the 15th annual

SIMULATION Symposium

Edited by

Robert Massarotti

RECORD OF PROCEEDINGS

the 15th annual SIMULATION symposium

Edited by

Robert Massarotti

acm Association for Computing Machinery

IEEE COMPUTER SOCIETY

SCS THE SOCIETY FOR COMPUTER SIMULATION (Simulation Councils, Inc.)

IMACS INTERNATIONAL ASSOCIATION FOR MATHEMATICS AND COMPUTERS IN SIMULATION

ANNUAL SIMULATION SYMPOSIUM   TAMPA, FLORIDA

IEEE Computer Society Catalog No. 407 IEEE Catalog No. 82CH1744-2 Library of Congress 71-149514 ISSN 0272-4715 ACM Order No. 577820

IEEE COMPUTER SOCIETY PRESS

Additional copies may be procured from the:

ANNUAL SIMULATION SYMPOSIUM

c/o Mr. Edward DeYoung
130 Jay Street
Morton, Illinois 61550

OR

IEEE COMPUTER SOCIETY
10662 Los Vaqueros Circle
Los Alamitos, CA 90720
Computer Society Order No. 407

OR

IEEE SERVICE CENTER
445 Hoes Lane
Piscataway, NJ 08854
IEEE Catalog No. 82CH1744-2

OR

ACM ORDER DEPARTMENT
P.O. Box 64145
Baltimore, MD 21264
ACM Order No. 577820

LIBRARY OF CONGRESS Catalog Card Number: 71-149514
ISSN 0272-4715

Published by
IEEE Computer Society Press
1109 Spring Street, Ste. 201
Silver Spring, MD 20910

ANNUAL SIMULATION SYMPOSIUM

Board of Directors

Mr. R. Dysart CoNine
TRW Defense & Systems Group

Mr. Phillip N. Adams
ARMCO, Inc.

Mr. Victor P. Boyd
U. S. Postal Services

Dr. A. Wayne Bennett
VPI & SU

Dr. W. Marvin Bunker
General Electric Co.

Mr. Edward A. DeYoung
Caterpillar Tractor Co.

Mr. James Gantt
AIRMICS

Mr. Ron Huhn
Harris Corp.

Dr. Oscar N. Garcia
University of South Florida

Mr. Alexander Kran
IBM Corporation

Mr. Robert A. Massarotti
Naval Underwater Systems Center

Mr. Paul F. Roth
Department of Commerce

Dr. Fred O. Simons, Jr.
University of Central Florida

Committee Chairmen:

SPEAKERS

Mr. Robert A. Massarotti
Naval Underwater Systems Center

PROGRAM

Mr. James Gantt
AIRMICS

GRANTS

Dr. W. Marvin Bunker
General Electric Company

PUBLICITY

Mr. Alexander Kran
IBM Corporation

LOCAL ARRANGEMENTS

Ms. Linda Holbrook
GTE Service Corporation

PROCEEDINGS OF THE ANNUAL SIMULATION SYMPOSIA

First:	January 18-19, 1968 (no longer available)
Second:	January 15-17, 1969 Annual Simulation Symposium
Third:	January 14-16, 1970 Annual Simulation Symposium
Fourth:	March 10-12, 1971 Gordon & Breach, Science Publishers, Inc.
Fifth:	March 8-10, 1972 Gordon & Breach, Science Publishers, Inc.
Sixth:	March 7-9, 1973 Annual Simulation Symposium
Seventh:	March 13-15, 1974 (no longer available)
Eighth:	March 12-14, 1975 Annual Simulation Symposium
Ninth:	March 17-19, 1976 Annual Simulation Symposium
Tenth:	March 16-18, 1977 Annual Simulation Symposium
Eleventh:	March 15-17, 1978 Annual Simulation Symposium
Twelfth:	March 14-16, 1979 Annual Simulation Symposium
Thirteenth:	March 19-21, 1980 Annual Simulation Symposium
Fourteenth:	March 17-20, 1981 Annual Simulation Symposium

PREFACE

The Annual Simulation Symposium is a non-profit corporation formed to provide a forum for the interchange of information related to digital computer simulation. Its objectives are:

- to provide a continuation of the forum for the exchange of working experiences in the field of digital computer simulation, to
- permit an opportunity to survey the state-of-the-art across a broad range of applications, to
- demonstrate the widest possible range of simulation languages, with their strengths for individual problems, and to
- furnish an opportunity for comprehensive understanding of techniques through organized question and answer periods and personal contact. It also aims to
- provide potential users of simulation with first-hand exposure to methods, to
- display, for library type perusal, the range of literature available in the field, to
- maintain objectivity to the art of simulation, through a non-commercial meeting without obligation to any specific langugage or hardware, and to,
- underwrite, through grants, the advancement of the art of simulation.

Membership is provided as a result of registration at the Annual Symposium. A Board of Directors is elected by the membership, one Director per year for a three year term.

The Symposium is indebted to those corporations and universities whose support, through their representatives, make this totally independent organization capable of serving the Art of Simulation. This year particular recognition is afforded to those organizations whose members served in offices and on committees as shown.

The Annual Simulation Symposium is sponsored by the IEEE Computer Society, the Association for Computing Machinery, the Society for Computer Simulation, and the International Association for Mathematics and Computers in Simulation.

AUTHOR INDEX

TABLE OF CONTENTS

REPLICAS
A NEW CONTINUOUS SYSTEM SIMULATION LANGUAGE

Peter McLaughlin
The Simulation and Modelling Workshop

Abstract. A new continuous system simulation language - REPLICAS, The Rational, Efficient Programming Language for the Implementation of Computerized Analysis and Simulation - is proposed for general engineering, scientific and econometric applications. The use of Gear's integration method coupled with a non-linear quasi-Newton solver relying on Broyden's method results in a reliable and efficient simulation system invoked by a language which requires only that the user define a mathematical model in terms of first-order, ordinary differential equations. Extensions to Gear's method accommodate discontinuities, extreme stiffness and steady-state within a single evaluation procedure.

INTRODUCTION

REPLICAS is a digital computer program and library by which mathematical models defined by a new continuous system simulation language are translated into operational simulation programs. This paper will describe the design and implementation of the REPLICAS system; especially with regard to the relationship between the form of the language and the evaluation techniques utilized. It will be divided into four sections: (1) Design objectives and methodology, (2) Language structure, (3) Numerical techniques and (4) Current status and future plans.

A continuous system simulation language (Reference 1) is intended to facilitate the creation of digital simulations of physical systems and processes which are based on their governing differential equations. These simulations utilize a single free variable, generally taken to be time, and so employ only ordinary differential equations in the definition of their mathematical models. They are utilized in the evaluation of initial value problems relative to the performance of the target system and also form the basis for the analysis of boundary value problems and design optimization studies. Simulation is characterized by the use of mathematical models as replications of physical entities of interest. Experiments or, in more concrete terms, tests utilizing the simulation then replace similiar procedures which would be applied to the real system in order to understand or improve upon it.

The REPLICAS system, in common with most other continuous system simulation languages, consists of a language processor which translates the user's model definition into the source language for a general purpose compiler, and a library of evaluation and modelling utilities from which the finished simulation program is synthesized. REPLICAS is a FORTRAN (Reference 2) based system because it is the most widely available compiler with the necessary characteristics.

DESIGN OBJECTIVES AND METHODOLOGY

The primary objective in the design and implementation of the REPLICAS system has been to provide an effective simulation and analysis tool which can be applied to any physical system or process by users who have no first-hand knowledge of numerical analysis or discrete simulation. This overall objective can be broken down into three distinct requirements. These are:

1. A language structure which allows the user to pose the model directly in the form of first-order ordinary differential equations.
2. A numerical integration algorithm which is unconditionally stable for any model that can be described by the language and has bounded outputs.
3. An evaluation procedure which is insensitive to the form of the model or the size of the discretization intervals and uses this property to minimize execution time while maintaining a given level of accuracy.

To the extent that it achieves these objectives, the design of the REPLICAS system serves to minimize the cost of the digital simulation process. This overall cost is made up of two factors: the labor and computer time required to create the simulation program, and the cost of exercising it. With a self-contained simulation package like REPLICAS, the former is the purchase price; that is the author's costs. The user's costs are made up of time spent learning how to use the system and programming models, and the computer time associated with the execution of the simulation program.

While the use of available computer time should be a significant constraint on the design of any computer software system, the unmistakable trend has been toward "user-friendliness" as a means of improving the productivity of these tools. This principal has been extensively applied to the design of REPLICAS. It is intended that the simple structure of the language will allow rapid training and early productivity for the user as well as having eased the task of programming the language processor and the library. The evaluation methodology has been chosen on the basis of generality and flexibility. It requires no user interaction and will compute the response of any mathematical model that can be described by the language. While it is anticipated that REPLICAS will be competitive in execution time with other languages in typical benchmark environments, its most important characteristic is its simplicity. By maintaining this concept throughout the design and execution of the system, its costs are transferred from the more expensive resource, people, to that which is becoming ever less costly, digital computing.

The REPLICAS language has been designed to be free of unnecessary punctuation and restrictive formats. By generalizing the required modelling functions, only a few statement types and keywords must be mastered in order to fully utilize its capabilities.

The evaluation procedure is fully implicit; that is, it accommodates models specified without regard for the order of their defining statements. The iterative evaluation technique employed has been designed to insure that any set of non-singular model equations will achieve convergence. It is intended to alleviate the effects poor modelling practices in that they are identified by unrealistic responses rather than by convergence failures. Computational efficiency is obtained through the use of an updating method of C. G. Broyden (Reference 3) which corrects the Jacobian matrix for model non-linearities without resort to direct reevaluation. Constraints and other discontinuous functions are accommodated by multiple continuous evaluations, if necessary. It is well suited for both transient and steady-state cases as well as the calculation of design parameters specified by system performance requirements.

The A-stable second order method of C. W. Gear (Reference 4) is the only integration method supported by REPLICAS. A selection algorithm, based on the relative stiffness of the individual state, chooses either the integral or a differential version of the formula in order to enhance the execution rate of the program. Transient response with respect to the free variable is obtained without the need to specify the discretization interval which is automatically controlled in order to achieve a given error condition.

The input/output facilities included in the REPLICAS system adhere to its requirement for simplicity in their design and use. Execution time input is generally limited to the selection of specific disturbances and forcing functions from a menu included with the model definition statements. In addition, all parameters of the user-defined model are addressable through execution time input. Appropriate numerical output is available under control of the user and an interface with the user's graphics software is provided through which tabular data can be transmitted for display.

The size of a REPLICAS program is limited only by the availability of computational resources. It can accommodate any size model and is as effective in large-scale applications as it is on smaller problems.

LANGUAGE STRUCTURE

In this section, the important language elements are described in terms of their application to models consisting of first-order ordinary differential equations. Their relation to the numerical techniques used to implement these functions are discussed in the next Section. Details of the more mundane modelling functions; such as constant parameter assignments, are omitted since they differ little from the standard practice.

The most important aspect of model definition is the specification of the states of the system (the arguments of the model's first-order differential equations). In REPLICAS, this function is provided by STATE, whose arguments serve to identify a state and its derivative with respect to the free variable. Its form is:

state variable = STATE (derivative, initial derivative)

The equations themselves consist of relationships in the form of first-order, ordinary differential equations:

derivative = f(state variables, input parameters)

These model functions can consist of any of the conventional FORTRAN arithmetic expressions or a comprehensive set of the FORTRAN utility and trigonometric functions.

In addition, three other statements are used to describe discontinuous or discretely defined functions. They are related by the fact that each of them employs a common technique which protects the evaluation procedure from the adverse effects of these necessary modelling elements. The form of the statement by which tables of data points are evaluated for given values of the input arguments is as follows:

result = output table name (first argument table name (first argument),
second argument table name (second argument, . . .))

where the values to be used as the basis for the interpolation and extrapolation of the data are specified as arguments of the table name:

table name = value, value, value, . . .

Model definitions which rely on conditional or alternative calculations of certain variables are represented by a statement of the form:

IF (logical expression, THEN, variable = expression,
ELSE, variable = alternative expression)

The logical expression controlling this function follows conventional FORTRAN practice. A generalized limiting function is also provided. It is of the form:

variable = CONSTR (minimum value, maximum value)

This modelling element is generally used to represent real physical constraints but can also serve to realize multiple performance limitations with the introduction of an appropriate system control function. It would use the CONSTR function to select the correct limit.

Communication of information between the model and the evaluation procedure is achieved through the use of literal or character arguments in the subroutine calls to the library functions. The language processor creates these arguments in the course of translating one of the four statement types discussed above. On execution, these constants are replaced by numerical pointers which are used to access the appropriate evaluation arrays. In order to insure that unique pointers exist for each REPLICAS function, the user is required to assign unique names to all variables which result from one of these operations. This restriction greatly simplified the task of designing the language processor and is probably good practice, anyway.

Model definition statements can appear under three different keywords. These define the mathematical model itself, its forcing functions or disturbances, and its design specifications. Forcing functions are defined by the occurrence of keywords whose arguments numerically identify a specific model disturbance. These can be defined by tables of data points read by the function described above. In addition, matrices of input parameter values can be specified.

Finally, the requested number of individual evaluations or, in transient cases, the length of time required to observe the event is specified under the forcing function keywords.

System or process design specifications can require that a particular parameter should assume a value such that a given condition is met. This can be described directly to the evaluation facility by the following statement placed under the design keyword:

parameter == required value - computed value

The double equal sign is used to indicate that the left-hand input parameter is to be varied until the value computed in the model matches the value required by the design specification. This statement, called an implicit equality, can also be used in the definition of the model.

NUMERICAL TECHNIQUES

The numerical integration formula of C. W. Gear are characterized by their implicit definition and their use of only the current value of the derivative. In this general class, the second-order form is given by:

$$X_n = \left(2\dot{X}_n DT + 4X_{n-1} - X_{n-2}\right)/3 \qquad (1)$$

where X are the system states and $\dot{X}$ are their derivatives. The subscripts refer to their discrete values displaced in time by multiples of DT, the discretization interval or time increment.

The selection of Gear's second-order method as the basis for REPLICAS' transient evaluation facility is based on several factors. First of all, its implicit nature is a necessary condition for absolute numerical stability which, in the case of this formula, is assured when it is applied to any model whose response is bounded. If all of the roots of the model's characteristic equation lie on or to the left of the imaginary axis of the complex frequency plane, this integration technique can be shown to provide bounded response, regardless of the size of the time increment employed. At the same time, the use of only the current derivative value minimizes the spurious component of the response associated with the simulation of stiff systems; that is, those whose models possess a wide range of response characteristics. While higher order, multistep Gear formula are also available, they can be unstable when applied to lightly damped systems. Therefore, their use is unsuitable in a simulation system intended for generalized applications. Finally, Gear's second-order formula can be readily expressed in a single-step form, as shown below. This has the effect of simplifying the design of the evaluation facility and minimizing the effect of discontinuities. If an average derivative is defined as:

$$\dot{X}_{av_n} = \left(X_n - X_{n-1}\right)/DT \qquad (2)$$

then, in terms of this variable's past value, the integration formula can be expressed as:

$$X_n = X_{n-1} + \left(2\dot{X}_n + \dot{X}_{av_{n-1}}\right)DT/3 \qquad (3)$$

Due to their implicit definition, Gear's formula require iterative evaluation when applied to the general, non-linear model:

$$\dot{X} = f(X,t) \qquad (4)$$

The quasi-Newton method of C. G. Broyden is utilized by REPLICAS to perform the iterative evaluation of the model under both transient and quiescent or steady-state conditions. In this method, the inverted Jacobian matrix of coefficients used to drive the iterative process is updated on the basis of information obtained directly from the results of previous iterations. Applied to Gear's implicit integration method, the iteration variables are the states of the model. The iteration errors E are, in general, the difference between the iteration variables and the state values computed from the integration formula:

$$E = X - X_n \qquad (5)$$

The successive iterates are obtained from the recursive relationship:

$$X^i = X^{i-1} - E^{i-1} \cdot A^{-1} \qquad (6)$$

where A is the Jacobian matrix $\Delta E/\Delta X$ and the superscripts are iteration indices.

The delta is used to indicate that the quantity is the discrete approximation of a differential. Convergence is attained when all of the elements of E are smaller than a given tolerance value. If A is reevaluated at each iteration, convergence is assured if the model functions are continuously differentiable, non-singular and single-valued. In REPLICAS, this expensive approach is replaced by Broyden's method which does not rely on model evaluations to update the coefficients. Rather, a matrix of changes is computed from the changes to the iteration variables and the errors as well as the elements of the current inverted Jacobian:

$$\Delta A^{-1} = Z^T \cdot \left(\Delta X^i - {A^{-1}}^i \cdot \Delta E^i\right) \tag{7}$$

where: $Z^T = {\Delta X^i}^T \cdot {A^{-1}}^i / \left({\Delta X^i}^T \cdot {A^{-1}}^i \cdot \Delta E^i\right)$

$$\Delta X^i = X^i - X^{i-1}$$

$$\Delta E^i = E^i - E^{i-1}$$

The updated matrix is then given by:

$${A^{-1}}^{i+1} = {A^{-1}}^i + \Delta A^{-1} \tag{8}$$

The Jacobian is evaluated once at the beginning of each simulation case. For the remainder of the case, the Broyden update algorithm corrects the inverted Jacobian for the effects of model non-linearities and changes to the time increment. This process is repeated following each step of the iterative process.

An important factor contributing to the generality of the REPLICAS evaluation facility is the use of a conditioning algorithm which is intended to insure reliable convergence properties, regardless of the form of the model. This technique first normalizes the coefficients of the initial Jacobian by the corresponding values of the iteration variables:

$$\frac{\Delta E}{\Delta \overline{X}} = \frac{\Delta E}{\Delta X} \cdot X = A \cdot X \tag{9}$$

Next, the largest element of each row of the matrix is selected as another