DIGITAL SIMULATION OF CONTINUOUS SYSTEMS AND THE NEW IMP PROGRAM

by Daniel M. Brandon, Jr.

INTRODUCTION

During the last two decades simulation has become a well established tool for the design and investigation of continuous systems. These types of systems are found throughout the engineering and science disciplines ranging from a simple servo system to very complex systems (as large electronic networks, the dynamics of chemical and transfer processes, nuclear processes, and physiological and environmental processes). Continued developments in simulation hardware and software have permitted the investigation of increasingly complex problems which in turn have stimulated further developments in simulation capabilities.

Continuous systems are characterized by a set of ordinary or partial differential equations. These types of problems were first solved on analog computers. For more than a decade the analog computer handled the major portion of this type of work. Simulation with the analog computer was limited and difficult. Some of this difficulty was removed by employing a digital computer along with the analog (hybrid simulation). As the general purpose digital computer evolved and became much faster, it was found that all of the work (integration included) could be done more accurately and cheaper with the digital equipment. The advantages of the digital computer are:

1. No need for magnitude scaling (within certain limits).
2. No need for time scaling (again within limits of numerical stability).
3. Much greater accuracy.
4. Much larger systems in terms of number of differential equations.
5. No hardware setup.
6. Easier maintenance.
7. Better matching utilization.
8. Rapid switching from one simulation to another.
9. Simpler and more compact storage of simulation models.
10. Availability of mathematical functions.
11. Greater flexibility in model content (functions, data, non-linearities, etc.)
12. Greater flexibility in simulation environment (time sharing, terminals, graphics) and man-machine symbiosis.

The current advance in digital technology, both hardware and software, is very rapid. As many more circuit functions and memory are being accomplished by smaller and smaller chips (and other elements), the times (by decreasing the distances signals have to travel) and cost (less raw material) of digital tasks are decreasing very quickly. Digital simulation is also becoming easier and cheaper due to improved software. These software improvements are due largely to new mathematical procedures and algorithms.

DIGITAL SIMULATION OF CONTINUOUS SYSTEMS

The digital simulation of continuous systems is the use of a general purpose digital computer to solve sets of simultaneous differential equations and associated relationships, functions, etc. Systems may be simulated in a manner similar to that of an analog or hybrid computer, but the manner of simulation need not be limited to the "block diagram" approach. Digital simulation, on other than a relatively small scale, came into wide usage initially to check the solutions produced by analog computers. In the sixties digital programs that simulated analog computers were developed to remove the scaling, inconvenience, and accuracy problems of analog computers and to ease the difficulty of direct digital programming. In the late sixties and seventies these digital programs have been continually refined to provide faster and more accurate solutions and greater ease of use.

For purposes of illustrating similarities and differences, digital simulation software may be divided into six classes as shown in Figure 1. Class one contains the standard general programming languages such as FORTRAN. To use Class one software for simulation one would have to be skilled in the programming of that language and also in the area of numerical methods and applied mathematics. If the individual were expert in these fields, an efficient program could be written for a particular application. However, this program would only be useful for this one application and could not be used for others. The FORTRAN program would simply be compiled into object code and executed as shown in Figure 2 (using FORTRAN as an example). Class two software includes subroutines written to handle specific tasks involved in simulation such as forcing functions, integration, etc. An individual with programming knowledge and some understanding of numerical methods could write a program with a general language (such as FORTRAN) and use the available subroutines to perform the necessary numerical tasks. The individual would have to control the entire process through his main program. Again, the same disadvantages as with Class one would apply here.

Class three software includes the early digital simulation programs, most notable of which was MIDAS (4). This was developed by the Air Force around 1964. It was designed to be an analog simulator and required a "block diagram" type of input. Milne and Reynolds pece type of explicit integration was used (fifth order). Another popular package in this class was developed by General Electric and called PACTOLUS. It used second order Runge-Kutta explicit integration. Software in this class usually involved its own special compiler as shown in Figure 4. These early simulation languages were much easier to use for small problems (less than twenty or so differential equations) than Class one or two, however, they lacked many mathematical capabilities and programming flexibility.

Figure 1

Software Types for Continuous System Simulation

Class I (basic languages)

FORTRAN
BASIC
PL-1
ALGOL
FORTRAN EXTENDED
ADVANCED PROGRAMMING LANGUAGE (APL)

Class II (scientific subroutines)

Class III (block diagram orientation)

MIDAS
PACTOLAS

Class IV (structured FORTRAN macros)

LEANS
IMP

Class V (continuous simulation languages)

MIMIC
DSL
CSMP
CSSL

Class VI (structured problem oriented programming)

SLANG
PROSE

The next software type developed was Class five simulation languages. These languages allow direct coding of equations (in a specific "Simulation Language") as well as an optional block diagram (or transfer function) approach. In addition, they provided libraries of functions, use of FORTRAN-like expressions, and some logical control over program execution by the user. The first such system of this type was MIMIC (7), which was developed by the Air Force. It used fourth-order Runge-Kutta integration. Later, others were developed by IBM: DSL (9,10) and CSMP (2, 11). The latest one (developed by Programming Sciences Corporation) is called CSSL III (3) and is an attempt to standardize on a single language for simulation with standards set by the SCI (Simulation Software Committee of Simulation Councils, Inc.) (8). DSL, CSMP, and CSSL offer a choice of many explicit integration methods. These types of programs use a "translator" (Figure 5) to produce a FORTRAN subroutine from the special "simulation language" of the system. These translators allow small problems to be set up very easily, however, their overhead burden of core storage, disc file usage, and central processor time can be quite expensive. The trade off is for man time at the expense of computer assets.

Class six software represents a true problem oriented programming language. For the specific problem of continuous simulation the features of these types of systems are similar to Class five. However, other problems of a related or dissimilar nature are also easily set up such as: optimization, discrete simulation, and other mathematical procedures. In addition, problems may be easily intermixed such as continuous and discrete simulation or continuous simulation and optimization. The only present members of this software class are SLANG and PROSE (12,13).

Class four software attempts to remove the difficulties of overhead burden and lack of flexibility from Class five and six software. This is done at the expense of programming ease for small problems. Figure 6 shows the operation of this type of approach. The user writes a FORTRAN (or other basic language) subroutine describing his particular problem. Thus the user is required to have knowledge of FORTRAN but not of numerical methods. This Class of software is usually used for problems that are large or where maximum efficiency and minimum computer assets are required. The IMP system (1) belongs to this class as does the LEANS (5,6) system.

THE IMP PROGRAM

There are still, however, major obstacles in the path of routine economic simulation of many systems. These types of systems are characterized by either or both very large numbers of differential equations (sometimes resulting from partial differential equations) or "stiff" (ratio of largest to smallest eigenvalue of system is large) differential equation sets. Unfortunately, in many disciplines, these are the types of problems that occur very frequently or whose solutions are the most valuable.

The recent IMP system was primarily designed to remove these difficulties by two novel procedures:

1. An A-stable fixed or variable step size implicit integration algorithm with improved accuracy and applicability of use over currently available implicit algorithms (1).

FIGURE 2

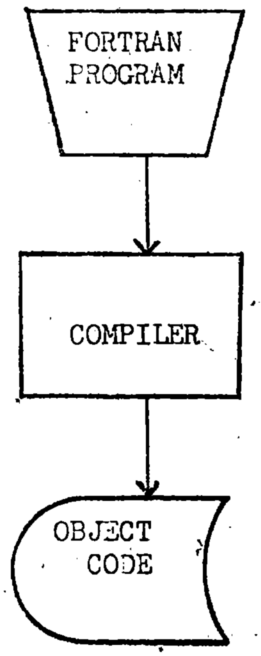


FIGURE 3

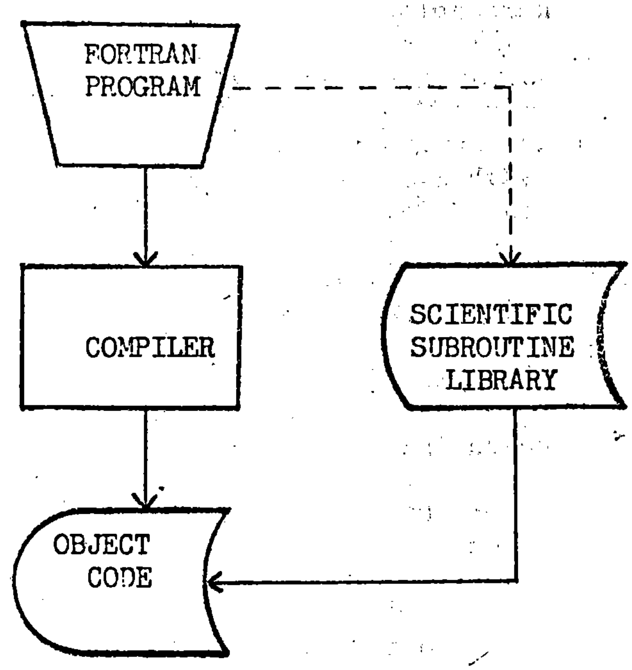


FIGURE 4

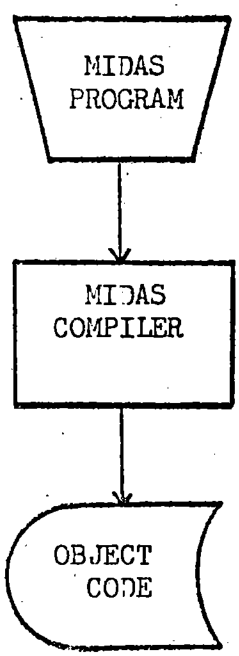
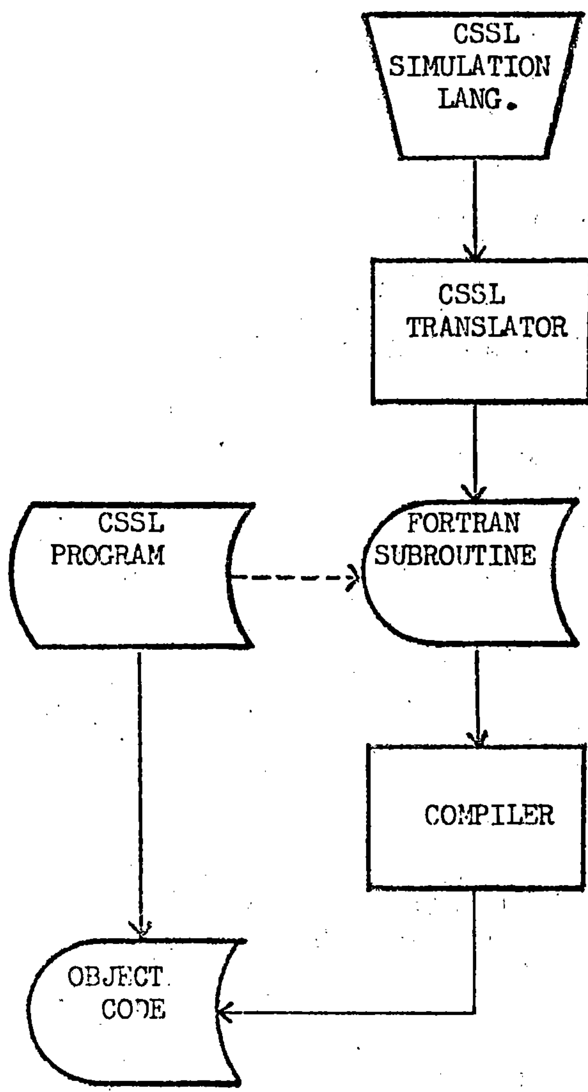


FIGURE 5



- - - - - TRANSFER OF CONTROL

———— JOB STEP TRANSFER

Figure 7

Procedural Capabilities

Related to Continuous System Simulation

Capability	Language: IMP	FORTRAN	APL	PROSE
no translation required	o	o	o	
free setup (overlays, etc.)	o	o	o	
relocatable subroutines	o	o		o
free field coding			o	o
free I/O (any device any time)	o	o	o	
dynamic storage allocation	*		o	o
problem oriented diagnostics	o		o	o
vector/matrix operations	o		o	o
sparse matrix storage and operations	o			o

*automatic core allocation

SEVENTH ANNUAL SIMULATION SYMPOSIUM

The Seventh Annual Simulation Symposium will be held in Tampa, March 13-15, 1974.
• Call for Papers: Initially you are to send: (a) working title for the paper; (b) an abstract of no less than 1000 words; (c) full name of author(s); (d) title or position and company or university affiliation; (e) complete mailing address of each author; (f) statement that author will personally present paper at the symposium. Send the data by April 6, 1973 {or blame it on the mail delay – Ed. } to:

Michael Brambier Hartford Insruance Group Hartford Plaza
Hartford CT 06115 | phone: 203 ∘ 547 - 5824 |

* For other questions about the Symposium, write to:

Annual Simulation Symposium
Post Office Box 22573
Tampa FL 33622

Figure 8

Mathematical Capabilities

Related to Continuous System Simulation

Capability	IMP	PROSE	CSSL
	Language		
explicit integration		o	o
implicit integration	o	o	
simultaneous equations	o	o	
automatic partial derivatives		o	
matrix operations	o	o	
parallel problems		o	
sparse matrix storage and operations	o		
optimization interface	o		o
automatic optimization		o	
local integration error control	o	o	o
automatic integration	o	o	o
round-off error monitors	o		
iterative matrix methods	o		
matrix scaling	o		
vector scaling	o		
optimal ordering	o		
limit stop interrupts	o		
end of integration step interrupt	o		
end of communication interval interrupt	o	o	o
optional nonlinear procedures	o		

Figure 9

Major Disadvantages of Simulation Programs

CSSL:

1. Translator required
2. Cannot adequately handle "stiff" problems
3. Lacks flexibility (I/O, overlays, interrupts)*
4. Difficult to program for large systems
5. Slow execution of large systems
6. Large storage requirements for large systems

PROSE:

1. Translator required
2. Lacks flexibility (I/O, overlays, interrupts)*
3. Difficult to program for large systems
4. Slow execution of large systems
5. Large storage requirements for large systems

IMP:

1. Relatively more difficult to use for small problems
2. Requires knowledge of FORTRAN
3. Requires problem formulation in matrix notation

* Very important for use in time sharing, real time, and interactive graphics systems.

REFERENCES

1. Brandon, D., "IMP - A Software System for the Direct or Iterative Solution of Large Differential and/or Algebraic Systems - General Manual", University of Connecticut, Chemical Engineering Department, 1972.

2. Brennan, R. D. and M. Y. Silverberg, "The System/360 Continuous System Modeling Program", Simulation, Dec. 1968

3. CSSL III - General Manual - Programming Sciences Corporation, 1971

4. Harnett, R. T., Sansom, F. J. and M. L. Warshawsky, "MIDAS - An Analog Approach to Digital Computation", Simulation, Vol. 3, No. 3, Sept. 1964

5. LEANS III - "An Advanced Programming System for Continuous System Simulation," Lehigh University - Department of Computer Science, 1971

6. Morris, S. M., "SALEM: A Programming System for the Simulation of Systems Described by Partial Differential Equations", Ph.D. Thesis, Department of Chemical Engineering, Lehigh University, 1967

7. Peterson, H. E. and F. J. Sanson, "MIMIC - A Digital Simulator Program", Sesca Internal Memo 65-12, Wright Patterson Air Force Base, Ohio

8. Strauss, J. C., D. C. Augustine, M. S. Fineberg, B. B. Johnson, R. N. Lineberger, and F. J. Sanson, "The SCI Continuous System Simulation Language", Simulation December 1967

9. Syn, W. M. and R. N. Lineberger, "DSL/90: A Digital Simulation Program for Continuous System Modeling," Proc. Spring Joint Comp. Conf., 1966

10. Syn, W. M., "DSL/90 User's Guide", IBM

11. System/360 CSMP User's Manual, IBM (360A-CX-16X), 1967

12. Thames, Jr., J. M., "SLANG: A Problem Solving Language for Continuous Model Simulation and Optimization", Proc. ACM Nat. Conf. 1969, San Francisco

13. Thames, Jr., J. M., "Introduction to PROSE", Solveware Associates, 1971