An efficient procedure for selecting equal probability samples

JOHN F. WALSH

Fordham University, Bronx, New York 10458

FORTTRAN code that implements an algorithm for selecting equal probability samples is given. The procedure is brief and decides to include or not include a record as it is called. Consequently, storage requirements are minimal. The procedure gives the exact number of values specified, with each record in the population having an equal probability for inclusion in the sample, including the last.

Many behavioral science research applications require the unbiased selection of n sample records from a population of records. The term, "record," is used to include Ss for an experiment, words for lists, or values to be used in sequencing material or Ss.

A direct approach to the problem of selecting records is to use a pseudorandom number generator as part of the procedure that indicates a record. The selected record is compared against all those previously called, and, if no duplicate is found, the record is added to a list. While this procedure works, it is limited in terms of memory size to store the records and is slow if the number of records

desired is at all close to the population size (Lehman & Bailey, 1968).

The procedure described below is an efficient algorithm for selecting n records. It describes whether to accept or reject each record as it is called. The algorithm selects the record in the $i^{th} + 1$ position with probability $(n - m)/(N - i)$ if m items have already been selected from the population, N . The procedure gives exactly n elements, and the sample is completely unbiased. In particular, the probability of the last element in the population being selected is n/N (Knuth, 1969).

Moreover, it usually will not be necessary to pass over all records in order to determine the n cases of

interest. With this approach, the repetitive search for duplicates common to most other techniques is eliminated, and no storage is required for the items selected. The procedure, however, requires that the population size be known and that the items in the population or an equivalent pointer system are in sequential order.

Table 1 presents the FORTRAN code implementing the procedure. In addition to the code for the main program, a subroutine to generate pseudorandom numbers uniformly in the unit interval is required. In the present example, the random number generator, RANDU, from IBM's (1969) Scientific Subroutine Package is utilized. However, the code for the subroutine that generates the random numbers will depend upon the word size of the computer at the user's installation.

Initially, the user provides the values for three variables—IY, NITEM, and NPOP. IY is a 7- to 9-digit odd integer that starts the random number generator. NITEM is the number of records, Ss, or words desired. NPOP is the size of the population from which the items are to be called. As a record meets the criterion for inclusion, the ordinal position of the record in the main file is written. For example, if it were desired to select a random sample of n Ss from a college registrar's list where the list is in sequential order by student identification number, the FORTRAN code in Table 1 would indicate the location of the student's number in the main list. The names of the students would then be collected.

The code also writes out the value, IY, to be used to start the random number generator for the next application. This feature insures that independent streams of random numbers will be utilized each time the program is run.

TABLE 1
FORTRAN CODE FOR SELECTING EQUAL PROBABILITY SAMPLES

```

      READ(1,2) IY, NITEM, NPOP
      2 FORMAT(19,2I6)
      NPOS=0
      NCALL=0
      10 CALL RANDU(IY,IY,YFL)
C   TEST FOR INCLUSION
      A=NPOP-NPOS
      B=NITEM-NCALL
      IF (A*YFL.GT.B) GO TO 5
      NCALL=NCALL+1
      NPOS=NPOS+1
      WRITE(3,3) NPOS
      3 FORMAT(17H SELECT RECORD , I6)
C   IS THE SAMPLE COMPLETE?
      IF ( NCALL-NITEM) 10, 15, 15
      5 NPOS=NPOS+1
      GO TO 10
      15 WRITE(3,4) IY
      4 FORMAT(36H INITIAL VALUE FOR NEXT APPLICATION ,I10)
      CALL EXIT
      END
      SUBROUTINE RANDU(IX,IY,YFL)
      IY=IX*65539
      IF ( IY.LT. 0) IY=IY+2147483647+1
      YFL=FLOAT(IY)*.4656613E-9
      RETURN
      END
  
```

REFERENCES

- IBM. Scientific Subroutine Package. White Plains, N.Y.: International Business Machines Corp., 1969, Manual H 20-0205-3.
- KNUTH, D. *The art of computer programming*. Vol. 2. *Seminumerical algorithms*. Reading: Addison & Wesley, 1969.
- LEHMAN, R., & BAILEY, D. *Digital computing: Fortran IV and its applications in behavioral science*. New York: Wiley, 1968.