

AGENCY FOR INTERNATIONAL DEVELOPMENT WASHINGTON, D. C. 20523 BIBLIOGRAPHIC INPUT SHEET		FOR AID USE ONLY <i>Batch 63</i>
1. SUBJECT CLASSI- FICATION	A. PRIMARY TEMPORARY	
	B. SECONDARY	
2. TITLE AND SUBTITLE Policy Analysis Language, version 2.3: guide to using a PAL program		
3. AUTHOR(S) Winer, Claudia; Wolf, Christopher		
4. DOCUMENT DATE 1976	5. NUMBER OF PAGES 14p.	6. ARC NUMBER ARC
7. REFERENCE ORGANIZATION NAME AND ADDRESS Mich.State		
8. SUPPLEMENTARY NOTES (<i>Sponsoring Organization, Publishers, Availability</i>) (In Computer Library for Agr.Systems Simulation. CLASS doc.no.5)		
9. ABSTRACT (LIBRARY & INFO.SCIENCE R&D)		
10. CONTROL NUMBER <i>PN-AAE - 008</i>		11. PRICE OF DOCUMENT
12. DESCRIPTORS Decision making Manuals Programming languages Simulation		13. PROJECT NUMBER
		14. CONTRACT NUMBER CSD-2975 Res.
		15. TYPE OF DOCUMENT

POLICY ANALYSIS LANGUAGE
VERSION 2.3
GUIDE TO USING A PAL PROGRAM

Claudia S. Winer

Chris Wolf

COMPUTER LIBRARY FOR AGRICULTURAL SYSTEMS SIMULATION

CLASS Document No. 5

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
Responding to the Computer	2
Standard and Current Values	3
Constraint Violations	3
Modes of Conversation	4
Entering Commands	5
Commands	6
1. <code>>var = value</code>	6
2. <code>>CHECK</code>	6
3. <code>>CONVERSATION (parm)</code>	7
4. <code>>EXIT</code>	7
5. <code>>EXPLAIN (var)</code>	7
6. <code>>INITIALIZE</code>	7
7. <code>>OK</code>	7
8. <code>>PRINT (parm, var)</code>	8
9. <code>>RESET</code>	8
10. <code>>RUN</code>	8
11. <code>>SAVE</code>	9
12. <code>>STORIV</code>	9
13. <code>>VARIABLES</code>	9

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 University 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 Policy Analysis Language (PAL) was developed in order to make computer simulation models easily accessible to nontechnical users. A PAL program will carry out a "conversation" with a user at a computer terminal. During the conversation, the user may be presented with explanations of a model and its characteristics, given choices as to what areas he would like to explore, asked for input values, and shown the results of a model run. Although developed for use with simulation models, PAL is suitable for use with almost any FORTRAN program. An experienced FORTRAN programmer can learn to write PAL programs in an hour or two.

A PAL program may be constructed to offer the user two alternative sets of messages--typically a set of detailed explanations for the novice user and a set of brief prompts for the experienced user. In addition, whenever the user is expected to enter input, he may choose to enter commands for changing and printing data values, running the model, or performing various other actions. If desired, all functions may be performed by commands, bypassing the conversation entirely.

The following instructions will tell you all you need to know about using a PAL program. The program will conduct a conversation with you at a computer terminal. All you need to do is to answer the questions it asks. If at any time you are confused about what to do enter a question mark (?).

Responding to the Computer

Whenever the computer is waiting for a response, it will let you know by printing a single or double asterisk (* or **) or an arrow (→) at the beginning of a line. When it does, type the proper response and hit the "return" key. Unless the machine indicates that it is waiting for a response, do not enter anything. Please be patient if the terminal pauses for a while.

When typing in numbers, there are a few rules to follow. You may put blank spaces before and after your answer, but there must be no blanks in the middle. You may include a minus sign and a decimal point where necessary. Commas are not allowed.

You will see three different types of questions. The first two types of questions are indicated by a single asterisk (*), and the third type by a double asterisk (**).

The first type is a multiple choice question with numbered answers. For these, enter the number that corresponds to the answer of your choice. A second type of question is a yes or no. Enter YES or Y or 1 for an affirmative answer and NO or N or 2 for a negative answer. If you want

a further explanation for a multiple choice or yes or no question, enter an E. If you are confused about what you may enter, type a question mark (?) and the possible responses will be explained to you.

The third type of question will ask you for the value of a particular item. Type in the value you want that item to have. If you enter a question mark (?), you will see a further explanation of the item whose value is being requested.

Standard and Current Values

The computer program will usually have a "standard" or "base-run" value assigned to each variable in it. When the model is run using these values, that is considered a "base run." For testing purposes one or more values may be changed and the results compared with those of the base run.

The value that a variable has at any given time is referred to as its "current" value. At the beginning all variables are initialized to their standard values so that the current and standard values are the same. Whenever a value is changed, it is only the current value that changes.

Constraint Violations

To ensure you of meaningful results, certain constraints may have been placed on the values that can be entered. For example, a percentage might be limited to values from 0 to 100. After you have entered all of your values, they will be checked against these constraints.

If any of your values have violated the constraints, a list of those that were violated will be printed. These will be in the form of FORTRAN

logical expressions. If you cannot interpret these, you should probably get help. On the line following the list an arrow (↔) will be printed.

At this point you may do one of two things. If you feel the values that you were warned about are reasonable, you may tell the computer to continue. This is done by typing >OK and hitting the carriage return. The second course you can follow is to enter commands to find out exactly what is wrong and then fix it. (See the section on commands.) The >PRINT command can be used to find out what the standard and current values of the offending variable are. The assignment command can be used to assign a new value to a variable. When you think you have fixed everything, use the >CHECK command to check the constraints again. If they are satisfied, the program will resume running where it left off. If there are still any violations, you must again proceed as indicated above. The program will not resume until you have either typed >OK or you have fixed the violations and typed >CHECK.

This covers all the basic necessities for using a PAL program. There is no need to read further unless you wish to use the more advanced features of PAL.

Modes of Conversation

There are three modes of conversation: long, short, and command. In the long mode a full text will be written out, as well as a list of answers for you to choose from. After you have done this several times and become familiar with it, you will no longer need the full text and

answers in order to know what to do. You may wish to change to the short mode where an abbreviated text is written and the answers are not printed. Should you get to the point where you do not need the conversation at all, you will want to change to the command mode. In the command mode, nothing is printed and you are expected to enter commands continuously. Unless you are told otherwise, assume you are in the long mode. For more information on changing the mode, see the section on commands.

If in the short mode you are asked a multiple choice or a yes or no question (these are indicated by a single asterisk), you can have the longer text of the question or the list of choices printed by typing in an L or a C, respectively.

Entering Commands

A command is an instruction that you give to the computer. It may be entered anytime the computer is waiting for a response from the terminal. Once the computer has performed the command, it will print an asterisk (*) or double asterisk (**) and wait for your response to the previous question. You may continue to enter commands for as long as you like.

Each command begins with a greater-than sign (>) which is immediately followed by the command name. If you are entering a command on a line that already has an arrow (->), as in command mode or when constraints have been violated, the greater-than (>) is not necessary. Some commands have a list of parameters which follows the name. If a list is to be used, it must be enclosed in parentheses. No blanks are allowed anywhere in a command, except between the command name and the opening parenthesis of the parameter list.

Only one command may be entered at a time. The entire command must fit on one line, and that line must be no longer than 72 characters. Some commands may be abbreviated by using just the part that is underlined in the list below.

The >PRINT command and the assignment command can be used to refer to more than one consecutive element of an array. This is done by specifying a range for any or all of the subscripts. For example, X(1-5) refers to X_1 through X_5 . A range can be specified for more than one of the subscripts. Y(2-4,4-5) refers to $Y_{2,4}$, $Y_{3,4}$, $Y_{4,4}$, $Y_{2,5}$, $Y_{3,5}$, $Y_{4,5}$.

A basic explanation of each command is given here. If you want more detailed information see Appendix C of the PAL Reference Manual (CLASS-4).

Commands

1. *>var = value*

The assignment command gives the value indicated to the variable specified. Its form is somewhat different from that of the other commands. The greater-than (>) is followed by a variable name, then an equal sign (=) and then the value that you would like the variable to have. If the variable is an array, the subscripts must be integers. If you wish to set several elements of an array to the same value, it can be done by using a subscript range as follows:

>X(2-4)=0;

2. *>CHECK*

This command will check the current values of the variables to see if they meet the requirements of the constraints section. Any violations will be printed. See the section on constraint violations for more information.

3. >CONVERSATION (*parm*)

This command will change the conversation mode as specified by *parm*.

<i>parm</i>	<i>conversation mode</i>
LONG or L SHORT or S COMMAND or C	long short command

4. >EXIT

This command stops the PAL program.

5. >EXPLAIN (*var*)

If there is an explanation for this variable, it will be printed. The variable name may include subscripts, but they will be ignored.

6. >INITIALIZE

This sets all type INITIAL variables to the values which they had at the time the last >STORIV command was executed. If no >STORIV command has been executed, this command will not work properly. This command differs from >RESET in that it affects only type INITIAL variables.

7. >OK

This command indicates your approval of any constraint violations which have occurred. The program will resume execution.

8. >PRINT (*parm*, *var*)

This command will print the value of a variable. The parameter specifies whether you want the current or the standard value printed. See the section on current and standard values. If *parm* is a C, the current value is printed; if it is an S, the standard value is printed.

If the variable is an array, then the subscripts must be specified and they must be integers or integer ranges. Six values are printed per line. If more than one subscript range is specified for an array, the values will be printed by columns. This means that with two subscript ranges, for example, the second subscript would be held constant at its starting value while the first one varies over its range. Then the second subscript is increased by one, and the first subscript varied over its entire range again. This continues until all of the values have been printed.

9. >RESET

This command will reset all variables back to their standard values; previous changes to current values will be lost.

10. >RUN

This command will cause the model to be executed. Some programs also require other commands, such as >STORIV, to be executed first. This command differs from all of the others, in that it can be used only when the program is operating in command mode. If this command is used when not in command mode an error message will be printed.

If the conversation you are using is overlaid and you attempt to change back to conversation mode after using a >RUN command, it may cause a fatal error in the program. If the program is not overlaid, there will be no problems. Talk to the model designer if you have any questions about this.

11. >SAVE

This command will write out all of the current values on a file in a form suitable for use as a standard value file. This means that at any point in your use of a PAL program you can save all of the current values for use at some later time. The values are written on TAPE8 and you must catalog this file after you are finished using the PAL program. Only one file of values can be saved during each use of the program because each use of the >SAVE command will write over any values written by a previous >SAVE command. So the values on the file at the end of the run will be those written by the last >SAVE command.

12. >STORIV

This takes the current values of all type INITIAL variables and stores them in a special table. It would generally be used immediately before a >RUN command. It is used in conjunction with the >INITIALIZE command.

13. >VARIABLES

This command prints the names and dimensions of all variables used in the PAL program. It is useful if you have some familiarity with the program but have forgotten the exact names used. If there are a lot of variables, it may produce a long listing.

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 and 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)