SHARE PROGRAM LIBRARY AGENCY

PROGRAM NUMBER

150005

University of Miami

1365 MEMORIAL DRIVE - CORAL GABLES, FLORIDA

(305) - 284-6257

SHARE PROGRAM LIBRARY SUBMITTAL FORM

SHARE PROGRAM LIBRARY AGENCY
Triangle Universities Computation Center
Post Office Box 12076
Research Triangle Park, North Carolina
27709 USA
Attention: Mr. Joe Ragland

SPLA CONTROL NUMBER: SHR-169

This form should be completed and submitted with the program package to the SHARE Program Library Agency at the address shown above. Standards and instructions for submitting programs are in the "SHARE Program Library Standards Manual".

(1) Program Number (to be filled in by SPLA) 360D-15.0.005

(2) System Type (machine) 370/155

(3) Search Key ..... Queues, Markov-chains

(4) Programming Language ..... FORTRAN G

(5) Author's Name and Address ..... Winfried K. Grassmann,
T.K. Ngai

(6) Direct Inquiries to Name and Address ..... Dept. of Computational Science,
(if different than Author) University of Saskatchewan,
Saskatoon, Saskatchewan, Canada.

(7) Title of Program ..... Transient solutions for Markov
Chains.

(8) Submitter's Installation Membership Code..... S.A.S.

(9) Submitter's Own Program Identification and Suffix(Optional)...

(10) Primary Subject Code..... 15 0

(11) Operating or Monitor System Required NO

(12) New or Revision Code (if revision, show prior Program Number in Item 1).. New

(13) Year Completed..... 1974

(14) Date of Submittal..... November 5

(15) Documentation (number of original pages submitted)..... 25

(16) Abstract (should contain sufficient information for a reader to determine the value of the program). Listed on the reverse side of this form are subjects which may serve as a guide for a descriptive abstract.

SHARE PROGRAM LIBRARY SUBMITTAL FORM

Subject Guide:

a. Purpose
b. Programming Language used
c. Version and modification level or release number
d. Field of application
e. Type of routine (main program, subroutine, etc.)
f. Specific description of machine requirements

ABSTRACT

The program finds transient solutions for continuous Markov-chains with sparse transition matrices. Such Markov-chains occur frequently in queueing theory, expecially in situations with more than one queue. The program is written in FORTRAN G. It consists of less than 200 statements and has no subroutines. The method employed is randomization. The algorithm is described by W. Grassmann in "Transient Solutions in Simple Queues", Working Papers of the Department of Computational Science, University of Saskatchewan, 74-R-2, page 7.

DISCLAIMER

Triangle Universities Computation Center (TUCC) **serves** solely as the distribution agent for contributed programs and does not test or maintain them. They are distributed essentially in the original form submitted by the author. Neither TUCC nor SHARE, INC., makes any warranty, expressed or implied, as to the documentation, function, or performance of the contributed programs.

(Please attach additional pages if necessary).....Total pages attached 0

Permission to Publish

"I hereby give the SHARE Program Library Agency permission to reprint, reproduce, and distribute this program."

(17) Signature of Submitter and Date Oct 31, 74 W. Grassmann

(18) Signature of Installation Addressee B.A. Mulcahy

PROGRAM DESCRIPTION FOR:

Transient solutions in continuous Markov chains.

by

W.K. Grassmann and T.K. Ngai,
Department of Computational Science,
University of Saskatchewan,
Saskatoon, Saskatchewan,
Canada.

TABLE OF CONTENTS

I. Introduction:

As much documentation as possible has been included in the program. Arrays and variables are also defined.

Since a program may contain over a hundred states, the tabulation could be difficult for the computer to do if using multi-dimensional matrices. So, throughout the program, only one-dimensional arrays have been employed.

II. The Method:

Let a_{ij} denote the transition rates of a continuous Markov process. We want to find $\pi_j(t)$, the probability of being in state j at time t, given that the probability of being in state i now is $\pi_i(0)$. This can be done using the following algorithm:

(a) find:

$$F = \max_i \left[\sum_j aij\right] \qquad (1)$$

and calculate:

$$P_{ij} = a_{ij} / F \qquad i \neq j \qquad (2)$$

$$P_{ii} = 1 - \sum_j aij/F \qquad (3)$$

Note that F exceeds all a_{ij}, and that consequently all P_{ij}, including P_{ii}, are between 0 and 1.

(b) Calculate π_j^n recursively as:

$$\pi_i^0 = \pi_i(0)$$

$$\pi_j^n = \sum_i \pi_i^{n-1} P_{ij} \qquad (4)$$

(c) Let $P_n(\mu)$ be the Poisson distribution with average μ. Then one obtains the final result as:

$$\pi_j(t) = \sum_{n=0}^{\infty} \pi_j^n P_n(Ft) \qquad (5)$$

To understand this algorithm, one notes first that π_j^n is the transient solution of a discrete Markov chain with the transition probabilities p_{ij}. (It is easily verified that the p_{ij} form the transition probabilities of such a discrete chain, since they are between zero and 1 and add up to 1). The points of transition of this discrete chain are not 1, 2, 3, 4 or other deterministically spaced epochs. Rather, the number of transition points in (0, t) is a random variable, having a Poisson distribution with average Ft. There is thus an average Ftp_{ij} transition from i to j during t, giving:
$Ftp_{ij} = FT(a_{ij}/F) = a_{ij}t$

The new process is also Markovian, as can be easily shown.

To sum up, one can say that the continuous Markov-chain is interpreted as a combination of a discrete Markov-chain and the Poisson distribution.

In this new process, the system is in state j at time t, given there are n points of transition during (0,t), and one is in state j after those n transitions. This argument leads directly to formula (5), π_j^n being the probability of being in state j after n points of transitions, and $P_n(Ft)$ being the probability of having n points of transitions.

follows: For each non-zero entry, one has to give the row, the column and the transition rate. Rows and columns have to be identified by the state identification number as given in (iii). Zero elements of the matrix need not be entered. The diagonal elements, i.e. elements with the same row and column number are calculated automatically and must not be entered.

(v) The time for which the probabilities should be calculated.

(vi) The initial probabilities, i.e. the probabilities of being in a certain state at time zero.

The number of states must not exceed 60, and the number of non-zero elements in the transition matrix (excluding the diagonal) must not exceed 840.

The data format is as follows:

Card 1, columns 1-5: FORMAT (I_5)

The number of problems to be run. After this card follows the first card specifying the problem.

First problem-card, columns 1-80: FORMAT (20A4)

Problem title (alphanumeric field).

Second problem card, columns 1-5: FORMAT (I5)

The number of different states.

State identification card(s): (FORMAT 16 I5)

An identification number for each state, each in a numeric field of 5 columns (i.e. column 1-5: first state number, column 6-10, second state number, ..., column 76-80: 16th state number, column 1-5 next card: 17th state number etc.).

Number of entries card, column 1-5: FORMAT (I5)

Number of non-zero entries in the transition matrix (excluding diagonal).

Transition card(s): FORMAT (4(2I5, F10.5))

The triplets: originating state (row state) destinating state (column state) and transition rate, each occupying 3 adjacent fields of 5 columns. Thus:

columns 1-5: First originating state

columns 6-10: First destinating state

columns 11-15: First transition rate.

columns 16-20: Second originating state

columns 21-25: Second destinating state

columns 26-30: Second transition rate,

etc.

In this way, the entire matrix is to be entered, always 3 entries per card. The transition rate must be entered as a real number, that is it must contain a decimal point.

The time card column 1-10: FORMAT (F10.5)

Time for which probabilities are to be calculated. Decimal assumed to be between column 5 and 6 unless punched otherwise.

Initial-probability cards: FORMAT (8 F10.5)

For all states, initial probabilities have to be provided which are to be punched 8 per card, each occupying 10 columns with the decimal point assumed between column 5 and 6. The order of the initial probabilities must coincide with the order of the state identification numbers on the state identification card.

V. Output

The output is a table showing the state, its initial probability and its final probability final time given in the time card (Appendix 1).

STATE	INITIAL PROBABILITY	FINAL PROBABILITY

Sample Deck

The following 4 sample problems were solved:

(1) The transient solutions of a Poisson-queue with arrival rate of 5 and a service rate of 8 and a maximum gueue length of 20;

(2) The probability of waiting longer than 4 time units in a Poisson-queue with LIFØ - discipline, an arrival rate of 13, a service rate of 20, and a maximum of 4 elements in the system;

(3) The transient solution for a system with two parallel queues. The arrival rate is 0.5, the service rate of the first queue is 0.75 and the service rate of the second queue is 0.5. Arrivals

The last 3 cards of Problem 1 give the initial probabilities. These are all zero except for state 0 (which is the first state since all initial probabilities must be given in the same order as the state numbers in Card 4 and 5). For state zero, the probabbility is 1, which means that we start for sure with an empty queue. The final results thus give the probability of having i elements in line after 10 time units, given the queue was empty initially. This output is given in Appendix 2. This completes the discussion of the first problem.

The second sample problem is really a waiting time problem. The first card of the second sample problem gives it the title, "LIFO-Case", because the waiting time under LIFO-discipline is to be calculated. There an 7 states (second card of second problem), each state being characterized by a two digit state number. The first digit always stands for the place in line the element in question occupies, the second number indicates how many elements are in the system. 11 means thus that our element's first in line, while one element in system, 12 for means our element first in line, while 2 elements are in the system etc. Because we allow only 3 elements in the queue, 33 is the last state. There is also a state 0, which just stands for "element has started service". What we really want to know is the probability that the element is in the state "started service" after time t, which obviously is equal to the cumulative waiting time distribution P (waiting time $<$t). After all the possible states (0, 11, 12, 13, 22, 23, 33) are listed, the next card indicates that there are a total of 9 transitions. We have a

arrival always joins the shorter queue, and if both queues are of the same length, it joins each queue with equal likelihood. The arrival rate is 0.5 and the service rates are 0.75 and 0.5 for the first and second queue respectively. This gives a total of 68 transitions (see Appendix 1).

Let us assume now the second queue is opened only if there are 4 elements in the first queue. What is the probability of having i element in the first and j in the second queue 4 time units after opening the second queue. In order to answer this question, we just have to punch a 4 on the time card, and a 1.00 in the field of the initial probability of state 40 which is the 20th probability. All other fields may be left empty, indicating a probability of zero.

The last example finally gives the case of 2 sequential queues as the title suggests. The first queue has an arrival rate of 1 and a service rate of 2. All elements having completed service in the first queue go to the second queue which has a service rate of 1.5. If the line of the second queue reaches 3, the first server is blocked and stops serving until the second line decreases. Also, if the number in the first queue reaches 3, all arrivals balk. What is the probability for the different states after 20 time units, given they are initially as indicated in the initial probability cards? Leaving off the details of the data cards, we just direct the reader to Appendix 1 and 2.

APPENDIX 1: Listing of Sample Problems

```
    4
POISSON QUEUE
   21
    0    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
   16   17   18   19   20
   40
    0    1        5.0    1    2        5.0    1    0        8.0    2    3        5.0
    2    1        8.0    3    4        5.0    3    2        8.0    4    5        5.0
    4    3        8.0    5    6        5.0    5    4        8.0    6    7        5.0
    6    5        8.0    7    8        5.0    7    6        8.0    8    9        5.0
    8    7        8.0    9   10        5.0    9    8        8.0   10   11        5.0
   10    9        8.0   11   12        5.0   11   10        8.0   12   13        5.0
   12   11        8.0   13   14        5.0   13   12        8.0   14   15        5.0
   14   13        8.0   15   16        5.0   15   14        8.0   16   17        5.0
   16   14        8.0   17   18        5.0   17   16        8.0   18   19        5.0
   18   17        8.0   19   20        5.0   19   18        8.0   20   19        8.0
     10.0
  1.000          .000          .000          .000          .000          .000          .000
   .000          .000          .000          .000          .000          .000          .000
   .000          .000          .000          .000          .000          .000          .000
LIFO CASE
    7
    0   11   12   13   22   23   33
    9
   11   22        13.0   11    0        20.0   12   23        13.0   12    0        20.0
   13    0        20.0   22   33        13.0   22   11        20.0   23   12        20.0
   33   22        20.0
     4.0
   .426          .277          .180          .117          .000          .000          .000          .000
TWO PARALLEL QUEUES
   25
   00   01   02   03   04   10   11   12   13   14   20   21   22   23   24   30
   31   32   33   34   40   41   42   43   44
   68
   00   01        .125   00   10        .125   01   11        .250   01   00        .500
   02   12        .250   02   01        .500   03   13        .250   03   02        .500
   04   14        .250   04   03        .500   10   11        .250   10   00        .750
   11   12        .125   11   21        .125   11   01        .750   11   10        .500
   12   22        .250   12   02        .750   12   11        .500   13   23        .250
   13   03        .750   13   12        .500   14   24        .250   14   04        .750
   14   13        .500   20   21        .250   20   10        .750   21   22        .250
   21   11        .750   21   20        .500   22   23        .125   22   32        .125
   22   12        .750   22   21        .500   23   33        .250   23   13        .750
   23   22        .500   24   34        .250   24   14        .750   24   23        .500
   30   31        .250   30   20        .750   31   32        .250   31   21        .750
   31   30        .500   32   33        .250   32   31        .500   32   22        .750
   33   34        .125   33   43        .125   33   23        .750   33   32        .500
   34   44        .250   34   24        .750   34   33        .500   40   41        .250
   40   30        .750   41   42        .250   41   31        .750   41   40        .500
   42   43        .250   42   32        .750   42   41        .500   43   44        .250
   43   33        .750   43   42        .500   44   34        .750   44   43        .500
     4.0
   .000          .000          .000          .000          .000          .000          .000          .000
   .000          .000          .000          .000          .000          .000          .000          .000
   .000          .000          .000        0.000        1.000          .000          .000          .000
```

```
      .000       .000       .000       .000       .000       .000       .000       .000
SEQUENTIAL QUEUES
   16
   00   01   02   03   10   11   12   13   20   21   22   23   30   31   32   33
   33
   00   10        1.0   01   11        1.0   01   00        1.5   02   12        1.0
   02   01        1.5   03   13        1.0   03   02        1.5   10   20        1.0
   10   01        2.0   11   21        1.0   11   02        2.0   11   10        1.5
   12   22        1.0   12   03        2.0   12   11        1.5   13   23        1.0
   13   12        1.5   20   30        1.0   20   11        2.0   21   31        1.0
   21   12        2.0   21   20        1.5   22   32        1.0   22   13        2.0
   22   21        1.5   23   33        1.0   23   22        1.5   30   21        2.0
   31   22        2.0   31   30        1.5   32   23        2.0   32   31        1.5
   33   32        1.5
      20.0
      .107       .004       .031       .009       .110       .090       .022       .080
      .128       .044       .022       .003       .201       .045       .066       .038
/*
```

APPENDIX 2: Output for Sample Problems

POISSON QUEUE

TIME = 10.00000

STATE	INITIAL PROBABILITY	FINAL PROBABILITY
0	0.100000E 01	0.375240E 00
1	0.000000E 00	0.234519E 00
2	0.000000E 00	0.146561E 00
3	0.000000E 00	0.915859E-01
4	0.000000E 00	0.572260E-01
5	0.000000E 00	0.357520E-01
6	0.000000E 00	0.223324E-01
7	0.000000E 00	0.139473E-01
8	0.000000E 00	0.870828E-02
9	0.000000E 00	0.543572E-02
10	0.000000E 00	0.339200E-02
11	0.000000E 00	0.211605E-02
12	0.000000E 00	0.131972E-02
13	0.000000E 00	0.822928E-03
14	0.000000E 00	0.513120E-03
15	0.000000E 00	0.197225E-03
16	0.000000E 00	0.122782E-03
17	0.000000E 00	0.764178E-04
18	0.000000E 00	0.475647E-04
19	0.000000E 00	0.296214E-04
20	0.000000E 00	0.184706E-04

TIME = 4.00000

STATE	INITIAL PROBABILITY	FINAL PROBABILITY
0	0.426000E 00	0.999997E 00
11	0.277000E 00	0.711522E-11
12	0.180000E 00	0.507258E-17
13	0.117000E 00	0.821214E-41
22	0.000000E 00	0.976773E-11
23	0.000000E 00	0.605803E-17
33	0.000000E 00	0.878406E-11

SEQUENTIAL QUEUES

TIME = 20.00000

STATE	INITIAL PROBABILITY	FINAL PROBABILITY
0	0.107000E 00	0.198473E 00
1	0.400000E-02	0.132324E 00
2	0.310000E-01	0.838435E-01
3	0.900000E-02	0.427548E-01
10	0.110000E 00	0.102525E 00
11	0.900000E-01	0.727381E-01
12	0.220000E-01	0.534425E-01
13	0.800000E-01	0.424481E-01
20	0.128000E 00	0.574177E-01
21	0.440000E-01	0.464868E-01
22	0.220000E-01	0.316808E-01
23	0.300000E-02	0.314016E-01
30	0.201000E 00	0.444651E-01
31	0.450000E-01	0.210077E-01
32	0.660000E-01	0.180256E-01
33	0.380000E-01	0.209372E-01

APPENDIX 3: Program Listing

```
C     ************************************************************
C     *                                                          *
C     *            T R A N S I E N T    S O L U T I O N S        *
C     *                                                          *
C     *              I N    S I M P L E    Q U E U E S           *
C     *                                                          *
C     ************************************************************
C
C         OBJECT : THIS PROGRAM FINDS THE PROBABILITY OF BEING
C                  IN STATE J AFTER N-POINTS OF TRANSITIONS FOR
C                  A SYSTEM OF MORE THAN ONE QUEUE.
C                  THE ALGORITHM FOR THE PROGRAM IS DESCRIBED
C                  ON PAGE 7, 'TRANSIENT SOLUTIONS IN SIMPLE
C                  QUEUES' BY W.K. GRASSMANN.
C
C     ************************************************************
C     *                                                          *
C     *    AUTHOR :      W.K. GRASSMANN & T.K. NGAI              *
C     *                                                          *
C     ************************************************************
C
C
C         LANGUAGE :          FORTRAN IV
C
C     ************************************************************
C     *                                                          *
C     *    ARRAYS IN THE PROGRAM :                               *
C     *              DESTIN - GIVES THE DESTINATING STATES IN THE*
C     *                       INITIAL MATRIX                     *
C     *              MARKOV - THIS GIVES THE INITIAL RATE FOR EACH*
C     *                       ORIGINATING STATE AND ITS RESPECTIVE*
C     *                       DESTINATING STATE IN THE INITIAL MATRIX
C     *              ORIGST - STORES THE ORIGINATING STATES IN THE*
C     *                       INITIAL MATRIX                     *
C     *              PII    - AN ARRAY TO STORE THE INITIAL      *
C     *                       PROBABILITY OF BEING IN A CERTAIN  *
C     *                       STATE.  THIS ARRAY IS FOR OUTPUT   *
C     *                       PURPOSE                            *
C     *              PIN    - NEW PROBABILITY OF BEING IN A STATE*
C     *              PIO    - INITIAL PROB OF BEING IN A STATE   *
C     *              PIT    - THIS TABLE GIVES THE SOLUTION.  IT *
C     *                       TELLS THE PROB OF BEING IN A CERTAIN
C     *                       STATE AFTER A GIVES AMOUNT OF ITERATIONS
C     *                       IN THE SPECIFIED TIME              *
C     *              STATE  - THIS ARRAY GIVES THE STATES IN THE *
C     *                       PROGLEM                            *
C     *              TRPROB - TRANSITION PROB FROM ONE STATE TO  *
C     *                       THE OTHER                          *
C     *                                                          *
C     *    VARIABLES IN THE PROGRAM :                            *
C     *              CITER  - A # TO CONTROL THE VALUE OF 'ITER' *
C     *                       (SEE BELOW)                        *
C     *              F      - AN ARBITRARY VALUE TO CONTROL THE  *
C     *                       VALUES OF TRPROB (SEE ABOVE).  THE *
C     *                       VALUE OF F IS TAKEN TO BE THE MAX  *
C     *                       OF MARKOV(J) IN THIS PROGRAM       *
C     *              ITER   - # OF ITERATIONS FOR CALCULATING THE*
C     *                       FINAL PROBABILITY                  *
C     *              J      - INDEX                             *
```

```
C      *             K       - INDEX                                          *
C      *             L       - INDEX                                          *
C      *             M       - INDEX                                          *
C      *             MARK    - SPECIAL INDEX TO MARK THE # OF ITEMS           *
C      *                       IN THE INITIAL MATRIX                          *
C      *             NPROB   - # OF PROBLEMS IN THIS RUN                      *
C      *             NSTATE  - TOTAL # OF STATES                              *
C      *             PN      - POISSON DISTRIBUTION                           *
C      *             PROKEY  - SPECIAL KEY TO CONTROL # OF PROBLEMS           *
C      *                       IN THIS RUN                                    *
C      *             TIME    - THE SPECIFIED TIME INTERVAL                    *
C      *             TITLE   - THE TITLE OF THE PROBLRM                       *
C      *                                                                      *
C      ************************************************************************
C
C          A NOTE ON THE PROGRAM :
C                    THE PROGRAM IS MAINLY DIVIDED INTO 3 PARTS :
C                    1. INPUT, INITIALIZATION AND FIND THE
C                       DIAGONAL ENTRIES OF THE INITIAL MATRIX;
C                    2. CALCULATE THE REQUIRES PROBABILITIES.
C                       THIS IS THE MAIN PART OF THE PROGRAM;
C                    3. PRINT OUT THE REQUIRED VALUES
C
C      ************************************************************************
C      *                                                                      *
C      *   REMARK : THE PROGRAM CAN BE FITTED INTO PROBLEMS WITH              *
C      *            VARIOUS #S OF STATES BY CHANGING THES SIZES               *
C      *            OF THE ARRAYS PROVIDED THAT THE TOTAL STORAGE             *
C      *            AREA FOR THE OBJECT CODE AND THE ARRAYS DOES              *
C      *            NOT EXCEED THAT PROVIDED BY THE SYSTEM                    *
C      *                                                                      *
C      ************************************************************************
C
C
C
 1        INTEGER CITER,ITER,J,K,L,M,MARK,NPROB,NSTATE,PROKEY
 2        INTEGER DESTIN(900),ORIGST(900),STATE(60),TITLE(20)
 3        REAL F,PN,TIME
 4        REAL MARKOV(900),PII(60),PIN(60),PIO(60),PIT(60),TRPROB(900)
C
C
C             READ IN :
C                1. THE TITLE;
C                2. # OF STATE & STATE NUMBERS;
C                3. INITIAL MATRIX, EXCEPT THE DIAGONAL ENTRIES;
C                4. INITIAL PROBABILITIES OF BEING IN A CERTAIN
C                   STATE
C
 5        READ 1000,NPROB
 6     1000 FORMAT(I5)
 7        DO 250 PROKEY=1,NPROB
 8        READ 1001,(TITLE(J),J=1,20)
 9     1001 FORMAT(20A4)
10        READ 1005,NSTATE
11     1005 FORMAT(I5)
12        READ 1010,(STATE(M),M=1,NSTATE)
13     1010 FORMAT(16I5)
14        READ 1005,MARK
C              PLEASE READ THE FORMAT STATEMENT 1005 ABOVE
15        READ 1015,(ORIGST(K),DESTIN(K),MARKOV(K),K=1,MARK)
16     1015 FORMAT(2I5,F10.5,2I5,F10.5,2I5,F10.5,2I5,F10.5)
17        READ 1020,TIME
```

```
 1020 FORMAT(F10.5)
      READ 1025,(PII(M),M=1,NSTATE)
 1025 FORMAT(8F10.5)
      DO 10 K=1,MARK
      DO 10 M=1,NSTATE
      IF(ORIGST(K).EQ.STATE(M))  ORIGST(K)=-M
      IF(DESTIN(K).EQ.STATE(M))  DESTIN(K)=-M
10    CONTINUE
      DO 12 K = 1,MARK
      IF(ORIGST(K).GE.0)   GO TO 9
      IF(DESTIN(K).GE.0)   GO TO 9
      ORIGST(K)=-ORIGST(K)
12    DESTIN(K)=-DESTIN(K)
      GO TO 100
9     PRINT 1111
1111  FORMAT('1 TRANSITION',2X,I10,2X,'INVALID')
      STOP
C
C
C          FIND THE DIAGONAL ENTRIES OF THE INITIAL MATRIX
C
100   DO 101 M=1,NSTATE
      ORIGST(MARK+M)=M
      DESTIN(MARK+M)=M
 101   MARKOV(MARK+M)=0.
      DO 102 K =1,MARK
      KORIG=ORIGST(K)+MARK
102   MARKOV(KORIG)=MARKOV(KORIG)-MARKOV(K)
      MARK=MARK+NSTATE
C
C
C          INITIALISE PIN(J), PIT(J), ARRAYS TO FIND THE
C          SUCCESSIVE PROBABILITY AND FINAL PROBABILITY
C          RESPECTIVELY
C
      DO 140 J=1,NSTATE
      PIO(J)=PII(J)
      PIN(J)=0.
 140  PIT(J)=0.
C
C
C          FIND F, AN ARBITRARY VALUE TO CONTROL TRPROB,
C          TRANSITION PROBABILITY FROM STATE I TO STATE J
C          AT THE TIME INTERVAL BETWEEN 0 AND TIME
C
      F=0.
      DO 150 J=1,MARK
      IF(ORIGST(J).NE.DESTIN(J))GO TO 150
      IF(ABS(MARKOV(J)).GT.F)F=ABS(MARKOV(J))
 150  CONTINUE
C
C
C          CALCULATE THE PROBABILITY OF TRANSITION FROM
C          STATE I TO STATE J
C
      DO 155 J=1,MARK
      IF(ORIGST(J).EQ.DESTIN(J))GO TO 152
      TRPROB(J)=MARKOV(J)/F
      GO TO 155
 152  TRPROB(J)=MARKOV(J)/F+1.
 155  CONTINUE
C
```

```
      C
      C           CALCULATE THE CONTROL # OF THE # OF ITERATIONS,
      C           CITER, FOR THE EVALUATION OF THE FINAL PROBABILITY
      C           AND THE PROBABILITY OF BEING IN STATE J AFTER
      C           N POINTS OF TRANSITIONS, PN
      C
   58         FT=F*TIME
   59         CITER=FT+4*SQRT(FT)+4.9
   60         INTER=FT-4*SQRT(FT)
   61         PN=0
   62         IF(INTER.LE.0) PN=1
   63         PNT=0
      C
      C
      C           EVALUATE THE SUCCESSIVE TRANSITION PROBABILITY,
      C           THAT IS, PIN(J), J=1,NSTATE
      C
   64         ITER=0
   65    167  DO 169 K =1,MARK
   66         L=ORIGST(K)
   67         J=DESTIN(K)
   68    169  PIN(J)=PIN(J)+PIO(L)*TRPROB(K)
      C
      C
      C           CALCULATE PIT(J), THE PROBABILITY IN STATE J
      C           AFTER CITER POINTS OF TRANSITIONS AT THE GIVEN
      C           TIME
      C
   69          IF(ITER.EQ.INTER) PN=1
   70         PNT=PNT+PN
   71         DO 175 J=1,NSTATE
   72    175  PIT(J)=PIT(J)+PIN(J)*PN
   73         ITER=ITER+1
   74         IF(ITER.GT.CITER)GO TO 185
   75         PN=PN*FT/ITER
   76         DO 180 J=1,NSTATE
   77         PIO(J)=PIN(J)
   78    180  PIN(J)=0.
   79         GO TO 167
      C
      C
      C           PRINT THE FINAL PROBABILITY
      C
   80    185  PRINT 1050,(TITLE(J),J=1,20)
   81   1050  FORMAT('1',20A4)
   82          DO 201 J=1,NSTATE
   83    201  PIT(J)=PIT(J)/PNT
   84         PRINT 1051,TIME
   85   1051  FORMAT('0','TIME = ',F10.5)
   86         PRINT 1052
   87   1052  FORMAT('-     STATE       INITIAL PROBABILITY       FINAL PROBABILITY
           *')
   88         DO 200 J=1,NSTATE
   89    200  PRINT 1055,STATE(J),PII(J),PIT(J)
   90   1055  FORMAT('0',4X,I4,9X,E14.6,10X,E14.6)
   91    250  CONTINUE
      C
      C
   92         STOP
   93         END
```