AGENCY FOR INTERNATIONAL DEVELOPMENT
WASHINGTON, D. C. 20523

BIBLIOGRAPHIC INPUT SHEET

FOR AID USE ONLY
Batch 63

1. SUBJECT CLASSIFICATION	A. PRIMARY	TEMPORARY
	B. SECONDARY	

2. TITLE AND SUBTITLE
Software standards manual

3. AUTHOR(S)
(101) Mich.State Univ. Dept.of Agr.Economics

4. DOCUMENT DATE	5. NUMBER OF PAGES	6. ARC NUMBER
1976	46p.	ARC

7. REFERENCE ORGANIZATION NAME AND ADDRESS
Mich.State

8. SUPPLEMENTARY NOTES (Sponsoring Organization, Publishers, Availability)
(In Computer Library for Agr.Systems Simulation. CLASS doc.no.2)

9. ABSTRACT
(LIBRARY & INFO.SCIENCE R&D)

10. CONTROL NUMBER PN-AAE-009	11. PRICE OF DOCUMENT
12. DESCRIPTORS Computer programs Libraries Simulation Software	13. PROJECT NUMBER 14. CONTRACT NUMBER CSD-2975 Res. 15. TYPE OF DOCUMENT

AID 590-1 (4-74)

SOFTWARE STANDARDS MANUAL

COMPUTER LIBRARY FOR AGRICULTURAL SYSTEMS SIMULATION

CLASS Document No. 2

Agricultural Sector Analysis and Simulation Projects
Supported under Contract AID/csd-2975
U. S. Agency for International Development

Department of Agricultural Economics
Michigan State University
East Lansing, Michigan

July 1976

TABLE OF CONTENTS

	Page
INTRODUCTION	1
SOFTWARE SELECTION STANDARDS	2
General Applicability	2
Theoretical and Empirical Validity	3
Compatibility	4
User Orientation	4
Documentation	4
DOCUMENTATION STANDARDS	5
General	5
Announcement	6
Abstract	7
Documentation	7
PROGRAMMING STANDARDS	11
Objectives	11
Language	11
Coding Format	12
Statement Labeling and Ordering	12
Coments	13
Variables	14
Statement Forms	14
Error Messages	16
Overlays	16
APPENDIX: SAMPLE DOCUMENTATION	17

COMPUTER LIBRARY FOR AGRICULTURAL SYSTEMS SIMULATION

The Computer Library for Agricultural Systems Simulation (CLASS) is one of the four major activities of the Agricultural Sector Analysis and Simulation Projects at Michigan State Unviersity under U. S. Agency for International Development Contract AID/csd-2975. The other three major interrelated project activities include theoretical and methodological research, the Development Analysis Study Program, and field activities, primarily in the Republic of Korea.

The project objective is to develop an approach to institutionalizing an analytical capacity for planning, policy formulation, program development, and project implementation for agricultural sector development within the public decision making structure of developing countries. A major component of the analytical capacity is a series of system simulation models tailored to the needs of the individual country. Much of the experience gained from the field activity and the knowledge gained from the theoretical and methodological research added to the present stock of knowledge about building and maintaining analytical capacities for agricultural sector development can be preserved and extended in the training provided through the Development Analysis Study Program and in the stock of model, component, and utility routine computer software documented in the Computer Library for Agricultural Systems Simulation.

In full operation, the Computer Library for Agricultural Systems Simulation (CLASS) acquires, catalogs, maintains and distributes computer programs and associated documentation. These computer programs are of generalized simulation models, components, and routines designed specifically for the analysis of agricultural development problems and processes. In particular, the library sets standards of admissibility for programs and documentation; catalogs and indexes programs and documentation so as to facilitate their retrieval by users seeking a set of programs to be used in a specific problem analysis; and distributes programs and documentation to users.

To enhance the effectiveness of the library, its functions also include identifying and soliciting needed models; actively bringing programs and documentation up to the library's standards; and providing limited consultation in identifying and implementing appropriate library programs for a particular application. A subsidiary function of the library in conjunction with the identification and solicitation of models is to survey and catalog ongoing research in agricultural systems modeling and simulation.

The CLASS document publication series is the main vehicle for informing potential users of the substance of CLASS holdings and activities.

July, 1976

George E. Rossmiller
Director
Agricultural Sector Analysis and
Simulation Projects

INTRODUCTION

The standards set out in this manual are for the Computer Library for Agricultural Systems Simulation (CLASS). The library contains models designed specifically to simulate processes and/or address problems related agriculture, components developed to simulate specific functional processes and which can be combined into models, and utility routines which perform generally useful mathematical or logical operations. Each software item is basically an abstract concept which is operationally realized in the library as computer programs and written documentations.

For CLASS to operate effectively and efficiently in carrying out its obligations of maintenance and distribution; and for the software to be of a standard which will allow the successful building of models, it is essential that a set of standards be defined and maintained. These standards pertain to the admissibility and continued inclusion of models, components, and routines.

The purpose of this manual is to specify the standards to be followed for definition, computer programming and documentation of library software. The manual is divided into three parts: software selection standards, documentation standards, and programming standards. Broad standards are given for the conceptual base of software items for selection, while standards for documentations and computer programs are specified in greater detail. These standards help to: (1) maintain a consistency of form, basic technical style, and type of content in the various classes of documents, and

(2) maintain a consistency of programming style and technical standards among software computer programs.

It must be noted that the standards described here apply to library software as individual system models, components, or utility routines. Not covered here is the relationship between individual software items and a simulation "language" system which is necessary for testing and executing the software items singly or in combination.

SOFTWARE SELECTION STANDARDS

Software are evaluated for admission to, and continued inclusion in, the library in five broad respects. These include general applicability, theoretical and empirical validity, compatibility with other software in the library, user orientation, and documentation. Since quantitative or objective criteria cannot be set for many of these concepts, subjective evaluations must be made. In the following two sections, however, more precise criteria are set for documentation and programming standards.

General Applicability

The quality of generality is actually an open-ended, one-dimensional continuum wherein movement in one direction is towards the more specific and in the other towards the more general. In this sense "complete" generality is impossible. Indeed, extreme generality in a model is as undesirable for the library as is extreme specificity. While a model of the latter type may be so pragmatic as to have no application but one, one of the former type may be so abstract and

unrealistic as to have no application at all. As usual, an optimum (qualitative if not quantitative) must be found somewhere in the middle range.

In this sense, library software is expected to be of general interest and applicability in the library's areas of application, i.e., agricultural development. That is, specialized models designed to simulate processes and/or address problems unique to a particular country or region within a country will not be included in the library. Models built for specific countries or regions will be eligible for admission to the library after reworking to generalize their structure and programming and otherwise be made to meet the library's standards and if the processes modeled or problems addressed are judged to be of broader interest than that for which the models were specifically written originally. Specific functional process components must meet similar criteria.

Similarly, utility routines will be included if they are judged to be of general usefulness. These may include matrix operations, simplex algorithms, gradient search procedures, distributed and discreate delays, table look-up functions, plot routines, etc.

Theoretical and Empirical Validity

Generalized models cannot be completely validated to the extent of faithfully representing a particular real-world system. Such a validation can only be made once the models have been adapted in a specific application. Nevertheless, a model's validity can and will be evaluated with respect to its own internal logical consistency and its conformity with empirical observations and with accepted social and economic theory.

Compatibility

Since models, components, and routines will be combined in a variety of configurations as required for specific applications, library software items are expected to be compatible with one another. Compatibility is insured and facilitated by the use of a common programming language, a standard programming style, clearly specified and compatible (as to definitions and formats) inputs and outputs, and documentation illustrating how software combinations may be made.

User Orientation

For CLASS to be used extensively and successfully, its software and associated documentations must be user oriented; that is, they must be both comprehensible and easy to use. The various forms of documentation and the computer program coding (see the following sections) must be clear and readable for their respective audiences. Necessary data--initial conditions and system parameters--must be clearly specified and input means and formats easy to use. In particular, the user's access to a model--including policy input and output formats--must be quick and uncomplicated and must allow the option of man-machine interactions where results may be evaluated and policies changed during the course of a simulation run.

Documentation

One of the most crucial elements of any library software item is its documentation. It is primarily through the documentation that users will be able to identify the software that best serve their purposes and then to implement, modify, test and interpret that software. While the

last section of this manual defines standards for another crucial software element, the computer program, the next section outlines standards which the software documentation must meet for inclusion in the library.

DOCUMENTATION STANDARDS

CLASS maintains three types of documents for each software item it inventories: announcement, abstract and documentation. Format and content standards for each of these is discussed separately following some general considerations.

General

Standard identifying information appears on the first page of each of the three document types for a software item. This information includes:

1. the software item's symbolic name--up to six alphanumeric characters, the first of which must be alphabetic, and preferably the same name as the primary subroutine of the computer program;
2. the full name, for which the symbolic name is a mnemonic acronym;
3. the software type, i.e., model, component, or routine;
4. the version number--a number and a letter representing the software item version and documentation release, respectively, where the original release is numbered 1A.

The layout of the above information appears on the first page of each type of document. The attached appendix shows a full sample documentation. In addition, the first (and only) page of the announcement and of the abstract contains the body of the announcement or abstract. The first page of the documentation contains a table of contents following the identifying information, with the body of the documentation beginning on the second page.

In addition, accuracy and, to a lesser degree, clarity and completeness is checked by at least two reviewers for each software item's set of documents. A computer-oriented reviewer evaluates primarily the documentation and the computer program itself in terms of:

(a) the accuracy and efficiency of the program as a computer model of the mathematical model described in the documentation; and

(b) the sufficiency of the documentation for implementing the computer program.

The other reviewer must have expertise in the subject areas (disciplines) covering the problem or process addressed by the software. His evaluation deals with:

(a) the theoretical and empirical validity of the mathematical model described in the documentation, and the appropriateness of the software item as a whole, in terms of its intended applications; and

(b) the sufficiency of the documentation for understanding the software and its intended applications (e.g., the completeness of the discussion of implicit and explicit assumptions of the model) and as a basis for deciding the appropriateness of the software for any particular application.

Announcement

Software item announcements will be published and distributed to actual and potential library users as a means of publicizing what CLASS has to offer. Thus, in most cases it will be on the basis of announcements that a user will decide, at least initially, whether certain software may be useful to him and, hence, whether to seek further information by requesting the abstract, documentation and/or consultation with the library staff.

An announcement, therefore, while it must be brief (a short paragraph), must include enough information so the above decision may be made. As a minimum, an announcement must give a verbal, non-technical statement of what the software does, i.e., the problem or process simulated by the

model, component, or utility function or algorithm performed by the routine, including any specific features of the software which may distinguish it from other possible models of the same process or similar processes. In addition, examples may be given of possible uses of the software item, e.g., larger models into which it may be embedded.

Abstract

Abstracts are intended to provide technical experts, analysts and advisors with brief (about half a page) technical descriptions of CLASS software items for use in preliminary software evaluation and selection. In addition to a more technical presentation, where appropriate, of the information contained in the announcement (the announcement itself is included in the abstract as the first paragraph), an abstract states the inputs required by the software item, the outputs it supplies, and the core storage and sample run execution time for the computer on which it was tested.

Documentation

There are actually two audiences for which documentations are intended: consultants and analysts, and programmers. One effective way of meeting the needs of these distinct audiences is to compartmentalize the documentation, either in three separate documents or in three clearly identifiable sections. The first document or section should include much explanatory material which would be require in order to understand what the software does and where it may properly be used; the second should go deeper into the theoretical and mathematical detail required by the professional analyst; and the third should provide the information a computer programmer would need in order to successfully implement the software at the local computer installation.

The documentation is essential in: (1) selecting appropriate software for a particular application, (2) validating models composed of a number of software items, (3) analyzing the results of simulation experiments, (4) operationalizing a software item on a local computer in the context of a larger program comprising a model, and (5) making modifications to software as necessary to suit a specific need.

In order to meet these objectives, each documentation adheres to the following guidelines on contents and formats as outlined:

Abstract

I. Problem or process description

II. Technical description

A. Mathematical model

B. Sample computer run and, for models, sensitivity test results

III. Program information

A. Program description

B, Program implementation

C. Program lists

1. Of sample run executive

2. Of software item

Bibliographic references

(See the appendix for a complete example.)

The abstract described above should be included in the software documentation as preliminary material.

I. This section presents a non-technical description of the problem or process the software simulates and suggests possible applications of it. In the case of a utility routine, the mathematical function or operation

performed is discussed. In essence, the objective of the discussion is to describe the need and rationale justifying the software item. The use of tables and figures is encouraged where such use contributes to the efficiency and clarity of the discussion. In addition, an example of the software item's use is developed and carried through the rest of the documentation. Any references cited are listed at the end of the documentation rather than in footnotes.

IIA. The technical description of the mathematical model is presented in the context of the example developed in section I and includes (in addition to equations) process flow diagrams, graphs, etc. as necessary to enhance the efficiency and clarity of the discussion of the model equations. The discussion itself states the assumptions embodied in the equations, identifies the input and output variables, and indicates limitations and weaknesses of the model. Again, any references cited appear in the bibliography at the end.

IIB. The sample run is based on the example developed in section I. Included in the discussion are the data input used, the equations and assumptions used to generate input variables exogenously and the output produced. In addition, for models, results of sensitivity tests using the sample run data as a base are reported so users will have information on the relative sensitivity of data input requirements.

IIIA. The program description discusses the:

a) operating environment

The machine, operating system, and compiler used for test runs are specified, including the sample run central processor compile and execution times and core storage requirements. All necessary FORTRAN library functions are

listed, and any tape and file requirements as well as special hardware requirements are stated.

b) program structure

This discussion includes a list of all subprograms supplied as part of the software item with a brief description of the purpose of each. COMMON block structure and usage are explained. Where necessary to meet the standards of clarity and completeness, appropriate subprogram flow charts and calling sequence diagrams are provided.

IIIB. The program implementation information includes:

a) internal data requirements

A list of all variables whose initial values are set within the program, either in DATA statements or assignment statements, is provided.

b) input

An explanation of all exogenous data requirements of the program--including the input units used, file descriptions and formats--is included.

c) output

A description of all intermediate and final output formats (i.e., WRITE statements), different types of output variable, output units used and a rough guide to the number of records of output produced are included.

d) error conditions and program restrictions

All known conditions which will cause improper program functioning are explained.

e) special instructions (if any)

IIIC. Shorter listings of the computer program including the executive routine producing the sample run are included in the documentation; longer listings in a supplement.

Bibliographic references, if any, appear at the end.

PROGRAMMING STANDARDS

Objectives

The objectives of programming standards are to:

(1) maintain a consistent programming style

(2) maintain compatibility among computer programs

(3) ensure adequate error checking

(4) facilitate further developments

(5) enhance readability

(6) ensure compatibility as much as possible across computer models and across compilers

These objectives have been set with the belief that their attainment will resolve a large number of problems which might otherwise be encountered by users of CLASS software.

Language

The language to be used is FORTRAN. In general, the version of FORTRAN which is used must be compatible with the compilers currently in use at MSU and on the IBM 360/370 series computers. Effort should be made to ensure that the FORTRAN standard used is compatible with as large a number of additional compilers as possible. This can be done by following ANSI FORTRAN standards as closely as possible. Within this general requirement, the specific points which follow are not at all to be considered exhaustive.

Coding Format

As a minimum for readability and consistency:

a) Continuation cards should be indented to the column following the end of the statement name (e.g., IF, CALL, FORMAT, etc.) or following the equal sign in an assignment statement.

b) Nested DO loops should be indented as follows:

```
      DO   30   I = 1,10
      TOTAL(I) = 0.
           DO   20   J = 1,20
                DO   10   K = 5,25   TOTAL(I) = TOTAL(I) + X(I,J,K)
10              CONTINUE
20         CONTINUE
30    CONTINUE
```

c) When continuation cards are required, the break in the statement should not occur within a symbolic name.

d) Blanks should be used before and after equal signs and after commas between symbolic entities in COMMON, DIMENSION, CALL, SUBROUTINE, and DATA statements. In particular, arrays in DATA statements may be arranged in tabular fashion for readability. A blank should appear before and after each relational or logical operator. Elsewhere, blanks are optional with due regard to readability.

Statement Labeling and Ordering

a) The order of statements in each subprogram should be:

(1) SUBROUTINE or FUNCTION statement
(2) COMMON
(3) type statements
(4) DIMENSION
(5) EQUIVALENCE
(6) DATA
(7) statement functions
(8) executable statements
(9) FORMAT statements
(10) END

b) Statement numbers should be in numerical order. Since FORMAT statements appear last, they must have the highest statement numbers in the routine.

c) Assignment statements for data initialization should appear at the beginning of the executable statements. They should be separated into groups by their function. Each group should be prefaced with a comment explaining what it is. Initialization for local use should appear last.

Comments

a) It is helpful if comments are arranged in some hierarchical structure. This may be done through the appropriate use of indentation, blank lines, and special characters (*, =, -, etc. for leaders and underlining). In order to help differentiate comments from regular FORTRAN statements, it is recommended that comments not start in column 7.

b) Comment cards should be designated by a C in column 1 (not a $ or *).

c) Title heading comments may include a brief description of what the routine does, all inputs to the routine, any formal arguments, and any technical restrictions as to variable type, array size, numeric or alphanumeric data.

d) Variable name definitions should appear in the primary subroutine of each component. These will contain information about each non-trivial variable which is used. The following format is suggested for these definitions (capitalized words will appear in the definition):

name (i,j,k). definition IN units. type.* (meaning of i, meaning of j, meaning of k). /Common block name/. DEF: subprogram where defined. USED: subprograms where used.

*only if the variable appears in a type statement.

e) A comment should be placed at the beginning of each logical section of the routine, briefly describing its function.

f) Any call to an entry point of a subroutine other than by the subroutine's name should have a comment telling what subroutine contains the entry point and on what line it can be found.

g) A comment should appear with each computed GO TO explaining what the different choices represent, if it is not clear from the context.

h) Any unusual circumstances or any information which would be of help to someone unfamiliar with the routine should be included in the comments. Also, any variations from these standards should be noted.

Variables

a) Symbolic Naming

(1) The names of variables, arrays, subroutines, functions and common blocks shall be no more than six characters long.

(2) Except for DO loop parameters, the choice of names should be an acronym reflecting as closely as possible the meaning of the variable.

(3) No symbolic name may be used in more than one of the following contexts: variable name, common block label, subprogram name.

(4) A variable which is used in more than one routine should, if possible, have the same name in all routines. Variables used in different routines, for different purposes, should never have the same name.

b) No array should have more than three FORTRAN dimensions. For a larger number of dimensions, an index-generating function should be used.

Statement Forms

a) All common blocks should be labeled. Separate COMMON statements should be used for each block. They should be ordered alphabetically.

b) All arrays in COMMON statements should be dimensioned there, not in separate DIMENSION statements.

c) Implied DO loops should not be used in DATA statements.

d) If variables in labeled COMMON are to be initialized in DATA statements, this must be done in a BLOCK DATA subroutine.

e) Data values should appear only in DATA statements, not in specification statements (DIMENSION, etc.).

f) No mixed mode arithmetic is permitted--use IFIX and FLOAT functions where necessary.

g) Parentheses should not be nested so deeply as to impair readability of a statement.

h) No more than one statement should appear on each card.

i) Multiple replacement statements should not be used.

j) If a statement is continued to following cards, no comments should appear between the continuation cards comprising the statement.

k) Hollerith constants should not appear in assignment statements.

l) Hollerith fields should be used in FORMAT statements rather than enclosing literal fields with asterisks or apostrophes.

m) Avoid transferring control to an earlier statement in the routine (except with a DO loop).

n) IMPLICIT statements should not be used.

o) NAMELIST statements should not be used.

p) Assigned GO TO statements should not be used.

q) PRINT statements should not be used. ALL READ and WRITE statements should use a variable to specify the I/O device number. Unit 5 should be used for the standard input device (card reader or terminal) and unit 6 should be used for the standard output device (line printer or terminal).

r) IF statements.

(1) The logical expression should reflect as closely as possible the "natural" way of looking at the question (i.e., do not optimize at the expense of readability).

(2) If more than a one-line IF is needed, the following structure should be used:

```
      IF  (logical expression) GO TO ℓ1
                 s1
                 :
                 sm
      GO TO ℓ2
ℓ1               t1
                 :
                 tm
ℓ2               u1
```

s) Loops

(1) Loops will be DO loops rather than programmed IF( ) GO TO. If the limit of the DO is not known, 99999 should be used.

(2) Each DO loop should end on a unique CONTINUE statement.

(3) Each DO loop should contain no more than one statement transferring control out of that loop.

Error Messages

a) Error messages will be included in association with data checking and any other checks made within the program.

b) Error messages will include the following information:

(1) The subprogram generating the error message

(2) Where in the program and at what stage of execution (usually represented by the current incremental value of some DO loop) the error occurred

(3) What the error was (incorrect data values, an incipient attempt to divide by zero, etc.)

(4) The actual value of the offending piece of data and if possible, what it should be.

Overlays

a) All variables to be saved will either be in common in the main overlay or be written out onto disk or tape and read back in after the subprogram has been replaced in core.

b) Except for unchanging variables, all variables initialized in data statements must be so initialized in the main overlay.

c) Variables in COMMON in the main overlay must not be in data statements in lower level overlays.

APPENDIX

SAMPLE DOCUMENTATION

This appendix includes the complete set of documents for one of the software items of CLASS. These documents serve to illustrate the types of documents and the documentation and programming standards of CLASS, as well as what is meant by a "generalized function-oriented component."

CLASS Document

1. Symbolic Name	2. Name	3. Type	4. Version No. and Date
DEMOGC	Demography with Distributed Age Cohorts	component	1A 5/1/74

DEMOGC Announcement

DEMOGC models the demographic cohort-survival processes of birth, death, migration and aging for a population disaggregated by age and sex. The length of a cohort is an average of the time individuals take to pass through the cohort; i.e., cohort staying times vary among individuals in the population. DEMOGC may be used, for example, for populations of trees, capital equipment, livestock, insects and humans --where "age" cohorts may be defined by such age-related characteristics as yield, reliability, fertility, instar and use of social services, respectively.

CLASS Document

1. Symbolic Name	2. Name	3. Type	4. Version No. and Date
DEMOGC	Demography with Distributed Age Cohorts	component	1A 5/1/74

DEMOGC Abstract

DEMOGC models the demographic cohort-survival processes of birth, death, migration and aging for a population disaggregated by age and sex. The length of a cohort is an average of the time individuals take to pass through the cohort; i.e., cohort staying times vary among individuals in the population. DEMOGC may be used, for example, for populations of trees, capital equipment, livestock, insects and humans --where "age" cohorts may be defined by such age-related characteristics as yield, reliability, fertility, instar and use of social services, respectively.

Age cohorts are modeled by distributed delays which assume individual staying times in a cohort are random variables following an Erlang distribution. The delays are non-conservative to handle storage losses due to deaths and migrations, and the means of the distributions, i.e., the average staying times, may be time varying. Finally, the user has the option of specifying whether births are to be endogenous (i.e., by natural reproduction) or treated as an input (e.g., overt plantings in the case of tree populations).

Inputs to DEMOGC are age-sex specific proportional fertility, death and net out-migration rates and mean delay times. DEMOGC computes population levels (total and by age and sex), births, deaths, migrations, the sex ratio, crude birth and death rates and the population's natural and net growth rates.

DEMOGC requires 429 decimal words of core storage and 6.0 milliseconds to execute (with three cohorts for each of two sexes) on a CDC 6500 computer.

CLASS Document

1. Symbolic Name	2. Name	3. Type	4. Version No. and Date
DEMOGC	Demography with Distributed Age Cohorts	component	1A 5/1/74

DEMOGC Documentation

Contents

I. Process Description 1

II. Technical Description 5

A. Mathematical model 5

B. Sample run 9

III. Program Information 13

A. Program description 13

B. Program implementation 14

C. Program lists 14

1. Of sample run executive 15

2. Of component 16

Bibliographic References 22

I. Process Description

One of the components most commonly found in large-scale socioeconomic systems models is a population dynamics model. Population models, at one extreme, are often simple models in which a constant exponential growth rate is applied to an initial population level. At another extreme are found complex demographic systems in which the population is disaggregated by age, sex, income class, education level, geographic area, employment, etc.; fertility and death rates for each category are determined endogenously by socioeconomic conditions in the larger model; and migrations among categories are also linked to factors in the larger system.

DEMOGC takes a middle course in order to increase its general usefulness. Aggregated exponential growth models, for anything but very macro analyses, limit too greatly considerations of socioeconomic feedback effects on population growth and of population distribution effects on demographic outputs to the larger system. Demographic models which, on the other hand, disaggregate the population along several dimensions and compute fertility, death and migration rates endogenously have limited applications--limited to situations where the designated dimensions are applicable and where the chosen factors and functional forms determining fertility death and migration rates are appropriate. In addition, where the population model is but one component of a much larger model, the use of computer resources (i.e., execution time and core requirements) becomes a real constraint to the use of the more disaggregated models. DEMOGC is general enough, however, to be used, if desired in a particular application, as a subcomponent of a larger population component which

computes the feedback effects to and from a still larger socioeconomic model and thus passes endogenous fertility, death and migration rates to DEMOGC as inputs. In addition, such a larger population component can use DEMOGC repetitively to treat a multidimensional disaggregation of a single population as a collection of populations with possible migrations among them.

DEMOGC itself disaggregates a population by age and sex, and cohort-specific fertility, death and migration rates are treated as possibly time-varying inputs. The user has the flexibility to collapse either the sex dimension or the age dimension or both if greater aggregation is desired. For example, for capital equipment populations, the sex dimension may not be appropriate. Even for cases where disaggregation by sex is inappropriate, the "sex" dimension may be redefined and used for another classification, e.g., modern and traditional biological varieties of cocoa trees. Further, the user has the option of specifying that births are not to be generated within the population by natural reproduction but rather by external factors or decisions. Again, capital equipment populations serve as an example, where "births" are investment decisions.

Finally, age cohorts in DEMOGC are modeled as distributed delays (using CLASS component DELLVF) wherein the length of time spent in each cohort varies from individual to individual in the population, e.g., due to genetic or environmental differences. DELLVF also accounts for cohort population losses and additions due to factors other than aging, e.g., deaths and migrations. The average staying time in each cohort is treated as an input to DEMOGC and may change over time depending on conditions in a larger model. In this way, age cohorts may be defined

not by chronological age but by some other age-varying characteristic. Continuing the capital equipment example, if it can reasonably be assumed that reliability (or productivity) and maintenance requirements vary with age in general but that individual pieces of equipment of the same age may exhibit different degrees of reliability and maintenance requirements (i.e., individuals "mature" at different rates)--then the use of distributed delays is appropriate. If fixed age cohorts are desired, CLASS component DEMOGD may be used in place of DEMOGC.

The technical description of DEMOGC, which follows, is presented with reference to the illustrative use of DEMOGC as a simplified version of a demographic cattle model developed for a region of Colombia [1]. Three age cohorts are defined (Figure 1). For females, the three cohorts represent heifers, producing cows and old cows, respectively. For males, the output of the second cohort is considered finished males, all of which are sold for slaughter, so the third cohort remains empty.

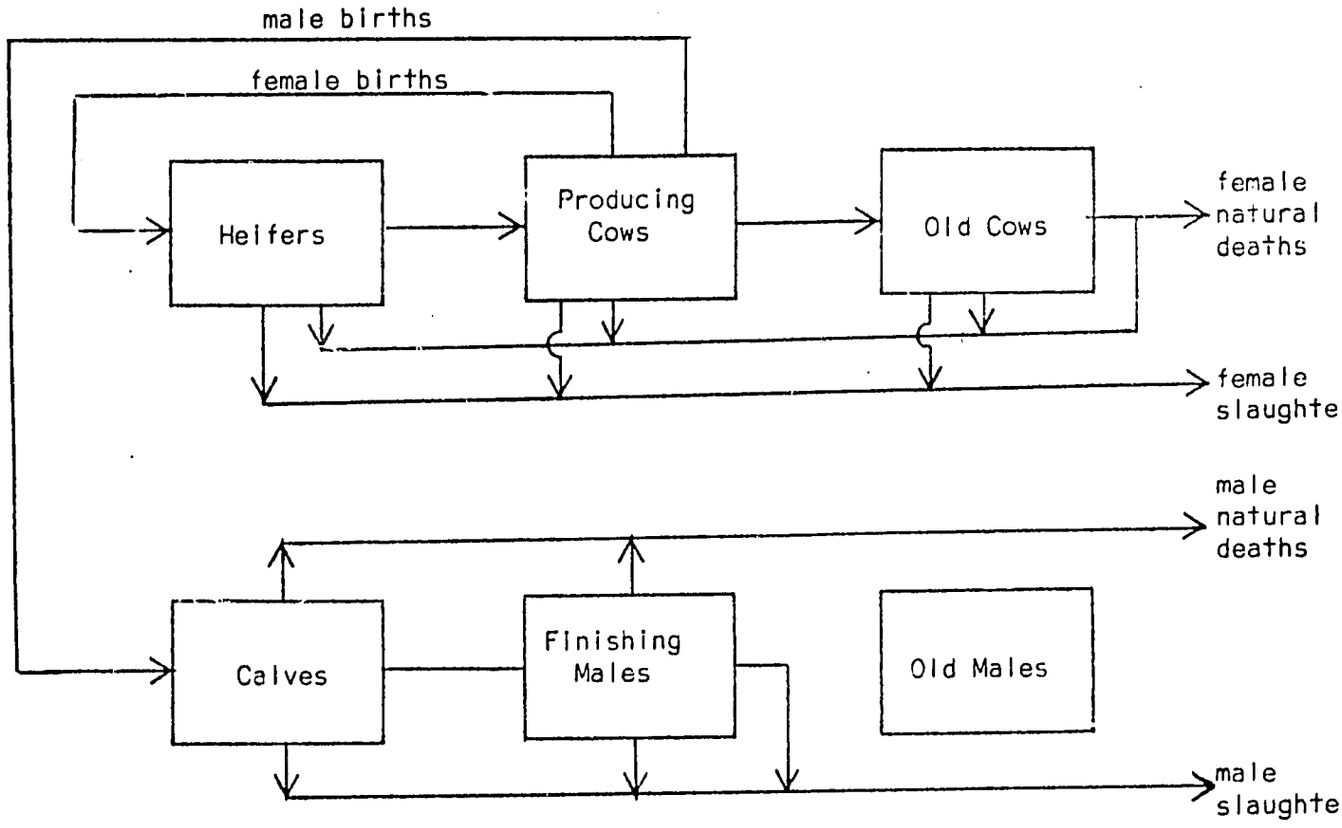

Figure 1

DEMOGC Cattle Demography Illustration

II. Technical Description

A. Mathematical model

DEMOGC is modeled in two parts. The first part uses CLASS component DELLVF to simulate the maturation process of individual animals aging through the distributed delay cohorts, and it computes the population for each age-sex cohort, for each sex, and the total. The delay for each cohort, e.g., of age j and sex k, is of order KC_{jk}. See the DELLVF documentation for derivations of equations (1) and (2). (The simulation model uses Euler integration to solve the differential equations.)

$$(1)\quad \frac{dRP_{ijk}(t)}{dt} = \frac{KC_{jk}}{DEL_{jk}(t)}\Big[RP_{(i+1)jk}(t) - RP_{ijk}(t)\Big(1 + \frac{1}{KC_{jk}}\frac{dDEL_{jk}(t)}{dt} + \frac{DEL_{jk}(t)}{KC_{jk}} \cdot PLR_{jk}(t)\Big)\Big], \quad i=1, \ldots, KC_{jk}$$

$$RP_{(KC+1)jk}(t) = RIN_{jk}(t)$$

$$ROUT_{jk}(t) = RP_{1jk}(t)$$

$$(2)\quad POP_{jk}(t) = \frac{DEL_{jk}(t)}{KC_{jk}} \sum_{i=1}^{KC_{jk}} RP_{ijk}(t)$$

$$(3)\quad POPS_k(t) = \sum_{j=1}^{NC} POP_{jk}(t)$$

$$(4)\quad TPOP(t) = \sum_{k=1}^{NSEX} POPS_k(t)$$

where:

RP -- intermediate rates of the delay

RIN -- population entering the cohort (animals/year)

ROUT -- rate at which population matures out of the cohort (animals/year)

POP -- population level in the cohort (animals)

PLR -- proportional loss rate due to deaths and migration[1/] (proportion/year) (equation (6))

DEL -- the average time for an individual to mature through the cohort--input to DEMOGC (years)

KC -- the order of the delay

POPS -- population by sex (animals)

TPOP -- total population (animals)

j -- indexes cohorts, j=1, ..., NC

k -- indexes sexes, k=1, ..., NSEX

NC -- the number of cohorts

NSEX -- the number of sexes (NSEX = 1 or 2).

Population entering cohort j of sex k, RIN, is defined as:

$$(5) \quad RIN_{jk}(t) = \begin{cases} BTH_k(t) & \text{for } j=1 \\ ROUT_{(j-1)k}(t) & \text{for } j=2, \ldots, NC \end{cases}$$

where:

BTH -- births by sex (animals/year) (equation (11)).

The DELLVF routine (refer to its CLASS documentation) simulates a distributed delay with time-varying mean delay time DEL (an input to DEMOGC) and storage losses PLR (equation (6)). The order of the delay KC is a measure of the variance σ^2 of the delay times of individual

---

[1/] Migrations are interpreted as slaughter in this illustration.

animals about the mean--the larger KC, the smaller the variance, where $\sigma^2 = DEL^2/KC$ and where individual delay times follow an Erlang distribution. In certain cases, KC and DEL may be estimated from observed or experimental data [2].

The second part of DEMOGC computes births, deaths and migrations from fertility, death and migration rates (inputs to DEMOGC). It also computes growth rate statistics.

The total proportional rate at which animals leave cohort jk for reasons other than aging, i.e., due to deaths and migrations, is PLR.

$$(6) \quad PLR_{jk}(t) = PMR_{jk}(t) + DR_{jk}(t)$$

where:

PMR -- proportional migration rate--input to DEMOGC (proportion/year)

DR -- proportional death rate--input to DEMOGC (proportion/year).

Deaths, then, are computed for each cohort and summed across cohorts, and total migration is calculated.

$$(7) \quad DTH_{jk}(t) = \begin{cases} DR_{jk}(t) \cdot POP_{jk}(t), & j=1, \ldots, NC-1 \\ DR_{NCk}(t) \cdot POP_{NCk}(t) + ROUT_{NCk}(t) & \end{cases}$$

$$(8) \quad TDTH(t) = \sum_{k=1}^{NSEX} \sum_{j=1}^{NC} DTH_{jk}(t)$$

$$(9) \quad TMR(t) = \sum_{k=1}^{NSEX} \sum_{j=1}^{NC} PMR_{jk}(t) \cdot POP_{jk}(t)$$

where:

DTH -- deaths from a cohort (animals/year)

TDTH -- total deaths (animals/year)

TMR -- total migration (animals/year).

Notice that animals aging out of the last, oldest cohort of the population are accounted as deaths (equation (7)).

Equations (10) and (11) compute total births and births by sex.

$$(10) \quad TBTH(t) = \sum_{j=1}^{NC} FR_j(t) \cdot POP_{j=1}(t)$$

$$(11) \quad BTH_1(t) = PBF \cdot TBTH(t)$$

$$BTH_2(t) = TBTH(t) - BTH_1(t)$$

where:

TBTH -- total births (animals/year)

BTH -- births by sex (animals/year)

FR -- fertility rate--input to DEMOGC (proportion/year)

PBF -- proportion of births which are female.

If DEMOGC is used to model a population which does not reproduce itself naturally (or for which it is not desired to model reproduction), and births are determined exogenously to DEMOGC, e.g., the planting of trees, then equation (10) is not executed and TBTH is considered an input to DEMOGC. It is also possible to use DEMOGC for populations not disaggregated by sex.

Finally, DEMOGC generates a number of population statistics.

$$(12) \quad SR(t) = POPS_2(t)/POPS_1(t)$$

$$(13) \quad CBR(t) = TBTH(t)/TPOP(t)$$

(14) CDR(t) = TDTH(t)/TPOP(t)

(15) PGRN(t) = 100(CBR(t) - CDR(t))

(16) PGRT(t) = PGRN(t) - 100•TMR(t)/TPOP(t)

where:

SR -- sex ratio (not computed if the population is not disaggregated by sex)

CBR -- crude birth rate (proportion/year)

CDR -- crude death rate (proportion/year)

PGRN -- natural growth rate (percent/year)

PGRT -- net growth rate (percent/year).

Initial conditions must be specified exogenously for the inputs FR_j, DR_{jk} and PMR_{jk} and for the cohort populations POP_{jk}. Outputs and internal variables (e.g., RP_{ijk}, RIN_{jk}) are initialized by DEMOGC. Parameters which must be given values are PBF, KC_{jk}, NC, NSEX, and the simulation step size DT.

B. Sample Run

The sample computer run using the cattle demography illustration (Figure 1) assumes constant proportional annual fertility, death and migration rates, where "migration" is interpreted to be slaughter. The fertility rate for the producing cows is .75. The death rates are, for the three female cohorts, .03, .02 and .02; and, for the two male cohorts, .03 and .02, respectively. Slaughter rates are .13, .02 and .5 for the three female cohorts, and .11 and .1 for the male cohorts. The mean lengths of the cohorts, also assumed constant for this illustration, are 2.5, 10 and 5 years respectively for the female cohorts, and 2.5 and 3 years for the male cohorts. Initial population levels are (in thousands of animals) 118, 242, and 43 for female cohorts, and 117 and 94 for male

cohorts. The output of a ten-year simulation appears in Figure 2.

The results of sensitivity tests changing parameters by +10% and DT by -50% (the latter to test integration accuracy) are tabulated in Table I.

```
TIME        TOT. POP.  FEM. POP.  MALE POP.  DEATHS      BIRTHS
 0.          .6140E+06  .4030E+06  .2110E+06  .2323E+05  .1815E+06

SEX RATIO  CRUDE BR   CRUDE DR   NAT. GRTH. NET GRTH.  MIGRN.
  .5236E+00  .2956E+00  .3783E-01  .2578E+02  .1026E+02  .9528E+05

    FEMALE POPULATION COHORTS          MALE POPULATION COHORTS
  .1180E+06  .2420E+06  .4300E+05  .1170E+06  .9400E+05 0.


TIME        TOT. POP.  FEM. POP.  MALE POP.  DEATHS      BIRTHS
  .2000E+01  .7508E+06  .4725E+06  .2783E+06  .2243E+05  .1944E+06

SEX RATIO  CRUDE BR   CRUDE DR   NAT. GRTH. NET GRTH.  MIGRN.
  .5890E+00  .2590E+00  .2988E-01  .2291E+02  .9145E+01  .1033E+06

    FEMALE POPULATION COHORTS          MALE POPULATION COHORTS
  .1761E+06  .2592E+06  .3718E+05  .1796E+06  .9863E+05 0.


TIME        TOT. POP.  FEM. POP.  MALE POP.  DEATHS      BIRTHS
  .4000E+01  .8932E+06  .5485E+06  .3447E+06  .2552E+05  .2290E+06

SEX RATIO  CRUDE BR   CRUDE DR   NAT. GRTH. NET GRTH.  MIGRN.
  .6283E+00  .2564E+00  .2857E-01  .2279E+02  .9624E+01  .1176E+06

    FEMALE POPULATION COHORTS          MALE POPULATION COHORTS
  .2077E+06  .3054E+06  .3543E+05  .2128E+06  .1319E+06 0.


            TOT. POP.  FEM. POP.  MALE POP.  DEATHS      BIRTHS
     100E+01  .1078E+07  .6560E+06  .4224E+06  .2991E+05  .2793E+06

SEX RATIO  CRUDE BR   CRUDE DR   NAT. GRTH. NET GRTH.  MIGRN.
  .6438E+00  .2590E+00  .2773E-01  .2313E+02  .9758E+01  .1442E+06

    FEMALE POPULATION COHORTS          MALE POPULATION COHORTS
  .2491E+06  .3724E+06  .3451E+05  .2550E+06  .1673E+06 0.


TIME        TOT. POP.  FEM. POP.  MALE POP.  DEATHS      BIRTHS
  .8000E+01  .1308E+07  .7967E+06  .5108E+06  .3561E+05  .3420E+06

SEX RATIO  CRUDE BR   CRUDE DR   NAT. GRTH. NET GRTH.  MIGRN.
  .6412E+00  .2616E+00  .2724E-01  .2343E+02  .1025E+02  .1723E+06

    FEMALE POPULATION COHORTS          MALE POPULATION COHORTS
  .3040E+06  .4560E+06  .3668E+05  .3113E+06  .1996E+06 0.


TIME        TOT. POP.  FEM. POP.  MALE POP.  DEATHS      BIRTHS
  .1000E+02  .1599E+07  .9748E+06  .6238E+06  .4317E+05  .4194E+06

SEX RATIO  CRUDE BR   CRUDE DR   NAT. GRTH. NET GRTH.  MIGRN.
  .6399E+00  .2624E+00  .2701E-01  .2354E+02  .1049E+02  .2086E+06

    FEMALE POPULATION COHORTS          MALE POPULATION COHORTS
  .3726E+06  .5592E+06  .4298E+05  .3914E+06  .2423E+06 0.
```

Figure 2
Sample Run Output

Table 1

DEMOGC Sensitivity Test Results at Year 10

un Definition	Standard Value Coh. 1	Standard Value Coh. 2	Standard Value Coh. 3	Test Run Value	Female Population	Male Population	Sex Ratio	Natural Growth Rates	Sales--Culls and Slaughters
					1000 animals	1000 animals	--	%/year	1000 animals/yr
1. Standard run					974.8	623.8	.6399	23.54	208.6
					% deviation from standard run				
2. Female initial cohort populations	118.0	242.0	43.0	+10%	+10.1	+10.0	0	0	+10.0
3. Male initial cohort populations	117.0	94.0	0	+10%	0	0	0	0	0
4. Female cohort maturation times	2.5	10.0	5.0	+10%	-3.10	-4.64	-1.60	-1.70	-4.02
5. Male cohort maturation times	2.5	3.0	--	+10%	0	+4.91	+4.92	-2.13	-1.44
6. Female cohort death rates	.03	.02	.02	+10%	-2.25	-1.72	+.5	-.8	-1.73
7. Male cohort death rates	.03	.02	--	+10%	0	-.609	-.594	-.170	-.527
8. Female cohort sales rates	.13	.02	.5	+10%	-5.73	-3.98	+1.86	-.212	-.911
9. Male cohort sales rates	.11	.1	--	+10%	0	-2.32	-2.31	+.9	+.8
10. Fertility rate		.75		+10%	+15.7	+19.7	+3.50	+6.37	+16.7
11. Proportion of births female		.5		+10%	+15.7	-2.02	-15.3	-3.99	+3.64
12. Simulation time increment DT		.25		-50%	+1.09	+1.47	+.391	-3.823	+1.39

III. Program Information

A. Program Description

DEMOGC was programmed and tested in the following operating environment:

Machine: CDC 6500
Operating system: MSU Hustler 2 L239 LSD 32--an extension of CDC Scope 3.2
Compiler: CDC FTN V3.0-P357
Core requirements: 429 decimal words
CP compile time: 2.5 seconds
CP execution time (for sample run): 6.0 milliseconds
Tapes: none
Direct access files: none
FORTRAN library subprograms used: FLOAT, AMAX1

DEMOGC is structured as two subroutines, DEMOGC and DELLVF, the former containing two calls to the latter, the second call being inside a DO-loop. No COMMON blocks are used.

DEMOGC models the cohort-survival process of a population disaggregated by age and sex, given fertility, death and migration rates. DELLVF models the flow of population through the cohorts as distributed delays. (See DELLVF Documentation.)

DEMOGC is divided in two parts which may be executed separately by specifying ICALL = 1 or ICALL = 2 in the argument list. This is significant only if fertility rates (FR), death rates (DR), or migration rates (PMR) depend on <u>current</u> population levels, which are computed in the first part of DEMOGC. If this is the case, two calls to DEMOGC (the first with ICALL = 1, the second with ICALL = 2) should be used, with FR, DR, and PMR calculated between the two calls. If this is not the case, i.e., if FR, DR and PMR are constant or depend only on past values of variables, a single call to DEMOGC (with ICALL = 0) will suffice. (Although in the sample program one call with ICALL = 0 rather than two calls would work

because FR, DR and PMR are constant, two calls are used for illustrative purposes as if FR, DR and PMR depended on current population levels.)

B. Program Implementation

All inputs and outputs are transmitted through the argument list, and no logical units are used by DEMOGC. See the list of the sample run executive program for an example of the call statement.

The program will <u>not</u> execute properly if NSEX has a value other than 1 or 2, or NOBTH has a value other than 0 or 1. If NSEX = 1, PBF need not be defined. If NOBTH = 1, TBTH must be defined outside of DEMOGC, and FR need not be defined. Finally, the following conditions must hold:

$DT > 0$

$NC \geq 1$ and integer

$DEL_{jk}(t) > 0$ for all t, $j=1, \ldots, NC$; $k=1, \ldots, NSEX$

$KC_{jk} \geq 1$ and integer, $j=1, \ldots, NC$; $k=1, \ldots, NSEX$.

C. Program lists

The computer programs for the sample run and for DEMOGC, including subroutines DEMOGC and DELLVF, are listed below:

```
      PROGRAM DEMTST(OUTPUT)
      DIMENSION BR(3), BTH(2), DEL(3,2), DELP(3,2), DR(3,2), DTH(3,2)
      DIMENSION KC(3,2), PLR(3,2), PMR(3,2), POP(3,2), POPS(2)
      DIMENSION ROUT(3,2), RP(10,3,2)
      DATA POP / 118000., 242000., 43000., 117000., 94000., 0. /
      DATA DEL / 2.5, 10., 5., 2.5, 3., 1. /
      DATA DR / .03, .02, .02, .03, .02, 0. /
      DATA PMR / .13, .02, .5, .11, .1, 0. /
      DATA IDTU / 1 /
      DATA KC / 10, 6, 3, 10, 6, 2 /
      DATA BR / 0., .75, 0. /
      DATA NK,NC,NSEX / 10, 3, 2 /
      DATA NOBTH,BCP,PBF / 0, 1., .5 /
      DATA DT / .25 /
      T = -DT
      DO 40 I=1,41
        T = T + DT
        ROUT(2,2) = 0.
        CALL DEMOGC(POP, ROUT, RP, DEL, DELP, KC, POPS, TPOP, SR, DR,
     +              CDR, DTH, TDTH, NOBTH, PBF, BR, CBR, BTH, TBTH,
     +              TMR, PMR, PLR, PGRN, PGRT, NK, NC, NSEX, DT, IDTU,
     +              T, 1)
        CALL DEMOGC(POP, ROUT, RP, DEL, DELP, KC, POPS, TPOP, SR, DR,
     +              CDR, DTH, TDTH, NOBTH, PBF, BR, CBR, BTH, TBTH,
     +              TMR, PMR, PLR, PGRN, PGRT, NK, NC, NSEX, DT, IDTU,
     +              T, 2)
        TMR = TMR + ROUT(2,2)
        PGRT = PGRN - 100.*TMR/TPOP
        IF(MOD(I-1,8) .NE. 0) GO TO 40
        PRINT90,T,TPOP,POPS,TDTH,TBTH
        PRINT91,SR,CBR,CDR,PGRN,PGRT,TMR
        PRINT92,POP
      CONTINUE
      STOP
      FORMAT(5H1TIME,7X,9HTOT. POP.,2X,9HFEM. POP.,2X,9HMALE POP.,2X,
     +       6HDEATHS,5X,6HBIRTHS/1H ,6E11.4)
      FORMAT(10H0SEX RATIO,2X,8HCRUDE BR,3X,8HCRUDE DR,3X,10HNAT. GRTH.
     +       1X,9HNET GRTH.,2X,6HMIGRN./1H ,6E11.4)
      FORMAT(30H0    FEMALE POPULATION COHORTS,9X,15HMALE POPULATION,
     +       8H COHORTS/1H ,6E11.4)
      END
```

```
SUBROUTINE DEMOGC(POP, ROUT, RP, DEL, DELP, KC, POPS, TPOP, SR,
+                 DR, CDR, DTH, TDTH, NOBTH, PBF, BR, CBR,
+                 BTH, TBTH, TMR, PMR, PLR, PGRN, PGRT, NK, NC,
+                 NSEX, DT, IDTU, T, ICALL)
```

TITLE - DEMOGRAPHY WITH DISTRIBUTED AGE COHORTS

VERSION - 1A

DATE - 1 MAY 1975

PURPOSE - DEMOGC MODELS THE COHORT-SURVIVAL DEMOGRAPHIC PROCESSES OF BIRTH, DEATH, MIGRATION AND AGING FOR A POPULATION DISAGGREGATED BY AGE AND SEX, WHERE AGE COHORTS ARE MODELED BY DISTRIBUTED DELAYS

INPUT REQUIREMENTS

VALUES TRANSMITTED THROUGH THE ARGUMENT LIST -- ALL INPUT VARIABLES

VARIABLES REQUIRING VALUES WHICH WILL NOT CHANGE DURING A RUN -- DT, IDTU, KC(I,J), NC, NOBTH, NSEX

VARIABLES REQUIRING VALUES WHICH MAY CHANGE DURING A RUN -- BR(I), DEL(I,J), DR(I,J), ICALL, PBF, PMR(I,J), T

VARIABLES COMPUTED BY THIS ROUTINE REQUIRING INITIAL VALUES -- POP(I,J)

OUTPUT VARIABLES

VALUES TRANSMITTED THROUGH THE ARGUMENT LIST -- BTH(J), CBR, CDR, DELP(I,J), DTH(I,J), PGRN, PGRT, PLR(I,J), POPS(J), ROUT(I,J), RP(K,I,J), SR, TBTH, TDTH, TMR, TPOP

VARIABLE DEFINITIONS

BR(I) - FERTILITY RATE IN PROPORTION/UNIT TIME. (AGE)

BTH(J) - BIRTHS IN INDIVIDUALS/UNIT TIME. (SEX)

CBR - CRUDE BIRTH RATE IN PROPORTION/UNIT TIME.

CDR - CRUDE DEATH RATE IN PROPORTION/UNIT TIME.

DEL(I,J) - LENGTH OF DELAY IN TIME UNITS. (AGE,SEX)

DELP(I,J) - LENGTH OF DELAY AT PREVIOUS TIME PERIOD IN TIME UNITS. (AGE,SEX)

DR(I,J) - DEATH RATE IN PROPORTION/UNIT TIME. (AGE,SEX)

DT - SIMULATION TIME INCREMENT IN TIME UNITS.

DTH(I,J) - DEATHS IN INDIVIDUALS/UNIT TIME. (AGE,SEX)

ICALL - MODE OF OPERATION OF THE COMPONENT. UNITLESS.

IDTU - NUMBER OF INTERVALS INTO WHICH DT IS DIVIDED FOR USE IN THE DELAY ROUTINE. UNITLESS.

KC(I,J) - ORDER OF COHORT DELAY. UNITLESS. (AGE,SEX)

NC - NUMBER OF AGE COHORTS. UNITLESS.

NOBTH - INDICATES WHETHER BIRTHS ARE GENERATED BY THE POPULATION OR NOT. UNITLESS.

NSEX - NUMBER OF SEXES (OR SUBPOPULATIONS). UNITLESS.

PBF - PROPORTION OF BIRTHS WHICH ARE FEMALE (OR FIRST SUBPOPULATION).

PGRN - NATURAL GROWTH RATE OF THE POPULATION IN PERCENT/ UNIT TIME.

PGRT - ACTUAL GROWTH RATE OF THE POPULATION IN PERCENT/ UNIT TIME.

PLR(I,J) - PROPORTIONAL LOSS RATE (MIGRATION AND DEATHS) IN PROPORTION/UNIT TIME. (AGE,SEX)

PMR(I,J) - PROPORTIONAL MIGRATION RATE IN PROPORTION/UNIT TIME. (AGE,SEX)

POP(I,J) - POPULATION IN INDIVIDUALS. (AGE,SEX)

POPS(J) - TOTAL POPULATION BY SEX IN INDIVIDUALS. (AGE,SEX)

ROUT(I,J) - COHORT OUTPUT RATE IN INDIVIDUALS/UNIT TIME. (AGE,SEX)

RP(K,I,J) - SET OF INTERMEDIATE RATES FOR EACH COHORT IN INDIVIDUALS/UNIT TIME. (STAGE OF DELAY,AGE,SEX)

SR - SEX RATIO. UNITLESS.

T - SIMULATION TIME IN TIME UNITS.

TBTH - TOTAL BIRTHS IN INDIVIDUALS/UNIT TIME.

TDTH - TOTAL DEATHS IN INDIVIDUALS/UNIT TIME.

TMR - TOTAL MIGRATION RATE IN INDIVIDUALS/UNIT TIME.

TPOP - TOTAL POPULATION IN INDIVIDUALS.

```
DIMENSION BR(NC), BTH(NSEX), DEL(NC,NSEX), DELP(NC,NSEX)
DIMENSION DR(NC,NSEX), DTH(NC,NSEX), KC(NC,NSEX), PLR(NC,NSEX)
DIMENSION PMR(NC,NSEX), POP(NC,NSEX), POPS(NSEX)
DIMENSION ROUT(NC,NSEX), RP(NK,NC,NSEX)

    DEMOGC IS DIVIDED IN TWO PARTS WHICH MAY BE CALLED SEPARATELY
        IF ICALL=0 THE ENTIRE ROUTINE IS EXECUTED
        IF ICALL=1 THE CURRENT POPULATION IS CALCULATED
        IF ICALL=2 BIRTHS, DEATHS, MIGRATION, ETC. ARE CALCULATED

IF(ICALL .GT. 1) GO TO 52
TPOP = 0.
DO 50 K=1,NSEX
  IF(T .NE. 0.) GO TO 20

    SET INITIAL VALUES FOR DELAY ROUTINE

  DO 15 J=1,NC
    DELP(J,K) = DEL(J,K)
    IKC = KC(J,K)
    DO 10 I=1,IKC
      RP(I,J,K) = POP(J,K)/DEL(J,K)
    CONTINUE
    ROUT(J,K) = RP(1,J,K)
  CONTINUE
  GO TO 30

    UPDATE POPULATION COHORTS

  JNC = NC-1
  DO 25 JJ=1,JNC
    J = NC + 1 - JJ
    CALL DELIVE(ROUT(J-1,K), ROUT(J,K), RP(1,J,K), POP(J,K),
+                PLR(J,K), DEL(J,K), DELP(J,K), DT, IDTU, KC(J,K))
  CONTINUE
  CALL DELIVE(BTH(K), ROUT(1,K), RP(1,1,K), POP(1,K), PLR(1,K),
+              DEL(1,K), DELP(1,K), DT, IDTU, KC(1,K))

    POPULATION TOTALS

  POPS(K) = 0.
  DO 40 J=1,NC
    POPS(K) = POPS(K) + POP(J,K)
  CONTINUE
  TPOP = TPOP + POPS(K)
CONTINUE
IF(ICALL .EQ. 1) RETURN

    IF ICALL=1 EXECUTION ENDS HERE
```

```
      IF ICALL=2 EXECUTION BEGINS HERE

      MIGRATION AND DEATHS

 TMR = 0.
 TDTH = 0.
 DO 54 K=1,NSEX
   DO 53 J=1,NC
     PLR(J,K) = PMR(J,K) + DR(J,K)
     DTH(J,K) = DR(J,K)*POP(J,K)

      THE OUTPUT OF THE LAST AGE COHORT IS CONSIDERED TO BE DEATHS

      IF(J .EQ. NC) DTH(J,K) = DTH(J,K) + ROUT(J,K)
      TDTH = TDTH + DTH(J,K)
      TMR = TMR + PMR(J,K)*POP(J,K)
   CONTINUE
 CONTINUE

      BIRTHS

      IF BIRTHS ARE NOT COMPUTED FROM THE POPULATION (NOBTH=1)
          THEN THE CALLING PROGRAM MUST SET TBTH TO THE DESIRED
          NUMBER OF BIRTHS

 IF(NOBTH .EQ. 1) GO TO 65
 TBTH = 0.
 DO 55 J=1,NC
   TBTH = TBTH + BR(J)*POP(J,1)
 CONTINUE
 IF(NSEX .NE. 2) GO TO 68

      FOR TWO SEXES BIRTHS ARE APPORTIONED BETWEEN THEM ACCORDING
          TO VARIABLE PBF

 BTH(1) = PBF*TBTH
 BTH(2) = TBTH - BTH(1)
 IF(POPS(1) .EQ. 0.) GO TO 70
 SR = POPS(2)/POPS(1)
 GO TO 75

      FOR ONE SEX ALL BIRTHS GO TO THAT SEX AND SEX RATIO IS ZERO

 BTH(1) = TBTH
 SR = 0.

      CRUDE BIRTH RATE, DEATH RATE, AND GROWTH RATES

 IF(TPOP .EQ. 0.) GO TO 80
 CBR = TBTH/TPOP
 CDR = TDTH/TPOP
 PGRN = 100.*(CBR - CDR)
 PGRT = PGRN - 100.*TMR/TPOP
 RETURN
 CBR = 0.
 CDR = 0.
 PGRN = 0.
 PGRT = 0.
 RETURN
 END
```

```
SUBROUTINE DELLVF(RIN,ROUT,R,STRG,PLR,DEL,DELP,DT,IDTU,K)

TITLE - DISTRIBUTED DELAY WITH VARIABLE DELAY TIME AND STORAGE
        LOSSES

VERSION - 1A

DATE - 1 NOV 1974

PURPOSE - SIMULATION OF A DISTRIBUTED TIME DELAY PROCESS WITH
          VARIABLE DELAY TIME AND STORAGE LOSSES

INPUT REQUIREMENTS

    VALUES TRANSMITTED THROUGH THE ARGUMENT LIST -- DEL, DELP,
              DT, IDTU, K, PLR, R(I), RIN, STRG

    VARIABLES REQUIRING VALUES WHICH WILL NOT CHANGE DURING A RUN
              -- DT, IDTU, K
    VARIABLES REQUIRING VALUES WHICH MAY CHANGE DURING A RUN
              -- DEL, PLR, RIN
    VARIABLES COMPUTED BY THIS ROUTINE REQUIRING INITIAL VALUES
              -- DELP, R(I), STRG

OUTPUT VARIABLES

    VALUES TRANSMITTED THROUGH THE ARGUMENT LIST -- R(I), ROUT,
              STRG

VARIABLE DEFINITIONS

    DEL - CURRENT LENGTH OF DELAY IN TIME UNITS.

    DELP - PREVIOUS LENGTH OF DELAY IN TIME UNITS.

    DT - SIMULATION TIME INCREMENT IN TIME UNITS.

    IDT - NUMBER OF SUBDIVISIONS OF DT REQUIRED. UNITLESS.

    IDTU - USER-SPECIFIED NUMBER OF SUBDIVISIONS OF DT. UNITLESS.

    K - ORDER OF DELAY. UNITLESS.

    PLR - PROPORTIONATE LOSS RATE IN PROPORTION/UNIT TIME.

    R(I) - INTERMEDIATE RATES IN UNITS/UNIT TIME.(STAGE OF DELAY)

    RIN - INPUT RATE IN UNITS/UNIT TIME.

    ROUT - OUTPUT RATE IN UNITS/UNIT TIME.

    STRG - STORAGE IN UNITS.
```

```
      DIMENSION R(1)
      FK = FLOAT(K)
      B = 1. + (DEL-DELP)/(FK*DT) + PLR*DELP/FK
      IDT = 1. + 2.*B*DT*FK/DELP
      IF(IDT .LT. IDTU) IDT = IDTU
      A = FK*DT/(DELP*FLOAT(IDT))
      DELP = DEL
      KM1 = K-1
      DO 20 J=1,IDT
        IF(K .EQ. 1) GO TO 15
        DO 10 I=1,KM1
          R(I) = R(I) + A*(R(I+1)-B*R(I))
10      CONTINUE
15      R(K) = R(K) + A*(RIN-B*R(K))
20    CONTINUE
      STRG = 0.
      DO 30 I=1,K
        STRG = STRG + R(I)*DEL/FK
30    CONTINUE
      ROUT = R(1)
      RETURN
      END
```

References

[1] Posada, Alvaro, A Simulation Analysis of Policies for the Northern Colombia Beef Cattle Industry, unpublished Ph.D. dissertation, Michigan State University, 1974.

[2] Manetsch, Thomas J. and Gerald L. Park, Systems Analysis and Simulation with Applications to Economic and Social Systems, Part II, Preliminary Edition (East Lansing: Division of Engineering Research, Michigan State University, 1973), Chapter 12.

CLASS Documents

Number	Title
1	"Computer Library for Agricultural Systems Simulation: A Progress Report," Michael H. Abkin and Tom W. Carroll. (July 1976)
2	"Software Standards Manual." (July 1976)
3	"Policy Analysis Language, Version 2.3, Programmer's Guide for CDC Cyber Computers," Claudia S. Winer and Chris Wolf. (July 1976)
4	"Policy Analysis Language, Version 2.3, Reference Manual," Claudia S. Winer anc Chris Wolf. (July 1976).
5	"Policy Analysis Language, Version 2.3, Guide to Using a PAL Program," Claudia S. Winer and Chris Wolf. (July 1976)
6	"DEMOGC: Demography with Distributed Age Cohorts," Michael H. Abkin and Chris Wolf. (July 1976)
7	"DEMOGD: Demography with Discrete Age Cohorts," Michael H. Abkin and Chris Wolf. (July 1976)
8	"Distributed Delay Routines: DEL, DELS, DELF, DELLF, DELVF, DELLVF," Michael H. Abkin and Chris Wolf. (July 1976)
9	"Table Functions: TABEL, TABEX, TABUL, TABUX," Michael H. Abkin, Chris Wolf, and Tom W. Carroll. (July 1976)
10	"AGACC: Accounting Routine for the Agricultural Sector," Dennis Pervis and Chris Wolf. (July 1976)
11	"User's Guide for the Beef Cattle Enterprise Simulation Model," Michael R. Jaske. (July 1976)
12	"User's Guide to SYSOPT: An Interactive System Optimization Computer Program," Marcus Buchner. (August 1976)