c_{a_i} ; $i=0,1,\ldots,5$	Background scaling factors
WEIGHT	Real	5 x 1	v_i ; $i=1,2,\ldots,s$	Long-defect weights
STACK	Real	5 x 1	θ^{t-i+1} ; $i=1,2,\ldots,s$	Long-defect stack

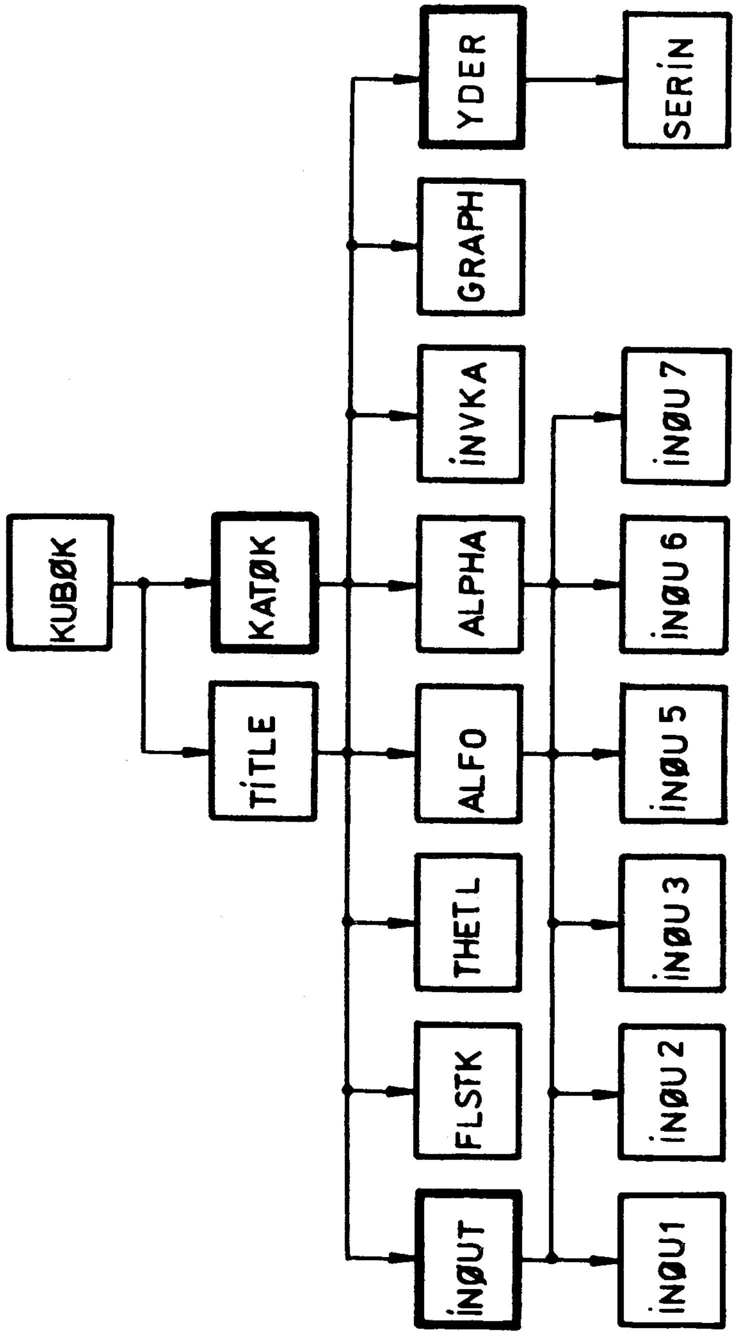

Fig.2. Hierarchical links between KATØK-F modules. Arrows are directed from calling modules to called ones (built-in functions not shown). Principal modules are given in darker lines.

block includes also certain auxilliary (number of section processed, current, iteration number, etc.) and limiting (maximum values of m , k , ℓ , n , t , superresolution coefficient, etc.) variables which are not listed in the table. For core economy parts of the largest array F1XT are also used as temporary working storage by means of EQUIVALENCE statements.

With these precautations the program's total core requirements amount to approximately 25K system-free HP words (i.e., ~ 50K bytes).

5.2. Program Status

KATØK-F is a software package with an intricate flow chart. The execution follows different paths which are selected in accordance with the numerical particularities of problems solved and with the current status of the program. The latter is described by two logical vectors PSV (Program Status Vector) and ØØV (Operator Option Vector) whose components have the following meaning (See Table 3).

All the components of PSV are automatically set.TRUE. at the beginning of processing each section; then, as computations progress, they may be gradually reset to.FALSE. according to process requirements and specific numerical conditions. Their current status is tested by means of program switches which select the suitable logical path at the stage reached. The operator has no means of controlling PSV.

On the contrary, the components of ØØV which define the output options in effect are set in direct dialogue via the operator's console. This is accomplished before processing the first spectrum section of the stream and remains in force untill completing the computations. If detailed output is not needed and it is undersirable to waste time for the dialogue itself, the output regime may be set automatically to its most economic version (all ØØV-components.FALSE.). This is achieved by means of a specific CALL (see next Section).

The structure and the action of regime switches should be clear enough from their FØRTRAN text.

5.3. Flow Charts

The general flow chart of the package is presented on Fig.3. Computations begin with setting the constant values in CØMMØN by the main program KUBØK (instead of the inadmissible BLØCK DATA); after printing out a title page (module TITLE),

Table 3

Significance of PSV and ØØV

Component	Regime at Value	
	.TRUE.	.FALSE
PSV(1)	Gauss-Newton method	Regularized iteration process
PSV(2)	Usual solution-defect	Long solution-defect
PSV(3)	Exponentially decreasing regularizer	Steady value of regularizer
PSV(4)	Iterations in course	Interrupt criterion satisfied
PSV(5)	No dump, normal computations	Dump, problem unsolvable
ØØV(1)	Output after each iteration	Output at solution point only
ØØV(2)	Output of program regime	No regime output
ØØV(3)	Section graph drawn after processing	Pure processing, no graphs

the KATØK subroutine is called. Here, according to the CALL-parameters, the dialogue possibility may be used. Output options are considered incorrect if detailed output is requested for streams consisting of 25 or more sections, and in such cases processing ends with an error message. When output options are correct, a section is read and chacked for internal consistency. Actually, the scheme 4.5.1.- 4.5.10 is executed by the part of Fig.3 which is enclosed in a dashed frame. More details of this part are given in the flow-chart on Fig.4 which shows the most essential blocks of the KATØK subroutine. The five program switches on Fig.4 represent each a series of IF-statements which sense the status of PSV-components and other numerical values (θ^t , Θ_s^t , det V , etc.) and direct accordingly the process of computations. A complete record of regime switching may be obtained by seting ØØV(2)= .TRUE. at the stage of the initial dialogue.

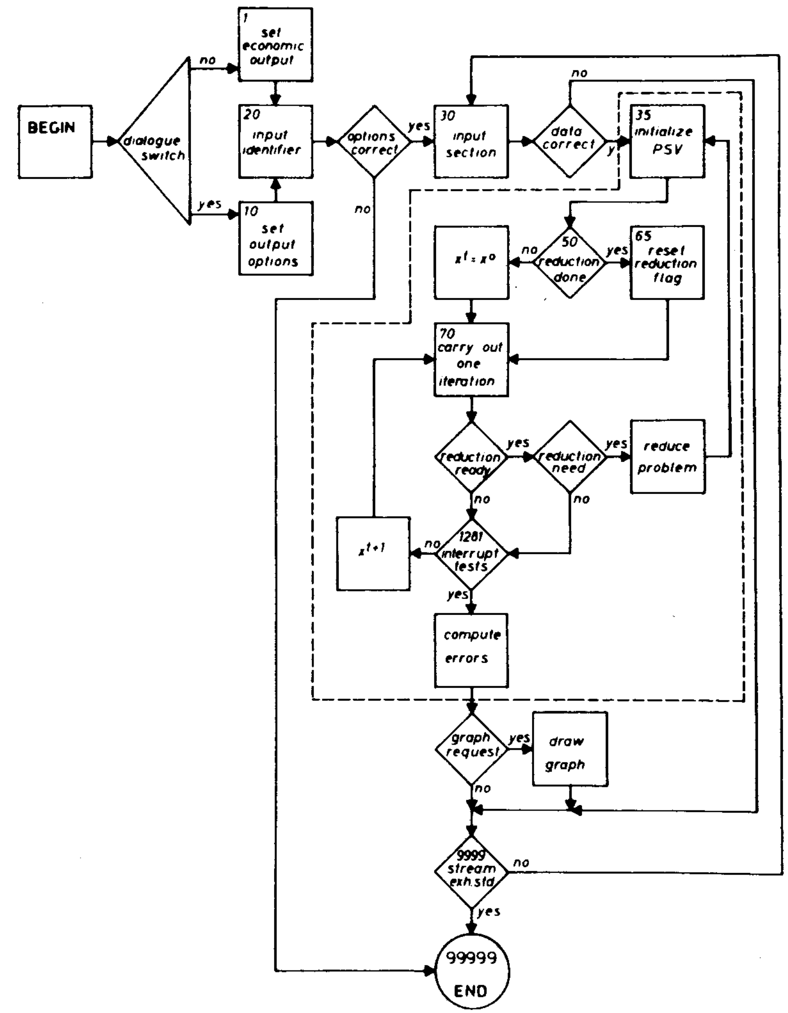

Fig.3. Flow chart of KATØK package (organization of calculations). Numbers in upper corner correspond to FØRTRAN statement labels in KATØK-subroutine. Auxiliary modules are not shown. For iteration-scheme details (framed part) see Fig.4.

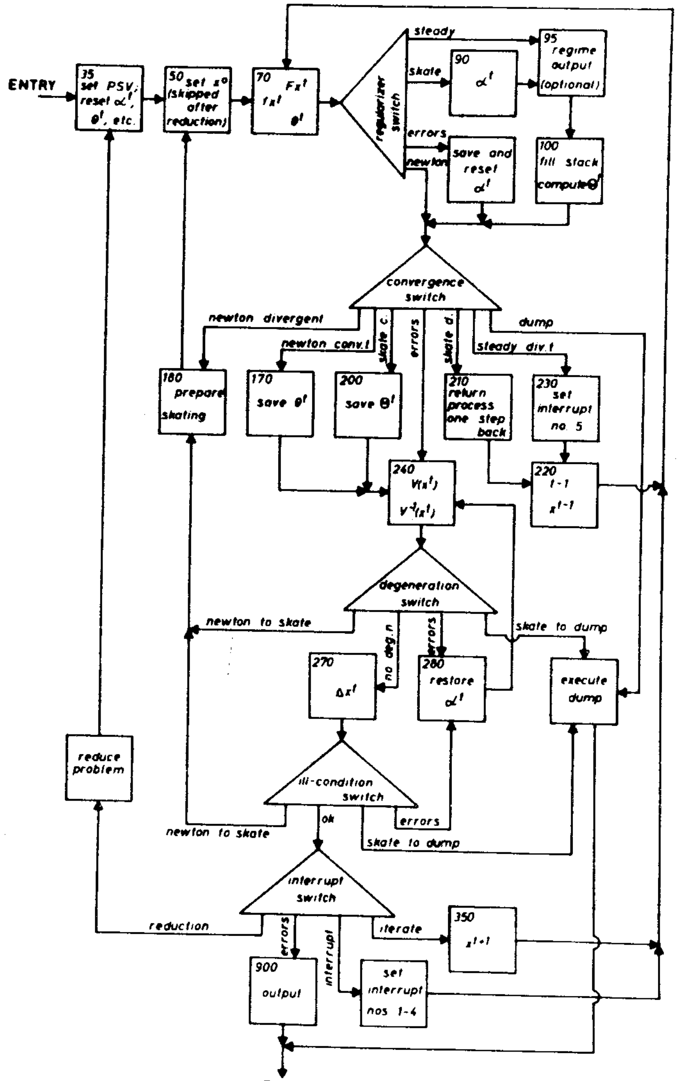

Fig.4. Flow chart of regime switching in KATØK-subroutine (framed part of Fig.3). Numbers in upper left corner corres- pond to FØRTRAN statement labels. A switch is programmed as a series of IF-statement.

The remaining modules (see Fig.2) have auxiliary functions which are briefly described in the next section.

5.4. Functions of Individual Modules

5.4.1. KUBØK (main) - sets values of constants in CØMMØN.

5.4.2. TITLE (subroutine) - prints out a title page.

5.4.3. KATØK (subroutine) - implements the iteration scheme; a chief module.

5.4.4. ALFO (function) - calculates the initial value of regularizer α^o .

5.4.5. ALPHA (function) - calculates the value of regularizer at iteration no. t .

5.4.6. FLSTK (subroutine) - fils in the stack of solution-defect values.

5.4.7. THETL (subroutine) - computes the value of long solution-defect.

5.4.8. GRAPH (subroutine) - draws a graph of section processed; a dummy module in this version.

5.4.9. INVKA (subroutine) - carries out the V-matrix inversion; an adapted to HP subroutine from the IBM SSP-package.

5.4.10. YDER (subroutine) - according to CALL-paratemeters:
(i) calculates actual number of unknowns n at given k and ℓ ;
(ii) computes spectrum value at given channel number q ;
(iii) calculates n values of derivatives at given channel number q ;
(iv) calculates pure spectrum contribution at given channel number q ;
(v) computes pure background at given channel number q .

5.4.11. SERIN (function) - calculates the error integral J(y).

5.4.12. INØUT - handles input-output and auxiliary operations in accordance with CALL-parameters. In particular:
(i) sets output options in dialogue via the system console.
(ii) inputs spectrum identifier and stream size M; prints out header;
(iii) inputs a data section for processing (with control print);
(iv) calculates and scales x^o ; checkes internal consistency of data;
(v) prints out point data (if detailed output is requested);

(vi) prints out spectrum parameters and errors;
(vii) prints out program status during execurion;
(viii) prints out error messages;
(ix) checks and - if needed - reduces peak number k.

5.4.13. INØU1 - carries out 5.4.12.i.
5.4.14. INØU2 - carries out 5.4.12.ii.
5.4.15. INØU3 - carries out 5.4.12.iii.
5.4.16. INØU5 - carries out 5.4.12.v.
5.4.17. INØU6 - carries out 5.4.12.vi.
5.4.18. INØU7 - carries out 5.4.12.vii.

5.5. Limitations

In the version reported the KATØK-F package can be used if the following limitations are observed.

$$1 \le M \le 999 \qquad (5.5.1)$$

$$-1 \le \ell \le 5 \qquad (5.5.2)$$

Although ℓ is generally the background-polynomial degree, when set to -1 it causes that the background be considered null throughout the section processed.

$$3 \le n \le 40 \qquad (5.5.3)$$

$$1 \le k \le 10 \qquad (5.5.4)$$

$$n+1 \le m \le 100 \qquad (5.5.5)$$

The violation of inequalities 5.5.2. - 5.5.5 leads to adequate error messages; 5.5.1 is simply ensured by format conventions (see Table 1 in /1/).

5.6. Development

The KATØK-F software package has been especially developed for streamline processing of discrete nuclear spectra which - mathematically - is equivalent to solving of overdetermined simultaneous non-linear equations. The type of nonlinearity dealt with is Gaussian. However, the composite iteration scheme (see Fig.4) is applicable to other types of nonlinearity as well (e.g. resonance curves, exponential regularities, etc.). To achieve this, one should only replace the present SYMGA (≡ YDER) subroutine and the respective I/Ø blocks (INØUT with subprograms) with suitable substitutes. The iteration scheme itself as implemented in KATØK subroutine needs no modifications.

5.7. The KATØK-F Text

The full FØRTRAN-IV text of the two chief modules is reproduced below. These are KUBØK (main program) and KATØK itself. The remaining auxiliary modules will appear shortly as a separate report in the same JINR-series.

```
      PROCRAM KUBOK
C     CARRIES OUT THE PROCESSINC OF DISCRETE SPECTRA WITH SYMMETRIC
C     GAUSSIAN SHAPE OF THE SINCLE ISOLATED LINE BY MEANS OF 'KATOK'
C     SUBROUTINE; THIS, IN TURN, SHOULD BE SUPPORTED BY:
C          A. 'SYMGAU' SUBROUTINE (OR EQUIVALENT SUBSTITUTE)
C          B. 'INOUT' SUBROUTINE (OR EQUIVALENT SUBSTITUTE)
C     THE USE OF GENUINE CODES RATHER THAN SUBSTITUTES IS RECOMMENDED.
C
C
      LOGICAL PSV,OOV
      COMMON PSV(5)
      COMMON OOV(3)
      COMMON MM,M,K,L,N
      COMMON         Q(100),YQ(100),X0(40),IDENTF(36),NFC,NLC
      COMMON         W(100),FXT(100),FIXT(100,40),YCT(100),XT(40),
     1               ETOT(40),C(40),BGPC(6),WEIGHT(5),STACK(5),DUM(280),
     2               V(40,40),DELTXT(40),
     3                    MSWICH,NSECT,ITER,THETAT,THETIG,THETLT,REG,
     4               THECPS,THECPL,INTERR,DET
      COMMON         LENGTH
      COMMON         THETLM,TLIM,LMIN,LMAX,NMIN,NMAX,KMIN,KMAX,MMAX,
     1               MMDMAX,DEGER,ITRMAX,DEVMIN,SRCOEF
C
C
C
      EXTERNAL INOUT,SYMGA
C
C SIMULATION OF 'BLOCK DATA'
      BGPC(1)=2.0**6
      BGPC(2)=2.0**3
      BGPC(3)=1.0
      BGPC(4)=2.0**(-4)
      BGPC(5)=2.0**(-7)
      BGPC(6)=2.0**(-11)
      THETLM=1.0E+20
      TLIM=.75
      LMIN=-1
      LMAX=5
      NMIN=3
      NMAX=40
      KMIN=1
      KMAX=19
      MMAX=100
      MMDMAX=25
      DEGER=1.0E-38
      ITRMAX=25
      SRCOEF=25.0
      LENGTH=3
      WEIGHT(1)=0.375
      WEIGHT(2)=0.4375
      WEIGHT(3)=0.1875
      WEIGHT(4)=0.0
      WEIGHT(5)=0.0
C
      CALL TITLE
      CALL KATOK(1,INOUT,SYMGA)
      STOP
      END
C
```

```
C
C
C THIS SUBROUTINE IS DESIGNED FOR AUTOMATIC STREAM PROCESSING
C OF DISCRETE SPECTRA. ITS DETAILED DESCRIPTION IS GIVEN IN A
C SEPARATE JINR-REPORT. PRESENT VERSION: IZOT310-F4/AUGUST"78
C
C
C DESCRIPTION OF ARGUMENTS:
C   IC    - OUTPUT CONTROL PARAMETER
C               WHEN IC=1 OUTPUT REGIME IS SET AUTOMATICALLY
C               WHEN IC=2 OUTPUT REGIME IS SET IN DIALOGUE
C                                VIA THE OPERATOR"S CONSOLE
C   INOUT - EXTERNAL SUBROUTINE FOR I/O HANDLING
C   YDER  - EXTERNAL SUBROUTINE WHICH CALCULATES THE VALUES
C             OF APPROXIMATING FUNCTION AND ITS DERIVATIVES
C
C
C DESCRIPTION OF COMMON BLOCKS:
C   /AP/      - PROGRAM-STATUS VECTOR PSV ("AUTOMATIC" PARAMETERS)
C   /OP/      - OPERATOR-OPTION VECTOR OOV (OPERATOR"S PARAMETERS)
C   /SIZEDT/ - SIZE DATA OF STREAM AND SPECTRA PROCESSED
C   /INDATA/ - INPUT DATA
C   /WSTORE/ - WORKING STORAGE
C   /THELNG/ - WEIGHTS, STACK AND LENGTH OF LONG SOLUTION-DEFECT
C   /RENORM/ - RENORMALIZING FACTORS OF BACKGROUND POLYNOMIAL
C   /LIMDT/  - LIMITING CONSTANTS OF VARIOUS SORTS
C   /OUTDAT/ - OUTPUT DATA
C
C
C IMPORTANT DIMENSIONS AND VARIABLES:
C ================================
C    MM     - NUMBER OF SECTIONS IN STREAM PROCESSED
C    NSECT  - ORDINARY NUMBER OF SECTION PROCESSED
C    M      - LENGTH OF SECTION PROCESSED
C    K      - NUMBER OF PEAKS IN SECTION
C    L      - DEGREE OF BACKGROUND POLYNOMIAL; MAY HAVE INTEGER VALUES
C                IN THE RANGE [0,5]; IN ADDITION, WHEN SET TO -1, THE
C                BACKGROUND IS CONSIDERED NULL THROUGHOUT THE SECTION
C    ITER   - NUMBER OF CURRENT ITERATION
C    THETAT - SOLUTION DEFECT AT ITERATION NO. "ITER"
C    THETLT - LONG SOLUTION-DEFECT AT ITERATION NO. "ITER"
C                (SET TO ZERO WHEN ITER < LENGTH-1)
C    LENGTH - WORKING LENGTH OF LONG SOLUTION-DEFECT (SET BY
C                "BLOCK DATA")
C
C
      SUBROUTINE KATOK(IC,INOUT,YDER)
C
C
      LOGICAL PSV,OOV
      COMMON PSV(5)
      COMMON OOV(3)
      COMMON MM,M,K,L,N
      COMMON        Q(100),YQ(100),X0(40),IDENTF(36),NFC,NLC
      COMMON        W(100),FXT(100),FIXT(100,40),YCT(100),XT(40),
     1              ETOT(40),C(40),BGPC(6),WEIGHT(5),STACK(5),DUM(280),
     2              V(40,40),DELTXT(40),
     3                  MSWICH,NSECT,ITER,THETAT,THETIG,THETLT,REG,
     4                THECPS,THECPL,INTERR,DET
      COMMON        LENGTH
```



```
      COMMON        THETLM,TLIM,LMIN,LMAX,MMIN,MMAX,KMIN,KMAX,NMAX,
     1              MMDMAX,DEGER,ITMAX,DEVMIN,SECCEF
C
C
C
      DIMENSION VVV(1600),CW(40,40),LOOLOO(40),MOOMOO(40)

      EQUIVALENCE (CW(1,1),FIXT(1,1)),(VVV(1600),FIXT(100,40))
      EQUIVALENCE (LOOLOO(1),FIXT(1,19)),(MOOMOO(1),FIXT(1,20))
C
C
      EXTERNAL YDER
C
      GO TO(1,10),IC
C
C
C AUTOMATIC SET OF CONTROLS (ECONOMIC VERSION)
    1 OOV(1)=.FALSE.
      OOV(2)=.FALSE.
      OOV(3)=.FALSE.
      GO TO 20
C
C
C CONTROLS SET BY OPERATOR (DIALOGUE VIA INOUT)
   10 CALL INOUT(1,YDER)
C
C
C INPUT SPECTRUM IDENTIFIER (MAX. 72 SYMBOLS) & STREAM SIZE MM
   20 CALL INOUT(2,YDER)
      NSECT=1
      MSWICH=0
      IF((OOV(1).OR.OOV(2).OR.OOV(3)).AND.(MM.GT.MMDMAX))MSWICH=5
      IF(MSWICH.EQ.5)CALL INOUT(8,YDER)
      IF(MSWICH.EQ.5)GO TO 99999
C
C
C INPUT A SPECTRUM SECTION TO BE PROCESSED
   30 CALL INOUT(3,YDER)
      CALL YDER(1,Q(1),YQ)
      MSWICH=0
      CALL INOUT(4,YDER)
      IF(MSWICH.NE.0)GO TO 9999
C
C
C IMPLEMENT THE KATOK ITERATION SCHEME: BEGIN
C ******************************************
C
C INITIAL SET OF PSV, DEFECT & REG (ALWAYS AUTOMATIC !)
   35 DO 40 I=1,5
      PSV(I)=.TRUE.
   40 CONTINUE
C
      ITER=0
      THETAT=0.0
      THETLT=0.0
      REG=0.0
      DET=0.0
      IF(OOV(2))CALL INOUT(7,YDER)
C
      DMN=FLOAT(M-N)
      THECPS=THETLM
```

```
      LMIN1=LENGTH-1
C
C SET VECTOR OF INITIAL GUESSES
C                                SETTING SKIPPED AFTER REDUCTION
   50 IF(MSWICH.EQ.11)GO TO 65
      DO 60 I=1,N
   60 XT(I)=X0(I)
      GO TO 70
   65 MSWICH=0
C
C COMPUTE APPROXIMATION VECTOR YQT, FXT & DEFECT THETAT
C                                  [ETOT(I) USED AS WORKING STORAGE]
   70 THETAT=0.0
      DO 80 I=1,M
      AUXIL=Q(I)
      CALL YDER(2,AUXIL,ETOT)
      YQT(I)=ETOT(I)
      FXT(I)=ETOT(I)-YQ(I)
   80 THETAT=THETAT+W(I)*FXT(I)**2
      THETAT=SQRT(THETAT/DMN)
      IF(ITER.EQ.0)THETIG=THETAT
      IF(ITER.EQ.0)T11111=THETIG
      IF((ITER.EQ.0).AND.(.NOT.PSV(1)))ALPH0=ALF0(T11111)
      IF(OOV(1))CALL INOUT(5,YDER)
C
C REGULARIZER SWITCH AND ADJOINT BLOCKS ("90" & "100")
      IF(PSV(1))GO TO 110
      IF(PSV(3))CO TO 90
      IF(PSV(4))GO TO 95
      SAVREG=REG
      REG=0.0
      GO TO 110
   90 REG=ALPHA(ALPH0,ITER)
   95 IF(OOV(2))CALL INOUT(7,YDER)
  100 CALL FLSTK(THETAT)
      IF(ITER.LT.LMIN1)GO TO 110
      CALL THETL(THETLT)
      IF(ITER.NE.LMIN1)GO TO 110
      PSV(2)=.FALSE.
      IF(OOV(2))CALL INOUT(7,YDER)
C
C CONVERGENCE SWITCH AND ADJOINT BLOCKS
  110 IF(PSV(1))GO TO 130
      IF(PSV(2))GO TO 140
      IF(PSV(3))GO TO 150
      IF(PSV(4))GO TO 160
      IF(PSV(5))GO TO 240
  120 CALL INOUT(7,YDER)
      CALL INOUT(5,YDER)
      GO TO 9999
C
  130 IF(THETAT-THECPS)170,180,180
  140 IF(THETAT-THECPS)170,145,145
  145 IF(THETAT.GT.4.0)GO TO 190
      GO TO 170
  150 IF(THETLT-THECPL)200,210,210
  160 IF(THETLT-THECPL)200,230,230
C
  170 THECPS=THETAT
      GO TO 240
```



```
C
  180 ITER=0
      THECPS=THETLM
      THECPL=THETIC
      PSV(1)=.FALSE.
      IF(OOV(2))CALL INOUT(7,YDER)
      GO TO 50
C
  190 DO 195 I=1,5
  195 PSV(I)=.FALSE.
      GO TO 120
C
  200 HOLD=THECPL
      THECPL=THETLT
      GO TO 240
C
  210 PSV(3)=.FALSE.
      THECPL=HOLD
      REG=ALPHA(ALPH0,ITER-1)
      DO 212 I=1,4
  212 STACK(I)=STACK(I+1)
  220 IF(OOV(2))CALL INOUT(7,YDER)
      ITER=ITER-1
      DO 225 I=1,N
  225 XT(I)=XT(I)-DELTXT(I)
      GO TO 70
C
  230 PSV(4)=.FALSE.
      INTERR=5
      GO TO 220
C
C COMPUTE JACOBI MATRIX (ETOT-ARRAY USED AS TEMPORARY WORKING STORAGE)
  240 DO 245 I=1,M
      CALL YDER(3,Q(I),ETOT)
      DO 245 KK=1,N
  245 FIXT(I,KK)=ETOT(KK)
C
C BUILD ITERATION-STEP MATRIX (SQUARE, SYMMETRIC, POSITIVELY-DEFINED)
      DO 250 I=1,N
      DO 250 KK=I,N
      AUXIL=0.0
      DO 248 J=1,M
  248 AUXIL=AUXIL+FIXT(J,I)*W(J)*FIXT(J,KK)
      IF(I.EQ.KK)AUXIL=AUXIL+REG/FLOAT(N)
      V(I,KK)=AUXIL
  250 V(KK,I)=AUXIL
C
C BUILD ITERATION-STEP VECTOR; STORE TEMPORARILY IN ETOT-ARRAY
      DO 260 I=1,N
      ETOT(I)=0.0
      DO 260 KK=1,M
  260 ETOT(I)=ETOT(I)+FIXT(KK,I)*W(KK)*FXT(KK)
C INVERT STEP MATRIX
      DO 261 I=1,N
      DO 261 KK=1,N
      JJ=(I-1)*N+KK
  261 VVV(JJ)=V(I,KK)
      CALL INVKA(VVV,N,DET,LOOLOO,MOOMOO)
C
      DO 267 I=1,N
```

```
      DO 267 KK=1,N
      JJ=(I-1)*N+KK
  267 V(I,KK)=VVV(JJ)
C DEGENERATION SWITCH (TOO SMALL A VALUE OF DETERMINANT)
      IF(DET.GT.DEGER)GO TO 270
      IF(.NOT.PSV(4))GO TO 280
      IF(PSV(1))GO TO 180
      GO TO 190
C
C COMPUTE VECTOR OF UNKNOWNS" INCREMENTS
  270 DO 272 I=1,N
      DELTXT(I)=0.0
      DO 272 KK=1,N
  272 DELTXT(I)=DELTXT(I)-V(I,KK)*ETOT(KK)
C
C COMPUTE VARIANCE VECTOR
C       NOTE: ILL-CONDITION & ERROR SWITCHES INCORPORATED IN CYCLE
      DO 278 I=1,N
      ETOT(I)=V(I,I)
      IF(ETOT(I))274,276,276
  274 IF(.NOT.PSV(4))GO TO 280
      IF(PSV(1))GO TO 180
      GO TO 190
  276 ETOT(I)=SQRT(ETOT(I))
      IF(.NOT.PSV(4))ETOT(I)=THETAT*ETOT(I)
  278 CONTINUE
      GO TO 281
C
C RESTORE REGULARIZER VALUE
  280 REG=SAVREG
      MSWICH=10
      CALL INOUT(8,YDER)
      MSWICH=0
      GO TO 240
  281 IF(OOV(1))CALL INOUT(6,YDER)
C
C INTERRUPT SWITCH (FOUR CRITERIA FOR NEWTON AND JUST ITRMAX FOR SKATE)
      IF(.NOT.PSV(4))GO TO 900
      IF((ITER.GE.5).OR.(THETAT.LT.1.0))GO TO 332
 1281 IF(ITER.GT.ITRMAX)GO TO 290
      IF(.NOT.PSV(1))GO TO 350
      IF(THETAT.LT.TLIM)GO TO 300
      DO 282 I=1,N
      IF((0.LT.(XT(I)-ETOT(I))).OR.((XT(I)+ETOT(I)).LE.0))GO TO 284
  282 CONTINUE
      GO TO 320
  284 DO 286 I=1,M
      IF(FXT(I).GT.DEVMIN)GO TO 350
  286 CONTINUE
      GO TO 330
C
  290 INTERR=1
      GO TO 340
  300 INTERR=2
      GO TO 340
  320 INTERR=3
      GO TO 340
  330 INTERR=4
      GO TO 340
  332 CALL INOUT(9,YDER)
```



```
      IF(MSWICH-11)1281,334,1281
  334 CALL INOUT(8,YDER)
      GO TO 35
  340 PSV(4)=.FALSE.
      IF(OOV(2))CALL INOUT(7,YDER)
      GO TO 70
C
C ACTUAL ITERATING
  350 DO 355 I=1,N
  355 XT(I)=XT(I)+DELTXT(I)
      ITER=ITER+1
      GO TO 70
C
  900 CALL INOUT(6,YDER)
C
C KATOK ITERATION SCHEME IMPLEMENTED: END
C ***************************************
C
C
C ANALYSIS OF PEAK POSITIONS AND INTENSITIES
      CALL INOUT(9,YDER)
      IF(MSWICH-11)9990,9980,9990
 9980 CALL INOUT(8,YDER)
      GO TO 35
C
C DRAW SECTION GRAPH
 9990 IF(.NOT.OOV(3))GO TO 9999
      DO 9995 I=1,M
      CALL YDER(5,Q(I),ETOT)
 9995 FXT(I)=ETOT(I)
      CALL GRAPH(M,Q,YQ,YQT,FXT)
 9999 NSECT=NSECT+1
      IF(NSECT.LE.MM)GO TO 30
99999 MSWICH=12
      CALL INOUT(8,YDER)
      MSWICH=0
      RETURN
      END
```

REFERENCES

1. Gadjokov V. JINR, E10-12352, Dubna, 1979.
2. Alexandrov L., Gadjokov V. J.of Radioanal.Chem., 1971, 9, pp.279-292.

Received by Publishing Department
on May 17, 1979