4-5

rid Computers

The scope of analog computers has been considerably extended by developments in solid-logic electronic devices, (1) and (11). Analog computers always used a few nonlinear elements, such as multipliers or function generators. Originally, such devices were expensive to make. Solid-logic devices, in addition to improving the design and performance of operational amplifiers, have made such nonlinear devices cheaper and easier to obtain. They have also extended the range of devices. Among the elements that can easily be associated with analog computers are circuits that carry out the logical operations of Boolean algebra, store values for later use, compare values, and operate switches for controlling runs.

The term *hybrid computer* has come to describe combinations of traditional analog-computer elements, giving smooth, continuous outputs, and elements carrying out such nonlinear, digital operations as storing values, switching, and performing logical operations. Originally, the term had the connotation of extending analog-computer capabilities, usually by the addition of special-purpose, and often specially constructed, devices. More recently, few purely analog computers are built. Instead, computers with large numbers of standard, nonlinear elements are readily available.

Hybrid computers may be used to simulate systems that are mainly continuous, but do, in fact, have some digital elements—for example, an artificial satellite for which both the continuous equations of motion, and the digital control signals must be simulated. The general technique of applying hybrid computers follows the methods outlined in Sec. 4-4, with blocks in the diagrams representing the nonlinear elements as special functions, rather than as purely mathematical operators. Hybrid computers are also useful when a system that can be adequately represented by an analog computer model is the subject of a repetitive study. It might, for example, be necessary to search for a maximum value. Using digital devices to save

4-6

Digital-Analog Simulators

To avoid the disadvantages of analog computers, many digital computer programming languages have been written to produce *digital-analog simulators*. They allow a continuous model to be programmed on a digital computer in essentially the same way as it is solved on an analog computer. The languages contain macro-instructions that carry out the action of adders, integrators, and sign changers. A program is written to link together these macro-instructions, in essentially the same manner as operational amplifiers are connected in analog computers.

More powerful techniques of applying digital computers to the simulation of continuous systems have been developed. As a result, digital-analog simulators are not now in extensive use. A history of their development will be found in Ref. (19).

4-7

Continuous System Simulation Languages (CSSLs)

Confining a digital computer to routines that represent just the functions of an analog or a hybrid computer, as is done with a digital-analog simulator, is clearly a restriction. To remove the restriction, a number of *continuous system simulation languages* (abbreviated to CSSLs) have been developed. They use the familiar statement-type of input for digital computers, allowing a problem to be programmed directly from the equations of a mathematical model, rather than requiring the equations to be broken into functional elements, (8). Of course, a CSSL can easily include macros, or subroutines, that perform the function of specific analog elements, so that it is possible to incorporate the convenience of a digital-analog simulator. In fact, most implementations of a CSSL, in addition to including a subset of standard digital-analog elements, allow the user to define special purpose elements that correspond to operations that are particularly important in specific types of applications.

In extending beyond the provision of simple analog functions, such as addition and integration, CSSLs include a variety of algebraic and logical expressions to describe the relations between variables. They, therefore, remove the orientation toward linear differential equations which characterizes analog methods. A general specification of the requirements of a CSSL has been published by a user group, (23). Several implementations have been produced, (3). One particular CSSL that will be described here, to illustrate the nature of these languages, is the *Continuous System Modeling Program*, Version III, (CSMP III), (14), and (22).

A CSMP III program is constructed from three general types of statements:

Structural statements, which define the model. They consist of FORTRAN-like statements, and functional blocks designed for operations that frequently occur in a model definition.

Data statements, which assign numerical values to parameters, constants, and initial conditions.

Control statements, which specify options in the assembly and execution of the program, and the choice of output.

Structural statements can make use of the operations of addition, subtraction, multiplication, division, and exponentiation, using the same notation and rules as are used in FORTRAN. If, for example, the model includes the equation

$$X = \frac{6Y}{W} + (Z - 2)^2$$

the following statement would be used:

```
X = 6.0*Y/W + (Z - 2.0)**2.0
```

Note that real constants are specified in decimal notation. Exponent notation may also be used; for example, 1.2E−4 represents 0.00012. Fixed value constants may also be declared. Variable names may have up to six characters.

There are many functional blocks which, in addition to providing operations specific to simulation, duplicate many of the mathematical-function subprograms of FORTRAN. Among these are the exponential function, trigonometric functions, and functions for taking maximum and minimum values. Figure 4-3 is a list of eleven of the functional blocks.

The following functional block is used for integration:

```
Y = INTGRL(IC,X)
```

where Y and X are the symbolic names of two variables and IC is a constant. The variable Y is the integral with respect to time of X, and it takes the initial value IC.

Of the data statements, one called INCON can be used to set the initial values of the integration-function block. Other parameters can be given a value for a specific run with the CONST control statement. A control statement called PARAM can also be used to assign values to individual parameters, but its chief purpose is to specify a series of values for one parameter (and only one). The model will be run with the specified parameter taking each of the values on successive runs of the same model. Examples of how these statements are written are

```
CONST  A = 0.5, XDOT = 1.25, YDOT = 6.22
PARAM  D = (0.25, 0.50, 0.75, 1.0)
```

As indicated, several values can be specified with each statement by separating the values with commas.

Among the control statements is one called TIMER, which must be present to specify certain time intervals. An integration interval size must be specified. For adequate accuracy, it should be small in relation to the rate at which variables change value. The total simulation time must also be given. Output can be in the form of printed tables and/or print-plotted graphs. Interval sizes for printing and plotting results need to be specified. The following is an example:

```
TIMER  DELT = 0.005, FINTIM = 1.5, PRDEL = 0.1, OUTDEL = 0.1
```

The items specified are

DELT	Integration interval
FINTIM	Finish time
PRDEL	Interval at which to print results
OUTDEL	Interval at which to print-plot

The set of structural, data, and control statements for a problem can be assembled in any order but they must end with an **END** control statement. However, control statements which define another run of the same model can follow an **END** statement if they also are terminated by another **END** statement. This can be repeated many times, until an **ENDJOB** statement signals end of all runs. A completely separate model can then follow the **ENDJOB** statement.

As an example, Fig. 4-4 shows a CSMP III program for the automobile wheel suspension problem. It has been coded for the case where $M = 2.0$, $F = 1$, and

```
TITLE AUTOMOBILE SUSPENSION SYSTEM
*
PARAM D = (5.656, 16.968, 39.592, 56.56, 113.12)
*
 X2DOT = (1.0/M)*(K*F - K*X - D*XDOT)
 XDOT = INTGRL(0.0,X2DOT)
 X = INTGRL(0.0,XDOT)
*
CONST M = 2.0, F = 1.0, K = 400.0
TIMER DELT = 0.005, FINTIM = 1.5, PRDEL = 0.05, OUTDEL = 0.05
PRINT X, XDOT, X2DOT
PRTPLT X
LABEL DISPLACEMENT VERSUS TIME
END
STOP
```

Figure 4-4. CSMP III coding for the automobile suspension problem.

```
TIME         X
 .0           .0       +         I         I         I         I         I         I
5.0000E-02   .20942    I---------+         I         I         I         I         I
 .10000      .65543    I---------I---------I---------I--+      I         I         I
 .15000     1.0819     I---------I---------I---------I---------I---------I---+     I
 .20000     1.3290     I---------I---------I---------I---------I---------I---------I------
 .25000     1.3620     I---------I---------I---------I---------I---------I---------I------
 .30000     1.2422     I---------I---------I---------I---------I---------I---------I-+
 .35000     1.0691     I---------I---------I---------I---------I---------I--+      I
 .40000      .92886    I---------I---------I---------I---------I-----+   I         I
 .45000      .86489    I---------I---------I---------I---------I--+      I         I
 .50000      .87583    I---------I---------I---------I---------I---+     I         I
 .55000      .93152    I---------I---------I---------I---------I------+  I         I
 .60000      .99472    I---------I---------I---------I---------I---------+         I
 .65000     1.0381     I---------I---------I---------I---------I---------I-+       I
 .70000     1.0517     I---------I---------I---------I---------I---------I--+      I
 .75000     1.0403     I---------I---------I---------I---------I---------I-+       I
 .80000     1.0171     I---------I---------I---------I---------I---------I+        I
 .85000      .99527    I---------I---------I---------I---------I---------+         I
 .90000      .98281    I---------I---------I---------I---------I--------+I         I
 .95000      .98138    I---------I---------I---------I---------I--------+I         I
1.0000       .98772    I---------I---------I---------I---------I--------+I         I
1.0500       .99667    I---------I---------I---------I---------I---------+         I
1.1000      1.0038     I---------I---------I---------I---------I---------+         I
1.1500      1.0070     I---------I---------I---------I---------I---------+         I
1.2000      1.0064     I---------I---------I---------I---------I---------+         I
1.2500      1.0034     I---------I---------I---------I---------I---------+         I
1.3000      1.0002     I---------I---------I---------I---------I---------+         I
1.3500       .99799    I---------I---------I---------I---------I---------+         I
1.4000       .99733    I---------I---------I---------I---------I---------+         I
1.4500       .99795    I---------I---------I---------I---------I---------+         I
1.5000       .99915    I---------I---------I---------I---------I---------+         I
```

Figure 4-5. Print-plot output for the automobile suspension problem.

4-9
Hybrid Simulation

For most studies the model is clearly either of a continuous or discrete nature, and that is the determining factor in deciding whether to use an analog or digital computer for system simulation. However, there are times when an analog and digital computer are combined to provide a *hybrid simulation*, (2). The form taken by hybrid simulation depends upon the application. One computer may be simulating the system being studied, while the other is providing a simulation of the environment in which the system is to operate. It is also possible that the system being simulated is an interconnection of continuous and discrete subsystems, which can best be modeled by an analog and digital computer being linked together.

Telelphone
ile

The introduction of hybrid simulation required certain technological developments for its exploitation. High-speed converters are needed to transform signals from one form of representation to the other, (13). As a practical matter, the availability of mini-computers has made hybrid simulation easier, by lowering costs and allowing computers to be dedicated to an application. The term "hybrid simulation" is generally reserved for the case in which functionally distinct analog and digital computers are linked together for the purpose of simulation. It should not be confused with the use of digital elements added to the operational amplifiers of an analog computer, as discussed in Sec. 4-5, when describing hybrid computers.

Def^n

4-10
Feedback Systems

A significant factor in the performance of many systems is that a coupling occurs between the input and output of the system. The term *feedback* is used to describe the phenomenum. A home heating system controlled by a thermostat is a simple example of a feedback system. The system has a furnace whose purpose is to heat a room, and the output of the system can be measured as room temperature. Depending upon whether the temperature is below or above the thermostat setting,