430

MADURAI KAMARAJ UNIVERSITY

(University with Potential for Excellence)

DISTANCE EDUCATION

www.mkudde.org

MBA

SECOND YEAR

MODELLING SIMULATION

430

DIRECTORATE OF DISTANCE EDUCATION

MASTER OF BUSINESS ADMINISTERATOIN

SECOND YEAR

MODELLING AND SIMULATION

MADURAI KAMARAJ UNIVERSITY
MADURAI- 625 021

Lesson Printed by :

Jagan Computers

4/107A, R.R. Compound
Viraganoor, Madurai

Year 2006 - 2007 Copies 3000

(3) MODELLING AND SIMULATION

1. Introduction

Contents of system-Systems – System environment – Starchiest activities-continuous and Discrete System-modelling and types-principles in Modelling.

2. System Studies

Subsystems-Types of systems study-system analysis system design – system postulation.

3. System Simulation

Techniques – Monte carlo method-comparison simulation and analytical methods-Experimental nature simulation types of simulation – lag models-cobweb models progress of a simulation study.

4. Continuous Systems Simulation

Continuous system model – Differential equations Analog methods-Analog and hybrid computers-digital analog simulators – CSSLS – Feedback systems-Interactivate systems – Real time simulations.

5. System Dynamics

Exponential models system Dynamics Diagrams. Won model

6. Discrete System Simulation

Discrete events – Time representation – gathering Statistics –discrete simulation languages model of telephone system study of GPSS.. SIMCRiPT languages.

Model of multi-user and multitasking computer system.

Review of probability concepts-arrival pattern and service times. Analysis of Simulation output.

Text

System Simulation – Geffey Gordon, PHI 2nd., 1987.

Reference

1. System Simulation with Digital computer-Narsingh dep Phi, 1987.

2. GPSS simulation made simple – T.M.O. Donovan, John Willey Sons, 1979.

1. INTRODUCTION

In this unit, we are going to divide the whole unit into two parts

- The first part is fully deals with the models that are basic to any simulation,
- The second part discuss about the topic of organizing a system study

1.1 CONTENTS OF SYSTEM

The term system is used in such a wide variety of ways that it is difficult to produce a definition broad enough to cover the many uses and, at the same time, concise enough to serve a useful purpose. We begin, therefore, with a simple definition of a system and expand upon it by introducing some of the terms that commonly used when discussing systems.

A system is defined as an aggregation or assemblage of objects joined in some regular interaction or interdependence.

The another way to define the system is a collection of Inter-related objects.

While this definition is broad enough to include static systems, the principal interest will be in dynamic systems where the interactions cause changes over time.

As an example of a conceptually simple system, consider an aircraft flying under the control of an autopilot (see fig. 1-1). A gyroscope in the autopilot detects the difference between the actual heading and the desired heading. It sends a signal to move the control surface movement, the airframe steers toward the desired heading

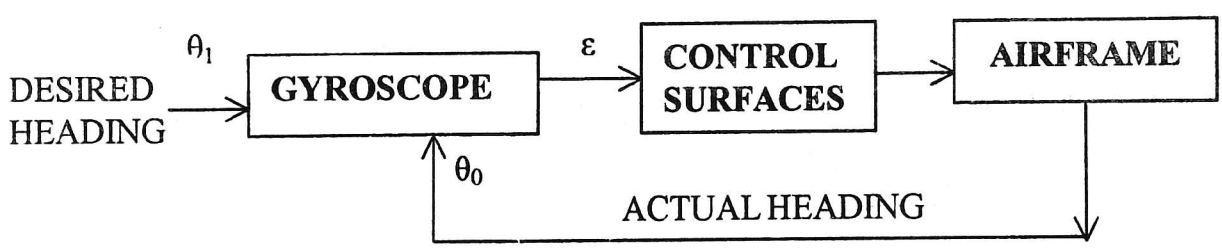


Figure 1-1. An aircraft under autopilot control.

As a second example, consider a factory that makes and assembles parts into a product (see Fig.1-2)

.Two major components of the system are

- The fabrication department making the parts
- The assembly department producing the products.

The components of the systems are

- A purchasing department maintains a supply of raw materials
- A shipping department Dispatches the finished products.
- A production control department receives orders and assigns work to the other department, so that the finished products are sent to the customer. This department will coordinate the activities of all the department

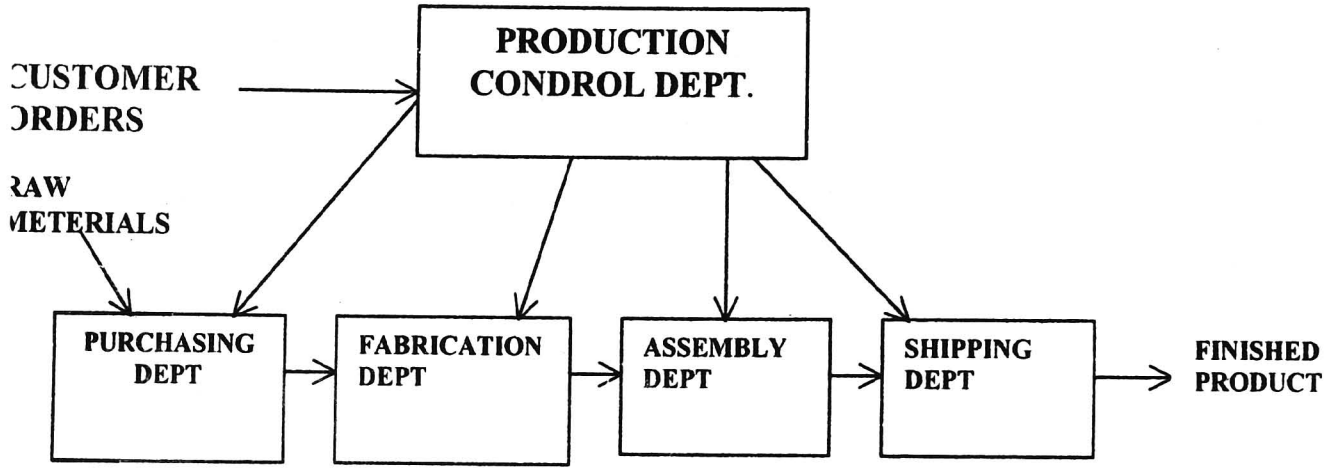


Fig 1.2 A factory system

In looking at these systems, we see that there are certain distinct objects, each of which possesses properties of interest. There are also certain interactions occurring in the system that cause changes in the systems.

Entity:

The term entity will be used to denote an object of interest in a system

Attribute:

The term attribute will denote a property of an entity. There can be, of course many attributes to a given entity.

Activity:

Any process that causes changes in the system will be called as an activity.

State of the System:

The term state of the system means the description of all the entities, attributes, and activities, as they exist at one point in time. The progress of the system is studied by following the changes in the state of the system.

In the description of the aircraft system

- The entities of the system are the airframe, the control surfaces, and the gyroscope.
- Their attributes are such factors as speed, control surface angle, and gyroscope setting.
- The activities are the driving of the control surfaces and the response of the airframe to the control surface movements.

In the factory system,

- The entities are the departments, orders, parts, and products.
- The activities are the manufacturing processes of the departments.
- Attributes are such factors as the quantities for each order, type of part, or number of machines in a department.

1.2 SYSTEMS

Figure 1-3 lists examples of what might be considered entities, attributes, and activities for a number of other systems. If we consider the movement of cars as a traffic system, the individual cars are regarded as entities, each having as attributes its speed and distance traveled. Among the activities is the driving of a car. In the case of bank system, the customers of the bank are entities with the balances of their accounts and their credit statuses as attributes. A typical activity would be the action of making a deposit. Other examples are shown in Fig. 1-3.

SYSTEM	ENTITES	ATTRIBUTES	ACTIVITIES
TRAFFIC	CARS	SPEED DISTANCE	DRIVING
BANK	CUSTOMERS	BALANCE CREDIT STATUS	DEPOSITING
COMMUNICATIONS	MESSAGES	LENGTH PRIORITY	TRANSMITING
SUPERMARKET	CUSTOMERS	SHOPPING LIST	CHECKING-OUT

Fig 1.3 Examples of systems

The figure does not show a complete list of all entities, attributes, and activities for the systems. In fact, a complete list cannot be made without knowing the purpose of the system description. Depending upon that purpose, various aspects of the system will be interest and will determine what needs to be identified.

1.3 SYSTEM ENVIRONMENT

A system is often by changes occurring outside the system. Some system activities may also produce changes that do not react on the system. Such changes occurring outside the system are said to occur in the *system environment*. An important step in modeling systems is to decide upon the boundary between the system and its environment. The decision may depend upon the purpose of the study.

In the case of the factory system, for example, the factors controlling the arrival of orders may be considered to be outside the influence of the factory and therefore part of the environment. However, if the effect of supply on demand is to be considered, there will be a relationship between factor output and arrival of orders, and this relationship must be considered an activity of the system. Similarly, in the case of a bank system, there may be a limit on the maximum interest rate that can be paid. For the study of a single bank, this would be regarded as a constraint imposed by the environment. In a study of the effects of monetary laws on the banking industry, however, the setting of the limit would be an activity of the system.

The term *endogenous* is used to describe activities occurring within the system and the term *exogenous* is used to describe activities in the environment that affect the system. A system for which there is no exogenous activity is said to be a closed system in contrast to an open system, which have exogenous activities.

1.4 STOCHASTIC ACTIVITIES

One other distinction that needs to be drawn between activities depends upon the manner in which they can be described. Where the outcome of an activity can be described completely in terms of its input, the activity is said to be *deterministic*. Where the effects of the activity vary randomly over various possible outcomes the activity is said to be *stochastic*.

The randomness of a stochastic activity would seem to imply that the activity is part of the system environment since the exact outcome at time is not known. However, the random output can often be measured and described in the form of a probability distribution .If however, the occurrence of the activity is random, it will taken for a machining operation may need to be described by a probability distribution but machining would be considered to be an endogenous activity. On the other hand, there may be power failures at random intervals of time. They would be result of an exogenous activity.

If an activity is truly stochastic, there is no known explanation for its randomness. Sometimes, however, when it requires too much detail or is just too much trouble to describe an activity fully, the activity is represented as stochastic. For example, in modeling elevator service in a building, the re-entry of people into the elevator, after they have been taken to a floor, could to be connected with their having left the elevator, by assigning the time they stay on the floor. In most models, however, leaving and re-entry would be treated as separate stochastic activities, connected only by the fact that the mean rates at which they transfer people are equal.

Assembling the data for a model will often involve an element of uncertainty that arises from sampling or experimental error. A value for some attribute of a model, which is known to be fixed, must be selected from a number of recorded values that contain random errors. Deciding on the best estimate is a statistical exercise. Usually, an arithmetic average will be considered sufficiently accurate.

1.5 CONTINUOUS AND DISCRETE SYSTEMS

The aircraft and factory systems used as examples in Sec.1-1 respond to environmental changes in different ways. The movement of the aircraft occurs smoothly, whereas the changes in the factory occur discontinuously. The ordering of raw materials or the completion of a product, for example, occurs at specific points in time.

Systems such as the aircraft, in which the changes or predominantly smooth, are called *continuous systems*. Systems like the factory, in which changes are predominantly discontinuous, will be called *discrete systems*. Few systems are wholly continuous or discrete. The aircraft, for example, may make discrete adjustments to its trim as altitude changes, while, in the factory example, machining proceeds continuously, even though the start and finish of a job are discrete changes. However, in most systems one type of change predominates, so that systems can usually be classified as being continuous or discrete.

The complete aircraft system might even be regarded as a discrete system. If the purpose of studying the aircraft were to follow its progress along its scheduled route, with a view, perhaps, to studying air traffic problems, there would be no point in following precisely how the aircraft turns. It would be sufficiently accurate to treat changes of heading at scheduled turning points as being made instantaneously, and so regard the system as being discrete.

In addition, in the factory system, if the number of parts is sufficiently large, there may be no point in treating the number as a discrete variable. Instead, the number of parts might be represented by a continuous variable with the machining activity controlling the rate at which parts flow from one state to another. This is, in fact, the approach of a modeling technique called System Dynamics, which will be discussed in chap.5.

There are also systems that are intrinsically continuous but information about them is only available at discrete points in time. These are called sampled-data systems. The study of such systems includes the problem of determining the effects of the discrete sampling, especially when the intention is to control the system on the basis of information gathered by the sampling.

This ambiguity in how a system might be represented illustrates an important point. The description of a system, rather than the nature of the system itself, determines what type of model will be used. A distinction needs to be made because, as will be discussed later, the general programming methods used to simulate continuous and discrete models differ. However, no specific rules can be given as to how a particular system is to be represented .The purpose of the model, coupled with the general principle that a model should not be more complicated than is needed, will determine the level of detail and the accuracy with which a model needs to be developed. Weighing these factors and drawing on the experience of knowledgeable people will decide the type of model that is needed.

1.6 Modelling and Types of models

MODELLING

To study a system, it is sometimes possible to experiment with the system itself. The objective of many system studies, however, is to predict how a system will perform before it is built. Clearly, it is not feasible to experiment with the system while it is in this hypothetical form. An alternative that is sometimes used is to construct a number of prototypes and test them, but this can be very expensive and time-consuming. Even with an existing system, it is likely to be impossible or impractical to experiment with the actual system. For example, it is not feasible to study economic systems by arbitrarily changing the supply and demand of goods. Consequently, system studies are generally conducted with a model of the system. For the purpose of most studies, it is not necessary to consider all the details of a system; so a model is not only a substitute for a system, it is also a simplification of the system.

We define a model as the body of information about a system gathered for the purpose of studying the system. Since the purpose of the study will determine the nature of the information that is gathered, there is no unique model of a system. Different models of the system will be produced by different analysts interested in different aspects of the system or by the same analyst as his understanding of the system changes.

The task of deriving a model of a system may be divided broadly into two subtasks: establishing the model structure and supplying the data. Establishing the structure determines the system boundary and identifies the entities, attributes, and activities of the system. The data provide the values the attributes can have and define the relationships involved in the activities. The two jobs of creating a structure and providing the data are defined as parts of one task rather than as two separate tasks, because they are usually so intimately related that neither can be done without the other. Assumptions about the system direct the gathering of data, and analysis of the data confirms or refutes the assumptions. Quite often, the data gathered would disclose an unsuspected relationship that changes the model structure.

To illustrate this process, consider the following description of a supermarket.

Shoppers needing *several items* of shopping arrive at a supermarket. They get a basket; if one is available, carry out their shopping, and then queue to checkout at one of the *several counters*.

After checking-out, they return the basket and leave.

Certain words have been italicized because they are considered to be key words that point out some feature of the system that must be reflected in the model. Essentially the same description is rewritten in Fig. 1-4 to identify the entities,

ENTITY	ATTRIBUTE	ACTIVITY
SHOPPER	NO OF ITEMS	ARRIVE GET
BASKET	AVAILABILITY	SHOP QUEUE CHECK-OUT
COUNTER	NUMBER OCCUPANCY	RETURN LEAVE

Figure 1.4 Elements of a supermarket model

Attributes, and activities. Notice that the concept of a supermarket as a whole does not appear as an entity. It defines the system boundary and therefore distinguishes between the system and its environment. The arrival of customers in this description of the system will be regarded as an exogenous activity affecting the system from the environment. If, in contrast, the study objectives include analyzing the effect of car parking facilities on supermarket business, the boundary of the system would need to include the parking lot. The arrival of a customer in the supermarket depends upon finding a parking space, which can depend upon the departure of customers. Customer arrivals in the supermarket then become an endogenous activity; the arrival of cars becomes an exogenous activity.

Other decisions about the system study objectives are implied in the model. The number of items of shopping is represented as an attribute of the shopper, but no distinction has been made about the type of items. Secondly, no provision has been made in the system model for the effects of congestion on shopping time. If these decisions are not in keeping with the study objectives, another from of model must be used. In the first case, where type of item is to be distinguished, it is necessary to define several attributes for each customer, one for each type of item to be purchased. In the second case, where allowance for congestion must be made, two approaches could be taken. It may be necessary to introduce new entities representing the various sections of the supermarket and establish as attributes the number of customers they can serve simultaneously. Alternatively, the activity of shopping could be represented by a function in which shopping time depends upon the number of shoppers in the supermarket.

It is not suggested that Fig. 1-4 represents a formal process by which a transition can be made from a verbal description of a system to the structure of a model. It merely illustrates the process involved in forming a model.

TYPES OF MODELS

Models used in system studies have been classified in many ways. The classifications that will be used here are illustrated in Fig. 1-5. Models will first be separated into physical models or mathematical models.

Physical model are based on some analogy between such systems as mechanical and electrical or electrical and hydraulic. In a physical model of a system, the system attributes are represented by such measurements as a voltage or the position of a shaft. The system activities are reflected in the physical laws that drive the model. For example, the rate at which the shaft of a direct current motor turns depends upon the voltage applied to the motor. If the applied voltage is used to represent the velocity of a vehicle, then the number of revolutions of the shaft is a measure of the distance the vehicle has traveled; the higher the voltage, or velocity, the greater is the buildup of revolutions, or distance covered, in a given time.

Mathematical models, of course, use symbolic notation and mathematical equations to represent a system. The system attributes are represented by variables and the activities are represented by mathematical functions that interrelate the variables.

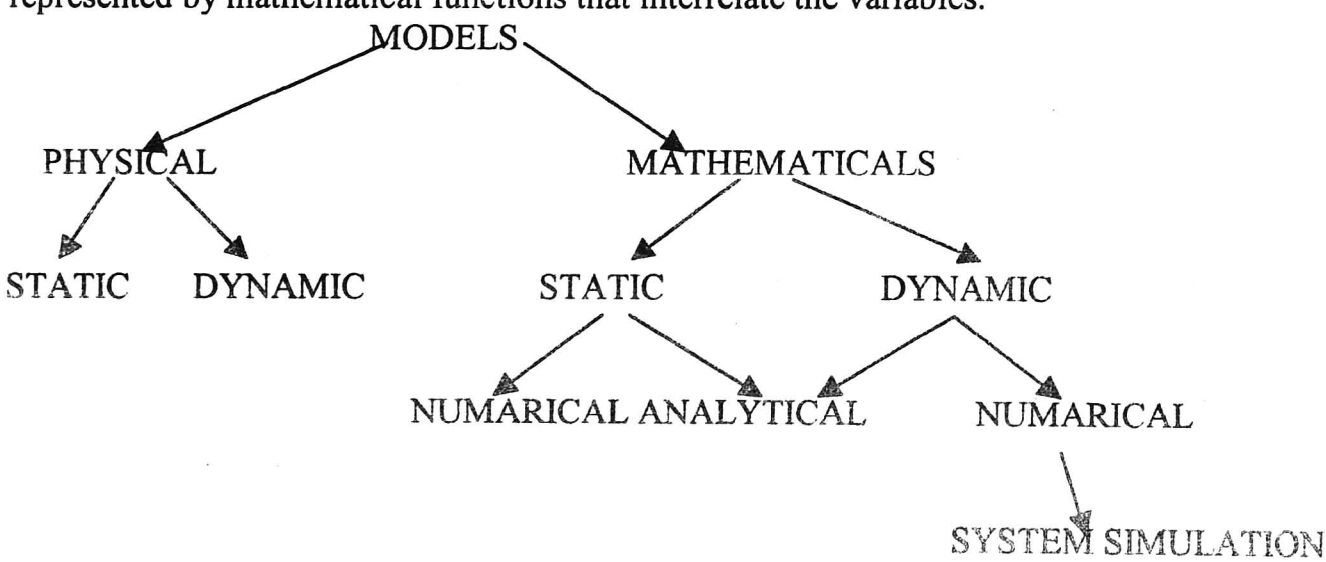


Figure 1.5 Types of models

A second distinction will be between static models and dynamic models. Static models can only show the values that system attributes take when the system is in balance. Dynamic models, on the other hand, follow the changes over time that result from the system activities.

In the case of mathematical models, a third distinction is the technique by which the model is "solved ", that is actual values are assigned to system attributes. A distinction is made between analytical and numerical methods. Applying analytical techniques means using the deductive reasoning of mathematical theory to solve a model. In practice, only certain forms of equations can be solved. Using analytical techniques, therefore, is a matter of finding the model that can be solved and best fits the system being studied. For example, linear differential equations can be solved. Knowing this, an engineer who restricts the description of a system the form will derive a model that can be solved analytically.

Numerical methods involve applying computational procedures to solve equations. To be strictly accurate, any assignment of numerical values that uses mathematical tables involves numerical methods, since tables are derived numerically. The distinction being drawn here is that analytical methods produce solutions in tractable form, meaning a form where values can be assigned form available tables. Making use of an analytical solution may, in fact, require a considerable amount of computation. For example, the solution may be derived in the form of a complicated integral, which then needs to be expanded as a power series for evaluation. However, mathematical theory for making such expansions exists, and, in principle, any degree of accuracy in the solution is obtainable if sufficient effort is expended.

As will be discussed more fully in chap. 3, system simulation is considered to be a numerical computation technique used in conjunction with dynamic mathematical models. Simulation models, therefore, are shown under that heading in Fig. 1-5.

Yet another distinction by which models are often classified is between deterministic and stochastic: the latter term meaning that there are random processes in the system. As will be discussed in chap. 14, the introduction of stochastic processes in a simulation model complicates the task of interpreting results, and it increases the amount of work to be done. It does not, however, change the basic technique by which simulation is applied, so this distinction has not been made in Fig. 1-5.

STATIC PHYSICAL

The best-known examples of physical models are scale models. In shipbuilding, making a scale model provides a simple way of determining the exact measurements of the plates covering the hull, rather than having to produce drawing of complicated, three- dimensional shapes. Scientists have used models in which spheres represent atoms and rods or specially shaped sheets of metal connect the shapers to represent atomic bonds. A model of this nature played an important role in the deciphering of the DNA molecule, work that was the subject of a Nobel Prize award. These models are static physical models. They are sometimes said to be *iconic* models, a term meaning "look-alike".

Scale models are also used in wind tunnels and water tanks in the course of designing aircraft and ships. Although air is blown over the model, or the model is pulled through the water, these are static physical models because the measurements that are taken represent attributes of the system being studied under one set of equilibrium conditions. In this case, the measurements do not translate directly into system attributes values. Well-known laws of similitude are used to convert measurements on the scale model to the values that would occur in the real system.

Sometimes, a static physical model is used as a means of solving equations with particular boundary conditions. There are many examples in the filed of mathematical physics where the same equations apply to different physical phenomena. For example, the flow of heat and the distribution of electric charge through space can be related by common equations. In general, these equations can only be solved for simple-shaped bodies. In practice, solutions are needed for specific, complicated shapes. The distribution of heat in a body, and measuring the charge in the space when the surface of the space has been electrified in a manner that reflects the way heat will be injected in to the body.

DYNAMIC PHYSICAL MODEL

Dynamic physical models rely upon an analogy between the system being studies and some other system of a different nature, the analogy usually depending upon an underlying similarity in the forces governing the behavior of the systems. To illustrate this type of physical model, consider the two systems shown in Fig. 1-6. Figure 1-6(a) represents a mass that is subject to an applied force F (t) varying with time, a spring whose force is proportional to its extension or contraction, and a shock absorber that exerts a damping force proportional to the velocity of the mass. The system might, for example, represent the suspension of an automobile wheel when the automobile body is assumed to be immobile in a vertical direction. It can be shown that the motion of the system is described by the following differential equation:

$$M\ddot{x} + D\dot{x} + Kx = K F(t)$$

Where x is the distance moved,
M is the mass,
K is the stiffness of the spring,
D is the damping factor of the shock absorber.

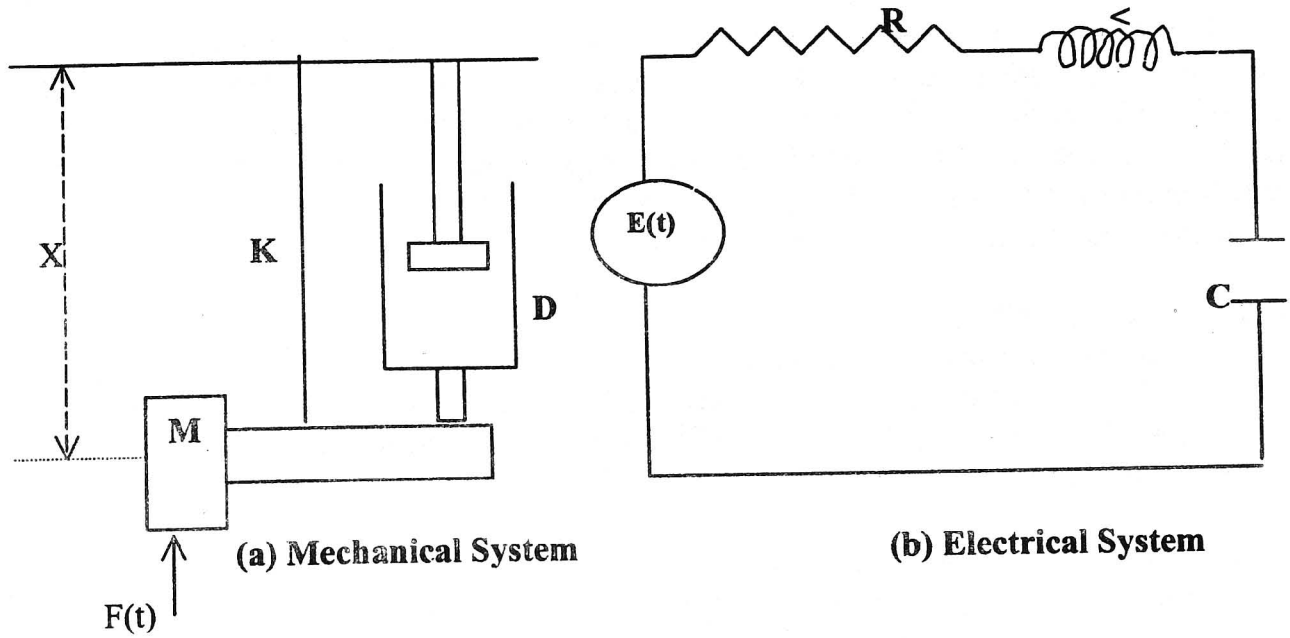


Figure 1.6 Analogy between mechanical and electrical systems

Figure 1-6 (b) represents an electrical circuit with an inductance L, a resistance R, and a capacitance C, connected in series with a voltage source that varies in time according to the function E (t). If q is the charge on the capacitance, it can be shown that the behavior of the circuit is governed by the following differential equation:

$$\longrightarrow \; L\ddot{q} + R\dot{q} + \frac{q}{C} = \frac{E(t)}{C}$$

Inspection of these two equations shows that they have exactly the same form and that the following equivalences occur between the quantities in the two systems:

Displacement	x	Charge	q
Velocity		Current	I (=q')
Force	F	Voltage	E
Mass	M	Inductance	L
Damping factor	D	Resistance	R
Spring stiffness	K	1/Capacitance	1/C

The mechanical system and the electrical system are analogs of each other, and the performance of either can be studied with the other. In practice, it is simpler to modify the electrical system than to change the mechanical system, so it is more likely that the electrical system will have been build to study the mechanical system. If, for example, a car wheel is considered to bounce too much with a particular suspension system, the electrical model will demonstrate this fact by showing that the charge (and, therefore, the voltage) on the condenser oscillates excessively. To predict what effect a charge in the shock absorber or spring will have on the performance of the car, it is only necessary to change the values of the resistance or condenser in the electrical circuit and observe the effect on the way the voltage varies.

If, in fact, the mechanical system were as simple as illustrated, it could be studied by solving the mathematical equation derived in establishing the analogy. However, effects can easily be introduced that would make the mathematical equation difficult to solve. For example, if the motion of the wheel is limited by physical stops, a non-linear equation that is difficult to solve will be needed to describe the system. It is easy to model the effect electrically by placing limits on the voltage that can exist on the capacitance.

STATIC MATHEMATICAL MODELS

A static model gives the relationships between the system attributes when the system is in equilibrium. If the point of equilibrium is changed by altering any of the attributes values, the model enables the new values for all the attributes to be derived but does not show the way in which they changed to their new values.

For example, in marketing a commodity there is a balance between the supply and demand for the commodity. Both factors depend upon price: a simple *market mode!* Will show what is the price at which the balance occurs.

Demand for the commodity will be low when the price is high, and it will increase as the price drops. The relationship between demand, denoted by Q, and price, denoted P, might be represented by the straight line marketed "Demand" in Fig. 1-7. On the other hand, the supply can be expected to increase as the price increases, because the suppliers see on opportunity for more revenue. Suppose supply, denoted by S, is plotted against price, and the relationship is the straight line marked "supply" in Fig. 1-7.If conditions remain stable, the price will settle to the point at which the two lines cross, because that is where the supply equals the demand.

Since the relationships have been assumed linear, the complete market model can be written mathematically as follows:

$$Q = Q - bp$$
$$S = c + dP$$
$$S = Q$$
$$Q \quad 1/p$$
$$S \quad P$$
$$S - Q = S$$
$$a - bp = C + dp$$
$$a - c = bp + dp$$
$$a - c = P(b+d)$$
$$p = \frac{(a-c)}{(b+d)}$$

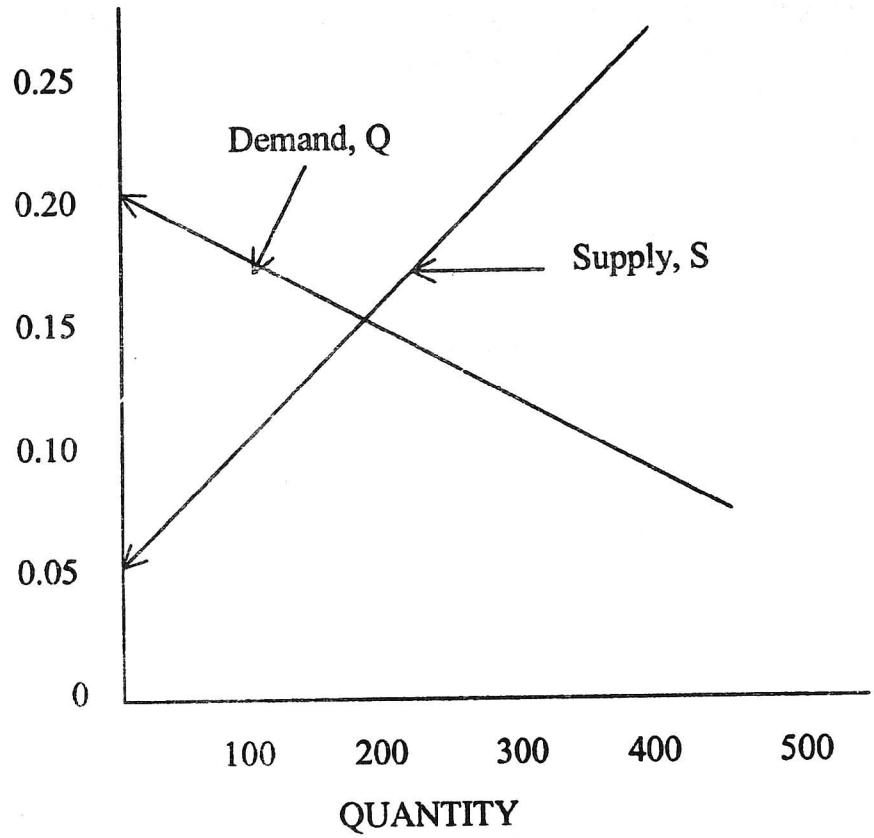


Figure 1.7 Linear market model

The last equation states the condition for the market condition for the market to be cleared; it says supply equals demand and, so, determines the price to which the market will settle.

For the model to correspond to normal market conditions in which demand goes down and supply increases as price goes up the coefficients b and d need to be positive numbers. For realistic, positive results, the coefficient a must also be positive. Figure 1-7 has been plotted for the following values of the coefficients:

A = 600
B = 3,000
C = -100
D = 2,000

The fact that linear relationships have been assumed allows the model to be solved analytically. The equilibrium market price, in fact, is given by the following expression.

$$P = (a - c) / (b + d)$$

With the chosen values, the equilibrium price is 0.14, which corresponds to a supply of 180.

More usually, the demand will be a curve that slopes downwards, and the supply by a curve that slops upwards, as illustrated in Fig. 1-8. It may not than be possible to express the relationships by equations that can be solved. Some numeric method is than needed to solve the equations. Drawing the curves to scale and determining graphically where they intersect is one such method.

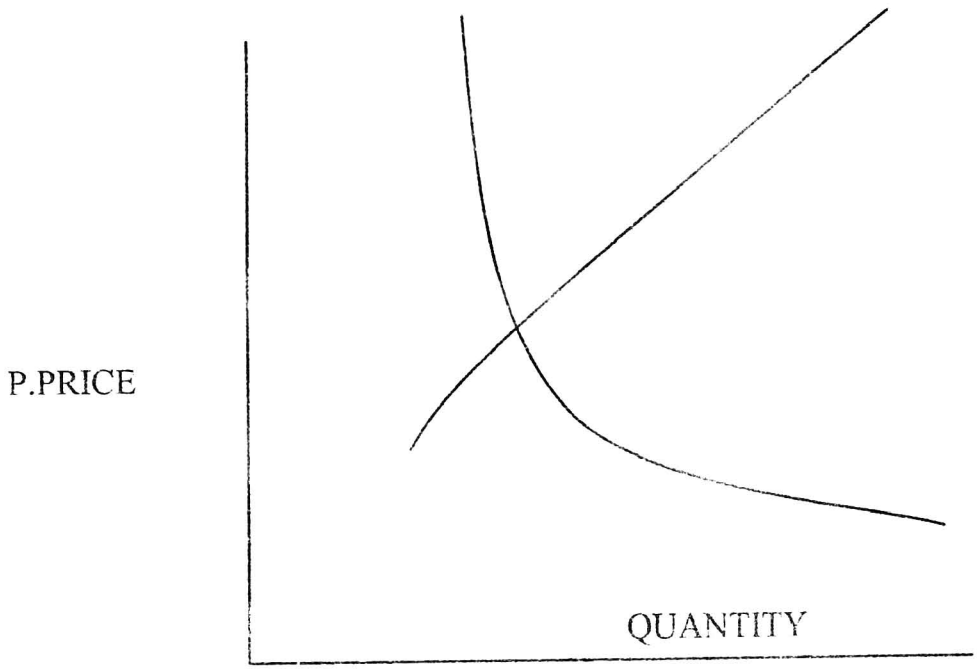


Figure 1.8 Non-linear market models

In practice, it is difficult to get precise values for the coefficients of the model. Observations over an extended period of time, however, will establish the slopes (that is, the values of b and d) in the neighborhood of the equilibrium point, and, of course, actual experience will have established equilibrium prices under various conditions. The values depend upon economic factors, so the observations will usually attempt to correlate the value with the economy, allowing the model to be used as a means of forecasting changes in market conditions for [illegible] economic changes.

Dynamic mathematical model

A dynamic mathematical model allows the change of system attributes to be derived as a function of time. The derivation may be made with an analytical solution or with a numerical computation, depending upon the complexity of the model. The equation that was derived to describe the behavior of a car wheel is an example of a dynamic mathematical model; in this case, an equation that can be solved analytically. It is customary to write the equation in the form

$$\ddot{X} + 2\zeta\omega\dot{x} + \omega^2 x = \omega^2 F(t)$$

Where $2\zeta\omega = D / M$ and $\omega^2 = K / M$.

Expressed in this form, solutions can be given in terms of the variable wt. X varies in response to a steady force applied at time t = 0 as would occur, for instance, if a load were suddenly placed on the automobile. Solutions are shown for several values of ζ, and it can be seen that when ζ is less than 1, motion is oscillatory.

The factor ζ is called the damping ratio and, when the motion is oscillatory, the frequency of oscillation is determined from the formula,

$$\omega = 2\pi f$$

Where f is the number of cycles per second.

Suppose a case is selected as representing a satisfactory frequency and damping. The relationships given above between ζ, ω, M, K, and D show how to select the spring and shock absorber to get that type of motion. For example, the condition for the motion to occur without oscillation requires that $\zeta \geq 1$. It can be deduced from the definition of and w that the condition requires that D^2 $4MK$.

1.7 PRINCIPLES OF MODELING

It is not possible to provide rules by which mathematical models are built, but a number of guiding principles can be stated. They do not describe distinct steps carried out in building a model. They describe different viewpoints from which to judge the information to be included in the model.

The description of the system should be organized in a series of blocks is to simplify the specification of the interactions within the system. Each block describes a part of the system that depends upon a few, preferably one, input variables and results in a few output variables The system as a whole can then be described in terms of the interconnections between the blocks. Correspondingly, the system can be represented graphically as a simple block diagram.

The description of a factory given in Fig. 1-2 is a typical example of a block diagram. Each department of the factory has been treated as a separate block, with the inputs and outputs being the work passed from department to department. The fact that the departments might occupy the same floor space and might use the same personnel or the same machines has been ignored.

The model should only include those aspects of the system that are relevant to the study objectives. As an example, if the factory system that are study aims to compare the effects of difficult operating rules on efficiency, it is not relevant to consider the hiring of employees as an activity. While irrelevant information in the model may not do any harm, it should be excluded because it increases the complexity of the model and causes more work in solving the model.

The accuracy of the information gathered for the model should be considered. In the aircraft system, for example, the accuracy with which the movement of the aircraft is described depends upon the representation of the airframe. IT may be suffice to regard the airframe as a rigid body and derive a very simple relationship between control surface movement and aircraft heading, or may be necessary to recognize the flexibility of the airframe and make allowance for

vibration in the structure. An engineer responsible for estimating the fuel consumption may be satisfied with simple representation. Another engineer, responsible for considering the comfort of the passengers, needs to consider vibrations and will want the detailed description of the airframe.

A further factor to be considered is the extent to which the number of individual entities can be grouped together into larger entities. The general manager of the factory may be satisfied with description that has been given. The production control manager, however, will want to consider the shops of the departments as individual entities.

In some studies, it may be necessary to construct artificial entities through the process of aggregation. For example, an economic or social study will usually treat a population as a number of social classes and conduct a study as though each social class were a distinct entity.

Similar considerations of aggregation should be given to the representation of activities. For example, in studying a missile defense system, it may not be necessary to include the details of computing a missile trajectory for each firing. It may be sufficient to represent the outcome of many firings by a probability function.

2. SYSTEM STUDIES

2.1 SUBSYSTEMS

We have given a simple definition of a system as a set of interacting objects. The objects might be considered the basic entities of the system, but, usually, the description of a system can be made at many levels of detail. It is customary to description of a system as consisting of interacting subsystems. Any subsystem might, itself, be considered a system consisting of subsystems at a still lower level of detail, and so on. A system study must begin by deciding on the level of subsystem detail to be used.

The block-building principle, mentioned in sec. 1-11, helps organize a system description by isolating subsystems and identifying their inputs and outputs. Each subsystem at a given level of detail is described as a block, giving relationships between the inputs and outputs. The relationships should be such that they are sufficient to determine the outputs from the inputs when the subsystem stands by itself; that is to say, there should be no need to use an endogenous variable within the block. The term "block box," borrowed from the engineering field, is often used to describe an element of this nature, which gives an output in response to an input without 0any need, or ability, to know how the transformation is made.

Each subsystem has its own inputs and outputs and, standing by itself, is response can be derived from the relationship that defines the subsystem. The interactions that occur in a system arise from the fact that the outputs of some subsystems become the inputs of others; they become endogenous variables of the system.

In the same way that a system breaks down into subsystems, so also a model of a system breaks into sub-models. When describing systems in terms of blocks, the terms block, subsystem, or sub-model tend to be used interchangeably, as will occur in the subsequent discussion. In fact, when concentrating on a particular part of a system or its model, we will often drop the prefix "sub," and talk about a part of a system or model as being itself, a system or model. The context should make it quite clear what is intended.

2.2 TYPES OF SYSTEMS STUDY

Having developed a model, there are various ways we can use it to study a system. Generally, system studies are of three types: system analysis, system design, and what will be called system postulation.

Many studies, in fact, combine two or three of these aspects or alternate between them as the study proceeds. The term system engineering is frequently used to describe system studies where a combination of analysis and design is aimed at understanding, first, how an existing system works and then preparing system modifications to change the system behavior.

System analysis aims to understand how an existing system or a proposed system operates. The ideal situation would be that the investigator is able to experiment with system itself. What is actually done is to construct a model of the system and investigate the behavior of the model. The results obtained are interpreted in terms of system performance.

In system design studies, the object is to produce a system that meets some specifications. Certain system parameters or components can be selected or planned by the designer, and, conceptually, he chooses a particular combination of components to construct a system. The proposed system is modeled and its performance predicted from knowledge of the model s behavior. If the predicted performance compares favorably with the desired performance, the design is accepted. Otherwise, the system is redesigned and the process repeated.

System postulation is characteristic of the way models are employed in social, economic, political, and medical studies, where the behavior of the system is known but the processes that produce they behavior or not. Hypotheses are made on a likely set of entities and activities that can explain the behavior. The study compares the response of the model based on these hypotheses with known behavior. A reasonably good match naturally leads to the assumption that the structure of the model bears a resemblance to the actual system, and allows is a system structure to be postulated. . Very likely, the behavior of the model gives a better understanding of the system, possibly helping to formulate a refined set of hypotheses.

In the resemblance this chapter we will demonstrate these three general approaches to system studies. In doing so, we will be using some simple, static models which can, in fact, be solved analytically. This will allow us to concentrate on the study technique. In each case, a realistic study would require a more complex, dynamic model. The precise, simple relationships derived from many simulation runs.

2.3 SYSTEM ANALYSIS

To demonstrate a system analysis study, we will consider a small part of the corporate model developed earlier. Looking at just the financial and be production models, and ignoring the influence of the economics that might affect results, gives the simple system illustrated in Fig. 2-1. The system involves four variabies denoted by the following symbols:

K	Capital investment
K	Capital investment
L	Labor
M	Machinery
S	Supply

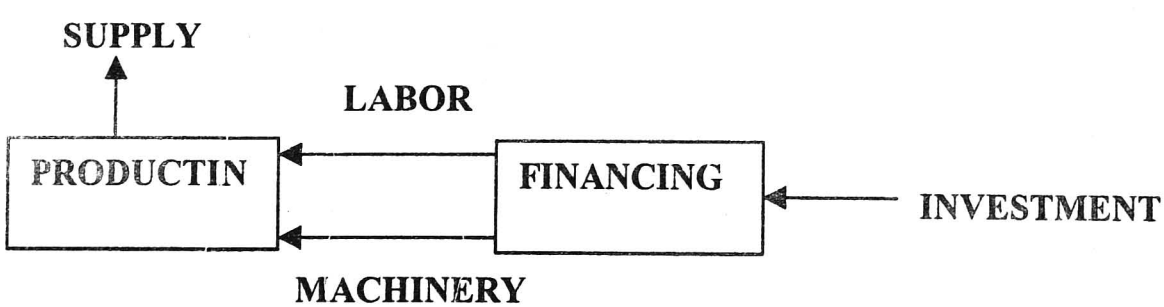


FIGURE 2-1. PORTION OF CORPORATE MODEL.

With regarded to production, economists have found that the output of an enterprise is often related to the investment in labor and machinery by an equation of the general form

$$S = f L^{a1} M^{a2}$$

Where f is a constant. Models of this form are known as Cobb- Douglas models. They imply that a given percentage increase in output. For simplicity, we will assume here that the exponents, a_1 and a_2, both equal to one. The production function then takes the form:

$$S = f LM$$

The financial model assumes that there are possible substitutions between labor and machinery. In the present case, we will assume a linear relationship, as follows:

$$K = eL + M$$

The coefficient e is a constant. It implies that one unit of investment in labor is equivalent to investing e units in machinery.

Note that both equations depend upon L and M. The financial model shows how a given amount of investment, K can be divided between L and M. The production model shows how the output, or supply S, depends upon the amount invested in L and M. It should be possible to find an assignment of L and M that maximizes supply for given investment.

The simple forms assumed for the equations permit differential calculus method to be applied to show that there is indeed, a maximum; but the result can also be demonstrated graphically. The horizontal axis plots M, and the vertical axis plot L. The straight lines of may show the relationship between L and M for a given value of K, when it is assumed that the value of the coefficient e is 0.75. One line is for $K = 50$, and the other is for $K = 100$. As can be seen, increasing the value of K moves the line farther from the origin.

The relationship between L and M for a constant value of S is not linear. It gives rise to the curves of Fig. 2-7, which are for the cases of S = 50 and 200 when f has the value 0.1. Again, larger values move the curve farther from the origin. The curve for $S = 50$ crosses the line for $K = 50$. This means that there are some combinations of value for L and M which produce a supply of 50 but add up to a K of less than 50. On the other hand, the curve for S = 200 does not cross the line for $K = 50$, meaning that it is impossible to produce a supply of 200 with an investment of only 50. The most that can be produced with $K = 50$ is given by the curve which just touches, or is tangent to, the line for $K = 50$. It happens to be the curve for S = 83.3. The maximum occurs when L 33.3 and M =25.

The model that has been analyzed is a small part of the full corporate model. An analysis of the full model would attempt to optimize some broader measure of the system performance, for instance, the rate of return on investment, which is the ratio of the profit to total capital investment. Such an analysis would of course, mean extending the set of equations to cover the other segments of the full model. It would not necessarily invalidate the analysis of the sub-model that has just been carried out.

What we have done is typical of many system analysis studies: we have sub-optimized a portion of a complete model. Sub-optimization is a convenient way of breaking the optimization of a total system model in to number of simpler (approximate) steps. It is assumed that when the system as a whole is operating at its maximum, its various parts are also operating at their maximum. The system maximum can therefore, be approached by maximizing the parts independently.

This is not usually true in the strict mathematical sense. In the present case, for example, it does not follow that the best rate of return, taking into account all parts of the system, is achieved when the supply is the maximum that can be achieved with the given investment. Realistically, it is unlikely that, when the rate of return is maximized, the supply is much different form that achieved with sub optimization; however, the uncertainly is the price paid for simplifying the solution.

Corporate models, particularly when used for the type of study just carried out, often are static models, like the one we have used. The models, however, will often be dynamic. When growth is being considered, and assumptions about such factors as competition and the state of the economy are included in the study, dynamic models are needed. Simulation methods are then needed for deriving the results on which the analysis is based.

2.4 SYSTEM DESIGN

To illustrate a system design problem, we consider the problem of designing a small part of an on-line computer system, that is, a computer that responds immediately to messages it receives. The function of the system is described by the flowchart of Fig. 2-8. Messages are being received over a communication channel at the rate of M messages a second. On the average, there are m characters in a message. They are received in a buffer that can hold a maximum of b characters. A fraction k of the messages needs replies, which have an average of r characters. The same buffer is used for both receiving and sending messages.

Processing a message when it has been received takes about 2,000 instructions. Preparing a reply requires a program that uses about 10,000 instructions. The finite size of the buffer means that messages and replies must be broken into section causes an interruption of the processing, requiring the execution of 1,000 instructions to transfer data either in or out of the computer.

Three computers are being considered as possible system components: a slow, a medium, and a fast computer, which have instruction execution rates of 25,000, 50,000, and 100,000 instructions a second. In addition four buffer sizes are being considered: there could be a one-, two-, five-, or ten-character buffer. The design problem is to determine which of the possible twelve combinations of computer speed and buffer size will be capable of carrying out the processing. Given prices, the cheapest design can then be determined.

The crux of the design problem is to see that the computer can keep up with the flow of data. We must calculate how many characters per second are transferred in and out of the computer, and compare this with the number of instructions that have to be

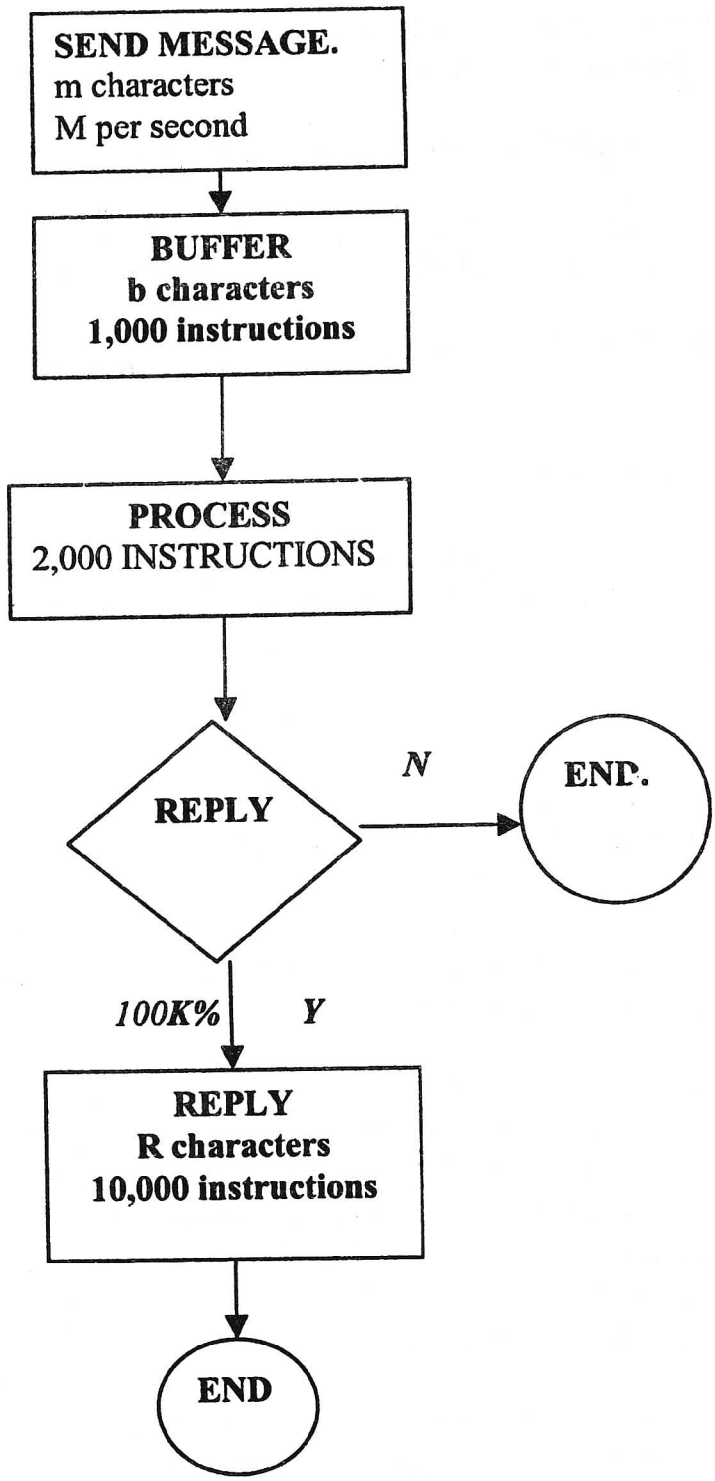


Figure 2.2 an input-output system

Executed every second. We need not be concerned with the exact sequence of events, as the use of the buffer is switched between these functions; we need only consider the total number of events involved. There are M messages coming into the computer and KM replies going out every second. This requires that Mm + kMr characters pass through the buffer every second. Since the buffer holds b characters, there will be M(m+kr)b interruptions every second. Adding

together the instructions to process messages, prepare replies, and service buffer interruptions, the number of instructions to be executed every second, denoted by N, is given by,

$$N = 2{,}000M + 10{,}000Mk + 1{,}000m(m + kr) / b$$

If we denote the number of instructions per second that computer can execute by s, the condition to be met for computer to keep up with the data flow as that N s.

To simplify the discussion, suppose the details of the message traffic are specified with the following values:

$M = 5, m = 15, K = 0.1, r = 50$

Only the buffer size, b, and the computer speed, s, remain to be selected. Putting the given values into the formula for N, the condition, after simplification, can be written

$$\frac{20}{b} \quad \frac{S}{5000} \quad -3$$

By trial, we can find which combinations of values satisfy the inequality. Alternatively, the solutions can be found by plotting. We can plot 20/b against b on log –log paper. Thee value of 20/b is plotted on the vertical scale, and the value of b, on the horizontal scale. The result is the diagonal line, slopping downwards to the right. The dotted vertical lines are drawn at the points 1, 2, 5, and 10, which are the possible buffer sizes. The horizontal dotted lines plot s/5,000 – 3 for the three values of computer speed.

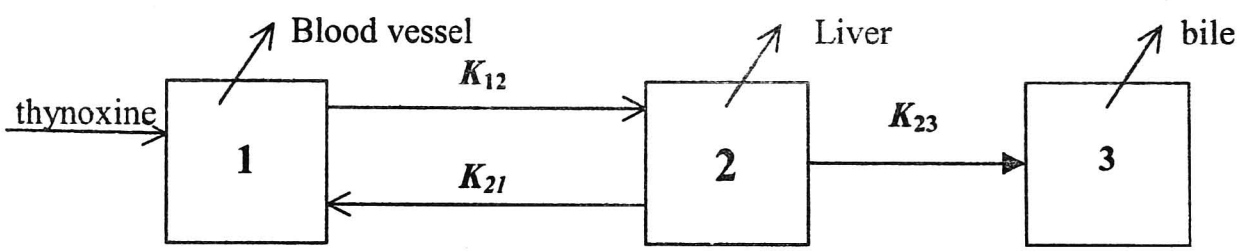
The twelve points of intersection of the vertical and horizontal lines represent the possible combinations of system components. The inequality, representing the conditions to be met, corresponds to saying the intersection must lie on or above the diagonal line. It can be seen that three combinations are satisfactory: the fast computer can operate with a two-character buffer, the medium-speed computer with a five-character buffer, and the low-speed computer will just keep up with a ten-character buffer. Realistic cost data will almost certainly result in the low-speed computer being the cheapest feasible design.

The problem in this case has been simplified to the point where a simple hand calculation solved the problem; further, the problem has been expressed in a form that allows a static model to be used. In practice, there will be mixtures of message types, with variations in sizes, even within a type; there will be congestion for getting access to data for generating replies; there will be competition for storage space to hold programs - all leading to delays in producing a replay. A simulation study would be needed to treat the more realistic problem.

2.5 SYSTEM POSTULATION

As a last example in the chapter, we illustrate the use of system postulation by looking at a study designed to investigate the function of the liver in the human body. When a chemical thyroxin, is injected into the blood stream, its carried to the liver. The liver can change thyroxin into iodine, which is absorbed into the bile. However, neither the conversion nor the absorption occurs instantaneously. Some of the thyroxin reenters the blood stream to be recalculated and returned to the liver. By using radioactive isotopes, it is possible to measure the rate at which thyroxin is removed from the blood stream, but the precise mechanism by which it is transferred from the blood to the liver and then to the bile was not known.

In the study, a mathematical model was constructed assuming that the body can as be represented as three compartments and that the rates at which thyroxin is transferred between the compartments are proportional to the concentration of thyroxin in the compartments. Figure 2-3, illustrates the model and shows the assumed transfer coefficients between compartments. The compartments 1, 2, and 3 represent the blood vessels, the liver, and the bile, respectively. The model leads to three simple differential equations, which are shown, together with their general solutions, in following figure 2.3.



$$dx_1/dt = -k_{12}x_1 + k_{21}x_2 \qquad x_1 = c_{11}e^{-b_1 t} + c_{12}e^{-b_2 t}$$

$$dx_2/dt = k_{12}x_1 - (k_{21} + k_{23})^x{}_2 \qquad x_2 = c_{21}e^{-b_1 t} + c^{22}e^{-b_2 t}$$

$$dx_3/dt = k_{23}x_2 \qquad x_3 = c_{31} + c_{32}e^{-b_1 t} + c_{33}e^{-b_2 t}$$

Figure 2.3 Mathematical model of the liver.

3. SYSTEM SIMULATION

3.1 THE TECHNIQUE OF SIMULATION

Given a mathematical model of a system, it is sometimes possible to derive information about the system by analytical means. Where this is not possible, it is necessary to use numerical computation methods for solving the equations. The factor that distinguishes analytical methods from numerical methods is that analytical methods directly produce general solutions-numerical methods produce solutions in steps; each step give the solution for one set of conditions, and the calculation must be repeated to expand the range of the solution.

Sometimes the term "simulation" is used to describe any procedure of establishing a model and deriving a solution numerically. However, in the case of static models, such as the corporate model, there seems to be little point in using the term "simulation" in preference to the general term "numerical computation," since no particular method of computation is being distinguished. In the case of dynamic models, however, such a distinction can be made.

Dynamic models can be solved analytically, as was seen in the case of the automobile wheel model discussed in Sec.1.10. If the model needs to be solved numerically, the particular technique that has come to be called simulation is the process of solving the equations of the model, step, by step, with increasing value of time. As a result, the current value at any step of the computation represent the state of the system being modeled at the point in time. We will be seeing some examples shortly. Before doing so, however, let's look at an example of solving a dynamic model numerically, which would not be considered simulation, using this point of view

Although the analytic solution to the automobile wheel model was not given in sec. 1-10 it was mentioned that the condition for the motion of the wheel not to oscillate is that $\zeta \geq 1$. Without going into details, this fact can be derived from knowing what value of p solve the following auxiliary equation,

$$P^2 + 2\zeta\omega p + \omega^2 = 0$$

This equation is related to the equation of motion describing the system (which is Eq.1-1 of sec. 1-10 by replacing the first and second derivatives of with p and p^2, respectively, and setting the value of x equal to 1. Because this equation is of the second order it can be solved analytically. A more complex model, however, might lead to a much higher order auxiliary equation. Nature of the response can still be determined by finding the solutions to the auxiliary equation but it may be impossible to the find the solutions analytically: instead, the equation must be solved numerically.

The information derived numerically in this way does not constitute simulation as it will be defined here: It does not involve knowing the actual motion of the system. If the information were derived by simulation, the motion of the system under various conditions would by determined and any conclusions about the nature of the motion would be drawn from these results-just as we observed from Fig.1-9 that, for the second- order system, oscillations appear to occur only when ζ is less than 1.

To distinguish this interpretation of simulation from more general uses of the term we will use the term "system simulation". We therefore define system simulation as the technique of solving problems by the observations of the performance, over time, of a dynamic model of the system. This definition is broad enough to include the use of dynamic physical models, in which case the results are derived from physical measurements rather than numerical computations. It is not intended here to discuss further the use of physical models, therefore, future references to simulation will be in terms of mathematical models and numerical computations.

In some problems the time element is not significant, but a step- by- step calculation representing the successive stages of development in a system is appropriate. For example one can evaluate the outcome of a sequential decision process, in which a number of successive choices must be made with given probabilities assigned to each choice, step-by-step, to find the probability of reaching each particular final outcome. Problems of this nature are often solved by simulation programs, in which use in made of the intrinsic step- by- step processing built in to the program. In such simulations, time is not explicitly involved. In effect, the model in use assumed to change at uniform intervals of time and, since only the results of the final step are of interest, there is no purpose in giving a specific value to the interval; it is therefore set to zero.

3.2 THE MONTE CARLO METHOD

A particular numerical computation method, called the *Monte carlo method*, consists of experimental sampling with random numbers. For example, the integral of a single variable over a given range corresponds to finding the area under the graph representing the function. Suppose the function, f (x) is positive and has lower and upper bounds a and b, respectively. Suppose , also, the function is bounded above by the value c. As show in Fig. 1-3, the graph of the function

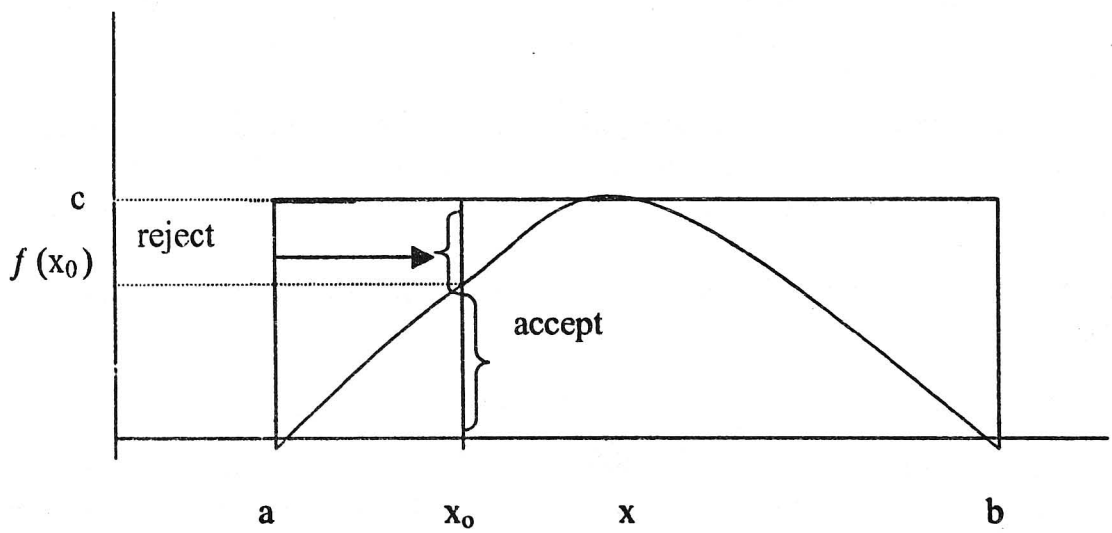


Figure 3-1. The Monte Carlo method

is then contained within a rectangle with side of length c, and b – a. If we pack points at random within the rectangle, and determine whether they lie beneath the curve or not, it is apparent that, proving the distribution of selected points is uniformly spread over the rectangle, the fraction of points falling on or below the curve should be approximately the radio of the area under the curve to the area of the rectangle. If *N* points are used and n of them fall under the curve, then, approximately,

$$\frac{n}{N} = \frac{\int_a^b F(x)dx}{C(b-a}$$

The accuracy improves as the number *N* increase. When it is decided that sufficient points have been taken, the value of the integral is estimated by multiplying n / N by the area of the rectangle, c(b – a).

The Computational technique is illustrated in Fig. 3-1. For each point, a value of x is selected at random between *a* and *b*, say x_o. A second random selection is made between O and c to give Y. If Y <*f* (x_o) the point is accepted in the court n, otherwise it is rejected and the next point is picked.

It is not likely that the Monte Carlo method would be used to evaluate an integral of a single variable: there are more efficient numerical computation methods for that purpose. However, the method is often used on integrals of many variables by using a random number for each of the variables,(16).It will noticed that, although random numbers have been used, the problem that has been solved is essentially determinate. There are many other applications of the Monte Carlo method, including examples where the problem being solved is of a statistical nature, such as in calculations concerned with reliability of nuclear reactors.

Monte Carlo applications are sometimes classified as being simulations. In addition, simulation is sometimes described as being an application of the Monte Carlo method, presumably because so many simulations involve the use of random numbers. Simulation and Monte Carlo are both numerical computational techniques. Simulation, as it will be described in this text, applies to dynamic models. The Monte Carlo technique is a computational technique applied to static models.

3.3 COMPARISON SIMULATION AND ANALYTICAL METHODS

Compared with analytical solution of problems, the n\main drawback of simulation is apparent: it gives specific solutions rather than general solutions. In the study of automobile wheel motion, for example, an analytical solution gives all the conditions that can cause oscillation. Each execution of a simulation tells only whether a particular set of conditions did or did not cause oscillations. To try to find all such conditions requires that the simulation be repeated under many different conditions. The mathematical solution is obviously preferable, particularly when the solution being sought is some maximizing condition. A single mathematical solution might give such a condition, whereas many simulation runs may be needed to find a maximum, and yet still leave undecided the question of whether it is a local or global maximum.

The range of problems that can be solved mathematically is limited, however. Mathematical techniques require that the model be expressed in some particular format-for example, that it be in the form of linear algebraic equations or continuous linear differential equations. The system must be approximated or abstracted in order to derive a model that fits the format of a mathematical technique. All to often, the process amounts, to making the problem fit the solution rather than the other way round.

In approaching a system study, it is essential to consider first what mathematical techniques might be applied to derived analytical solutions. It is a matter of judgment to decide whether the degree of abstraction required to apply analytical methods is too server. To make that judgment it is necessary to consider carefully what questions need to be answered in the system study, and to what degree of accuracy the answers need to be known.

When the decision is made to simulation in order to use a more realistic model, it is still important to limit the amount of detail in the model to the level necessary. The step-by-step nature of the simulation technique means that the amount of computation increases very rapidly as the amount of the detail increases. Coupled with the need to make many runs to explore the range of conditions, the extra realism of simulation models can result in a very extensive amount of computing.

In many ways, the ideal way of using simulation is as an extension of mathematical solutions that might have been obtained at the cost of too much simplification. There are many simple limitations on a system, such as physical stops, finite time delays, or nonlinear forces, which render what would other wise be a soluble mathematical model insoluble. Simulation easily removes these limitations, and it can then provide a powerful extension of known mathematical solutions.

Even when a model with a known solution is considered adequate, there are still times when a simulation will provide a quicker of more convenient way of deriving results. Many analytical results occur in the form of complex series or integrals that still require extensive evaluation. It is often more convenient to use simulation to obtain results directly from a model with specific values rather than to perform the numerical evaluation of the analytical solution.

3.4 EXPERIMENTAL NATURE OF SIMULATION

The simulation technique makes no specific attempt to isolate the relationships between any particular variables; instead, it observes the way in which all variables of the model change with time. Relationships between the variables must be derived from these observations. The relationship between D, k, and M to prevent oscillation, which was previously discovered analytically, would have to be discovered by observing the values that result in the motion being non-oscillatory. Simulation is, therefore, essentially an experimental problem -solving technique. Many simulation runs have to be made to understand the relationships involved in the system, so the use of simulation in a study must by planned as a series of experiments.

3.5 TYPES OF SIMULATION

In Sec. 1-4 we distinguished between continuous and discrete systems as being systems in which smooth or sudden changes occur. We went on, however, to comment that the important distinction, from the point of view of a system analyst, is whether the model selected to represent the system is continuous or discrete, since there is not a unique correspondence between types of systems and models.

The distinction between continuous and discrete models, however, was not made in the classification of models shown in Fig. 1-5, because this distinction does not determine whether analytical or numerical techniques will be applied to the model, a fact which was made a point of distinction. The distinction between continuous and discrete models becomes important when it is decided to use simulation, particularly when the simulation is to be carried out on a computer and programming system is to be selected to perform the task. The general computational techniques used with the two kinds of model differ significantly. We will be discussing continuous and discrete simulation methods more fully in later chapters. Before doing so, however, we will show, in the next two sections, the general nature of the computational techniques that are used.

NUMERICAL COMPUTATION TECHNIQUE FOR CONTINUOUS MODELS

To illustrate the general numerical technique of simulation based on a continuous model, consider the following example, (6). A builder observes that the rate at which he can sell houses depends directly upon the number of families who do not yet have house. As the number of people without hoses diminishes, the rate at which he sells houses drops. Let H be the potential number of households, and Y be the number of families with houses. The situation is represented in Fig. 3-2;

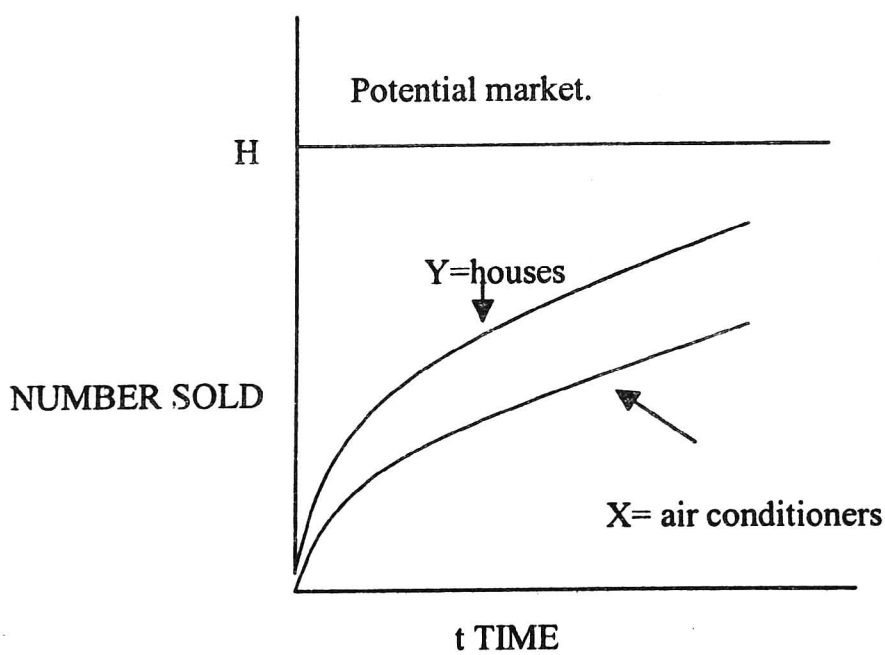


Figure 3-2.sales of houses and air conditioners.

The horizontal line at H is the total potential market for houses. The curve for y indicates how the number of houses sold increases with time. The slope of the curve for (i.e., the rate at which y increases) decreases as H – y gets less. This reflects the slowdown of sales as the market becomes saturated. Mathematically, the trend can be expressed by the equation

$$\dot{y} = k_1(H - y), \qquad y = 0 \text{ at } t = 0$$

Consider now a manufacturer of central air conditioners designed for houses. His rate of sales depends upon the number of houses built. (For simplicity, it is assumed that all houses will install an air conditioner.). As with houses sales, the rate of sales diminishes as the unfilled market diminishes.

Let x be the number of installed air conditioners. Then the unfilled market is the difference between the number of houses and the number of installed air conditioners. The sales trend may be expressed mathematically by the equation

$$\dot{x} = k_2(y - x), \qquad x = \theta \text{ at } t = \theta$$

The change of x with time is also illustrated in Fig [illegible]. The two equations constitute a model of the growth of air conditioner sales. Because of its simplicity, it is in fact possible to solve the model analytically. However, it quickly becomes insoluble if it is expanded to become more representative of actual marketing conditions. The market limit, for example, may not be stable. It could grow with population growth or fluctuate with economic conditions. The coefficients that determine the rates of growth could be influenced by the amount of many spent on advertising, and there could be competitive influences, such as mobile homes or apartment housing. These influences could also depend upon the population growth or prevailing economic conditions, and so further complicate the model.

The simple model, however, will serve to illustrate the general methods applied in continuous simulation. The simulation technique is to compute the output, step by step. Suppose that the computation is made at uniform intervals of time and that the calculation has already progressed to the time t_i, when the two variables of the problem have the values y_i and x_i figure 3-3 shows the next step in the calculation

The calculation steps forward an interval Δt to $t_{i+1} = t_i + \Delta t$. The rates of sales are

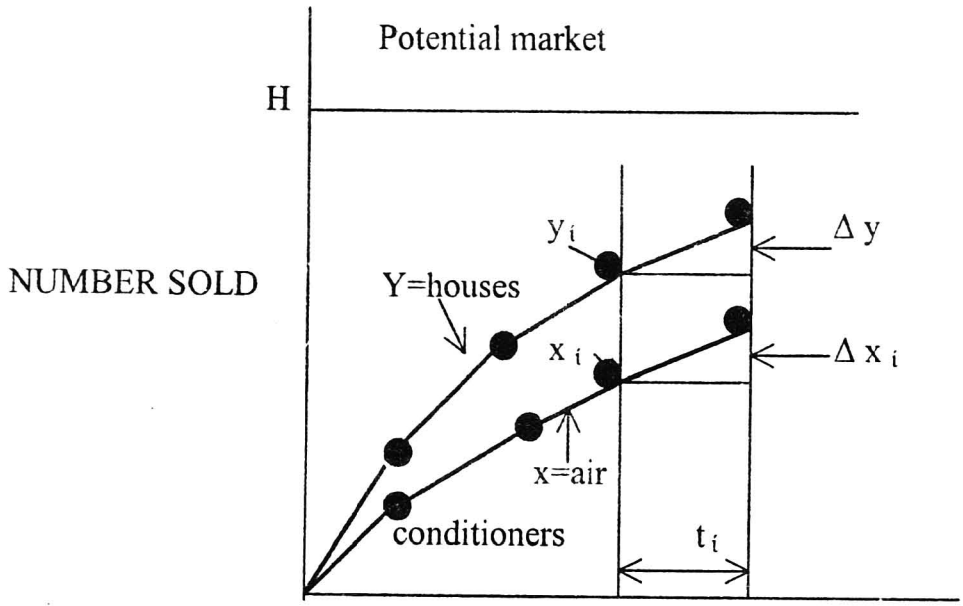


Figure 3-3. Calculations for air conditioner sales model

assumed to be constant over the interval. The rates can be interpreted as the amount of change per unit time. That is,

$$\text{Rate of change of } y \quad \frac{\Delta y_i}{\Delta t}$$

$$\text{Rate of change of } x \quad \frac{\Delta x_i}{\Delta t}$$

From the equations of the model, these may be written

$$\Delta y_i = k_1(h - y_i)\,\Delta t$$

$$\Delta x_i = k_2(y_i - x_i)\,\Delta t$$

Since y_i and x_i are known, it is a simple matter to get the values of y and x At time t_{i+1}. However, it will be noticed that the equation for Δy_i must be solved first to get the value of y_i needed in the equation for x_i. In preparation for the solution of a continuous system model, therefore, there is must be a careful starting of the equations to establish a workable order.

Repetition of the calculation using the new values of y and x produces the output at the end of the next interval. As illustrated in Fig. 3-3, the calculation is equivalent to calculating the slope at each point and projecting a short straight line at that slope. The simulation output is a series of such line segments, approximating the continuous curve that represents the true output of the model.

The method described is a very simple way of integrating different equations numerically, but it is not a very accurate method, unless small steps are used, compared with the rate at which the variables change. There are other much more accurate, and often more efficient, ways of integrating numerically which do not rely simply upon the last-known value of the variables. Rather, they use several previous values to predict the rate at which the variables are changing. (Special methods are used to supply initial values to start the process.) In addition, the computation interval is often adjusted in size to match the rate at which the variables are changing

There are many programming systems available that incorporate continuous system simulation languages. They usually include a number of computational methods for the user's selection.

NUMERICAL COMPUTATION TECHNIQUE
For Discrete Model

To illustrate the general computational technique of simulation with discrete models, consider the following example. A clerk begins his day's work with a pile of documents to be processed. The time taken to process them varies. He works through the pile, beginning each document as soon as he finishes the previous one, except that he takes a five minutes break if, at the time he finishes a document, it is an hour or more since he began work or he last had a break. We assume the times to process the documents are given. We will also keep a count of how many documents are left The count will be initially set to the number of documents at the beginning of the day, and, in this simple model, we will assume that no documents arrive during the day. The count will be decremented for each completed job, and work will stop if the count goes to zero.

The computations involved can be organized as shown in Table 3-1. The ith row corresponds to the *i*th document. The first column numbers the documents. There are then four

TABLE 3-1 Simulation of document processing

Document Number i	Start Time t_b	Work Time t_w	Finish Time t_f	Cumul. Time t_e	Break Flag F	Number of jobs N
1	0	45	45	45	0	57
2	45	16	61	61	1	56
3	66	5	71	5	0	55
4	71	29	100	34	0	54
5	100	33	133	67	1	53
6	138	25	163	25	0	52
7	163	21	184	46	0	51

columns giving various times, measured in minutes from time zero. The second column gives the time the clerk begins to work on a document, denoted by t_b (i),The third column gives the time

required to work on the document, denoted by $t_w(i)$ and the fourth column gives the time each document is finished. The fifth column contains the cumulative time since work started or since the last break, measured at the time each job is completed. This is denoted by $t_c(i)$. There is a sixth column which contains a flag, denoted by F, that takes the value I if the clerk should take a break after the document, and the value 0 if he should not The clerk works until there are no more documents, or the time he finishes a document goes beyond some time limit

The computation proceeds row by row, and from left to right .The first row shows that work starts on the first document at time zero. The processing time is 45 minutes, so the job is finished at 45, with a cumulative time (in this case, since the start of work) of 45. This is not long enough for a break, so the flag is set to 0. The count, which was initialized to 57 jobs, is dropped to 56.

Because the flag is zero and the count is not zero, the second document is begun at 45.It needs 16 minutes for processing, which leads to a cumulative time of 61, so the flag is set to 1 to indicate that a break should be taken. Because of the five minutes break, the third document starts at 66. The computation continues in this manner until either, N, the count of documents to be processed, drops to zero, or the finish time, t_f, reaches some limit representing the end of the work period.

3.6 DISTRIBUTED LAG MODELS

The computation for the model of the previous section was straightforward, but it can be imaged that the record keeping quickly becomes burdensome as the model becomes larger or more complex. The aid of a computer and a suitable programming language then becomes important. There are some applications of simulation, however, where the simple technique illustrated in the previous section can be applied without difficulty, even for large models. If the events all occur synchronously, at fixed intervals of time, the computation remains simple.

Models that have the properties of changing only at fixed intervals of time, and of basing current values of the variables on other current values and values that occurred in previous intervals, are called distributed lag models. They are used extensively in econometric studies where the uniform steps correspond to a time interval, such as month or a year, over which some economic data are collected. As a rule, these models consist of linear, algebraic equations. They represent the continuous system, but one in which the data is only available at fixed points in time.

As an example, consider the following simple mathematical model of the national economy. Let,

C be consumption,
I be investment,
T be taxes,
G be government expenditure,
Y be national income,

$$C = 20 + 0.7(Y - T)$$
$$I = 2 + 0.1Y$$
$$T = 0.2Y$$
$$Y = C+I+G$$

All quantities are expressed in billion of dollars.

This is a static mathematical model, but it can be made dynamic by picking a fixed time interval, say one year, and expressing the current values of the variables in terms of values at previous intervals. Any variable that appears in the form of its current value and one or more previous intervals is said to be a *lagged variable*. Its value in a previous interval is denoted by attaching the suffix – n to the variable, where n indicates the interval, with I denoting the previous interval, 2 denoting the one prior to that, and so on. In the present example we will only go back one interval.

The static model could be made dynamic by lagging all the variables, as follows:

$$I = 2 + 0.1Y_{-1}$$
$$T = 0.2Y_{-1}$$
$$Y = C_{-1} + I_{-1} + G_{-1}$$
$$C = 20 + 0.7(Y_{-1} - T_{-1})$$

Given an initial set of values for all variables, the values of the variables at the end of one year can be derived. Talking these values as the new values of the lagged variables, the values can than be derived at the end of the second year, and so on. There are only four equations in five unknowns; so one variable must be specified for each interval.

It is not necessary, however, to lag all the variables as has just been done. Suppose there is one equation that expresses a single current variable in terms of lagged variables only. When this equation is solved, a second equation can be solved that involves the current value just derived from the first equation plus any lagged variables, and so on. Using this principle, a slight rearrangement of the model just given makes it depend upon only one lagged variable. Substituting for T and C in the original equation for Y gives

$$Y\ 45.45 + 2.27(I + G)$$

The set of equations can then be written in the form:

$$I = 2 + 0.1Y_{-1}$$
$$Y = 45.45 + 2.27(I + G)$$
$$T = 0.2Y$$
$$C = 20 + 0.7(Y - T)$$

The only lagged variable is Y, and to solve the model an initial value of Y_{-1} must be given. With the lagged variable, the current value of I can be derived from the first equation. Suppose the values of G are supplied for all intervals. The next equation then gives the current value of Y from the current value of I. The next equation similarly gives current value of T from the current value of Y, and the last equation gives the current value of C from current values of

both Y and T. Taking the current value of Y as the new value of the lagged variable Y_{-1}, the calculations can then be repeated for the next interval of time.

In practice, of course, the variables are not arbitrarily lagged to produce a dynamic model. Extensive statistical analysis to need to establish that the dependencies do indeed exist, as well as to produce values for the coefficients in the equations, (12).

Although distributed lag models are conceptually simple, and they can be computed be hand, computers are extensively used to run them. However, the value of the computer is its more conventional data- processing capability. Unlike other types of model used for simulation purposes, there is no need for a special programming language to organize the simulation task.

Econometric models of this nature have been built fort the national economics of most of the major industrial countries. Many large corporations also use models reflecting the effects of the national economy on their industry. Among the better known model of the U.S. economy are the models developed at the Wharton school of the university of Pennsylvania, Professor Klein's models at the University of Chicago, and a model at the Brooking Institute in Washington, D.C,.

3.7 COBWEB MODELS PROGRESS OF A SIMULATION STUDY

A particularly simple, but nevertheless useful, distributed lag model can be constructed from the static market model that was given in sec.1-9. This related supply, S, and demand, D, to market price, P. To be more realistic, the supply should be made dependent upon the price from the previous marketing period, since that is the only figure available to the supplier at the time of marketing future plans. The demand, however, will respond to current price. Again assuming the market is cleared, the market model in distributed lag form is as follows:

$$Q = a - bP$$
$$S = c + dP_{-1}$$
$$Q = S$$

Given an initial value of price, P_o, the value of S at the end of the first interval can be derived. This determines the value of Q, since the market is cleared and from this the new value of P can be derived. This becomes the value of P_{-1} used to calculate the values for the second interval, and so on. Figure 3-4 shows the fluctuations of price for the following two cases:

(a)	(b)
$P_o = 1.0$	$P_o = 5.0$
$A = 12.4$	$a = 10.0$
$b = 1.2$	$b = 0.9$
$c = 1.0$	$c = 2.4$
$d = 0.9$	$d = 1.2$

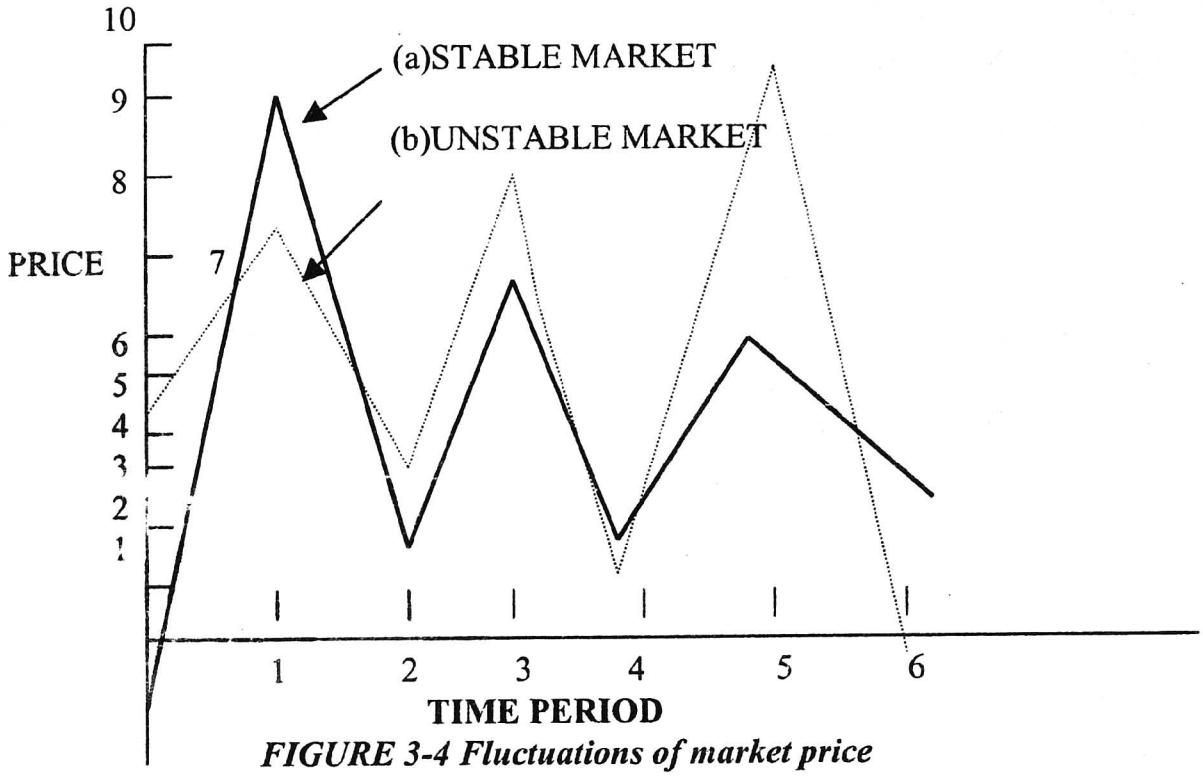


FIGURE 3-4 Fluctuations of market price

As can be seen, case (a) represents a stable market in which the market settles tom a price of 5.43 units. Case (b) is unstable, with price fluctuating with increasing amplitude.

Models of this type are called cobweb models because of a particular way they can be solved graphically. The method is illustrated in Fig. 3-5 for the stable case just considered. Two straight lines plot the linear relationships representing supply and demand, in the same way as they were plotted in Fig. 1-7 for the static market model. Beginning from the initial price of 1, a horizontal line to the supply line determines the supply that would be produced at that price level: about 1.8 units. A vertical line to the demand line reflects the fact that the market is cleared by making the demand equal the supply. With this small supply the price immediately rises to about 8.7 units. Taking that price as a basis for production, the supply in the next period jumps to about 8.8 units, as show by the horizontal line, but, at this price, the supply is too big, therefore the price must drop to about 3.0 units in order to sell that amount. Continuing in this manner, each successive half- turn of the spiraling pattern produces the price in successive time periods. As can be seen, the spiral will eventually reach the equilibrium price of 5.43, from which time the market will remain unchanged.

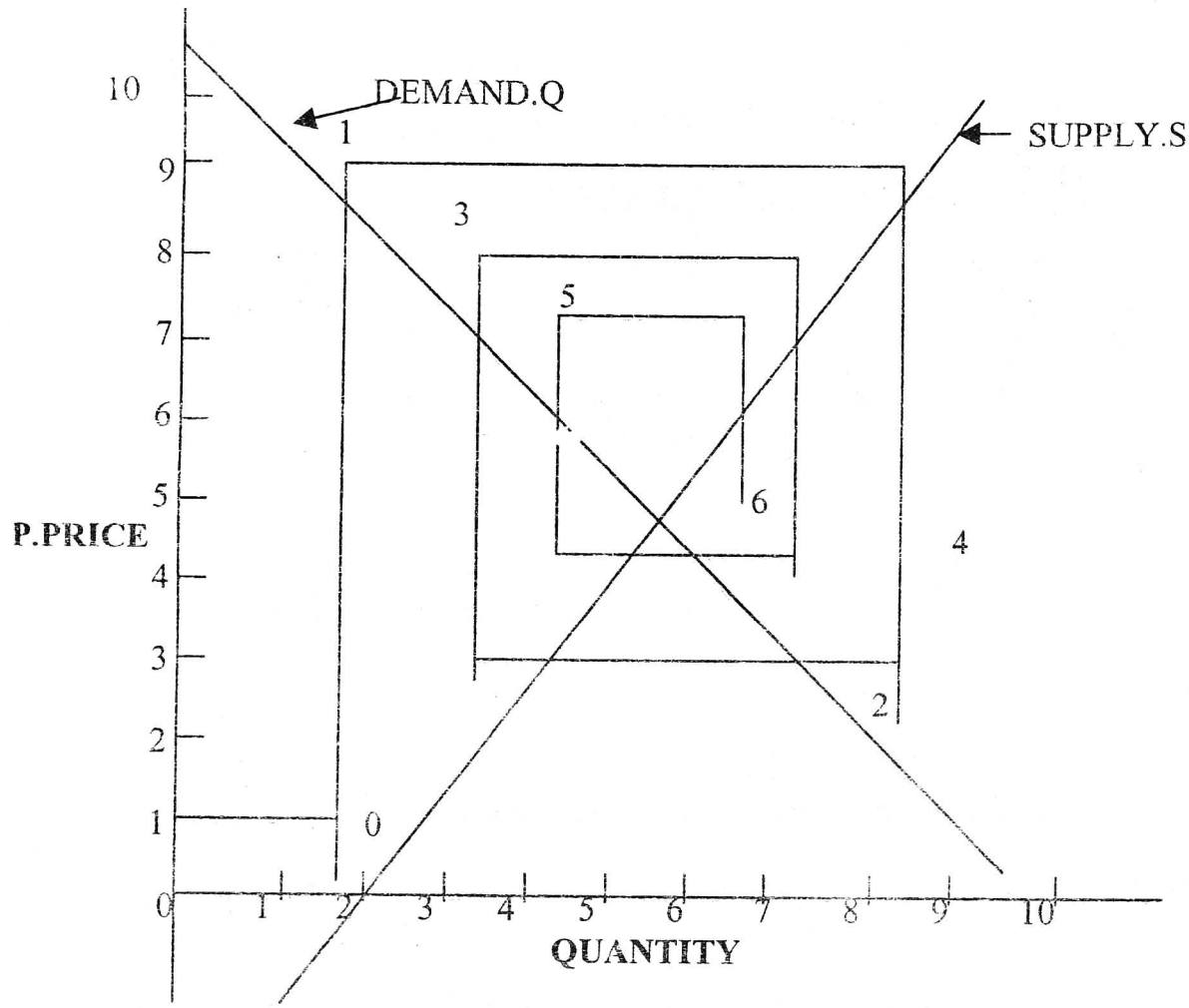


Figure 3-5 cobweb model of a market economy.

Progress of simulation Study

It is apparent from general discussion of the preceding sections that simulation is a very general method of studying problems. No formal procedure can be given for showing how a simulation study will proceed. There is not even a simple way of deciding whether to simulate or not, or, if simulation is used, whether to use a continuous or discrete model. However, some of the steps involved in the progress of a simulation study are illustrated by the flowchart of 3-6. The flowchart is illustrative: it is not indented to suggest a formal procedure.

An initial step is to describe the problem to be solved in as concise a manner as possible so that there is a clear statement of what questions are being asked and what measurements need to be taken in order to answer those questions. Based on this problem definition, a model must be defined. It is at this point that it becomes apparent whether the model can be a form that allows analytical techniques to be used.

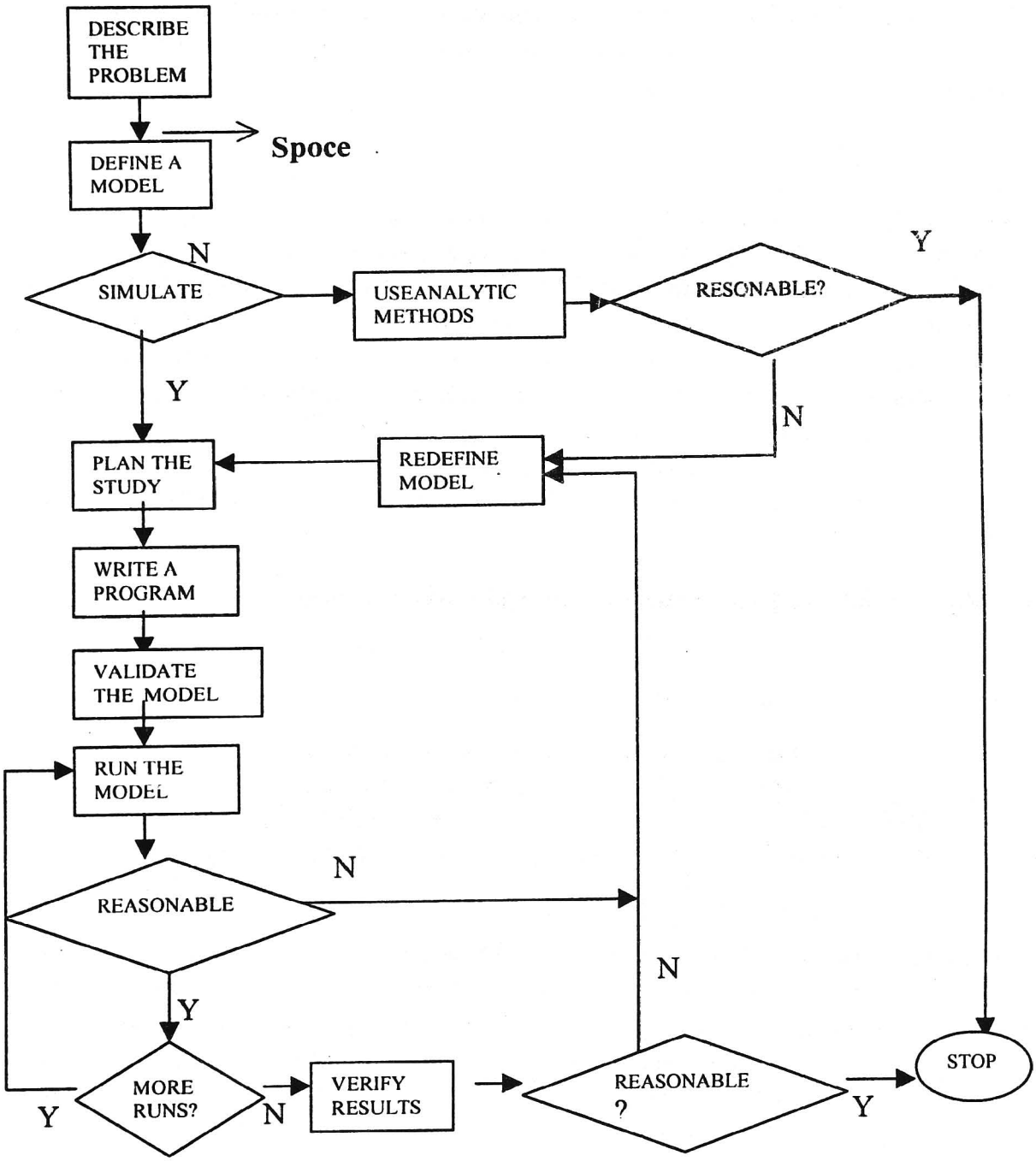
It should be understood that there is not, in fact, a single model for any given system. In the course of a study many different models are, likely to be constructed as understanding of the system behavior increases. A possible course, which is to be recommended if it does not entail too much extra effort, is to explore first a model that can be solved analytically, even though it is clear that the simplifications required to produce the model are too drastic. The results will help guide the simulation study.

When it is decided to simulate, the experimental nature of the simulation technique makes it essential to plan the study by deciding upon the major parameters to be varied, the number of cases to be conducted, and the order in which runs are to be made. This procedure will help gauge the magnitude of the simulations are to be made. This procedure will help gauge the magnitude of the simulation effort and may cause a reappraisal of the model. Of course, it is not possible to plan every step ahead of time, and the study plan may need to be revised periodically as results become available; but the plan should always be to show the direction in which the study is going.

Given that the simulation is to be on a digital computer, a program must be written. Actually this step is likely to be carried out in parallel with the study planning, once the model, structure has been decided. Figure 3-6 does not show the choice between using a continuous or discrete model. The figure does show, however, the step of establishing the validity of the model before beginning the major sets runs. This is an important step that requires good judgment. It is worth bearing in mind the need for validation at the time of deciding upon the model. It could be that some of the earlier, oversimplified models may provide guidelines for establishing the reasonableness of the final model.

The study will then move into the stage of executing a series of runs according to the study plan. As has already been intimated, the study is not likely to proceed in the orderly steps implied by the flowchart of Fig, 3-6.As results are obtained; it is quite likely that there will be many changes in the model and the study plan. Frequently, the main value of the early runs is a simulation study is to gain insight into the general behavior of the system, rather than to extract data. In the construction of a model certain parameters of the system are clearly going to be important and others unimportant – to the point where they can be neglected or simplified.

Figure 3-6. The Process of Simulating.



There will also be parameters whose importance, either by themselves or in interaction with other parameters, is not so clear. The early runs may make their significance clear, and so lead to a reassessment of the model. The blocks in Fig 3-6 that ask the question of whether the results are reasonable imply this or not. This should not, however, be interpreted as an incentive to include numerous parameters in the model on the assumption that the unimportant ones can be quickly eliminated. As was emphasized in sec. 3-4 it is essential to keep the model as simple as possible.

A major factor that needs to be considered in the presence of random variables is the problem of statistically verifying the results. The flowchart of Fig 3-6 shows this step as following the runs, but in fact, it must also be considered on each planned run. Once random variables enter a model, almost all measures of interest will also be random variables. The result of a single run will give just one sample of the measured variable. It is thus essential to repeat the run with a different set of random numbers so that more than one sample is available; it is then possible to judge the variability of the results. The number of repetitions needed depends upon the nature of the system and the importance to be attached to the results.

Sometimes it is useful to repeat runs so that parts of the model have different random numbers while the rest use exactly the same random numbers on each run. By comparing the results have different random numbers; it is then possible to analyze the cause of the variance in the results. The problem of verifying simulation results will be discussed in detail elsewhere.

4.CONTINUOUS SYSTEM SIMULATION

4.1 CONTINUOUS SYSTEM MODEL

A continuous system is one in which the predominant activities of the system cause smooth changes in the attributes of the system entities. When such a system is modeled mathematically, the variables of the model representing the attributes are controlled by continuous functions. The distributed lag model discussed in sec. 3-8 is an example of a continuous model. That model described how the attributes of the system were related to each other in the form of linear algebraic equations. More generally, in continuous systems, the relationships describe the rates at which attributes change, so that the model consists of differential equations.

The simplest differential equation models have one or more linear differential equations with constant coefficients. It is then often possible to solve the model without the use of simulation. Even so, the labor involved may be so great that it is preferable to use simulation techniques. However, when non-linear ties are introduced in to the model, it frequently become impossible or, at least, very difficult to solve the models. Simulation methods of solving the models do not change fundamentally when non-linear ties occur. The methods of applying simulation sat the continuous model can therefore be developed by showing their application to models where the differential equations are linear and have constant coefficients, and then generalizing to more complex equations.

4.2 Differential equations

An example of a linear differential equation with constant coefficients was given in sec. 1-8 to describe the wheel suspension system of an automobile. The equation derived was

$$M\ddot{X} + D\dot{X} + Kx = KF(t) \text{ (or) } \frac{Md^2x}{dt} + \frac{Ddx}{dt} + Kx = k\,f(t)$$

Note that the dependent variable x appears together with it is first and second derivatives and, and that the terms involving these quantities are multiplied by constant coefficients and added. The quantity F (t) is an input to the system, depending upon the independent variable t. A linear differential equation with constant coefficients is always of this form although derivatives of any order may enter the equation. If the dependent variable or any of its derivatives appear in any other form, such as being raised to a power, or are combined in any other way – for example, by being multiplied together, the differential equation is said to by nonlinear.

When more then one independent variable occurs in a differential equation, the equation is said to be a partial differential equation. It can involve the derivatives of the same dependent variables. An example is an equation describing the flow of heat in a three dimensional body. There are four independent variables, representing the three dimensions and time, and one dependent variable, representing temperature. The general method of solving such equations numerically is to use finite differences to convert the equations into a set of ordinary (that is, non-partial) differential equations, which can be solved by the methods that are about to be described. There are programming languages specially designed for this type of equation, which will perform the function of constructing the finite differences.

Differential equations, both linear and nonlinear, occur repeatedly in scientific and engineering studies. The reason for this prominence is that most physical and chemical processes involve rates of change, which require differential equation for their mathematical description. Since a differential coefficient can also represent a growth rate, continuous models can also be applied to problem of a social or economic nature where there is a need to understand the general effects of growth trends. We shall be discussing such socio-economic systems in the next chapter. For now, we concentrate on the solution of scientific and engineering types of problems which generally require a higher degree of accuracy than just establishing trends. This, however, is only a separation of types of application which have different emphases. The continuous system simulation methods that will be described in this chapter can be used for either type of application.

To illustrate how differential equations can represent engineering problems we will show how the equation describing the automobile wheel suspension system is derived from mechanical principles. If we pick a point of the wheel as a reference point from which to measure the vertical displacement of the wheel, the variable x can represent the displacement of the point; taking x to be positive for an upward movement. The velocity of the wheel, in the vertical direction, is the rate of change of position, which is the first differential. The acceleration of the wheel, in the vertical direction, is the rate of change of the velocity. which is the second differential,

The mechanical law that determines the relationship between applied force and movement of a body states that the acceleration of the body is proportional to the force. In particular, if there is no force there is no acceleration. The body then remains stationary, if no force has been acting upon it, or continues moving at a constant velocity-a fact that, at one time would have seemed to be merely an abstraction but is familiar in this age of space travel, since the ideal condition of no force acting on a body is obtainable in space.

The coefficient of proportionality between the force and acceleration is the mass of the body; so, in the case of the automobile wheel, where the mass is M and the applied force is KF(t), the equation of motion in the absence of other forces would as follows:

$$M\ddot{X} = KF(t)$$

However, the shock absorber exerts a resisting force that depends on the velocity of the wheel: the force is zero when the wheel is at rest, and it increases as the velocity rises. If we assume the force is directly proportional to the velocity, it can be represented by $D\dot{X}$, where D is a measure of the viscosity of the shock absorber. Similarly, the spring exerts a resisting force which depends on the extent to which it has been compressed. (Assume that x is defined so that it is zero when the spring is uncompressed.) Again, if the force is directly proportional to the compression, it can be represented by Kx, where K is a constant defining the stiffness of the spring. Since both these force oppose the motion, they subtract from the applied force to give the following equation of motion:

$$M\ddot{X} = KF(t) - D\dot{X} - Kx$$

Transposing terms, the equation is seen to be Eq. (4-1), which was previously given. It is a linear differential equation, with constant coefficients, and, as mentioned before, it can be solved analytically. A more accurate model, however, would not assume the restraining forces are linear functions of the motion variables, and so the coefficients would not be linear. Also there are physical limits to the amount of movement that is possible, which places another no linearity on the equation. The more realistic model is unlikely to be soluble by analytic methods, so it would become the basis of a continuous system simulation study.

In Sec. 1-8 the equation that has just been derived was also used to describe an electrical system. We will not derive the equation for that interpretation, but the same general principle applies. The quantity being measured in that case is the electrical charge of the condenser. The physical laws of electricity relating current and voltage to charge introduce the first two derivatives of the charge, and so lead to a linear differential equation of the same form as that for the automobile wheel, but with different interpretations of the constant coefficients. Similar examples can be taken from many other fields of engineering.

4.3 Analog Methods

The general method by which analog computers are applied can be demonstrated using the second-order differential equation that has already been discussed:

$$M\ddot{X} + D\dot{X} + Kx = KF(t)$$

Solving the equation for the highest order derivative gives

$$M\ddot{X} = KF(t) - D\dot{X} - Kx$$

Suppose a variable representing the input F(t) is supplied, and assume for the time being that there exist variables representing –x and - . These three variables can be scaled and added with summer to produce a voltage representing M . Integrating this variable with a scale factor of 1/M produces , which supplies one of the variables initially assumed; and a further integration produces –x, which was the other assumed variable. For convenience, a further sign inverter is included to produce + x as an output.

A block diagram to solve the problem in this manner is shown in Fig 4-1. The symbols used in the figure are standard symbols for drawing block diagrams representing analog computer arrangements. The circles indicate scale factors applied to the variables. The triangular symbol at the left of the figure represents the operation of adding variables. The triangular symbol with a vertical bar represents an integration, and the one containing a minus sign is a sign changer.

$$M\ddot{X} + D\dot{X} + Kx = KF(t)$$

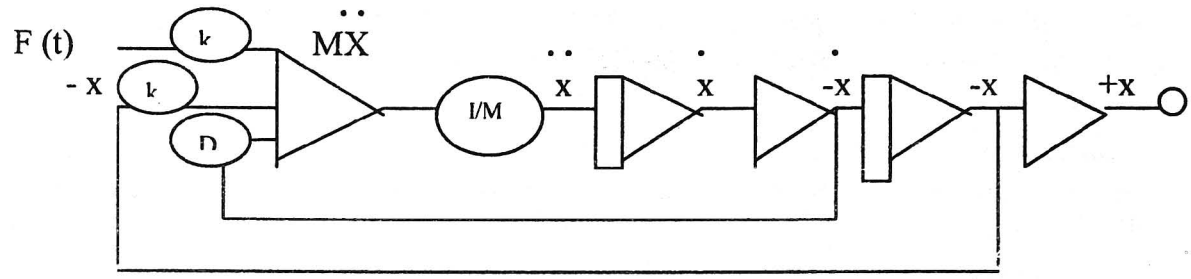


Figure 4-1. Diagram for the automobile suspension problem.

The addition on the left, with it is associated scaling factors, corresponds to he addition of :he variables representing the three forces on the wheel, producing a variable representing M. The scale is changed to produce, and the result is integrated twice to produce both and x. Sign :hangers are introduced so that variables of the correct sign can be fed back to the adder, and the)utput can be given in convenient form.

With an electronic analog computer, the variable that have been described would be /oltages, and the symbols would represent operational amplifiers arranged as adders, integrators, ind sign changers. Figure 4-1 would then represent how the amplifiers are interconnected to ;olve the equation. It should be pointed out, however, that there can be several ways of drawing a liagram for a particular problem, depending upon which variables are of interest, and the size of he scale factors.

When a model has more than one independent variable, a separate block diagram is lrawn for each independent variable and, where necessary, interconnections are made between he diagrams. As an example, Fig 4-2 shows a block diagram for solving the model of the liver

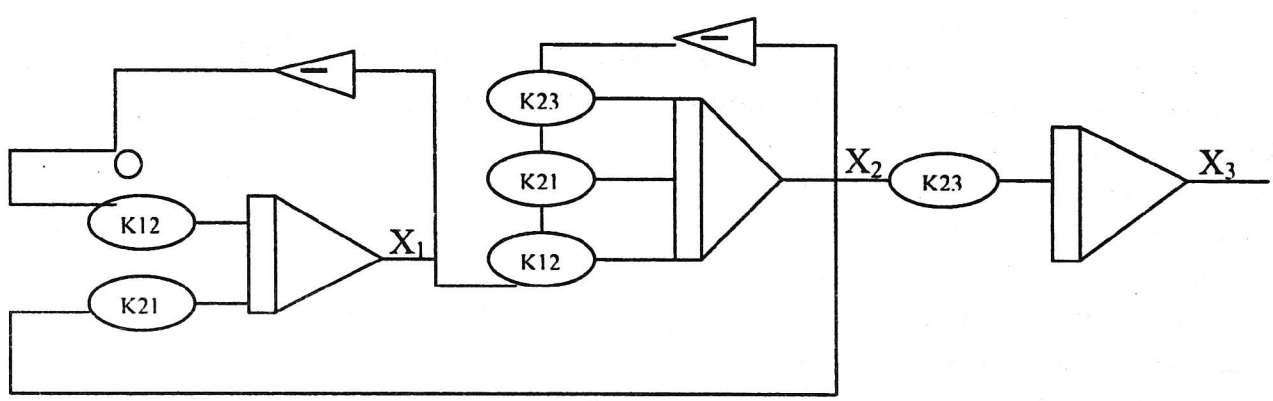


Figure 4-2. Analog computer model of the liver.

Shown in Fig 2-10. There are three integrators, shown at the bottom of the figure. Reading from left right, they solve the equation for x_1, x_2, and x_3. Interconnections between the three integrators, with sign changers where necessary, provide inputs that define the differential coefficients of the three variables. The first integrator, for example, is solving the equation

$$\dot{x}_1 = - K_{12}x_1 + K_{21}x_2$$

The second integrator is solving the equation

$$_2 = - K_{12}x_1 - (K_{21} + K_{23})x_2$$

In this case, the variable x_2 is being used twice as an input to the integrator, so that the two coefficients K_{21} and K_{23} can be changed independently. The last integrator solved the equation

$$_3 = K_{23}X_2$$

4.4 ANALOG AND HYBRID COMPUTERS.

Analog Computers

Historically, continuous system simulation was in general use for studying complex systems long before discrete system simulation was similarly applied. The main reason was that, before the general availability of digital computers, needed for the numerical computation of discrete system simulation, there existed devices whose behavior is equivalent to a mathematical operation such as addition or integration. Putting together combinations of such devices in a manner specified by a mathematical model of a system allowed the system to be simulated. By their nature, the devices gave continuous outputs and so lent themselves readily to the simulation of continuous systems. Specific devices have been created for particular systems but with so general a technique it has been customary to refer to them as analog computers, or, when they are primarily used to solve differential equation models, as differential analyzers. Physical models based on analogies were described in chapter1. The specific example discussed there described the analogy between electrical and mechanical systems. Simulation with an analog computer, however, is more properly described as being based on a mathematical model than as Being a physical model.

The most widely used form of analog computer is the electronic analog computer, based on the use of high gain dc (direct current) amplifiers, called operational amplifiers. Voltages in the computer are equated to mathematical variables, and the operational amplifiers can add and integrate the voltages. With appropriate circuits, an amplifier can be made to add several input voltages, each representing a variable of the model, to produce a voltage representing the some of the input variables. Different scale factors can be used to on the inputs to represent coefficients of the model equations. Such amplifiers are called summers. Another circuit arrangement produces an integrator for which the output is the integral with respect to time of a single input voltage or the sum of several input voltages. All voltages can be positive or negative to correspond to the sign of the variable represented. To satisfy the equations of the model, it is sometimes necessary to use a sign inverter, which is an amplifier designed to cause the output to reverse the sign of the input.

Electronic analog computer limited in accuracy for several reasons. It is difficult to carry the accuracy of measuring a voltage beyond a certain point. Secondly, a number of assumptions are made in deriving the relationships for operational amplifiers, none of which is strictly true; so amplifiers do not solve the mathematical model with complete accuracy. A particularly troublesome assumption is that there should be zero output for zero input. Another type of difficulty is presented by the fact that the operational amplifiers have a limited dynamic range of output, so that scale factors must be introduced to keep within the range. As a

consequence, it is difficult to maintain and accuracy better than 0.1% in an electronic analog computer. Other forms of analog computers have similar problems and their accuracies are not significantly better.

A digital computer is not subject to the same type of inaccuracies. Virtually any degree of accuracy can be programmed and, with use of floating-point representation of numbers, an extremely wide range of variations can be tolerated. Integration of variables is not a natural capability of digital computer, as it is in an analog computer, so that integration must be carried out by numerical approximations. However, methods have been developed which can maintain a very high degree of accuracy.

A digital computer also has the advantage of being easily used for many different problems. An analog computer must usually be dedicated to one application at a time, although time-keeping sections of a analog computer has become possible.

In spite of the widespread availability of digital computers, many users prefer to use analog computers. There are several considerations involved. The analog representation of a system is often more natural in the sense that it directly reflects the structure of the system; thus simplifying both the setting-up of a simulation and the interpretation of the results. Under certain circumstances, an analog computer is faster than a digital computer, principally because it can be solving many equations in a truly simultaneous manner; whereas a digital computer can be working only on one equation at a time, giving the appearance of simultaneity by interlacing the equations. On the other hand, the possible disadvantages of analog computers, such as limited accuracy and the need to dedicate the computer to one problem, may not be significant.

HYBRID 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 outline 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 values, and other devices that will change initial conditions, together with switches to control runs, a hybrid computer can be arranged to carry out large portions of the study without human intervention.

4.5 DIGITAL ANALOG SIMULATIONS

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. The languages contain macro – instructions that carry out the action of adders, integrators, and sign changers. A program is written to link together this macro – integrators, 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.

4.6 CSSLS

Confining a digital computer to routines that represent just the functions of an analog or 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. Of course, a CSSL can easily 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 subset of standard digital-analog elements, allow the user to define special purpose elements that correspond to operations that are particularly important in specific of applications.

In extending beyond that 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, move the orientation toward linear differential equations, which characterizes analog methods. A user group has published a general specification of the requirements of a CSSL. Several implementations have been produced.

4.7 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 phenomenon. 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, the furnace will be turned on or off, so that information is being feedback from the output to the input. In this case, there are only to states, either the furnace is on or off.

An example of a feedback system in which there is continuous control is the aircraft system discussed in Sec.1-1 and illustrated in Fig. 1-1. Here, the input is a desired aircraft heading and the output is the actual heading. The gyroscope of the autopilot is able to detect the difference between the two headings. A feedback is established by using the difference to operate the control surfaces, since change of heading will then affect the signal being used to control the heading. The difference the between the desired heading θ_i and actual heading θ_0 is called the error signal, since it is a measure of the extent to which the system deviates from desired condition. It is denoted by .

Suppose the control surface angle is made directly proportional to the error signal. The force changing the heading is then proportional to the error and, consequently, it diminishes as the aircraft approaches the correct heading. Consider what happens when the aircraft is suddenly asked to change to a new heading θ_i. The subsequent changes are illustrated in fig. 4-6. The upper curves represent the control surface angle and the lower curves represent the aircraft heading.

Consider first the solid curves of fig. 4-6.upon receipt of a signal to change direction, the error signal, and therefore the control surface angle, suddenly takes a non-zero value. The aircraft heading, shown in the lower curves of fig.4-6, responds to the control surface by moving toward the new heading but, because of inertia, it takes time to respond. As the aircraft turns, the control surface angle decreases so that less force is applied as the aircraft approaches the required heading. Ultimately, when the aircraft reaches the required heading, the control surface angle will be zero but the inertia of the aircraft can carry it beyond the desired heading. As a result, the control surface turns in the opposite directions in order to bring the aircraft back from its overshoot. The correction from the overshoot produces an under-shoot and the motion follows a series of oscillations of decreasing amplitude as illustrated in fig. 4-6.

Suppose the system is changed so that for the same size error signal there is a larger control surface angle. The dotted curves of Fig. 4-6 represent this case. For the same conditions, the restoring force is larger and the aircraft responds more rapidly but, correspondingly, the initial overshoot will be larger. The aircraft will oscillate more widely and rapidly, as is illustrated in Fig. 4-6. The feedback loop in the second case is said to have greater amplification, since the same error produces a larger correction force. Under certain conditions, it is possible for the amplification to be so great that the initial overshoot exceeds the initial error; in which case, the undershoot from the correction becomes even larger and the system becomes unstable because of ever-increasing oscillations

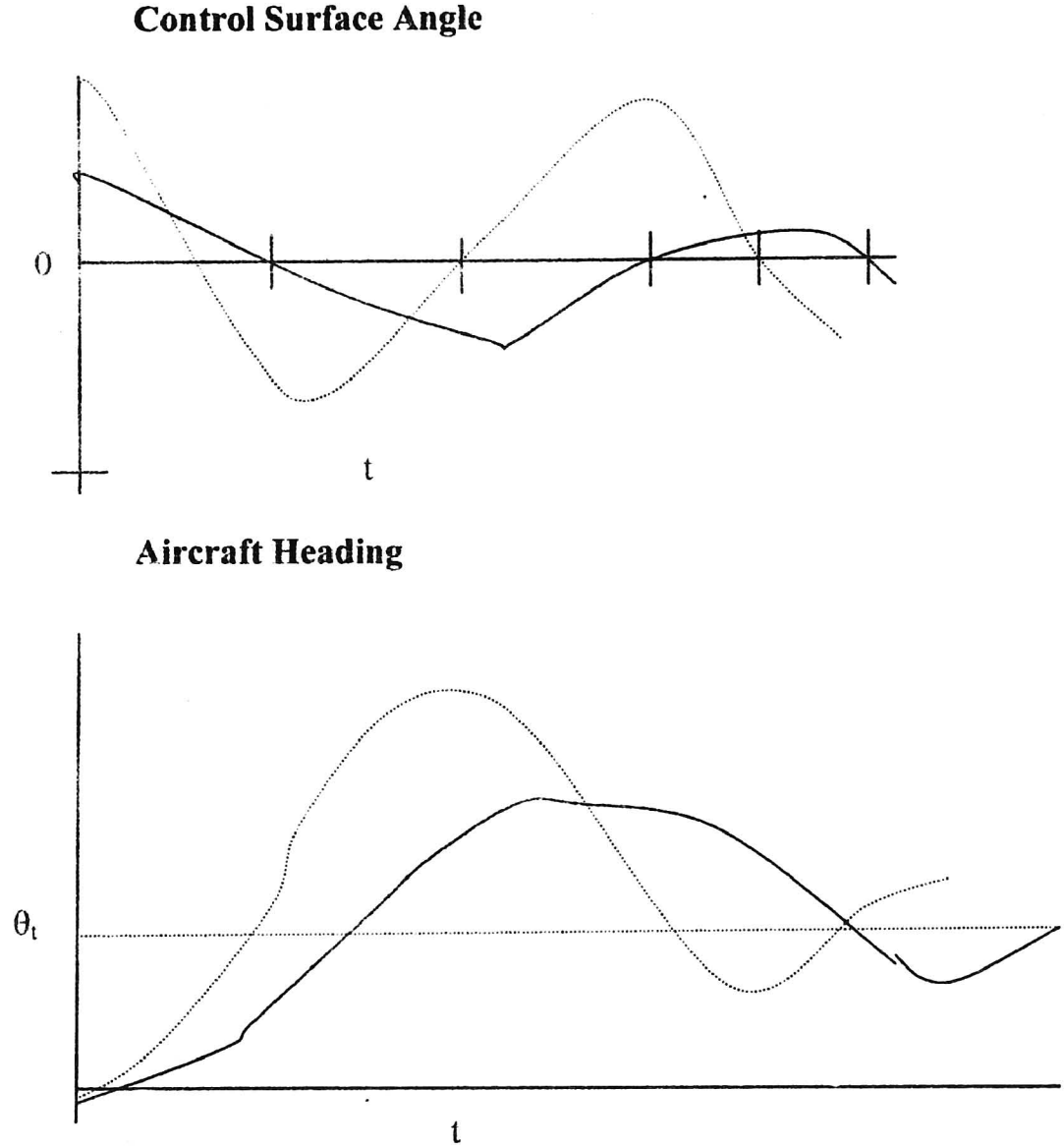


Figure 4-6. Aircraft response to autopilot system.

The feedback in the autopilot is said to be negative feedback. The more the system output deviated from the descried value the stronger is the force to derive it back.

Generally, negative feedback is a stabilizing influence, although, as we have just seen, it can become strong enough to be a destabilizing influence. There are occasions when a system experiences positive feedback, in which case the force tends to increase the deviation. Any prolonged positive feedback will; make a system unstable.

Many simulation studies of continuous systems are concerned primarily with the study of servomechanisms, which is the general name given to devices that rely upon feedback for their operation. The field of control theory provides the theoretical background with which to design such systems, but simulation is extensively used to carry out detailed studies. As an example, we will simulate the aircraft under the control of the autopilot.

4.8 INTERACTIVE SYSTEMS

While all system simulation, as we have defined it, is concerned with the response of a system over time, the observation of the response over time is particularly important for continuous system simulation. Where as the output of a discrete system simulation is likely to be a specific measure, such as a mean waiting time, the result of a continuous system simulation is likely to be judged by such factors as whether the system is stable or whether its output oscillates. Visual inspection of the results as they develop is therefore very valuable. Most CSSLs provid the ability to display the outputs on a screen, or a plotter. In particular, CSMP III provides this capability.

Given this capability, a user can judge whether the output is developing satisfactorily, and if not, interrupt a simulation run to try for better condition. Most graphic terminals serve as both input and output devices by providing a keyboard for entering data, and, perhaps, a light pen, which can be used to identify a particular output to the computer. It is a simple matter, therefore, to instruct the computer to take another case.

4.9 REAL TIME SIMULATION

We illustrated in Sec. 4-2 the general process by which a mathematical model of an engineering system in constructed. It involves understanding physical or chemical laws, and implies a number of experiments or measurements to derive the coefficients of the model. This can be particularly time-consuming if the model is not being simplified by assuming linearly.

It must have occurred to the reader that this detailed, preliminary work could be avoided if an actual device can be used, rather than constructing a model. This approach is, in fact, followed when using real-time simulation. With this technique, actual devices, which are part of a system, are used in conjunction with either a digital or hybrid computer, providing a simulation of the parts of the system that do not exist or that cannot conveniently be used in an experiment. Real-time simulation will often involve interaction with a human being, thereby avoiding the need to design and validate a model of human behavior.

Along with the technological developments needed for hybrid simulation, real –time simulation requires computers that can operate in real-time; this means that they must be able to respond immediately to signals sent from the physical devices, and sent out signals at specific points in time. These capabilities have been available for many years, and so real-time simulation is used in many areas. The aerospace industry in particular makes extensive use of real-time simulation.

There are devices called simulators, whose main function is to provide human beings with a substitute for some environment or situation, for example, devices for training pilots by giving them the impression they are at the controls of an aircraft. A well-known example was used to train astronauts. They were suspended in spring harnesses, which reduced their apparent weight to that which would occur on the moon surface. Since the main purpose of such devices is to duplicate some sort of sensory stimulus, they cannot properly be classified as computers.

5. SYSTEM DYNAMICS

5.1 EXPONENTIAL MODELS

Control theory is concerned with understanding how the response of a system can be changed by modifying the signals occurring in the system. Maintaining stability and controlling oscillations are prime concerns.

There are many examples outside the field of engineering in which some from of instability or oscillation can occur. Business cycles cause fluctuations in the general level of economic activity. In individual industries there are similar cycles where prices and supplies fluctuate or temporarily appear to go out of control. There are also many examples in the field of biology of oscillations, or explosion, in populations. It seems natural to speculate on whether these phenomena can be controlled. Clearly, the precision obtained in engineering cannot be duplicated in economic or social systems. Nevertheless, insight gained from control theory can be used to analyze the systems and, perhaps, suggest changes that will improve performance.

Initial studies of this nature dealt with industrial systems, and were called industrial dynamic studies. The dramatic changes in urban centers that began to occur in the 1950s led to Urban Dynamics studies. More recently, there has been great concern about many social and environmental problems, which, it can be seen, are world -wide in their scope. Study of these problems has been the subject of World Dynamics. Since there are major differences in the techniques used in these different areas, the general name System Dynamics has come to be used.

The principal concern of a System Dynamics study is to understand the forces operating in a system in order to determine their influence on the Stability or growth of the system. The output of the study will, it is hoped, suggest some reorganization, or change in policy, that can solve an existing problem or guide developments away from potentially dangerous directions. It is not usually expected that a System Dynamics study will produce specific numbers for redesigning a system, as occurs with engineering systems. Correspondingly, many of the coefficients in the models of System Dynamics studies consist of estimates or best guesses, particularly since the models must sometimes reduce such qualitative factors as personal performances, or social tension, to quantitative form. Nevertheless, the lack of precision that may have to be tolerated does not destroy the value of the study. The model can establish the relative effectiveness of different policies under the same assumptions, or mark out ranges of values that can be expected to produce a given type of output.

EXPONENTIAL GROWTH MODELS

Growth implies a rate of change. Consequently, mathematical models describing growth involve differential equations. Consider, for example, the growth of a capital fund that is earning compound interest. If the growth rate coefficient is k (i.e., 100k% interest rate), then the rate at which the fund grows is k times the current size of the fund. Expressed mathematically, where x is the current size of the fund.

$$\dot{X} = kx$$
$$x = x_0 \quad \text{at} \quad t = 0$$

This is a first-order differential equation whose solution is the exponential function. The solution, in terms of the mathematical constant e, which has the approximate value 2.72, is

$$X = x_0 e^{kt}$$

Fig.5-1 plots x for various values of k and an initial value of 1. It can be seen that the fund grows indefinitely, whatever (positive) value of k is used, and it grows faster with greater values of k. Looking at the curve for k =0.2, and picking the point x=2, the corresponding slope at the point A has a certain value. Later, at the point B, where x has become twice its value at A, the slope has become twice as great as at A. Since the slope is measured as the first order differential coefficient, this fact is simply the reflection of the law defining exponential growth: the growth rate is directly proportional to the current level.

Another way of describing the exponential function is to say that logarithm of the variable increases linearly with time. To test whether any particular set of data represents exponential growth, the logarithms of the data should be plotted against time. If the data then appear to fall on a straight line, the growth exponential, and the slope of the straight line will be greater for larger growth rate coefficients.

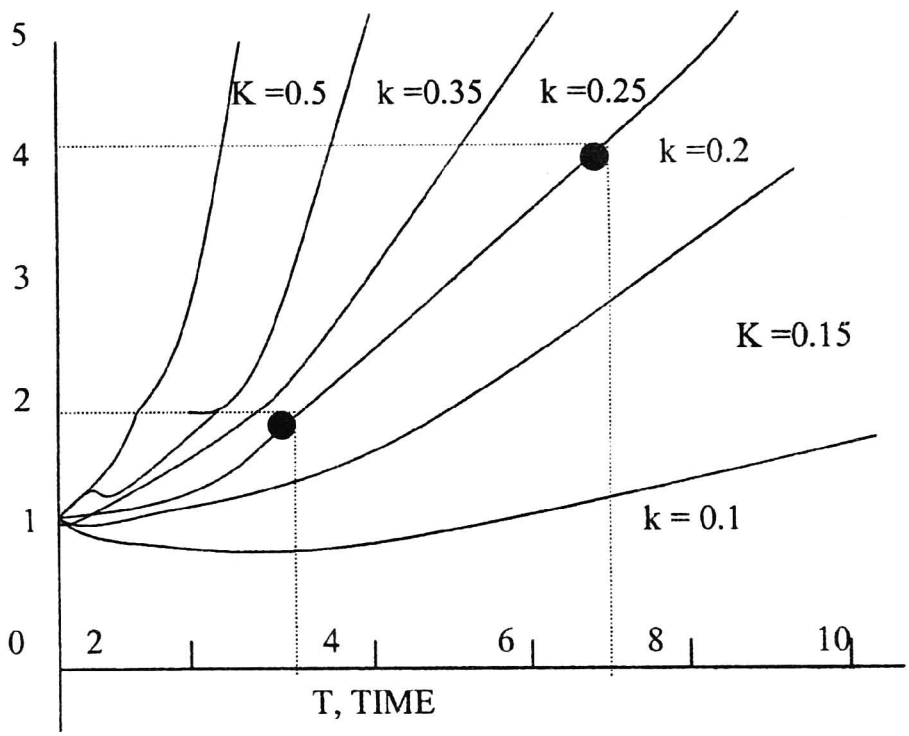


Figure 5-1. Exponential growth curves.

Alternatively, the data can be plotted on semi-logarithmic graph paper where the horizontal lines are placed at logarithmic intervals. Plotting data on such paper is equivalent to taking the logarithm of the data and then plotting on normal linear graph paper. For example, Fig 5-2 shows the gross national product figures for several countries plotted against year on semi-logarithmic paper. The point fall reasonably well on straight lines, indicating exponential growth rates.

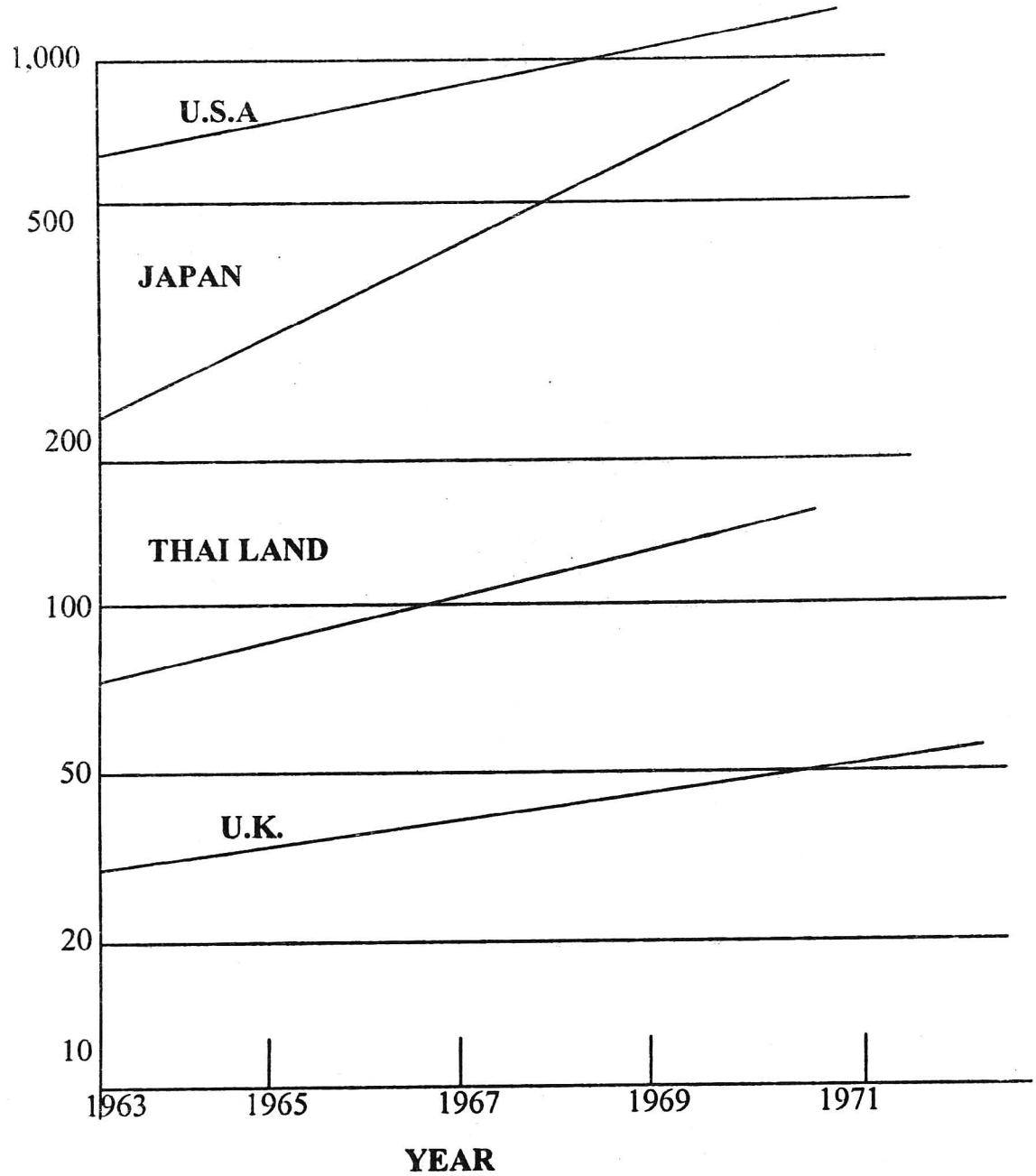


Figure 5.2 Growth of gross national products

The growth rate coefficient can be estimated by picking two points of the straight line that best fits the data, and taking the (natural) logarithm of the ratio of the values. If the points are x_1 at t_1 and x_2 at t_2 ($t_2 > t_1$), the result is

$$\ln \frac{X_2}{X_1} = (t_2 - t_1)\, k$$

From which it is possible to derive K. In terms of the more familiar logarithms to the base 10, the corresponding result is

$$\text{Log} \frac{X_2}{X_1} = 0.434(t_2 - t_1)\, k$$

For example, looking at the figures for the U.K., in Table 5-1, the values foe 1965 and 1971 are 35.3 and 55.6, respectively. The radio is 1.575, and, since the logarithm to the base 10of this number is 0.1973, the growth rate coefficient is 0.085, or 8 ½ %. [2]

Sometimes the coefficient K is expressed in the form 1/T, so that

$$K = \frac{1}{T}$$

The solution for the exponential growth model then takes the form

$$X = x_o e^{t/T}$$

TABLE 5-1 NATIONAL GROSS DOMESTIC PRODUCTS

YEAR	UK	USA	THAILAND	JAPAN
1963	30.2	596	68.1	245
1965	35.3	692	84.3	321
1966	37.7	759	101.4	369
1967	39.8	804	108.3	437
1968	42.9	863	116.7	518
1969	45.7	929	128.6	605
1970	49.9	983	135.9	712
1971	55.6	1060	145.3	794
1972	61.1	1159	160.2	907

Figures are in units of milliards (10^9) of the countries' monetary units, except For Japan's, which are in hundreds of milliards yen.

The constant T is said to be a time constant since it provides a measure of how rapidly the variable x grows. For example, when t equals T, the variable is exactly e times its initial value x_o. If T is small, say 2, x reaches this level after 20 time units. The level of e times the initial value has only been chosen as a convenient example. Whatever level is chosen as a basis of comparison, the ratio of the time constants of the time different exponential growth models will give the relative times that the two systems will take to reach that level.

The inverse relationship between K, the growth rate coefficient, and T, the time constant, means that a large coefficient is associated with a small time constant and therefore, a more rapid rate of increasing. For example, a growth rate of 0.05, or 5%, will double the size of a population in a little under 14 year, while a growth rate of 10% will do the same in just under 7 years. Doubling the growth rate again, to 20%, will reduce the time to 3 ½ years. We can say the values 14, 7, and 3 ½ are the time constants of the three cases.

Another model, closely related to the exponential growth model, is a model in which a variable decays from some initial value, x_o, at a rate proportional to the current value. The model can, in fact, be interpreted as a negative growth model. The equation for the model is

$$\dot{X} = -kx$$

$$x = x_o \quad \text{at } t = 0$$

The solution is

$$X = x_o e^{-kt}$$

The response is shown in Fif.5-3 for various values of k. As with the exponential growth model, the constant k is sometimes expressed in the form 1/T. The characteristic of the model is that the level, x, is divided by a constant factor for a given interval of time. In the interval of T time units, the level is divided by e. Since e is approximately 2.72, the level is reduced by a factor of 0.37. Each successive interval of T reduces the level by the same factor. Again, there is nothing significant about the value e, but comparing values of T for different models measures the relative times they will take to decay by a given fraction.

An example of a decay model is an again population, which being diminished by deaths or breakdowns. Another example is the manner in which radioactive material decays.

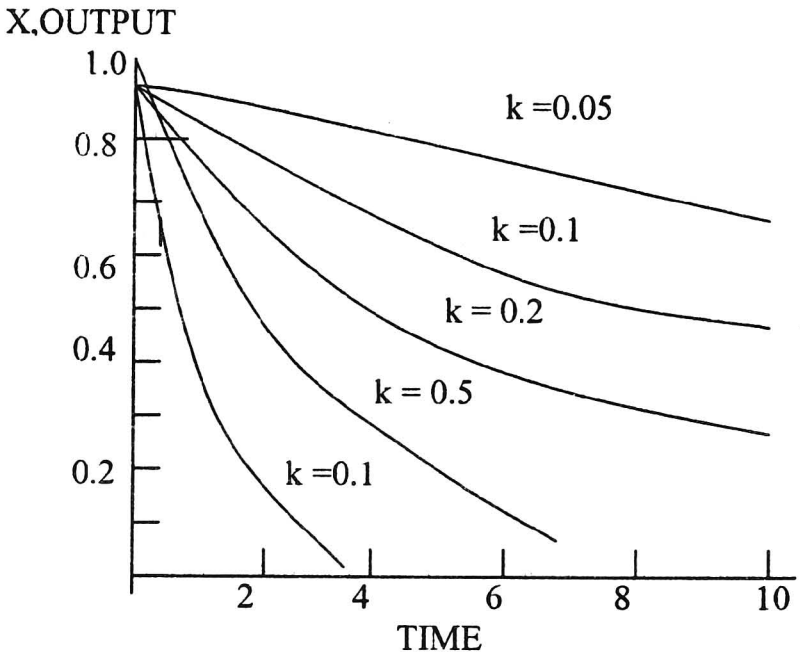


Figure 5-3. Exponential decay curves.

A market model attempts to describe how many items of a product will be sold. In practice, there is limit to the market, if we ignore replacement sales. The exponential growth model, therefore, is not a satisfactory market growth model because it shows an unlimited growth.

The exponential growth model assumes the rate of growth is proportional to the existing level. A more realistic assumption for a market model is that the growth rate is proportional to the number of people who have not yet bought the product. Suppose the market is limited to some maximum value, **x:** the number of people who might buy the product, in our example. Let x be the number of people who have bought the product. Then the number who have not is X x, and if we use k as the coefficient of proportionality, the equation for the model is

$$\dot{X} = k\,(X - x)$$
$$x = 0 \quad \text{at } t = 0$$

and the solution is

$$x = X(1 - e^{-kt})$$

Figure 5-4 plots the solution for a number of values of k. This type of curve is sometimes referred to as a modified exponential curve. As can be seen, the maximum slope occurs at the origin and the slope steadily decreases as time increases. As a result, the curve approaches the limit more slowly the closer it gets, and never actually reaches the limit. In marketing terms, the sales rate drops as the market penetration increases. The constant k plays the same role as growth rate coefficient in the exponential growth model. As k increase, the sales grow more rapidly. As with the exponential model, the constant is sometimes expressed in the form 1/T, in which case, it can be interpreted as a time constant.

$$x = X(1 - e^{-kt})$$

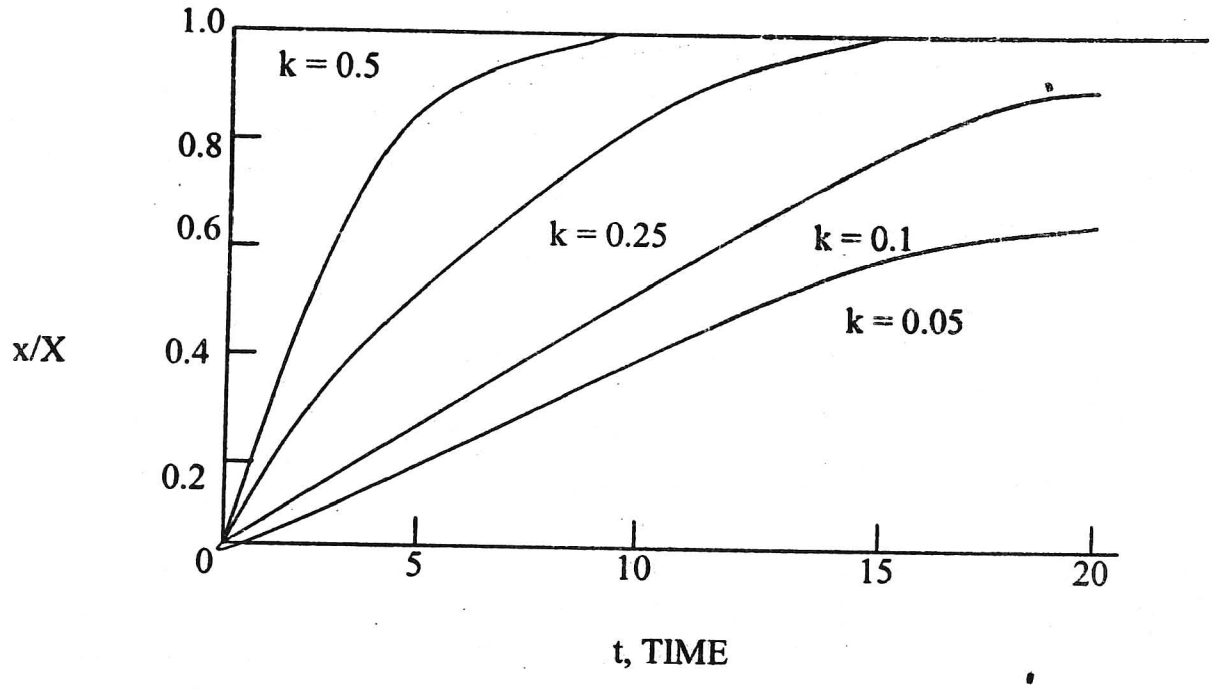


Figure 5-4. Modified exponential curves.

5.2 SYSTEM DYNAMICS DIAGRAMS.

A System Dynamic view of a system concentrates on the rates at which various quantities change and expresses the rates as continuous variables. The flow of orders from customers will be described as a certain rate of orders and the delivery of orders will similarly be represented by a rate. The difference between the two rates will be integrated to give the level of unfilled orders at any time.

The basic structure of a System Dynamics model is illustrated in Fig. 5-7. It consists of a number of reservoirs, or levels, interconnected by flow paths. The rates of flow are controlled by decision functions that depend upon conditions in the system. The levels represent the accumulation of various entities in the system, such as inventories of goods, unfilled orders, number of employees, etc. The current value of a level at any time represents the accumulated difference between the input and the output flow for that level. Rates are defined to represent the instantaneous flow to or from a level. Decision functions or, as they are also called, rate equations determine how the flow rates depend upon the levels.

A set of symbols has been established for indicating the various factors involved in a System Dynamics model and some of these are illustrated in Fig 5-7. Levels are represented b y boxes; and rates are indicated by a symbol which, as a matter of interest, is adapted from the symbol used in diagrams of hydraulic systems to represent a value. It is sometimes necessary to indicate a source or sink of entities, as is shown in Fig 5-7. For completeness, the need for a

constant in the model is indicated with a special symbol. Sometimes an auxiliary variable needs to be defined, usually by combining other variables mathematically by some relationship other than the integration implied by a rate equation and a level. Auxiliary variables are depicted by circles. Solid lines in the diagram indicate the flow of tangible objects. Dotted lines indicate causal relationships.

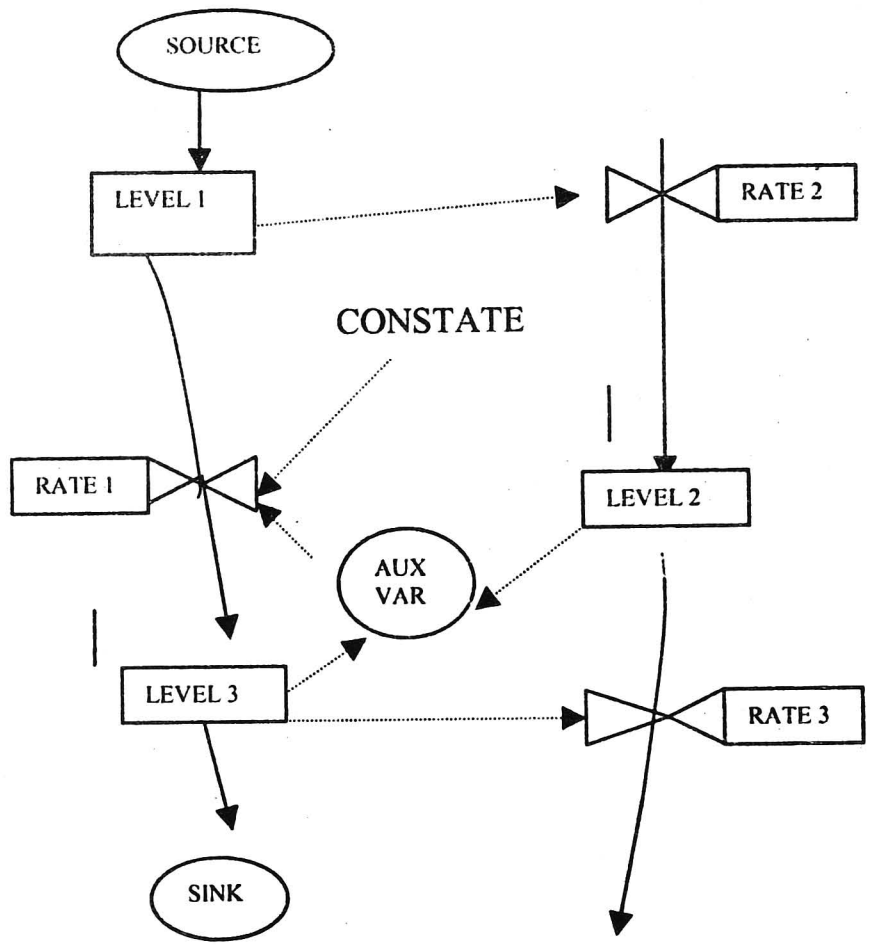


Figure 5-7. Structure of a System Dynamics model.

Levels would appear to be dimensionless quantities since they represent a count. However, the term level is also applied to some quantities that do have dimensions. For example, the rate of recording goods may depend upon the average number of orders over some period of time, which is a quantity that has the dimensions of number over time. Nevertheless, the average number of orders per week would be regarded as a level in a System dynamics model. The simplest test for determining which quantities are to be regarded as levels is to imagine the system brought to rest. Any quantity that is a rate is then automatically zero, while any quantity that maintains a magnitude should be regarded as a level.

The simple exponential growth model is concerned with a single level representing, say, a population.The model would be represented by the System Dynamic diagram (a), shown on the left of Fig 5-8. The level, x, representing the population, is being increased by a rate factor, representing the excess of the birth rate over the death rate. The rate equation is Eq. (5-1), that is = kx. The dependence of the rate on the level, x, is indicated by the dashed line of Fig. 5-8 (a). The coefficient k is shown as a constant.

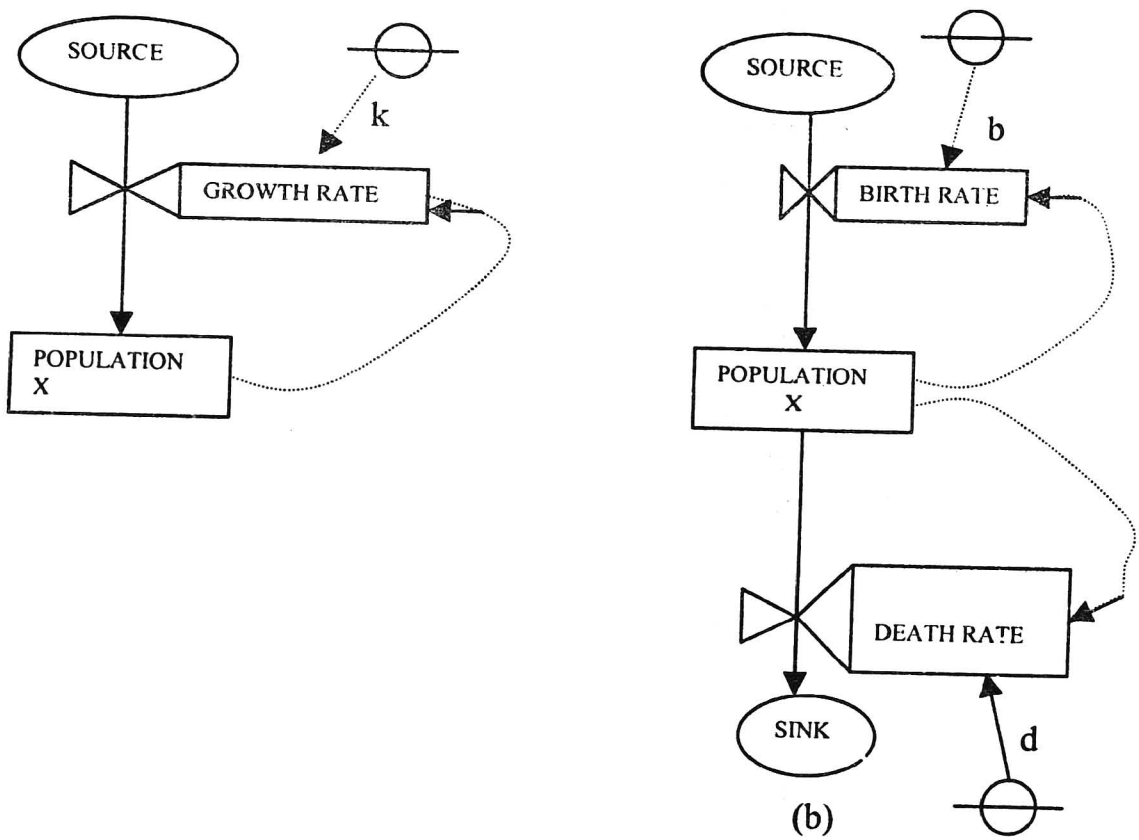


figure.5-8. System Dynamics diagrams of population growth.

An alternative way of modeling the population is to represent the births and deaths separately, as is done in diagram (b) of Fig.5-8.The level is now affected by two rate variables, one for births to increase the level, and one for deaths to decrease the level. If we assume that both rates depend upon the current level of the population, the rate equation for the births is the exponential growth model of Eq. (5-1) with the birth rate coefficient, b, and the rate equation for the deaths is the decay model of Eq. (5-2) with the coefficient set to the death rate, d.

If we consider the market models that were discussed before, we could use the two diagrams shown in Fig. 5-9. Diagram (a), on the left, represents the modified exponential model. A constant x, the market limit, has been introduced, and the rate is shown as depending upon the constant k and the current level of the market. The rate equation is Eq. (5-3).

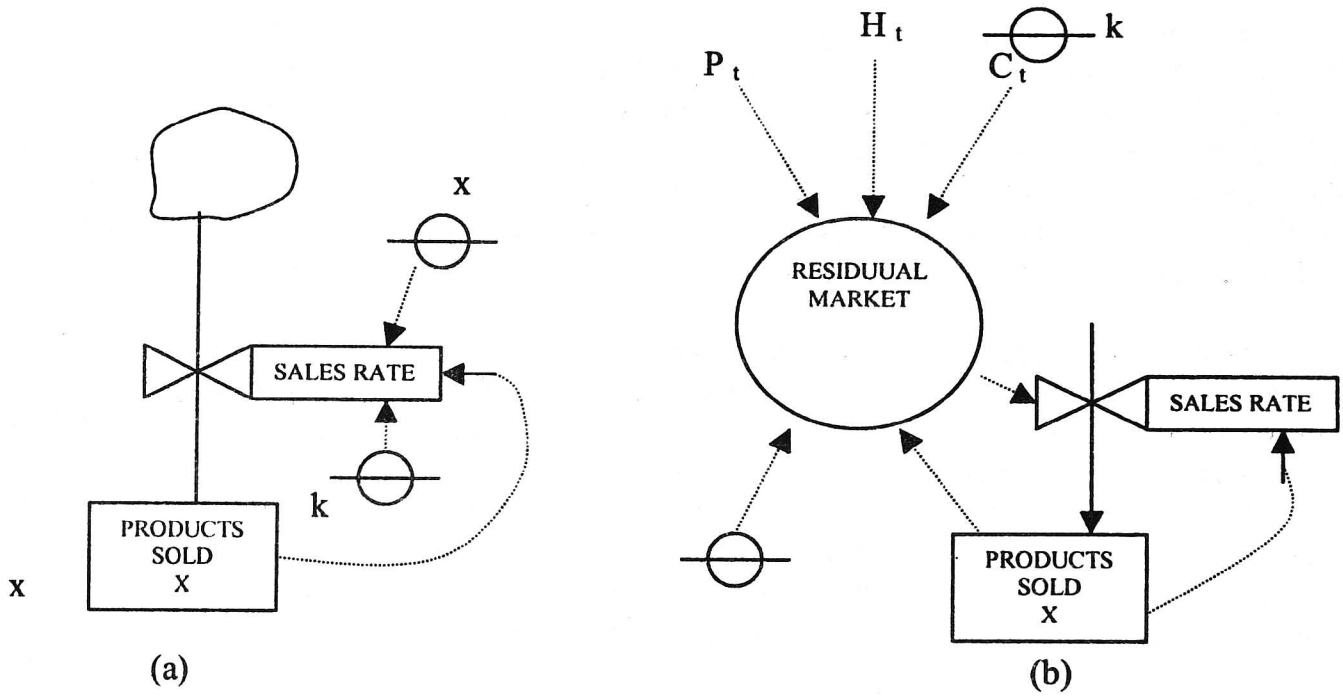


Figure. 5-9. System Dynamics diagrams of market model

The logistic model could be represented in exactly the same way, with the rate equatior being given by Eq. (5-4). Where the remaining market depends upon three economic variables P_t, H_t, and C_t, an auxiliary variable is needed to represent the residual market. Where it is implie that the three economic factors are derived from levels that are not shown. If the auxiliar variable is denoted by z, it is defined by the right-hand side of Esq. (5-7), and the rate, affecting the level is kzx.

5.3 DIAGRAMS WORLD MODELS

The most widely known example of a system Dynamics model is the world mode studied under the auspices of The Club of Rome. There have, in fact, been several similar model produced by other investigators, as a result of the Club of Rome work. The Club of Rome is ar informal, international group of individuals concerned with the future of mankind The sponsored a study based on a model of the world, a model originally proposed by Professor Jay W. Forrester. The results of the study have been reported, and their generally gloomy tone has created a great deal of discussion

The Club of Rome model investigates five of global development: population growth, the spread of industrialization, pollution, the depletion of natural resources, and the incidence of malnutrition. The main conclusions drawn are that present trends place a limit on growth, the limit is likely to be reached in the next hundred years, and creating that limit will result in a disastrous decline in both population and living conditions. It was also concluded that it is feasible to establish conditions that can lead to state of equilibrium with satisfactory living conditions for everybody. (However, establishing these conditions requires an urgent cooperation among the nations of the world to implement some major changes in present practices.)

A typical section of the model uses a population growth model similar to that of Fig. 5-8 (b), but with the birth and death rates affected by many factors. For example, the natural death rate is modified by the availability of food, the level of pollution, and the degree to which people are crowded as a result of industrialization. Each of these factors is influenced, directly or indirectly, by the existing population level, leading to a set of feedback loops, mostly of a positive type. Conflicts between industrialization and food production cause competition for capital and land, resulting in other forms of interaction between the variables of the model

Inevitably, attempts to reflect the trends and their interactions in a mathematical model require some speculation and judgment, particularly in the matter of providing specific numbers for the model. Critics of the model have questioned the assumptions and data of the model; while advocates have pointed out the persistence of the main conclusions, with relatively small changes in the scale, for a wide range of input values. Among the major topics of controversy has been the role of future technological changes, with the critics being unable to say what these will be, and the advocates finding it difficult to see feasible changes that would produce sufficient corrective action without changes in existing practices.

Some other work, based on input-output models rather than System Dynamics, however, has suggested ways in which adequate resources can be developed for future growth.

6 - DISCRETE SYSTEM SIMULATION

6.1 DISCRETE EVENTS

We demonstrated the general nature of the numerical computations involved in discrete system simulation, using a simple model describing the processing of documents. We now discuss discrete system simulation more fully, in terms directed toward executing the simulation with digital computers.

We saw that the model used in discrete system simulation has a set of numbers to represent the sate of the system, such as the numbers of Table 3-1, representing the processing of documents. A number used to represent some aspect of the system state is called a state descriptor. Some state descriptors range over values that have physical significance, such as the number representing the count of documents. Others represent conditions, such as the flag denoting whether a break in work is due.

As the simulation proceeds, the state descriptors change value. We define a discrete event as a set of circumstances that causes an instantaneous change in one or more system state descriptors. Implicit in the definition is the assumption that the system change is one that has been considered of sufficient importance to be represented in the model. In addition, it is possible that two different events occur simultaneously, or are modeled as being simultaneous, so that not all changes of state descriptors occurring simultaneous necessarily belong to a single event.

The term simultaneous, of course, refers to the occurrence of the changes in the system, and not to when the changes are made in the model-in, which, of necessity, the changes must be made sequentially. A system simulation must contain a number representing time. The

simulation proceeds by executing all the changes to the system descriptors associated with each event, as the events occur, in chronological order. The way in which events are selected for execution, particularly when there are simultaneous events, is an important aspect of programming simulations. It will be discussed in Chap.13, when discussing programming techniques used in simulation programming languages.

6.2 TIME REPRESENTATION

A number referred to as clock time records the passage of time. It is usually set to zero at the beginning of a simulation and subsequently indicates how many units of simulated time have passed since the beginning of the simulation. Unless specifically stated otherwise, the term simulation time means the indicated clock time and not the time that a computer has taken to carry out the simulation. As a rule, there is no direct connection between simulated time and the time taken to carry out the computations. The controlling factor in determining the computation time is the number of events that occur. Depending upon the nature of the system being simulated, and the detail to which it is modeled, the radio of the simulated time to the real time taken can very enormously. If a simulation were studying the detailed workings of a digital computer system, where real events are occurring in time intervals measured in fractions of microseconds, the simulation, even when carried out by a high speed digital computer, could easily take several thousand times as long as the actual system operation. On the other hand, for the simulation of an economic system, where events have been aggregated to occur once a year, a hundred years of operation could easily be performed in a few minutes of calculations.

Two basic methods exist for updating clock time. One method is to advance the clock to the time at which the next event is due to occur. The other method is to advance the clock by small (usually uniform) intervals of time and determine at each interval whether an event is due to occur at that time. The first method is referred to as event-oriented, and the second method is said to be interval-oriented. Using the event-oriented method, while continuous system simulation normally uses the interval-oriented method usually carries out discrete system simulation.

It should be pointed out, however, that no firm rule could be made about the way time is represented in simulation for discrete and continuous systems. An interval-oriented program will detect discrete changes and can b made to follow continuous changes by artificially introducing events that occur at regular time intervals. Further, the event-oriented method is not necessarily faster than the interval-oriented method for discrete systems.

Another approach to representing the passage of time has been called significant event simulation. The method is applicable to continuous systems in which there are quiescent periods. The interval between events in the event-oriented approach is, of course, a quiescent period, but it involves having the model's representation of the system activities create a notice of the event that terminates the interval. The significant event approach assumes that simple analytic functions, such as polynomials of low order, can be used to project the span of a quiescent period. The event that ends the period could be any one of several alternatives, each of which has projected span. The significant event is the one with the least span. Determining this event, by simple comparisons of the projections, allows the clock to be update by an extended period of time; achieving the same thing would otherwise have cost the effort of executing the updating of many fruitless intervals of fixed size.

An example, which is essentially the one used in Ref. (1), is an automobile traveling at constant acceleration. Its movement might result in a significant event for several reasons: the automobile might approach the preceding vehicle closer than a specific limit, it might reach the end of the road, its velocity might reach some for each of these possible events can be calculated from simple formulae. Conditions relating to all other vehicles must, of course, also be evaluated before deciding on the next significant event for the entire system.

The method includes, however, selecting the next significant event, and not just its time of occurrence. As such, it is more than a timing mechanism: it should more properly be described as an event-scanning method. It is, in fact, closely related to the activity-oriented scanning methods.

An important aspect of discrete system simulation is the generation of exogenous arrivals. It is possible that an exact sequence of arrivals has been specified for the simulation. For example, discrete system simulation is extensively used for testing the design of logic circuits, such as components of digital computers. A particular sequence of signals might as the simulation input to see if the deigns reacts as expected.

The sequence of inputs may have also been generated from observations on a system. In particular, computer system deigns, and especially the programming components of the system, are often tested with record, gathered from a running system, that is representative of the sequence of operations the computer system will have to execute. This approach has been called trace driven simulation. Program monitors can be incorporated with, or attached to, the running system to extract the data with little or no disturbance of the system operation.

When there is no interaction between the exogenous arrivals and the endogenous events of the system, it is permissible to create a sequence of arrivals in preparation for the simulation. Usually, however, the simulation proceeds by creating new arrivals, as they are needed.

The exogenous arrival of an entity is defined as an event and the arrival time of the next entity is recorded as one of the event times. When the clock time reaches this event time, the event of entering the entity in to the system is executed, and the arrival time of the following entity is immediately calculated from the inter-arrival time distribution. The term bootstrapping is often used to describe this process of making one entity create its successor. The method requires keeping only the arrival time of the next entity; it is therefore, the preferred method of generating arrivals for computer simulation programs.

The arriving entity usually needs to have some attribute values generated, in which case, attention must be paid to the time the arrival time is calculated, or they could be generated when the entity actually arrives. If there is no interaction between the attributes and the events occurring within the system, the generation may be done at either time. If however, the attributes Values depend upon the state of the system, it must be remembered that, at the time of generating the arrival time, the actual arrival is still an event in the future. The generation of the attribute values must then be deferred until the arrival event is executed. For example, telephone calls are generated. The call length and the origin of the call need to be generated. There is no interaction between the distribution of call length and the state of the system, so the call length

can be generated at the time the arrival time is decided or when the call arrives. However, a call cannot come from a line that is already busy, so the choice of origin must be left until the call arrives. To select the origin when the arrival time is decided carries the risk that some other call will have made the proposed origin busy before the call arrives.

6.3 GATHERING STATISTICS

Most simulation programming systems include a report generator to print out statistics gathered during the run. The exact statistics required from a model depend upon the study being performed, but there are certain commonly needed statistics are:

(a) Counts giving the number of entities of a particular type or the number of times some event occurred.
(b) Summary measures, such as extreme values, mean values, and standard deviations.
(c) Utilization, defined as the fraction (or percentage) of time some entity is engaged.
(d) Occupancy, defined as the fraction (or percentage) of a group of entities in use on the average.
(e) Distributions of important variables, such as queue lengths or waiting times.
(f) Transit times, defined as the time taken for an entity to move from one part of the system to some other part.

When there are stochastic effects operating in the system, all these system measures will fluctuate as a simulation proceeds, and the particular values reached at the end of the simulation are taken as estimates of the true values they are designed to measure. Deciding upon the accuracy of the estimates is a problem that will be taken up in Chap. 14. For the time being, we discuss the method used to derive the estimates.

6.4 DISCRETE SIMULATION LANGUAGES

A number of programming languages have been produced to simplify the task of writing discrete system simulation programs.

Essentially, these programs embody a language with which to describe the system, and a programming system that will establish a system image and execute a simulation algorithm. Each language is based upon a set if concepts used for describing the system. The term world-view has come to be used to describe this aspect of simulation programs. The user of the program must learn the world-view of the particular language he is using and be able to describe the system in those terms. Given such a description, the simulation programming system is able to establish a data structure that forms the system image. It will also compile, and sometimes supply, routines to represent the activities. Routines are supplied to carry out such functions as scanning events, updating the clock, gathering statistics, and maintaining events in time and priority sequence. These are needed to affect the simulation algorithm. Most programs also provide a report generator.

There is, however, a great variety in both the world-views of the languages and the degree to which the programming systems relieve the user of programming details. In general, most languages view the world in terms of entities with attributes and activities.

It is not feasible to discuss here all the simulation languages that are available. Instead, the discussion will be limited to two languages, GPSS and SIMSCRIPT. The next two chapters are devoted to GPSS, and the following two chapters are given to SIMSCRIPT. The reasons for choosing these two specific languages are that they are among the most widely used languages, and they illustrate the divergence in design considerations. They also represent two major approaches to the problem of implementing a simulation algorithm.

GPSS has been written specifically for users with little or no programming experience, while SIMSCRIPT, along with many other simulation languages, requires programming skill to the level where the user is able to program in FORTRAN or ALGOL. The simplifications of GPSS result in some loss of flexibility; so that, while SIMSCRIPT requires more programming skill, it is capable of representing more complex data structures and can execute more complex decision rules. Both GPSS and SIMSCRIPT appear to be general enough to be equally applicable to a wide variety of systems. The differences are summed up by saying that the greater programming flexibility of SIMSCRIPT means that, in more complex models, SIMSCRIPT is able to produce a more compact model that requires less storage space and, generally, will be executed more rapidly.

The general purpose Simulation System language has been developed over many years, principally by the IBM Corporation. Originally published in 1961 it has evolved through several versions to the latest version. It has been implemented on several different manufacturers' machines, and there are variations in the different implementations. With regard to the successive IBM versions of the language, all but GPSS/360and GPSS V are obsolete. Of the two current versions GPSS/360 models can operate with GPSS V, with some minor exceptions and modifications. GPSS V, however, is more powerful and has more language statements and facilities. The differences are described in Appendix 7 of (3). If these extensions are avoided, GPSS V models with run under GPSS/360.

The description given in this and the next chapter will be of GPSS V as implemented by the IPM Corporation. Some of the more significant differences between GPSS/360 and GPSS V will be noted. For simplicity, the language will be called GPSS, except where comparisons are being made.

The system to be simulated in GPSS is described as a block diagram in which the block represent the activities, and lines joining the blocks indicate the sequence in which the activities can be executed. Where there is a choice of activities, more than one line leaves a block and the condition for the choice is stated at the block.

The use of block diagrams to describe systems is, of course, very familiar. However, the form taken by a block diagram description usually depends upon the person drawing the block diagram. To base a programming language on this descriptive method, each block must be given a precise meaning. The approach taken in GPSS is to define a set of 48 specific block types, each

of which represents is a characteristic action of systems. The program user must draw a block diagram of the system using only these block types.

Each block type is given a name that is descriptive of the block action and is represented by a particular symbol. Figure 6-1 shows the symbols used for the block types that will be described in this and the next chapter. To assist the reader, the block diagrams drawn in this chapter will name the block types, although a programmer familiar with GPSS does not usually do this. Coding instructions for al the described block types are similarly brought together in table 6-1. Table 6-2 describes the control statements. Each block type has a number of data fields. As the blocks are described, the fields will be referred to as field A, B, C, and so on, reflecting the order in which they are specified.

Moving through the system being simulated are entities that depend upon the nature of the system. For example, a communication system is concerned with the movement of messages, a road transportation system with motor vehicles, a data processing system with records, and so on. In the simulation, these entities are called transactions. The sequence of events in real time is reflected in the movement if transactions form block to block in simulated time.

Transactions are created at one or more GENERATE blocks and are removed from the simulation at TERMINATE blocks. There can be many transactions simultaneously moving through the block diagram. Each transaction is always positioned at a block and most blocks can hold many transactions simultaneously. The transfer of a transaction from one block to another occurs instantaneously at a specific time or when some change of system condition occurs. A GPSS block diagram can consist of many blocks up to some limit prescribed by the program (usually set to 1,000).

An identification number called a location is given to each block, and the movement of transactions is usually from one block to the block with the next highest location. An assembly program within GPSS assigns the locations automatically so that, when a problem is coded, the blocks are listed in sequential order. Blocks that need to be identified in the programming of problems (for example, as pointes to which a transfer is to be made) are given a symbolic name. The assembly program will associate the name with the appropriate location. Symbolic names of block and other entities of the program must be from three to five non-block characters of which the first three must be letters.

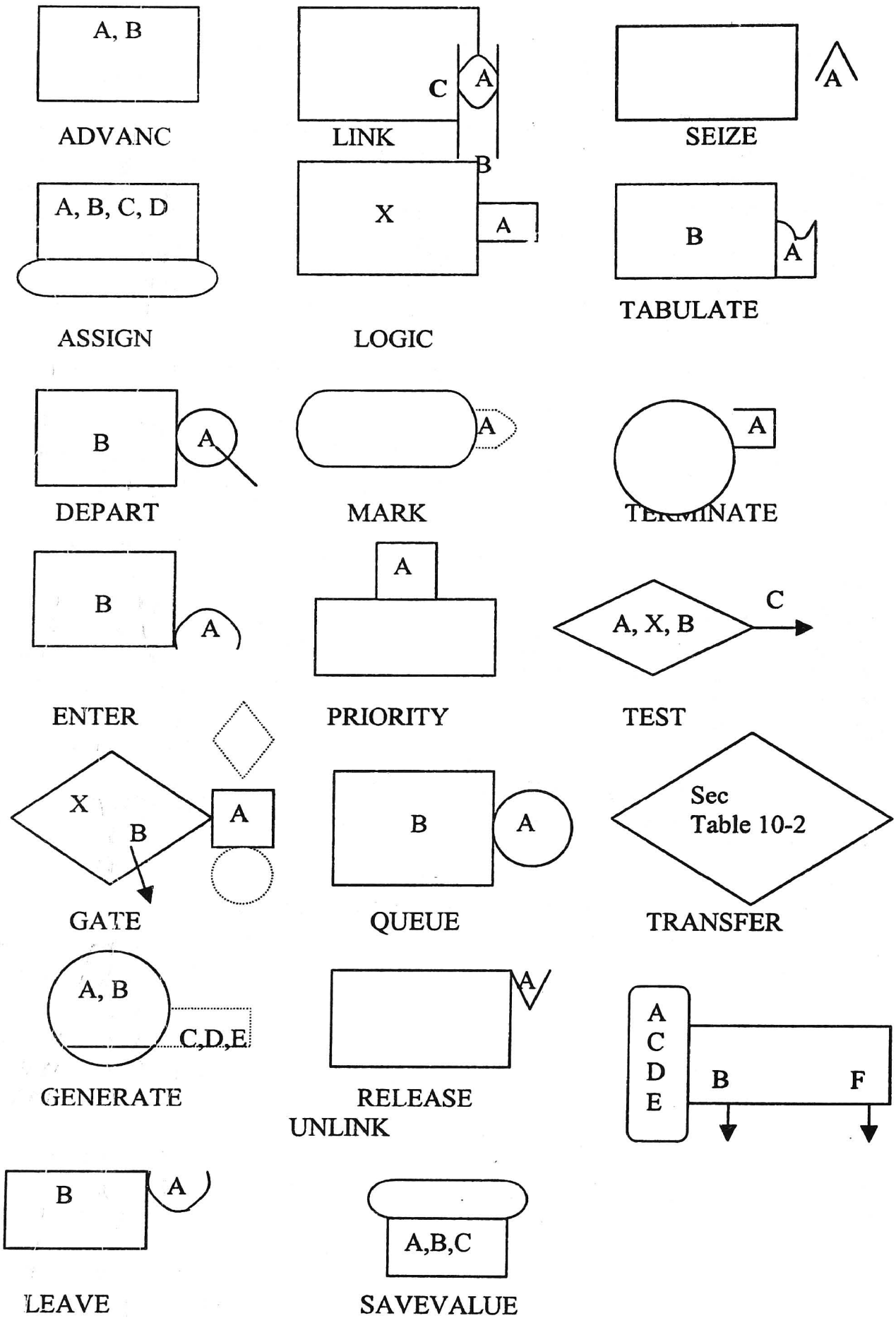


Figure 6-1. GPSS block-diagram symbols.

TABLE 6-1 GPSS Block Types ( ) indicate optional field.

Operation	A	B	C	D	E	F
Advance	Mean	Modifier				
Assign	Param. No.(±)	Source	(Funct. No.)	Param. Type		
Depart	Queue No.	(Units)				
Enter	Storage No	(Units)				
Gate†	Item No	(Next block B)				
Generate	Mean	Modifier	(Offset)	(Count)	(Priority)	(Parms.)
Leave	Storage No.	(Units)				
Link	Chain No	Order	(Next block B)			
Logic {R, S, I}	Swtich No					
Mark	(Parms No)					
Priority	Priority					
Queue	Queue No	(Units)				
Release	Facility No					
Savevalue	S.V.No	SNA				
Seize	Facility No					
Tabulate	Table No	(units)				
Terminate	(Units)					
Test†	Arg.1	Arg.2	(Next block B)			
Transfer	Select. Factor	Next block A	Next block B			
Unlink	Chain No	Next block A	Count	(Param.No.)	(Arg)	(Next Block B)

TABLE 6-2 GPSS Control Statements

Location	Operation	A	B	C	D
	CLEAR				
	END				
Function No	FUNCTION	Argument	{C, D, L} No. of Points		
	INITIAL	Entity	Value		
	JOB				
	RESET				
	←SIMULATE				
	START	Run Count	(NP)		
Storage No.	STORAGE	←Capacity			
Table No.	TABLE	←Argument	Lower limit	Interval	No.of Intervals

Clock time is represented by an integral number, with the interval of real time corresponding to a unit of time chosen by the program user. The unit of time is not specifically stated but is implied by giving all times in terms of the same unit. One block type called ADVANCE is concerned with representing the expenditure of time. The program computes an interval of time called an action time for each transaction as it enters an ADVANCE block, and the transaction remains at the block for this interval of simulated time before attempting to proceed. The only other block type that employs action time is the GENARATE block, which creates transactions. The action time at the GENARATE block controls the interval between successive arrivals of transactions.

The action time may be a fixed interval (including zero) or a random variable, and it can be made to depend upon conditions in the system in various ways. An action time is defined by giving a mean and modifier as the A and B fields for the block. If the modifier is zero, the action time is a constant equal to the mean. If the modifier is a positive number (mean), the action time is an integer random variable chosen from the range mean modifier, with equal probabilities given to each number in the range. Sometimes, this uniform distribution is an accurate representation of a random process in the system, but the principal purpose in providing this way of representing a random time is to allow for cases where randomness is known to exits but no detailed information is available on the probability distribution.

It is possible to introduce a number of functions, which are tables of numbers relating an input variable to an output variable. By specifying the modifier at an ADVANCE or GENARATE block to be function, the value of the function controls action time. The action time is derived by multiplying the mean by the value of the function. Various type of input can be used for the functions, allowing the functions to introduce a variety of relationships among variables of a system. In particular, by making the function an inverse cumulative probability distribution, and using as input a random number, which is uniformly distributed, the function can provide a stochastic variable with a particular non-uniform distribution in the manner described.

The GENARATE block normally begins creating transactions from zero time, and continues to generate them throughout the simulation. The C field, however, can be used to specify an offset time as the time when the first transaction will arrive. If the D field is used, it specifies a limit to the total number of transactions that will come from the block.

Transactions have a priority level and they can carry items of data called parameters. The E field determines the priority of the transactions at the time of creation. If it is not used, the priority is of the lowest level.

Parameters can exist in four formats. They can be signed integers of full word, half word, or byte size, or they can be signed floating-point numbers. If no specific assignment of parameter types is made, the program creates transactions with 12 half-word parameters. Any number of parameters (including zero) of any type, up to a limit of 255, can be specified by using fields F, G, G, and 1 of the GENERATE block. The symbol nix will call for n parameters of type x, where x takes the value F, H, B, or L, for full word, half word, byte size, and floating-point, respectively.

For the examples used here, there are no special requirements for parameter types. The default case of 12 half-word parameters, therefore, will be used. The C, D, and E, fields also will not be needed. In such cases, where only the A and B fields are needed, the flag attached the symbol for the GENRATER block.

The program maintains records of when each transaction in the system is due to move. It proceeds by completing all movements that are scheduled for execution at a particular instant of time and that can logically be performed. Where there is more than one transaction due to move, the program processes transactions in the order of their priority class, and on a first come, first-served basis within priority class.

Normally, a transaction spends no time at a block other that ant an ADAVNCE block. Once the program has begun moving a transaction, therefore, it continuous to move the transaction through the block diagram until one of several circumstances arises. The transaction may enter an ADVANCE block with a non-zero action time, in which case, the program will turn its attention to other transactions in the system and return to that transaction when the action time has been expended.

Secondly, the conditions in the system may be such that the action the transaction is attempting to execute by entering a block cannot be performed at the current time. The transaction is said to be blocked and it remains at the block it last entered. The program will automatically detect when the blocking condition has been removed and will start to move the transaction again at the time. A third possibility is that the transaction enters a TERMINATE block, in which case it is removed from the simulation. A fourth possibility is that a transaction may put on a chain. This concept will be explained in the next chapter.

When the program gas moved one transaction as far as it can go, it turns its attention to any other transactions due to move at the same time instant. If all such movements are complete, the program advances the clock to the time of the next most imminent event and repeats the of executing events.

6.5 MODEL OF TELEPHONE SYSTEM STUDY OF GPSS

To illustrate the use of logic switches, a GPSS model of the telephone system discussed in chap. 8 will be derived. The system is one in which a series of calls come from a number of telephone lines and the system is to connect the calls by using one of a limited number of links. Only one call can be made to any one line at a time and it is assumed that calls are lost if the called party is busy or no link is available. A logic switch represents each line whose number is the line number. The line is considered busy if the switch is set.

Each call is represented by one transaction; the unit of time chosen is 1second. It will be assumed that the distribution of arrivals is Poisson with a mean inter-arrival time of 12 seconds. The length of the calls will also be assumed to have an exponential distribution. As in the pervious examination of this system, it will assumed that each new call can come from any of the non-busy lines with equal probability, and that its destination is equally likely to be any lime other than it self.

A GENARATE block is used to create a series of transactions representing calls. The modifier at the block is the same function, number 1, used in the supermarket example. The mean of the GENARATE block is set to 12. Parameters 1 and 2 will be used to carry the origin and destination of the call. Each transaction is sent to two ASSIGN blocks to select and record the values. The source of the information is a VARIABLE statement, number 1, which will select a line at random by the method described.

The number of lines N is multiplied by a random number between 0 and 1, and the integral part plus 1 is taken to represent a choice of 1 out of N. (The value zero is not allowed for any GPSS entity.) It is not necessary to take any action to extract the integral part because (with certain exceptions) GPSS works with integral numbers. Any evaluation of a VARIABLE statement or function that results in a fractional number is converted to an integer by dropping the fractional part. The number of lines will be varied on different runs so the desired number of lines is placed in half-word save value number 1. An INITIAL statement defines the desired value at the beginning of the program. It can appear any where in the coding

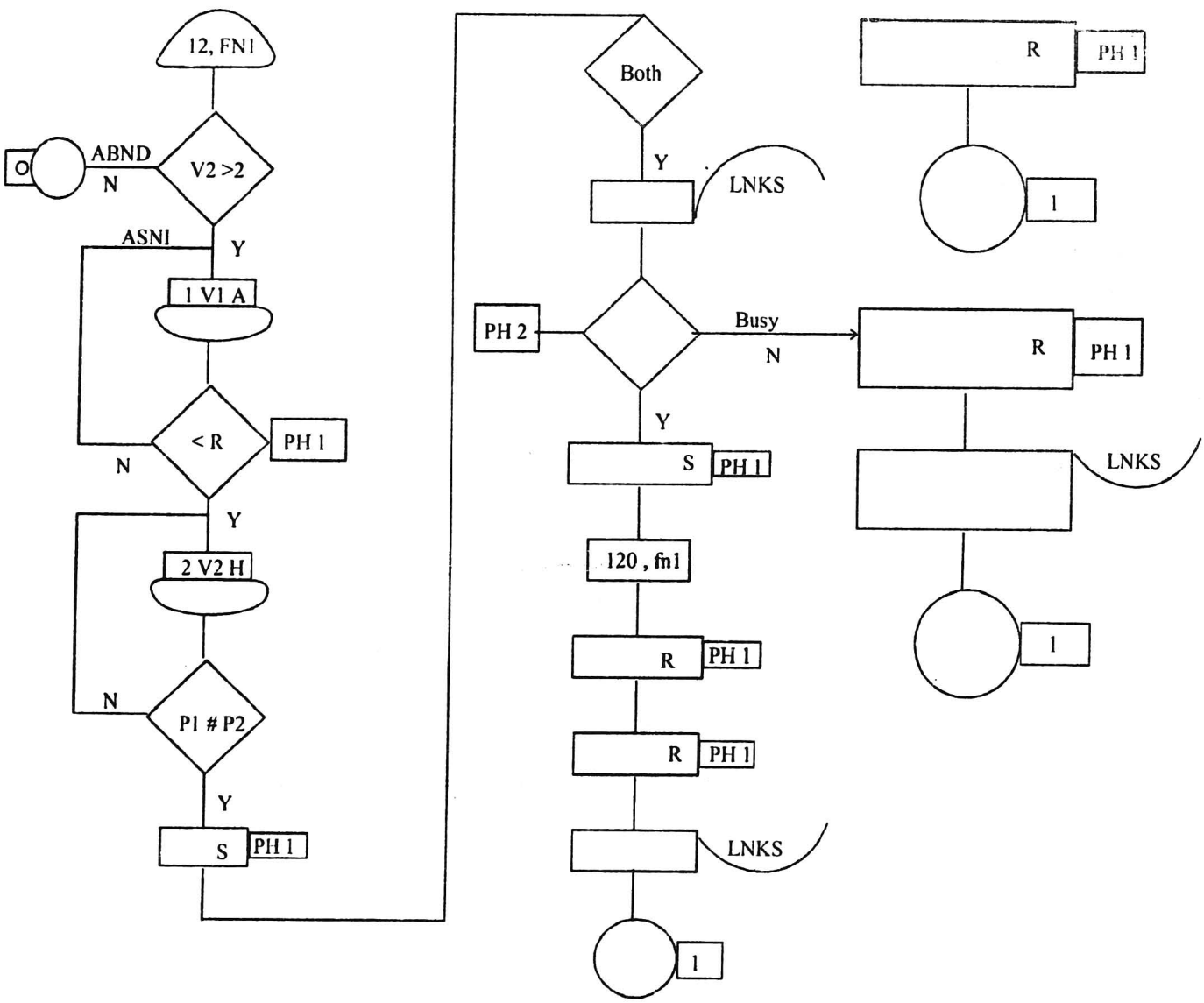


Figure 6.2Telephone system in GPSS – model 1.

- SIMULATION OF TELEPHONE SYSTEM – 1

```
*
    1      FUNCTION   RN1,C24      FUNCTION          FOR          EXPONENTIAL
DISTRIBUTION
       0. 0, 0, 0 / 0, 1, 0, 104 /  0, 2, 0, 222 / 0, 3, 0, 355 /  0,  4, 0,  509 / 0,  5, 0, 69
       0, 6, 0, 915 / 0, 7, 1. 2 /  0.75, 1.38 /  0.8, 1.6 /  0.84, 1.83 / 0.88. 2.12
       0.9,  2.3 / 0 .92 , 2.52 / 0.94, 2.81 / 0.95 , 2.99 / 0.96, 3.2 / 0.97, 3.5
       0.98, 3.9 / 0.99, 4.6 / 6, 0.995, 5.3 / 0.998, 6.2 / 0.999, 7 / 0.9997, 8
*

1              GENERATE   12, FN1              CREATE CALLS
2.             TEST G          V2, 2, ABND  TEST IF SYSTEM IS FULL
3.      ASNI   ASSIGN     1,V1, PH      PICK ORIGIN
4.             GATE LR    PH1, ASN1                  TEST FOR BUSY
                                                     PICK  DESTINATION
5       ASN2   ASSIGN     2,V1,PH                    RETRY   IF   DEST   =
6.              TEST NE   PH1,PH2,ASN2
       ORIGIN
7.             LOGIC S    PH1                        MAKE ORIGIN BUSY
8.             TRANSFER   BOTH, , BLKD                  TRY FOR LINK
9.      GETL   ENTER            LNKS                    GET LINK
10.            GATE LR    PH2, BUSY                  TEST FOR BUSY
11.            LOGIC S    PH2                        MAKE DEST. BUSY
12.            ADVANCE    120, FN1                   TALK
13.            LOGIC R    PH1                        ORIGIN HANGS UP
14.            LOGIC R    PH2                        DESTINATION HANGS UP
15.            LEAVE           LNKS                        FREE LINK
16.  TERM      TERMINATE      1
        *
17.  ABND      TERMINATE  1                          ABANDON CALL
        *
18.  BLKD      LOGIC R    PH1                        ORIGIN HANGS UP
        *
19.            TERMINATE  1                          BLOCKED CALLS
        *
20.  BUSY      LEAVE           LANKS                      FREE LINK
21.            LOGIC R    PH1                        ORIGIN HANGS UP
22.            TERMINATE  1                          BUSY CALLS
     *
      LNKS     STORAGE    10                         NO. OF LINKS
     *
        1      VARIABLE   XH1*RN1/1000+1             PICK A LINK
        2      VARIBALE   XH1 -  2*S $ LNKS – 2           COUNT   NO.OF   FREE
 LINKS
     *
               INITIAL    XH1, 50                    SET NO. OF LINKS
     *
```

```
START       10, NP          INITIALIZE
RESET
START       1000            MAIN RUN
```

FIGURE 6.3 Coding of telephone system – model 1

The following coding will place line numbers chosen at random from 50 lines in parameters numbers 1 and 2:

```
            ASSIGN      1, V1, PH
            ASSIGN      2, V1, PH
1           VARIABLE    XH1 * RN1/1000+1
            INITIAL     XH1, 50
```

The same variable statement may be used for both assignments because each reference to the VARIABLE will produce a different random number. Note that RN1 is not being used as the input to a function. It therefore is a number in the range 0 to 999 and must be reduced to the range 0 to1 by being divided by 1,000.

With this method of generating the call origin and destination, it is possible that the origin of the call is already busy. A GATE block checks whether the selected origin is busy b y using indirect addressing. If it is, the call is returned for reassignment. Should all lines be busy, this could cause an endless loop, so before assigning the origin, a check is made at a TEST block to ensure that at least two lines or not busy. The TEST block uses VARIABLE 2 to make the check and, if it finds the conditions unsatisfactory, the call is abandoned. It is also possible that the second ASSIGN block will choose the destination to be the same as the origin. This is checked at another TEST block, which compares parameter 2 with parameter 1. If they are equal, the transaction is returned to the second ASSIGN block to reassign the destination.

When a valid call has been generated, the model makes the calling line busy by setting a switch, using the SNA PH1. It then checks whether a link is available by attempting to enter the storage called LNKS whose capacity equals the number of links. If the transaction cannot enter, the call is sent to a TERMINATE block called BLKD and the call is lost after the calling line switch is reset. Transactions that get a link check whether the called party is busy, by using a GATE block.

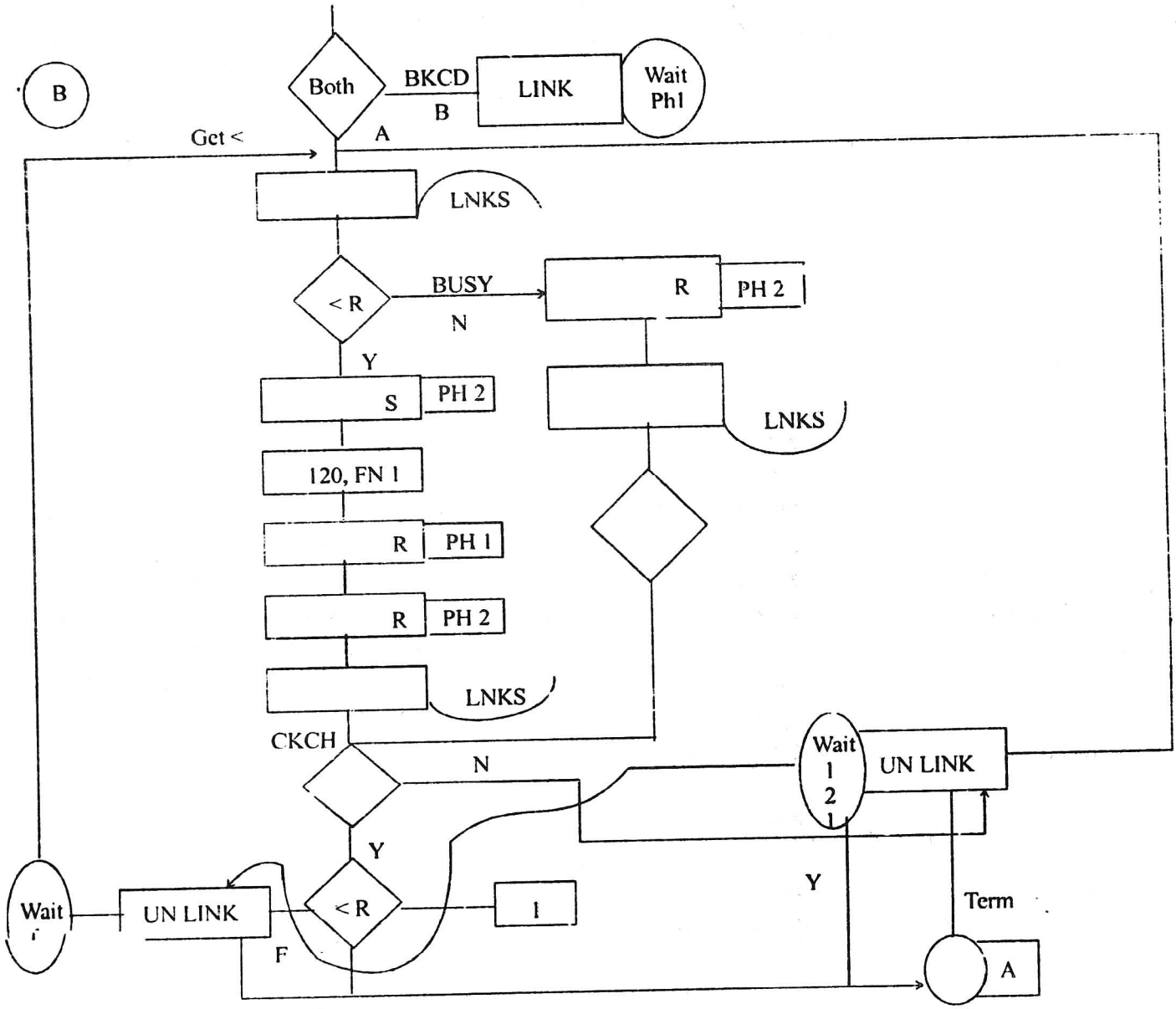
If the line is busy, the call is lost and it goes to a location BUSY where the transaction terminates after resetting the calling line switch and returning the link. Otherwise, the call is established by setting the logic switch corresponding to the destination. An ADVANCE block represents the expenditure of time during the call, using function number 1 with a mean of 120. When the transaction leaves the ADVANCE block, the call is finished and the transaction proceeds to disconnect the call by resetting both logic switches, releasing the link, and terminating.

An important requirement in a simulation language is the capability of handing sets of temporary entities, which have some common property. In GPSS, transactions that are blocked are automatically entered and removed from sets with a FIFO discipline. The program also has a way of controlling sets so that more complex queuing disciplines can be simulated.

A number of user chains are made available and a transaction is placed on a chain when it enters a LINK block. Field carries the number (or name) of the chain and field B indicates the queuing discipline. The words FIFO or LIFO result in the disciplines they name. If Pxn is used, the transactions are ordered by ascending values of parameter number n, with a FIFO rule for transactions having the same value. While on the chain, the transactions remain at the link block.

To correspond to the LINK block, there is an UNLINK block which allows a transaction (not on the chain) to remove transactions from the chain. The block names the chain in field A, and in field B gives the location to which unlinked transaction are to go. Field C says how many transactions are to be removed. The count can be an integer, the value of an SNA, or the word ALL can be used to remove all the transactions. If only these first three fields are specified, the program removes transactions from the beginning of the chain. However, removal can be made to depend upon the value of any parameter of the transactions on the chain. Field D carries the number of the parameter to be examined and field E carries the value the parameter must have for the transaction to be removed. If fields F is used, in provides a location to which the transaction entering the UNLINK block will go if it does not find a transaction on the chain that meets the conditions for removal. If field F is not used, the unlinking transaction goes to the next block, as it always does if it removes a transaction.

As an example of how these blocks are used, suppose that in the telephone system blocked calls wait for a link to become free with the following services rules. Line 1 belongs to the company president. If there is an incoming call for line 1 and line 1 free, the next free link goes to the call. Otherwise, the link goes to the call with lowest origin number.



Tl generation of transactions representing calls is the same as before so that the line of
locks. N , when the calls are blocked because they cannot enter the storage LNKS, they are
ent to a k block at BLKD> This puts them on a chain called WAIT in ascending order of the
rigin lin umber, because the LINK block has $1PH^2$ in field B. If the call is successful proceeds
s before.

N , when a call finishes, the program must check whether a call is waiting for the
rigin or stination lines that have just become free. It begins this checking at the block called
CKCH. T check must also be made by any transaction that gets a link but then finds the called
line busy. hese calls are still going to be lost. Since they release the origin line, as they are lost.
it is nece: ıry to check the waiting calls. (Note that this call might have been waiting for a link
so its cou have been holding the origin line for some time.)

The checking begins with a TEST block examining the SNA called CHSWAIT, which is the number of transactions on the chain WAIT. If there are no waiting calls, no further action is required, and the transaction is terminated. If there is a waiting call, the transaction goes to see if line number 1, the president's line, is free by checking whether logic switch number 1 is reset. If it is, the program is to connect any incoming call to the president. This means searching the chain WAIT for a transaction with parameter 2 equal to 1. The transaction goes to an UNLINK block coded as follows:

```
UNLINK          WAIT, GETL, 1,2,PH, 1,GETF
```

This tells the program to search chain WAIT and send to location GETL one transaction such that its second half-word parameter has the value 1. If the transaction fails to find such a transaction, field F instructs it to go to the location GETF. If it is successful in its search, the transaction that was in the UNLINK block goes to the next location, which is the TERMINATE block. A transaction that is unlinked takes over the link that just become free.

If the president's line is busy, or if it is free but there is no call waiting to be connected to it, service is to be offered to the waiting call with a lowest origin number. It will be recalled that when the blocked transaction where put on the chain, they where placed in ascending order of parameter number 1, which is the origin number. The first transaction on the chain, therefore, comes from the lowest origin of the waiting calls. The need is, therefore, to unlink the first transaction of the chain. This done by the transaction that tried to connect a call to the president's line but failed. It goes to the UNLINK block at GETF, which is coded as follows:

```
UNLINK          WAIT, GETL, 1
```

Figure 6-4 shows the coding for the problem in its new form. Note that variable 2 has also been changed so that the TEST block, which checks whether it is feasible to generate a new call, now takes account of the waiting calls.

Suppose letting the president have two telephones, on lines 1 and 2, modifies the system and a call to either is to get first priority. The UNLINK block could be used with another type of SNA called a Boolean variable. These variables are similar to the variables used before but, instead of combining simple SNA's, they use the conditional test phrases of the GATE and TEST blocks. They are distinguished by the operation name BVARIABLE. Each term of a Boolean variable is a single test that could be made by a GATE or TEST block. The terms are combined with the operators * and +for AND and inclusive OR, respectively. The individual terms take the value 1 or 0 according to whether the test is true or false. The values are combined by the rules of the operators. For example, consider the following coding:

```
1       BAVARIBLE           PH2'E'1*LR1+PH2'E'2*LR2

        •   SIMULATION OF TELEPHONE SYSTEM - 2
*
1
    FUNTION   RN1,24          FUNCTION      FOR      EXPONENTIAL
    DISTRIBUTION
    0. 0, 0, 0 / 0, 1, 0, 104 /  0, 2, 0, 222 / 0, 3, 0, 355 /  0,  4, 0,  509 / 0,  5, 0, 69
    0, 6, 0, 915 / 0, 7, 1. 2 /  0.75, 1.38 /  0.8, 1.6 /  0.84, 1.83 / 0.88. 2.12
    0.9,  2.3 / 0 .92 , 2.52 / 0.94, 2.81 / 0.95 , 2.99 / 0.96, 3.2 / 0.97, 3.5
    0.98, 3.9 / 0.99, 4.6 / 6, 0.995, 5.3 / 0.998, 6.2 / 0.999, 7 / 0.9997, 8
*

1           GENERATE  12, FN1              CREATE CALLS
2.          TEST G          V2, 2, ABND   TEST IF SYSTEM IS FULL
3.          ASNI  ASSIGN    1,V1, PH      PICK ORIGIN
4.          GATE LR   PH1, ASN1           TEST FOR BUSY
5   ASN2    ASSIGN    2,V1,PH             PICK  DESTINATION
6.          TEST NE   PH1,PH2,ASN2        RETRY IF DEST = ORIGIN
7.          LOGIC S   PH1                 MAKE ORIGIN BUSY
8.          TRANSFER  BOTH, , BLKD        TRY FOR LINK
9.    GETL  ENTER           LNKS          GET LINK
10.         GATE LR   PH2, BUSY           TEST FOR BUSY
11.         LOGIC S   PH2                 MAKE DEST. BUSY
12.         ADVANCE   120, FN1            TALK
13.         LOGIC R   PH1                 ORIGIN HANGS UP
14.         LOGIC R   PH2                 DESTINATION HANGS UP
15.         LEAVE     LNKS                FREE LINK
16.  CKCH   TEST G    CH$WAIT, O, TERM    TEST IF CALLS ARE WAITING
17.         GATE LR   1, GETF             SEE IF LINE ONE IS FREE
18.         UNLINK    WAIT, GETL, 1, 2PH,1, GETF    CONNECT CALL TO ONE
19.  TERM   TRERMINATE          1
20.  GETF   UNLINK    WAIT, GETL, 1       CONNECT 1 WAITING CALL
21.         TRANSFER  , TERM
    *
22.  ABND   TERMINATE  1                  ABANDON CALL
    *
23.  BLKD   LINK          WAIT, 1 PH      LINK IN ORDER OF CALL ORIGIN
    *
24.  BUSY   LEAVE             LNKS        FREE LINK
25.         LOGIC               PHI       ORIGIN HANGS UP
26.         TRANSFER          ,CKCH       GO TO TEST FOR WAITING CALLS
    *
LNKS STORAGE   10                         NO. OF LINKS
  *
1     VARIABLE  XH1*RN1/1000+1            PICK A LINK
```

```
      2     VARIBALE   XH1 - 2*S $ LNKS – 2          COUNT   NO.OF   FREE
LINKS
*
            INITIAL    XH1, 50                       SET NO. OF LINKS
*
            START          10, NP                    INITIALIZE
            RESET
            START          1000                      MAIN RUN
```

FIGURE 6.5 SHOWS CODING OF MODEL 2

The Boolean variable will be equal to 1 (or true) if parameter number 2 equals 1 and logic switch number 1 is reset, or if parameter number 2 equal 2 and logic switch number 2 is reset. Note that the TEST block conditions are placed between single quotes.

If the first UNLINK block of Fig. 10-3 has BV1 in field D (and nothing in field E), it will unlink transactions that meet the stated condition. With this form of the UNLINK block, it is not necessary to include the GATE block checking for line number 1 to be free.

Two other ways of organizing sets in GPSS will be briefly described, but the block types employed will not be described in detail. Transactions that are on a chain remain static until they are unlinked. It is sometimes necessary to identify members of a set that continue to move around the system. For example, it may be necessary to identify all the jobs in a factory for one customer, or all the cars of a given make. The GPSS program defines a number of groups for forming such sets A block type JOIN allows a transaction to make itself a number of a group and then to proceed to the next block. Another block type REMOVE allows one transaction to remove others from the group, in much the same manner as transactions are unlinked from a chain. It is possible to SCAN the group for particular members, ALTER the parameters of the group members, or EXAMINE a transaction's group membership.

A common reason for wanting to form mobile transaction sets is that they represent inter-related tasks that must be coordinated. The making of a product, for example, may involve many operations some of which can proceed independently; but some, such as an assembly, require that other operations be finished first. A block type called SPLIT allows one transaction to produce many copies which are automatically linked as a set but may proceed independently. A block type called ASSEMBLE will gather a given number of copies and merge them into one. It is also possible to synchronize the movement of copies with the use of a MATCH block, normally used in pairs. A transaction arriving at a MATCH block must wait until a copy has arrived at another MATCH block. At that time, both transactions can proceed.

6.6 SIMSCRIPT LANGUAGES

SIMSCRIPT is a very widely used language for simulating discrete systems. Beginning with two early versions, now obsolete, it has progressed through two major, intermediate versions, SIMSCRIPT II, to the current and most powerful version, SIMSCRIPT II. It has been implemented on several different manufacturers' computers. SIMSCRIPT II is generally compatible with SIMSCRIPT However, there are some important differences, some of which are dependent on the particular machine implementation. The differences are described in the preface. For brevity, however, the program will be called, simply, SIMSCRIPT.

The language can, fact, be considered as more than just a simulation language since it can be applied to general programming problems. Beginning with a simple teaching level to introduce the concepts of programming, the description adds levels corresponding to a scientific programming language, comparable to ALGOL or PL/I; a list processing language, needed for creating the data structures of a simulation and gather statistics. The description given here will be of the language. The present the simpler telephone system model that was described. Programmed as model 1in GPSS. This model is for a lost-call system and does not have any sets. Discuss the telephone system, which includes sets of calls waiting for connection.

The viewpoint to be taken by the SIMSCRIPT user corresponds essentially to that used in Sec. 1-1 to discuss the general nature of systems. That is to say, the system to be simulated is considered to consist of entities having attributes that interact with activities. The interaction causes events that change the state of the system. In describing the system, SIMSCRIPT uses the terms entities and attributes. For reasons of programming efficiency, it distinguishes between temporary and permanent entities and attributes. The former type represents entities that are created and destroyed during the execution of a simulation, while the latter represents those that remain during the run. A Special emphasis is placed on the manner in which temporary entities forms sets. The user can define sets, and facilities are provided for entering and removing entities into and from sets.

Activities are considered as extending over time with their beginning and (usually) their end being marked as events occurring instantaneously. Each type of event is described by an event routine, each of which is given a name and programmed as a separate closed subroutine. A distinction is made between endogenous (or internal) events, which result from actions with in the system, and exogenous (or external) events, which arise from actions in the system environment. A scheduling statement in some event routine causes an endogenous event. The event marking the beginning of an activity will usually schedule the event that marks the end of the activity. If the beginning or ending of an activity implies the beginning of some other activity, then the event routine marking the beginning of end of the first activity will schedule the beginning of the second.

Exogenous events require the reading of data supplied by the user. Among the data is the time at which the event is to occur. There can be many data sets representing different sets of external events. Event routines are needed to execute the changes that result when an external event becomes due for execution. Part of the automatic initialization procedure of SIMSCRIPT is to prepare the first exogenous event from each data set.

In practice external events, such as arrivals, are often generated using the "bootstrap" method that was described. Given the statistical distribution of the inter-arrival time (together with the statistical properties of any attributes that need to be represented) an endogenous event routine continuously creates one arrival from its predecessor. An exogenous event routine is essential only when specific data are needed, as in the cases where historical data are being used, or several runs are being made with exactly the same data In order to minimize variance between runs comparing different system designs.

In summary, the concepts used in SIMSCRIPT are

Entities
- Permanent
- Temporary

Sets

Event routines

Since the event routines, some means must be provided for transferring control between them. The transfer is affected by the use of event notices, which are created when it is determined, that an event is scheduled. At all times, an event notice exists for every endogenous event scheduled to occur, either at the current clock time or in the future. Each event notice records the time the event is due to occur and the event routine that is to execute the event. If the event is to involve one of the temporary entities, of which there may be many copies, the event notice will usually identify which one is involved.

The general manner in which the simulation proceeds is illustrated in Fig 11-1. The event notices are field in chronological order. When all events that can be executed at a particular time have been processed, the clock is updated to the time of the next event notice and control is passed to the event routine identified by the notice. Theses actions are automatic and do not need to be programmed. Event notices do not usually go to more than one activity in the manner of a GPSS transaction. Having activated the routine, the event notice has served its purpose and is usually destroyed. However, it is possible to save reschedule an event notice.

If the event executed by a routine results in another event, either at the current clock time or in the future, the routine must create a new event notice and file it with the other notices. For example, in the telephone system, the connection of a call implies its disconnection at a later time, so the routine responsible for connecting the call will be responsible for producing the event notice that schedules the disconnection.

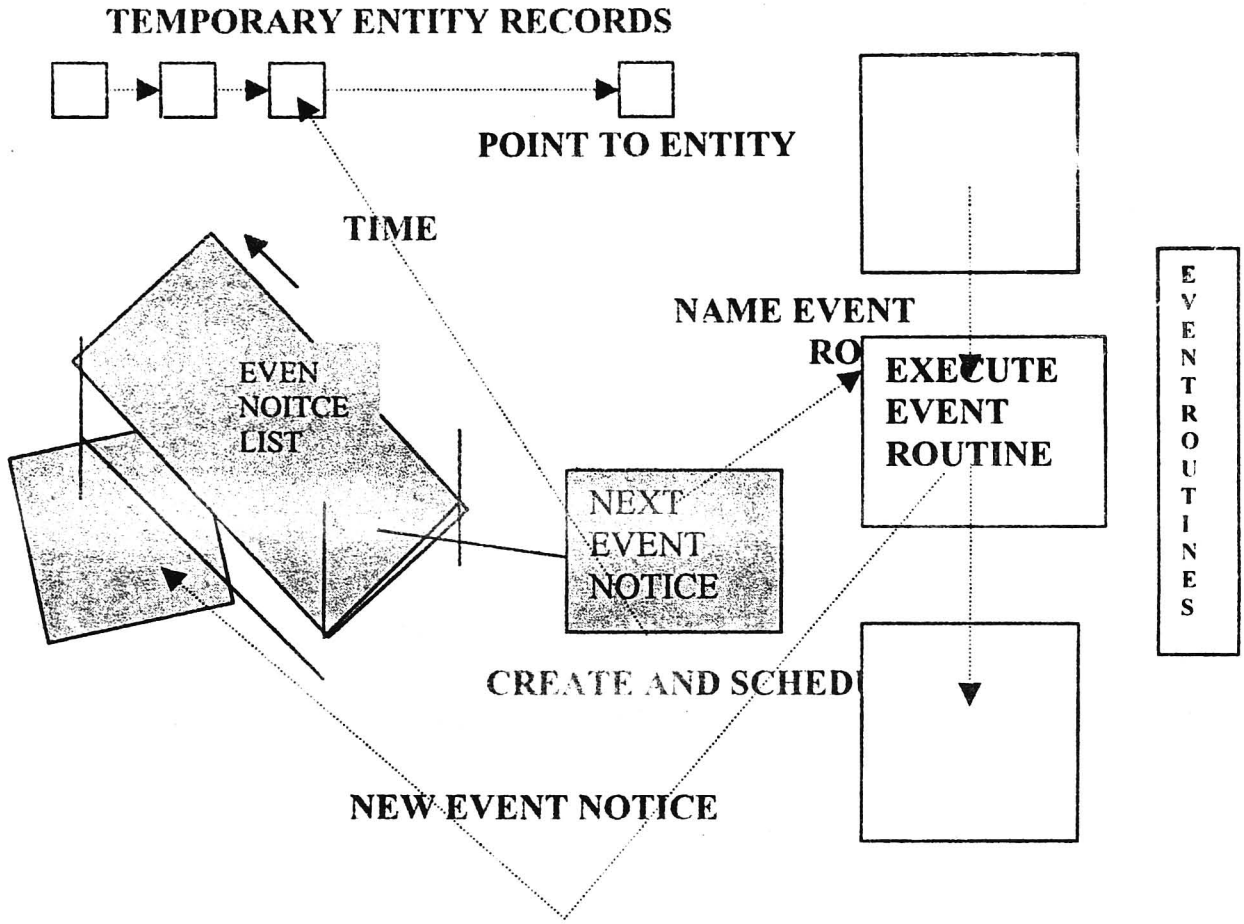


FIGURE 6.6 SIMSCRIPT execution cycle.

In the case of the exogenous events, a series of exogenous event statements are created; one for each event. The statements are similar to event notices in that they give the time the event is to occur and identify the exogenous routine to execute the event. All exogenous event statements are sorted into chronological order and the program reads them when the time for the event is due.

The user must describe all the entities by giving a name to each entity and is attributes. Names may consist of any combination of letters and digits provided there is at least one letter. (Some names are reserved for system use.) It is also permissible to use periods in a name so that compound names can be constructed. For example: if an entity type is to represent persons, it could be named PRESON; if age and education are to be attributes, they could be named age and EDUCATION, or they could be named PERSON. AGE and PERSON. EDUCATION. In practice, the specific machine implementation of the language will only recognize a certain number of initial characters (typically eight) within any name, but for documentation purposes the name can be of any length.

Labels for identifying programming statements similarly consist of any combination of etters and numbers, without the restriction that at least one be a letter. They also may make compound by using periods. Labels are identified by bring enclosed between single quotation marks.

SIMSCRIPT statements are written in a form closely resembling the English language. To enhance the similarity, there are many alternative terms or modes of expression in which will be nterpreted as equivalent statements calling for n lines of text to be printed could be written in either of the following ways:

PRINT n LINES AS FOLLOWS
PRINT n LINES THUS

The n lines of text then follow, exactly as they are to be printed. If the values of some variables are to be included in the text, the statement could read as follow s:

PRINT n LINES WITH X AND Y LIKE THIS

The symbols x and Y represent the variables or expressions to be printed. The lines of text that follow the statement will indicate with asterisks and, if necessary, decimal place would be indicated by , and a four digit real, number with one decimal place would be indicated by . In this short account of SIMSCRIPT no attempt will be made to point out all the alternative forms that can be taken by statements.

Reading the program can be made easier by including comments indicated by beginning a statement with two single quotes (not a double quote mark). Another pair of single quotes can conclude the comment, but, if the entire line is to be nothing but a comment, the trailing quotes are not necessary. Comments can, in fact, be included within a statement, in which case it is necessary to mark the end of the comment. For example, a printing statement might say

PRINT " AS THE REPORT HEADING" n LINES THUS

Blank spaces may also be freely used to make program listings easier to read, by indenting statements or spacing words.

Table 11-1 lists the SIMSCRIPT statements that will be used in programming the event routines.

TABLE 11-1 SIMSCRIPT Statements

CREATE	temporary entity or event notice (CALLED variable)
DESTROY	
	IN
SCHEDULE AN	event notice (CALLED variable) expression
	AT
LET	variable = expression
FILE	variable IN set
	FIRST
REMOVE	variable FROM set
	LAST
FOR	variable = expression.1 TO expression.2 BY expression.3
FOR EACH	permanent entity
FOR EACH	variable OF set
WITH	expression.1 compression.2
OR	
AND	
UNTIL	
IF	expression.1 comparison expression.2, statement
	IS
IF	set EMPTY, statement
	IS NOT
GO TO	statement labels
DO	
LOOP	
FIND THE FIRST CASE, IF NONE	statement ELSE

A SIMSCRIPT program begins with a preamble section that defines the system structure and establishes the conditions under which the simulation will be run. Variables can be real, integer, or alphabetical. The size depends upon the particular machine implementation and on whether single or double precision is chosen. It is also possible to pack variables so that more than one is placed in one word. In addition, it is possible to process character strings, in what is called the TEXT mode of operation. If no specific declaration is made, the program operators with single precision in a real mode. The statement

NORMALLY, MODE IS INTEGER (or ALPHA)

Changes the mode. Whichever mode is established as normal, individual variables can be defined as being in another mode.

Names must be given to every entity and its attributes. Also, the simulation will need to define certain words or tables for each purpose as carrying out computations, gathering statistics, or holding initialization values. If these are to be global variables (that is, available to all routines) they are called system variables. Those that are to be in the normal mode of the program can be listed after the statement

THE SYSTEM HAS

Items in this are any other SIMSCRIPT list can be separated by commas or the word AND (or both). If the mode is to be other than the normal mode, a DEFINE statement lists the variables and declares their mode, as follows:

DEFINE name .1 name.2... AS INTEGER VARIABLES

If the DEFINE statement does not mention the mode, the variables are taken to be in the normal mode.

Events are represented by individual routines. Within each of these event routines it is possible to use local variables, accessible only to that routine. These must not be defined in the preamble. They do not, in fact, have to be specifically defined in the event routine, unless their mode differs from the normal mode, in which case the DEFINE statement is used within the event routine.

The permanent entities, temporary entities, and the event notices are defined in lists, following the statements PERMANENT ENTITIES, TEMPORARY ENTITIES, and EVENT NOTICES, respectively. Following the opening statement is a line for each entity, taking the following form:

EVERY ENTITY HAS AN attr.1 AND AN attr.2...

Where entity is the name of the entity and attr.*i* is the name of its *i*th attribute. There can be any number of attributes. If any SIMSCRIT statement needs to go beyond 80 characters it can be extended to another line.

Event notices have the same definition format as temporary entities, although a number of words within the record are automatically reserved for use by the program. The event notices can also carry data, supplied by the user, just like the attributes of temporary entities. If they are needed, the attributes are defined with a list of EVERY statement following an EVENT NOTICES statement, just as for permanent or temporary entities. Often there is no need for data other than the automatically defined system data. In that case, definition of the event notices can be simplified by listing the names of those not needing user-defined attributes, in a sequence, after the statement

EVENT NOTICES INCLUDE...

If any array or multidimensional table is needed, a DEFINE statement, giving the name of the entity, defines the dimensions, in addition to declaring the format, if that is necessary. For example:

DEFINE table AS AN INTEGER, n-DIMENSIONAL VARIABLE

Where table is the table name, and n is the number of dimensions (for which there is no specific limit.)

There are other controls that can be exercised by statements in the preamble of a SIMSCRIPT program, such as whether single or double precision arithmetic is to be used. One control that is often needed is to redefine the name of the time unit. If no specification is made, units of days, hours, and minutes are used by following any specification of a time with one of the words DAYS, HOURS, or MINUTES.

To illustrate the use of SIMSCRIPT, the telephone system that was described and later programmed in GPSS, will be programmed as a SIMSCRIPT model. The program will be for a last call system, in which calls that are unable to make a connection immediately are abandoned.

The telephone lines are obviously entities, and each needs and attribute indicating whether the line is busy or not. The name TLINE will be used to represent telephone lines, and the attribute will be called STATE. It is an integer variable, and, in fact, takes only the values 0 or 1, to represent a free or busy state. Calls are to be represented by temporary entities, each with two attributes for carrying the origin and destination.

As in the previous implementations of this model, he links will be not being represented individually. It is not necessary, therefore, to define a permanent entity for their representation. Instead, two system variables will be needed to carry the maximum number of links and the number currently in use. The maximum number will be initialized from data read into the program. So also will be the mean inter-arrival time, the mean call length, and the time the simulation is to run, and so system variables need to be defined for these quantities. Other system variables will be used to collect statistics on how many calls are processed, and how many are abandoned because they are blocked or find the called party busy.

The preamble for model 1 of the telephone system is shown in Fig. 11-2. The figure actually shows part of the printed output of a compilation, which lists the input. The line numbers shown at the left in the figure are placed there by the compilation: the user does not enter them.

```
1    " TELEPHONE SYSTEM – MODEL 1
2    PREMABLE
3    NORMALLLY, MODE IS INTEGER
4    EVENT NOTICES INCLUDE ARRIVAL AND CLOSING
5    EVERY DISCONNECT HAS A DIS.CALL
6    TEMPORARY ENTITIES
7    EVERY CALL MAS AN ORIGIN AND A DESTINATION
8    DEFINE LINKS.IN.USE, MAX. LINKS, BLOCKED, BUSY, FINISHED
9    AND STOP. TIME AS VARIABLES
10 DEFINE INTER. ARRIVAL. TIME AND MEAN. LENGTH AS REAL
VARIABLES
11   PERMANENT ENTITIES
12   EVERY TLINE HAS A STATE
13   DEFINE SECS TO MEAN / 60 MINUTS
14   END
```

Figure 6-2 Telephone System – model 1,preamble.

Line 1 is simply a comment. Because it contains nothing but a comment, it is not essential to use the two quote marks at the end of the comment. Only the inter- arrival time and the mean call length need to be specified as real variables, so it is convenient to define the normal mode as being integer. The statement of line 3 does that.

As will be explained shortly, three event routines will be needed; therefore, there must also be three event notices. Line 4 and 5 define the event notices. By convention the names of the event notices must be same as the event routines they initiate. In this case, the routines are going to be called ARRIVAL, DISCONNRCT, and CLOSING. Because the ARRIVAL and CLOSING event notices do not need attributes, their definition is included in the event notice header. The DISCONNECT routine, however, will need to know which particular call is to disconnected, so the DISCONNECT event notices needs an attribute with which to identify the call.

Lines 6 and 7 define the temporary entity representing the calls. It has two attributes for recording the origin and destination of the call. The next three lines of the listing define the system variables, some as integer (by implication) and some as real variables. Lines 8 and 9 illustrate how statements can be carried over more than one line. Lines 11 and 12 define the permanent entities representing the telephone lines. Line 13 defines a time unit of seconds, and line 14 is an obligatory END statement for indicating the end of the preamble section.

The purposes of the system variables are apartment from their names. In addition, there are many system variables automatically provided by the program. These carry names defined by the system and do not have to be specified in the preamble. In particular, every permanent entity definition implies a system variable called N.name, where name is the name given to the permanent entity by the user. The variable holds the number of permanent entities of that type, and a value must be read into the program at the time of intialization.

7. REVIEW OF PROBABILITY CONCEPTS

7.1 ARRIVAL PATTERNS AND SERVICE SYSTEMS

Most systems of interest in a simulation study contain processes in which there is a demand for service that causes congestion. There may, for example, be customers trying to check-out at a supermarket counter, work-pieces waiting for a machine to become available, ships waiting for a berth, and so on. The system can service entities at a rate in which, general, is greater than the rate at which entities arriver, but there are random fluctuations either in the rate of arrival, the rate of service, or both. As a result, there are times when more entities in the system —those being served, and those waiting for service —will be called a queue.

Congestion may be described in terms of three main characteristics. These are

(a) THE arrival pattern, which describes the statistical properties of the arrivals.
(b) THE service process, which describes how the entities are served.
(c) THE queuing discipline, which describes how the next entity to be served is selected.

The service process, in turn, is described by two main factors: the service time and the capacity. The service time is the time required to serve an individual entity. The service capacity, or, more simply, the capacity, is the number of entities that can be served simultaneously.

A third factor that may need to be described in discussing the service is its availability. The service may not be available at all times. For example, a machine may break down or be periodically removed from service for inspection. In that case, the availability will be a function of the system conditions. The availability may also be an intrinsic property of the service, as occurs, for example, with an elevator which is only able to admit people when it is stopped with its doors open.

To model a system, the probability functions that describe the arrival patterns and service times must be given. Most system models consist of several activities, interconnected by having the output of one become the input for another. The arrival patterns that result from the transfer of entities between these activities arise from endogenous events and do not have to be described; they are generated as the simulation proceeds. The exogenous arrivals coming to the system from its environment, however, do need to be described.

When measurements have been taken of an arrival pattern, the approximation methods described can be used to reproduce the distribution. This is likely to be done when studying a specific system. When studying general types of systems, it is desirable to have some fundamental representation of the distribution in which, while it may be theoretical, has the general characteristics of the traffic occurring in that type of system. A number of theoretical distributions that serve this purpose will be discussed. In addition, the introduction of stochastic variables into a model makes most of the system performance measures vary stochastically. The measurement of queues is among the most important outputs of a simulation, and the ways of describing and measuring queues will also be discussed.

In simple cases, the statistical properties of the arrivals and the service are independent of time, in which case they are said to be stationary. There may, however, be effects that cause these factors to vary with time, in which case the process is said to be non-stationary or time-variant. For example, the arrival rate may depend on time of day, resulting in peak load periods or the rate of service may speed up when there are long queues. where this happens, the effect can be simulated by making some parameter of the distribution, such as the mean, vary according to conditions

The usual way of describing an arrival pattern is in terms of the inter-arrival time, defined as the interval between successive arrivals. For an arrival pattern that has no variability, the inter-arrival time is, of course, a constant. Where the arrivals vary stochastically, it is necessary to define the probability function of the inter-arrival times. Two or more arrivals may be simultaneous. If n arrivals are simultaneous, than n — 1 of them have zero inter-arrival times.

In discussing arrival patterns, the following notation will be used:

T_a Mean inter-arrival time

λ Mean arrival rate

They are related by the equation

$$\lambda = {}^{1}/_{T_a}$$

For example, records might show that an office, working an eight-hour day for five days a week, gets about 8000 telephones calls a week. Suppose a model of the office during working hours is to be constructed, using a time scale of minutes. A week has 40 working hours is to be constructed , using a time scale of minutes. A week 40 working hours, so 800 calls implies an average inter-arrival time of 40*60 / 800 = 3 minutes; which is equivalent to an arrival rate of 1/3 = 0.333 calls per minute.

Probability distributions have been described so far as either probability density functions or cumulative distribution functions. When describing arrival patterns, it is common practice to express the distribution in terms of the probability, that an inter-arrival time is greater than a given time. We define $A_0(t)$ as the arrival distribution, so that

$A_0(t)$ is the probability that an inter-arrival time is greater than t.

Since the cumulative distribution function F (t) is the probability than an it inter-arrival time is less than t, it is related to the arrival distribution by

$$A_0(t) = 1 - F(t)$$

From its definition, the function $A_0(t)$ takes a maximum value of 1 at t=0 and it cannot increase as t increases.

Poisson Arrival Patterns

A common situation is that the arrivals are said to be completely random. Speaking loosely, this means that an arrival can occur at any time, subject only to the restriction that the mean arrival rate be some given value. More formally, it is assumed that the time of the next arrival is independent of the previous arrival, and that the probability of an arrival is an interval. Δ_t Is proportional to Δ_t. If, in fact, λ is the mean number of arrivals per unit time, then the probability of an arrival in Δ_t λ Δ_t .With these assumptions, it is possible to show that the distribution of the inter-arrival times is exponential. The probability density function of the inter-arrival time is given by

$$f(t) = \lambda e^{-\lambda t} \qquad (t \geq 0)$$

It follows that the arrival distribution is

$$A_0(t) = e^{-\lambda t}$$

The number λ is the mean number of arrivals per unit time. The actual number of arrivals in a period of time t is a random variable. It can be shown that with an exponential distribution of inter-arrival times, the probability of n arrivals occurring in a period of length t is given by

$$p(n) = \frac{(\lambda t)\, e^{n-\lambda t}}{n!} \quad (n = 0,1,2,\ldots)$$

This distribution, which is called the **Poisson distribution**, is **discrete**. The exponential distribution is, of course, continuous, since the inter-arrival time can take any non-negative value. Because of this connection between the two distributions, a random arrival pattern is often called a Poisson arrival pattern. Where this term is used, it will mean that the inter-arrival time is exponentially distributed.

As an instruction, Table lists the times at which 20 successive customers arrive at a store. Column 1 lists the customer number, column 2 gives the time of arrival to the nearest tenth of a minute. The third column gives the inter-arrival times between the customers, the inter-arrival time for the first customer being measured from time zero.

Table 7.1 Arrival Data

Arrival Number	Arrival Time	Inter arrival Time	Arrival Number	Arrival Time	Inter arrival Time
1	12.5	12.5	11	136.4	21.4
2	15.1	2.6	12	142.7	6.3
3	44.1	29.0	13	151.2	8.5
4	62.6	18.5	14	162.5	11.3
5	65.3	2.7	15	167.2	4.7
6	67.6	2.3	16	172.9	5.7
7	71.0	3.4	17	179.8	6.9
8	92.5	21.5	18	181.6	1.8
9	106.5	14.0	19	185.0	3.4
10	115.0	8.5	20	194.9	9.9

We discussed exponential growth models and described a method for testing whether data are exponentially distributed. The method consists of ranking the data by magnitude and plotting it on semi-logarithmic paper. We use the same method here. The inter-arrival times range from a low of 1.8 to high a 29.0. Plotting the values against rank on semi-logarithmic paper gives the results. The data fall close to the straight line shown in the figure (which has been drawn freehand.) Accepting that the data are exponentially distributed, the mean inter-arrival time, T_a, is estimated by taking the average of the observed values to get a value of 9.74. The estimated arrival rate, therefore, is $\lambda = 1/T_a = 0.103$.

To demonstrate the Poisson distribution of the arrivals, consider the interval 0 to 200 minutes broken into adjoining 20minute periods, and count the number of arrivals in each successive period. (It is assumed that the next arrival, after number 20, occurs after time 200.) Table , we collect together the results arranged in time period order. Table rearranges the data of Table 7-2. Column 1 gives the number of arrivals, ranging form 0 to 5.The second column shows how many periods had that many arrivals.

TABLE 7-2 Number of Arrivals in Successive Periods

Time Period	Number of Arrivals	Time Period	Number of Arrivals
0 – 20	2	100 – 200	2
20 – 40	0	120 - 140	1
40 – 60	1	140 - 160	2
60 – 80	4	160 - 180	4
80 – 100	1	180 - 200	3

Referring now to Eq. which defines the Poisson distribution: for the present case,

$\lambda = 0.103$, and $t = 20$. Computing the value of p (n) for n = 0,1,2,3,4 and 5 $_t$
gives the value shown in column 3 of Table These figures are the probabilities of getting the given number of arrivals in a 20minute period when the arrival rate is 0.013 per minute. The particular sample used in this case was for intervals, so, multiplying column 3 by 10, gives the expected number of intervals.

an exponential distribution. In that case, the associated Poisson distribution represents the number of entities served within a given interval.

Frequently, the service time of a process is constant; but where it varies stochastically, it must be described by a probability function. In discussing sevice times, the following notation will be used:

T_s = Mean service time
μ = Mean service rate
$S_o(t)$ = Probability that service time is > t

If the service time is considered to be completely random, it may be represented by an exponential distribution or, if the coefficient of variation is found to differ significantly from 1, an Erlang or hyper- exponential distribution may be used. A common situation is that, although the service time should be constant, there are random fluctuations due to uncontrolled factors. For example, a machine tool may be expected to take a fixed time to turn out a part, but random variations in the amount of material to be removed and the toughness of the material cause fluctuations in the processing time. The normal, or Gaussian, distribution is often used to represent the service time under these circumstances.

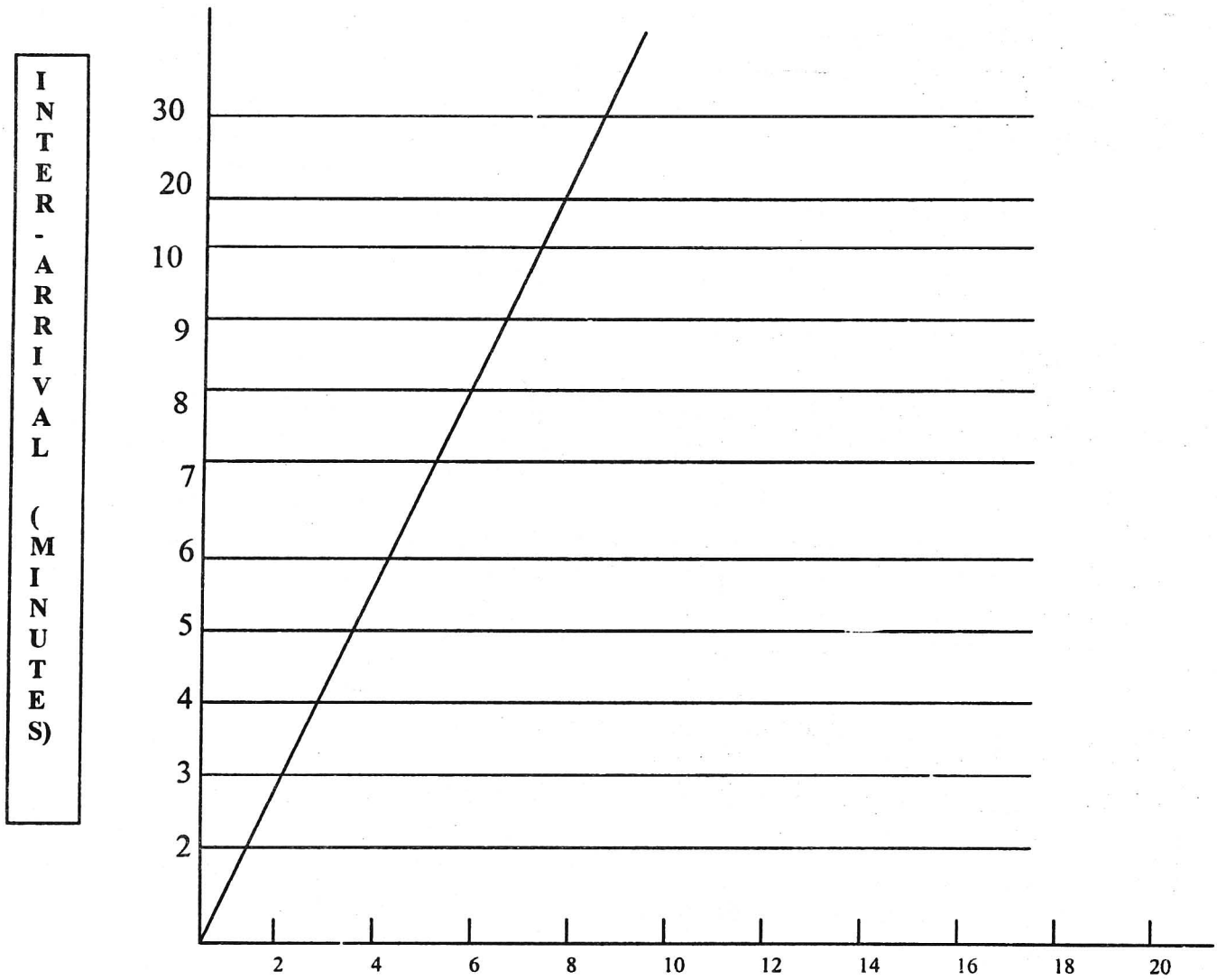


Figure 7.1. Experimental inter-arrival data.

Table 7 -3 Poisson Distribution Data

Number of Arrivals In 20 Mins.	Actual Number of Occurrences	p(n)	Expected Number of Occurrences
0	1	0.128	1.3
1	3	0.266	2.7
2	3	0.272	2.7
3	1	0.187	1.9
4	2	0.096	1.0
5	0	0.040	0.4

7.2 ANALYSIS OF SIMULATION OUTPUTS

Nature of the Problem

Once a stochastic variable has been introduced into a simulation model, almost all the system variables describing the system's behavior also become stochastic; because of the way endogenous events make one variable depend upon another. The value of most, if not all, of the system variables will fluctuate as the simulation proceeds, so that no one measurement can be arbitrarily taken to represent the value of a variable. Instead, many observations of the variable's value must be made, in order to make a statistical estimate of its true value. Some statement must also be made about he probability of the true value falling within a given interval about estimated value. Such a statement] defines a confidence interval. Without it, simulation results are of little value to a system analyst.

A large body of statistical methods has been developed over the years to analyse results in science, engineering, and other fields where experimental observations are made. Because of the experimental nature of system simulation, it seems natural to attempt applying these methods to simulation results. Unfortunately, most of them presuppose that all observations being made are mutually independent-a reasonable assumption when an experiment is being repeated, or independent samples are being selected.

Simulation results, however, are not likely to be mutually independent. A single simulation run will produce many "observed" value of a variable, but the value observed at one time is likely to be influenced by the value at some earlier time. For example, in the simulation of a waiting line, the time one entity spends waiting depends upon the number of entities that happened to be on the waiting line at the time it arrived. The need to account for the resultant interactions requires careful application of the established statistical methods. This need has also lead to the development of new statistical methods, and it is the subject of much current research work.

One concern of this newly developing statistical methodology is to ensure that the statistical estimates are consistent, meaning that, as the sample size increases, the estimate tends

to the true value. Another concern is to control bias in measures of both mean values and variances. Bias causes the distribution of estimates based on finite samples to differ significantly from the true value population statistics, even though the estimates may be consistent. A third, practical aspect of current research work is the attempt to develop sequential testing methods that will allow automatic controls to determine how long a simulation should be run in order to obtain a given level of confidence in its results.

The purpose here is to review the problems involved, and no describe some of the methods being used to analyse simulation results. A more comprehensive discussion of the statistical aspects of simulation will be found.

Estimation Methods

We first review some of the statistical methods commonly used to estimate parameters from observations on random variables. Usually, a random variable is drawn from an infinite population that has a stationery probability distribution with a finite mean, μ, and finite variance, σ^2 . This means that the number of samples already made does not affect the population distribution, nor does it change with time. If, further, the value of one sample is not affected in any way by the value of any other sample, the random variables are mutually independent. Random variables that meet all these conditions are said to be independently and identically distributed, abbreviated to i.e. Under broad conditions that can be expected to hold for simulation data, the central limit theorem can be applied to i.e. Data. The theorem can be applied to i.e. data. The theorem states that the sum of n i.e. variables, drawn from a population that has a mean of μ and a variance of Σ^2 , is approximately distributed as a normal variable with a mean of n and a variance of $n\Sigma^2$.

As was described in any normal distribution can be transformed into a standard normal distribution that has a mean of 0 and a variance of 1. Let x_i (i= 1,2,...,n) be the i.i.d random variables. Using the central limit theorem, and applying the transformation, gives the following (approximate) normal variant:

$$z = \frac{\sum_{i=1}^{n} x_i - n\mu}{\sqrt{n}\,\sigma}$$

Dividing top and bottom of this fraction by n, and defining $\bar{X} = \frac{1}{n} \sum_{i=1}^{n} x^i$ we have

$$Z = \frac{\bar{x} - \mu}{\sigma / \sqrt{n}}$$

The variable $\bar{X}$ is the sample mean. It can be shown to be a consistent estimator for the mean of the population from which sample is drawn. Since the sample mean is the sum of

random variables, if is itself a random variable. As a result, a confidence interval about its computed value needs to be established.

The probability density function of the standard normal variant is illustrated in figure 7.2.1. The integral from – ∞ to a value is the probability that z is less than or equal to u. The integral is usually denoted by

Φ (u) and tables of its value are widely available. Suppose the value of u is chosen so that Φ (u) = 1 – α / 2, where α are some constant less than 1, and denote this value of u by $u_{\alpha/2}$.

The probability that z is greater than $u_{\alpha/2}$ is then α /2.The normal distribution is symmetric about its mean so the probability that z is less than-. $u_{\alpha/2}$ is also α /2. Consequently, the probability

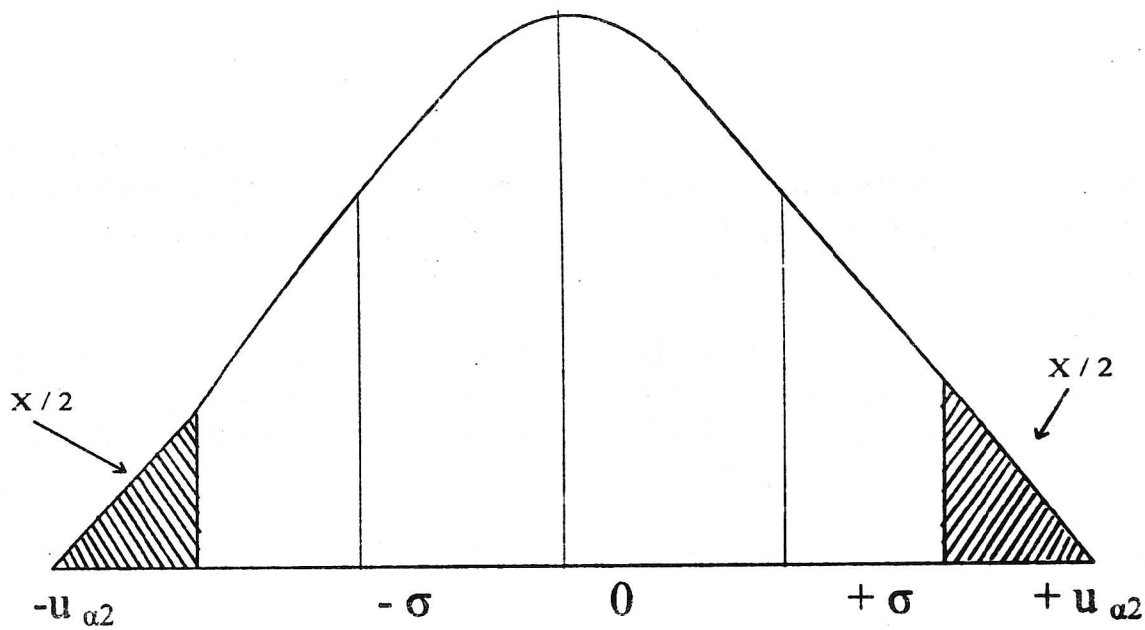


Figure 7.2.1 Probability function of standard normal variation

That Z is lies between $-u_{\alpha/2}$ and $u_{\alpha/2}$ is 1 – α. That is

$$\text{Prob}\{-u_{\alpha/2} \leq z \leq u_{\alpha/2}\} = 1 - \alpha$$

In terms of the sample mean, this probability statement can be written

$$\text{Prob}\left\{ \bar{X} + \frac{\sigma}{\sqrt{n}} u_{\alpha/2} \geq \mu \geq \bar{x} \frac{\sigma}{\sqrt{n}} u_{\alpha/2} \right\} = 1 - \alpha$$

The constant 1 – (usually expressed as a percentage) is the confidence level and the interval

$$\bar{X} \pm \frac{\sigma}{\sqrt{n}} u_{\alpha/2}$$

Is the confidence interval The size of the confidence interval depends upon the confidence level chosen. Typically, the confidence level might be 90%, in which case $u_{\alpha/2}$ is 1.65. The statement then says that μ will be covered by the confidence interval $\bar{x} \pm 1.656 \sqrt{n}$ with probability 0.9; meaning that, if the experiment is repeated many times, the confidence interval can be expected to cover the value μ of on 90% of the repetitions.

In practice, the population variance σ^2 is not usually known; in which case, it is replaced by an estimate by an estimate calculated from the formula

$$S^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \qquad (7.2.1)$$

The normalized random variable based on σ^2 is replaced by a normalized random variable based on S^2. This has a student –t distribution with n – 1 degrees of freedom. The quantity $u_{\alpha/2}$ used in the definition of a confidence interval given above, is replaced by a similar quantity, $t_{n-1, \alpha/2,}$ based on the Student –t distribution, for which tables are also readily available.

The Student–t distribution is strictly accurate only when the population from which the samples are drawn is normally distributed. It is common practice, however, to rely upon this distribution when it is being assumed that the distribution is normal, as when the central limit theorem is being invoked.

Expressed in terms of the estimated variance, S^2, the confidence interval for is defined by

$$\bar{X} \pm \frac{s}{\sqrt{n}} t_{n-1, \alpha/2} \qquad (7.2..2)$$

Simulation Run Statistics

In addition to the assumption of normally inherent in the use of the central limit theorem, the method of establishing confidence intervals, outlined in the previous section, is based on two other assumptions. It is assumed that the observations are mutually independent, and it is assumed that the distribution from which they are drawn is stationery. Unfortunately, many statistics of interest in a simulation do not meet these conditions. To illustrate the problem that arise in measuring statistics from simulation runs, a specific example will be discussed.

Consider a single-server system in which the arrivals occur with a Poisson distribution and the service time has an exponential distribution. The queuing discipline is first in, first out with no priority. Suppose the study objective is to measure the mean waiting time, defined as the time entities spend waiting to receive service and excluding the service time itself. The problem can be solved analytically. This system is commonly denoted by M/M/1, which indicates; first,

that the inter-arrival time distributed exponentially; second, that the service time is distributed exponentially; and, third, that there is one server. (The M stands for Marko Ian, which implies an exponential distribution.)

In a simulation run, the simplest approach is to estimate the mean waiting time by accumulating the waiting time of n successive entities and dividing by n. This measure, the sample mean, is denoted by $\bar{x}(n)$ to emphasize the fact that its value depends upon the number of observations taken. If x_i $(i = 1,2,\ldots, n)$ are the individual waiting times (including the value 0 for those entities that do not have to wait), then

$$\bar{X}(n) = \frac{1}{n} \sum_{i=1}^{n} x_i \qquad (7.2.3)$$

Waiting times measured this way are not independent. Whenever a waiting line forms, the waiting time of each entity on the line clearly depends upon the waiting times of its predecessors. Any series of data that has this property of having one value affect other values is said to be auto correlated. The degree to which the data are auto correlated can be measured in ways that will be briefly described in a later section.

Under broad conditions that can normally be expected to hold in a simulation run, the sample mean of auto correlated data can be shown to approximate a normal distribution as the sample size increases. The usual formula for estimating the mean value of the distribution, Eq. (8.2.3) remains a satisfactory estimate for the mean of auto-correlated data. However, the variance of the auto-correlated data is not related to the population variance by the simple expression as occurs for independent data. A term must be added to account for the auto-correlation. The term is usually positive in the case of the M/M/1 system, so that, if it is ignored, the variance is underestimated, but, in other systems, it can be negative, resulting in an overestimate.

Another problem that must be faced is that the distributions may not be stationery. In particular, a simulation run is started with the system in some initial state, frequently the idle state, in which no service is being given and no entities are waiting. The early arrivals then have a more than normal probability of obtaining service quickly, so a sample mean that includes the early arrivals will be biased. As the length of the simulation run is extended, and the sample size increases, the effect of the bias will die out. For a given sample size starting from a given initial condition, the sample mean distribution is stationery; but, if the distributions could be compared for different sample sizes, the distributions would be slightly different. The analytical solutions previously quoted are for the steady state values to which the distributions converge as the sample size increases, but even with a sample size of

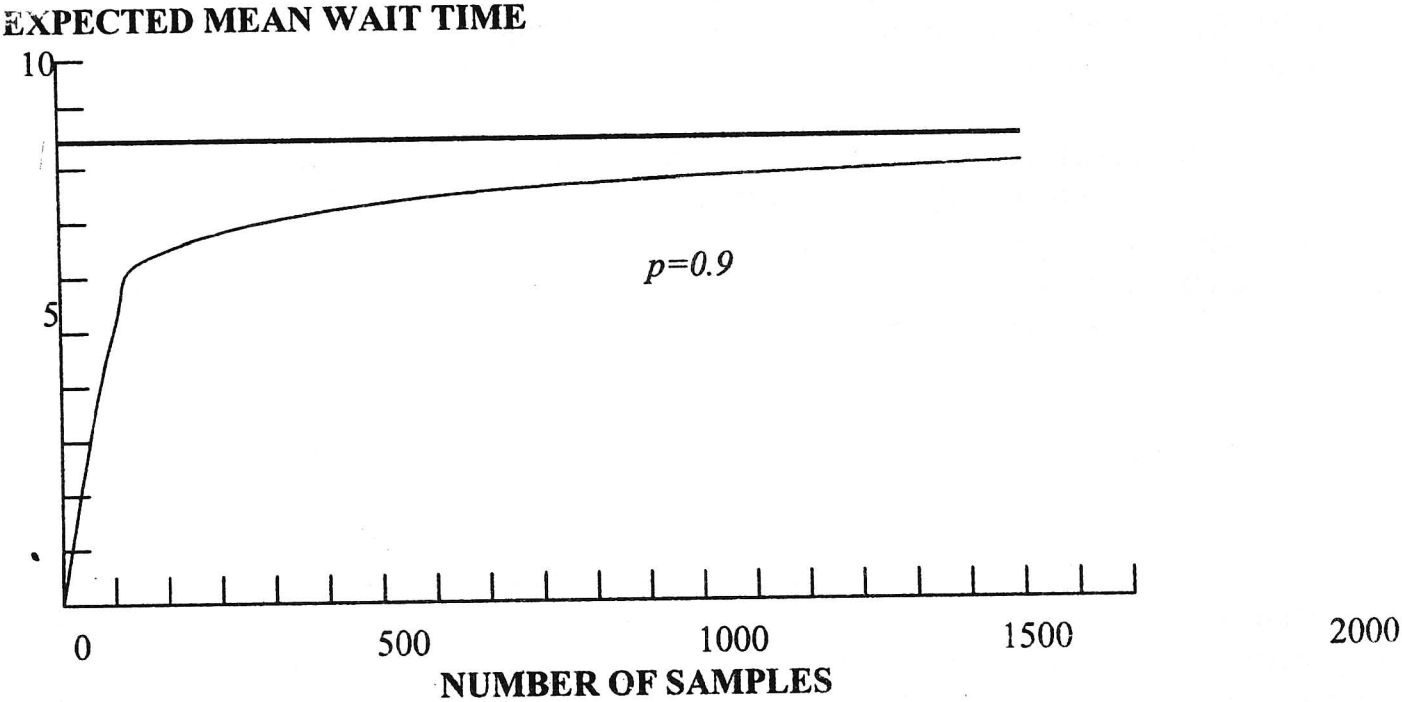


Figure 7.2.2. Mean wait time in M / M / 1 system for different sizes.

2,000 , the mean has still only reached about 95% of the steady state value. The steady state value will be approached more rapidly for lower levels of server utilization but unfortunately, high server utilization cases are usually the ones of interest in simulation studies.

Replication of Runs

One way of obtaining independent results is to repeat the simulation. Repeating the experiment with different random numbers for the same sample size n gives a set of independent determinations of the sample mean (n). Even though the distribution of the sample mean depends upon the degree of auto correlation, these independent determinations of the sample mean can properly by used to estimate the variance of the distribution. Suppose the experiment is repeated p times with independent random numbei series. Let x_{ij} be the observation ith in the jth run, and let the sample mean and variance for the jth run be denoted by $_j$ (n) and $s^2_j(n)$, respectively. For that j th

run the estimates are:

$$\bar{X}_j(n) = \frac{1}{n} \sum_{i=1}^{n} x_{ij} \qquad (7.2.4)$$

$$S_j^2 = \frac{1}{n-1} \sum_{i=1}^{n} [X_{ij} - \bar{X}_j(n)]^2$$

Combining the results of p independent measurements gives the following estimates for the mean waiting time, , and the variance, S^2, of the population:

$$\bar{x} = \frac{1}{p} \sum_{j=1}^{p} \bar{x}_j \ (n) \qquad (7.2.5)$$

$$S^2 = \frac{1}{P} \sum_{j=1}^{p} S^2_j \ (n) \qquad (7.2.6)$$

The value of $\bar{x}$ is an estimate for the mean waiting time, and S^2 can be used to establish a confidence interval, according to the expression degrees of freedom.

The result of applying the procedure to experimental results for the M/M/1 system. Results are shown for server utilizations of 0.1, 0.3, 0.4, 0.5, and 0.6. In each case, the experiment has been repeated from an initial idle state, different random numbers being used on each repetition. The results show the estimated mean waiting time calculated from Eq 7.2.3. as a function of sample size, n. Measurements were made in steps of n = 5 for p = 0.2, 0.3, and 0.4, 10 for p = 0.5 and 0.6. Each case is for 100 repetitions (p = 100). Also shown are the 90 lence intervals calculated for the highest value of n used in each case, and the known eady state values of the mean waiting times.

The p repetitions of n observations involve a total of N = pin observations. In a computer based simulation, the total time spent carrying out calculations will be roughly proportional to N. The question of how best to divide the N observations between the number of repetitions and the length of the individual run has given rise to much discussion. Increasing the number of repetitions decreases the size of the confidence interval, since the variance estimate is approximately inversely proportional.

To p. Normally, this is a desirable effect, since it reduces the range of uncertainty about the estimate of the mean. However, the desirably assumes the estimate of the mean is unbiased. A short confidence limit, centred on a biased estimate, can easily fail to cover the true value being estimated. In the replication of simulation runs, if the number of runs is increased at the cost of shortening the individual runs (in order to keep N constant), the estimate of the mean will be more biased, as a result of the initial empty state.

Law for example, reports results of experiments with the M/M/1 system, at a server utilization of 0.9. The entire process of computing a confidence interval by replication of runs starting from an empty state, in this case at a 90% confidence level, was repeated 400 times for different combinations of n and p. For each combination of n and p, the total number of observation used in each determination of a single confidence interval, N was kept at 12,800.

At a 90% confidence level, it is to be expected that 90% of the 400 independently determined confidence intervals would cover the true mean. Taking the steady state mean as the true mean, the results showed that for 5 replications, each of 2,560 samples (n = 2,560 and p = 5), 83%

the confidence intervals covered the true mean. At the other extreme of the experiment, 40 replications, each of 320 samples (n = 320 and p = 40), the figure dropped to 9 %. It can be estimated form Fig 14.2. That, at a sample size of 2,560, the mean wait time is about 7.8, which is only 0.3 below the steady state mean. At a sample size of 320, the mean is about 6, which is 2.1.below the steady state mean. At a sample size of 320, the mean is about 6, which is 2.1.below the steady state mean. The shorter confidence interval, achieved the greater number of replications, is bought at the cost of a much greater bias resulting in the larger difference in the accuracy.

From this evidence, we see that it is preferable to keep the number of repetitions as low as possible, bearing in mind the need to approximate a normal distribution with the sample means. These results, of course, are for a simple system. However, they are indicative of the type of research work being conducted in the study of the simulation process.

Elimination of Initial Bias

The experimental results given together with those reported) clearly show the need to remove the initial bias, reduce its effect. Two general approaches can be taken to remove the bias; the system can be started in a more representative state, or the first part of the simulation run can be ignored.

In some simulation studies, particularly of existing systems, there may be information available on the expected conditions that make it feasible to select better initial conditions. The ideal situation is to know the steady state distribution for the system, and select the initial condition from that distribution for the system, and select the initial condition from that distribution. In the study previously discussed, Law repeated the experiments on the M/M/1 system, supplying an initial waiting line for each run, selected at random from the known steady state distribution of the waiting line. The case of 40 repetitions of 320 samples, which previously resulted in coverage of only 9%, was improved to coverage of 88%. Of course, the theoretical knowledge on which this technique is biased is not usually available. However, experience with an existing system, or similar type of system, could provide a reasonable approximation.

The more common approach to removing initial bias is to eliminate an initial section of the run. The run is started from an idle state and stopped after a certain period of time. The entities existing in the system at that time are left as they are. The run is then restarted with statistics being gathered up to the point of restart. No simple rules can be given to decide how long an interval should be eliminated. It is advisable to use some pilot runs starting from the idle state to judge how long eh initial bias remains. Plotting the measured statistic against run length as has been done can do this.

Another disadvantage of eliminating the first part of a simulation run is that the estimate of the variance, needed to establish a confidence limit, must be based on less information. The reduce in bias, therefore, is obtained at the price of increasing the confidence interval size.

Batch Means

Another approach to the problem of establishing confidence intervals for simulation results does not rely upon replication, but uses a single long run, preferably with the initial bias removed. The run is divided into a number of segments to separate the measurements into batches of equal size. The sample means are then treated as independently distributed variables. A complete run consists of N observations that are broken into p batches of size n. (It is assumed that N is exactly divisible by p.) In effect, the experiment is equivalent to repeating an experiment is equivalent to repeating an experiment of length n a total of p times, with the final state of one run becoming the initial condition for the next. Denoting the observation in the batch by Esq. (14-3) through can be used to estimate the mean and standard deviation of the variable being measured.

This way of repeating a run is preferable to starting each run from an initial idle state, since the state at the end of a batch is a more reasonable initial state than the idle state. However, the connection between the batches introduces correlation. Sometimes, the batches are separated by intervals in which measurements are discarded in order to eliminate the correlation. Clearly, this throws away useful information. Conway (2) demonstrated that the variable to be expected by using all the data and accepting the correlation between batches is less than that obtained from the reduced amount of data obtained by separating the batches. It seems to be preferable, therefore, to work with adjoining batches.

The batch mean method has the advantage of the repetition method without the necessity of eliminating the initial bias on each repetition. However, it is necessary to assume that the individual batch means are independent. The assumption can be justified if the batch length is sufficiently long. The effect of auto-correlation is that value of one piece of data affects the value of following data. The effect usually diminishes as the separation between the data increases, and beyond some interval size it may reasonably be ignored. If the batch size is greater than this interval, the batch means may be treated as independent. It remains a matter of judgement to choose a suitable batch size. It might reasonably be speculated that the interval excluded from the beginning of a run to remove initial bias. If that value has been determined, it can also be sued as a batch size. However, the only safe procedure is to use a test which to try a batch size and test for the presence of correlation in the results. Another approach is to repeat the calculations with several batch sizes and test for consistency of results. By making the batch sizes multiples of each other, it is possible to perform the operation in a single run.

An important practical aspect of the batch method is that it does not entail the simultaneous presence of all the data to carry out the calculations. The batch means can be calculated as the simulation run proceeds. Computer space is only required to accumulate the sum of the batch means and the sum of their squares, together with an accumulation of the numbers forming the current batch mean. The batch method, therefore, forms a good basis for sequential testing methods.

In discussing the replication method, it was pointed out that there is a trade-off between the number of repetitions and the run length. With the batch method there is a similar trade-off between batch size and number of batches. Since the number of batches corresponds to the number of samples of an assumed normal distribution, it is again advisable to hold this number

to a reasonable limit to meet the assumption, and maximize the batch size, in order to reduce the correlation between batches.

The experiments reported in tested the batch mean method against the replication method for many combinations of n and p. The results showed that, in almost all cases, the batch mean method was superior to the replication method, and that the difference was statistically significant. Again, the results were for simple systems, although, in this test, more complex systems than the M/M/1 system were tested.

Regenerative Techniques

We have seen that a discrete system is described by a number of state descriptors that are changing value at specific points in time. To know the state of the system (to the extent that it is modelled), it is necessary to know the values of the state descriptors. Suppose all the values are known for one point of time. Most discrete systems are such that exactly the same set of values will occur some time later. In fact, that set of values will continue to recur at certain random intervals of time.

A system with this property is said to be regenerative. A particular set of values is chosen as a reference; the times at which the same set of values recurs are called regeneration points. Between successive regeneration points, the system is said to execute a tour. The time the system spends in a tour is called an epoch. Consider, again, measuring the waiting time in the M/M/1 system as an example. Suppose the system starts from the empty state, and measurements begin from the time the first entity arrives for service. We use the return to the empty state to define the regeneration points.

At the time the measurements begin, the system becomes busy with the first entity. If a second entity arrives before the first completes service, the system will immediately begin serving the second entity when the first completes service. The server will then remain busy, and this may happen again with there will be no other entity waiting when service finishes. The system then becomes idle, and remains so until the next arrival occurs. The point at which the system becomes busy again marks a regeneration point. This process is repeated as the system moves through successive cycles of a busy period followed by an idle period. Each of these cycles is a tour.

The number of entities served in a tour varies. It could happen that an entity finding the system idle, makes the system busy, but there in no other arrival by the time service for that entity finishes. The system will fall back to the idle state, and there will have been a tour involving only one entity (with a waiting time of zero).

As such busy period begins, the system regenerates itself. That is, its future behaviour is independent of its past behaviour. The system behaviour in any tour is independent of its behaviour in any other tour. As a result, samples of any statistic taken from the tours are i.i.d. In particular, the epochs are i.i.d. As a result, the behaviour, and the estimates that are based on the tour statistics are free of the initial bias that proved to be so troublesome with replicated tests.

Suppose there are p complete tours when measuring an M/M/1 system. Let y_j be the sum of the waiting times for the entities served in the jth the tour, and let n_j be the no.of entites served in the tour. Because of the regeneration principle, both Y and n are i.i.d. samples, each from its own distributions. Note, however, that $Y_j$and $n_j$are not mutually independent on the contrary, they are strongly, positively correlated, since a larger value of Y can be expected to occur with a larger value of n_j .

Disregarding the tours for the time being, suppose there is a total of N observations, and the individual waiting times are $w_1, w_2, \ldots\ldots w_n$. Then, a consistent estimate of the mean waiting time, denoted by _given by

$$\overline{w} = \frac{W_1 + W_2 + \ldots W_n}{N}$$

If the individual waiting times are now grouped according to their tours, this expression may be written in the form

$$\overline{w} = \frac{Y_1 + Y_2 + \ldots\ldots\ldots + Y_p}{n_1 + n_2 + \ldots\ldots + n_p}$$

If further, we define

$$\overline{Y} = \frac{1}{P} \sum_{j=1}^{p} Y_j \qquad (7.2.7)$$

$$\overline{n} = \frac{1}{P} \sum_{j=1}^{p} Y_j \qquad (7.2.8)$$

The estimate of the mean waiting time is given by

$$w = \frac{\overline{Y}}{n} \qquad (7.2.9)$$

The number and n are, of course, the estimates of the means for the sum of the waiting times in a tour, and the number of entities served in a tour. Equation (14-9) states that the estimate of the mean waiting time is the ratio of these two numbers. Using the notation E (x) to denote the expected value of a random variable x, it also follows that

$$E(W) = \frac{E(Y)}{E(n)} \qquad (7.2.10)$$

To construct a confidence interval for W, it is necessary to estimate the variance of the statistic w . To avoid dealing directly with the ratio of random variables, we define the following variable:

$$V_j = Y_j - E(W)n. \qquad (7.2.11)$$

Note that E (W), the expected value of the waiting time, is a constant (even though its value is not known). This makes V_j a linear combination of two random variables that are i.i.d. It follows that V_j is also i.i.d. If we define as the sample mean of V_j, it can also be seen that

$$\overline{V} = \overline{Y} - E(\overline{w})\,5 \qquad (7.2.12)$$

In addition, we have

$$E(V) = E(Y) - E(W)\,E(n)$$

Substituting the value of E (W) form Eq. (14-10), it follows that the expected value of the variable V is zero.

Since the values of V_j are i.i.d., the central limit theorem can be applied to their sum. Letting σ^2 be the variance of V, we construct the following standard normal variate (since the value E (V) is zero)

$$z = \frac{\overline{v}}{\sigma / \sqrt{P}}$$

This leads to the following probability statement, (using $u_{\alpha/2}$ and α as defined already

$$\text{Prob}\left[-u_{\alpha/2} \le \frac{\overline{v}}{\sigma / \sqrt{P}} \le u_{\alpha/2} \right] = 1 - \alpha$$

Substituting for $\bar{v}$ from Eq. (14-12), and multiplying the expression by $\sigma / \sqrt{P}$ result in

$$\text{Prob}\left\{\frac{\sigma u_{\alpha/2}}{\sqrt{P}} \leq \bar{Y} - E(w)n \leq \frac{\sigma u_{\alpha/2}}{\sqrt{P}}\right\} = 1 - \alpha$$

This can be further rearranged to give

$$\text{Prob}\left\{\frac{\bar{Y}}{n} + \frac{\sigma u_{\alpha/2}}{n\sqrt{P}} \leq E(W) \geq \frac{\bar{Y}}{n} - \frac{\sigma u_{\alpha/2}}{5\sqrt{P}}\right\} = 1 - \alpha$$

We have, therefore, constructed a confidence interval about the expected mean wait time defined by the points

$$w \pm \frac{\sigma u_{\alpha/2}}{n\sqrt{P}} \qquad (7.2.13)$$

The variance of V_j σ^2 however, is not known, but it can be estimated from Eq.(8.2.11). which defines V_j in terms of Y_j and n_j . since Y_j and n_j are not mutually independent, we must take note of their covariance. Let s^2_v denote the sample variance of V, based on the p pairs of measurement take from the tours; then it can be shown that

$$S2v = S^2_{11} - 2\bar{w}S^2_{12} + \bar{w}^2 S^2_{22} \qquad (7.2.14)$$

Where S^2_{11} , S^2_{22} and S^2_{12}
are the sample variance of Y_j, the sample variance of n_j and the sample covariance of (Y_j, n_j) respectively? From the definition of V_j given by Eq. (14-11), the expression for S2v should involve E (W). Since this is not known, it is replaced by its estimated value, w
In terms of the observed data, the values of S^2_{11}, S^2_{22} and S^2_{12} are

$$S^2_{11} = \frac{1}{p-1}\sum_{j=1}^{p}(Y_j - \bar{Y})^2 = \frac{1}{p-1}\sum_{j=1}^{p} Y^2_j - \frac{p\bar{y}^2}{p-1} \qquad (7.2.15)$$

$$S^2_{22} = \frac{1}{p-1}\sum_{j=1}^{p}(n_j - n)^2 = \frac{1}{p-1}\sum_{j=1}^{p} n^2_j - \frac{p\bar{n}^2}{p-1} \qquad (7.2.16)$$

$$S^2_{12} = \frac{1}{P-1}\sum_{j=1}^{p}(Y_j - \bar{Y})(n_j - n) = \frac{1}{p-1}\sum_{j=1}^{p} Y_j n_j - \frac{p\bar{y}\,\bar{n}}{p-1} \qquad (7.2.17)$$

Replacing the unknown variance of V by its estimate, the estimate of the mean waiting time, and its confidence interval are

$$\bar{w} = \frac{\bar{y}}{\bar{n}} \qquad (7.2.18)$$

$$\bar{w} \pm \frac{S_v u_{a/2}}{\bar{n}\sqrt{p}} \qquad (7.2.19)$$

Summarizing the application of the regeneration method, we see that the following steps are involved:

1. Gather simulation data for p tours
2. For each tour, j, record the total wait time, Y_j and the number served, n_j
3. Compute the statistics , , n, S^2_{11}, S^2_{22} ,S^2_{12} and S^2_v
4. Substitute in Eq. (7.2.18) for the estimate of the mean waiting time, and in Eq. (7.2.19) for a confidence interval.

Time Series Analysis

The three methods for making estimates of the variance of the sample mean that have been described so far- the replication of runs, batch means, and the regenerative method- have been based on the principle of getting (or closely approximating) independent samples. Another approach is to accept the presence of auto correlation, and to estimate the variance of the sample mean by methods that have been developed in the study of time series.

With such studies, it is not usually possible to rerun a stochastic process. Furthermore, the auto correlation contains valuable information, since it describes the underlying nature of the process producing the time series. This has led to the development of techniques for measuring the auto correlation. Given measures of the auto correlation, it is possible to compensate for its presence, and deduce estimates of the variance of the population from which the observations are drawn.

Virtually all time series are based on data observed at uniform intervals of time. To apply the time series techniques to simulation results, therefore, a single run is made, with observations being made at uniform intervals of time. For convenience, suppose the observations are made at unit time intervals, and the run is for a time T time units, so that there are T observations.

Autocorrelation is measured by a series of auto co variances that show the extent to which values separated by s time units affect each other. The auto co variance between values of a variable made at two different times, X_t, and X_u is defined as

$$R_{tu} = E[(X_t - \mu)(X_u - \mu)]$$

If we assume that the value of μ is independent of the indices t and u, and that R_{tu} depends only upon the separation $s = t - u$, the stochastic process is said to be covariance stationery. The auto co variance then exists for all integer values of t and u, but has the property of symmetry that $R_{tu} = R_{ut}$. Since the value then depends only on the separation, the auto covariance is usually denoted by R, and the property of symmetry says that $R_s = R_{-s}$. For convenience, in the following discussion, is allowed to take all the values 0, ± 1, ± 2, ...,but the case of $s = 0$ is actually the definition of the variance of X_t

Given a set of observations at T uniform intervals of time, the auto- co variances

$$R_s = \frac{1}{T} \sum_{t=1}^{T-s} (X_t - \bar{X})(X_{t=s} - \bar{X}) \quad (s = 0,1,\ldots,T-1) \qquad (7.2.20)$$

$$\bar{X} = \sum_{t=1}^{T} X_t$$

The estimation of an unbiased sample mean, V ( ), from the autocovariarce values involves some computational subtlety, but an acceptable formula is
The estimate is taken to have 3k /2n degrees on freedom.

$$V(\bar{X}) = \frac{T}{T-K}\left\{ R_o + 2 \sum_{s=1}^{k-1} \{1 - \frac{s}{T}\} R_s \right\} \qquad (7.2.21)$$

It will be noticed that only the first k-1 auto co variance estimates are involved. Normally, the values of the auto co variances decreases as increases. The value used for k should be large enough to include the auto co variances of significant size, but, as s increases, the accuracy of the auto co variances estimate decreases. For a fixed number of observations, the number of pairs of points available for the estimate decreases as increases, leading to difficulties in judging the accuracy of the sum of the auto co variances. One approach is to carry out the estimation for different values of s, looking for the estimate to stabilize. An example discussed, for the M/M/1 system at a server utilization of 0.9, required a k of about 300.

It is apparent that the method involves a considerable amount of calculation. It also suffers the disadvantage of requiring that all the data be simultaneously available – unlike the batch mean approach, which is able to summarize results as the run proceeds.

Spectral Analysis

The analysis of time series through autocorrelation is intimately related to another way of viewing time series. A time series may also be regarded as the summation of oscillations of different frequencies. The spectrum of frequencies and the amplitudes of the oscillations can be formally related to the auto-covariance. Essentially the same calculations involved in the estimation of autocorrelation can be applied to derive a spectral analysis of a simulation run. The results can be used to estimate sample mean variances.

A spectral analysis, however, can provide more information than is contained in the estimate of a mean value. Comparing two systems on the basis of mean values of such factors, as waiting time or queue length is a rather gross comparison. Two systems may show no significant difference in their mean values, but their transient behaviour may be significantly different. One system may respond slowly to deviations from its mean values while the other may respond rapidly. Depending upon the purpose of the system, one performance may be preferable to the other. A simple comparison of mean will mask the difference but a spectral analysis can distinguish the difference by showing whether the spectrum emphasizes low or high frequencies.

Another interesting use of spectral analysis has been to test a simulation model by comparing the spectral pattern of simulation output against the corresponding pattern derived from the system itself. This, of course, can only be done when analysing an existing system. However, in complex systems, where it is unlikely that a simulation could produce a precise event-by-event replication of the true system response, a reasonable match of the spectra can indicate that the system response is being reproduced correctly.

Autoregressive Processes

Among all possible stochastic processes, there is a sublet, called autoregressive processes, having a structure that makes the evaluation of its auto-correlation properties relatively easy to calculate. The values of an auto-regressive variable X_t are defined by the equation.

$$X_t = -b_1 \quad X_{t-1} \quad -b_2 \quad -X_{t-2} - b_1 \quad --\ldots.- b_p \quad X_p + \varepsilon_t \qquad (7.2.22)$$

Where ε_t is a random variable (usually assumed to be normally distributed), and the Following conditions hold

$$\mu = E(X_t)$$
$$E(\varepsilon_t) = 0$$
$$E(\varepsilon^2_t) = \sigma_2$$
$$E(\varepsilon_3, \varepsilon_t) = 0, \quad S > t$$
$$E(\varepsilon_3, \varepsilon_t) = 0, \quad S \quad t$$

The process is determined by p coefficients b $_{I(I}$ = 1,.....,p),together with the first p – 1 values of X_t, which must be given as initial conditions. Equation (14-22) states that the value of t th value of X_t(t p) - 1 is a linear sum of the previous p –1 value, plus a random component. ε_t.
The conditions state that the random variable ε_t has a zero mean, finite variance, an all its auto-co- variances are equal to zero. Further, the random component is not correlated with X_t the definition can be written more compactly in the form:

$$\sum_{s=0}^{p} b_s \, (X_{t-s} - \mu) = \varepsilon_t \qquad b_0 = 1$$

It can be shown that the auto-co-variances, which are still defined by Eq. (14-20), are related by the following set of equations, known as the Yule-Walker equations:

$$\sum_{s=0}^{p} b_s R_{s-r} = \begin{cases} \sigma^2 & r = 0 \\ 0 & r \neq 0 \end{cases}$$

This is a set of p + 1 linear equations involving the p coefficients $b_s (s = 1,\ldots, p; b_0 = 1)$, 2p + 1 auto-co-variances, $R_s (s = -p,\ldots,-1, 0, 1,\ldots, p)$, and σ^2. However, the symmetry in the auto-co-variances mean there are only p + 1 distinct auto-co-variances.

Assume that a series of observations, X_t is generated by an autoregressive process of order p. Equation provides estimates of the p + 1 auto-co-variances.$R_s (s = 0,\ldots,p)$. Taking the p Yule-Walker equations that correspond to r 0 gives a means of deriving estimates of the coefficients of the autoregressive process, b_s, (s = 1,...,p). The Yule-Walker equation corresponding to r = 0 gives an estimate for σ^2 as follows:

$$\sigma^2 = \sum_{s=0}^{p} b_s R_s \qquad (7.2.23)$$

It can be shown [(10), pp. 286-287] that, by forming the quantity,

$$b = 1 + \sum_{s=1}^{p} b_s \qquad (7.2.24)$$

An estimate of the variance of the sample mean, V (X), is given by

$$m = \frac{\sigma^2}{b^2} \qquad (7.2.25)$$

The degrees of freedom of the estimate are taken to be f, where

$$f = \frac{nb}{(2p + 1)\, b - 4 \sum_{s=0}^{p} s b_s}$$

The question of what value to assume for p, the order of the autoregressive process, is left open. However, experiments with this approach have suggested that small values of p are adequate: certainly, the values needed for a good approximation appear to be much less than the large number of auto-co-variance calculations needed for the approach described, where no assumptions were made about the nature of the stochastic process. In any event, it is possible to establish statistical tests for deciding whether to accept a particular value of p. There are also ways of computing the auto-co-variance estimates recursively, that is, deriving higher order values from lower order values. When combined, these techniques provide a means of applying sequential tests that can decide, automatically, how long a simulation should be run in order to measure a statistic to given degree of confidence.

8. BIBLIOGRAPHY

"System Simulation" – Geoffrey Gordon – Second Edition – 1996 Prentice Hall of India

---

Model Questions

Any FIVE Questions 5*20 = 100 marks

1. Discuss entities, attributes & activity of a system
2. What are the different types of models? Explain in detail
3. Explain with flowchart about the principles of modeling
4. Discuss System analysis & Design concept with an example
5. Explain with a suitable example about system postulation
6. Discuss in detail about Monte-carol method
7. Explain **a)** Distributed lag model
   b) Cobweb model
8. Explain the concept of System Dynamics with an Example
9. Explain with an example about discrete systems
10. Discuss
    a) Interactive system
    b) Gathering Statistics

MODEL QUESTIONS

Unit – I

1. Define System with an example
2. Discuss the term Entity, Attribute, Activity and the State of the System with an example
3. What do you meant by stochastic Activities? explain
4. Discuss in detail about the different types of Models and Explain in detail
5. Explain in detail about the principles of modeling

Unit – II

1. Discuss in detail about System Analysis with an example
2. Explain in detail about System Design with an example
3. What do you meant by System postulation explain with an neat example.

Unit – III

1. Explain Monte Carlo method in detail
2. Discuss the numerical computation technique for continuous models
3. Elaborate Distributed Lag model with an example
4. Explain in detail about the Cobweb model
5. Explain with neat diagram about the progress of Simulation study

Unit – IV

1. Define Analog methods
2. What do you meant by hybrid computers
3. Discuss in detail about CSSLS
4. What do you meant by Real Time simulation explain with an example
5. Define Interactive system

Unit – V

1. Explain in detail about the Exponential Growth models with an example
2. Draw the Curve for Exponential Decay model and explain it
3. Discuss the System Dynamics diagram in detail

Unit – VI

1. Discuss Gathering Statistics in detail
2. Elaborate Discrete Simulation Languages in detail
3. Discuss 10 GPSS block diagram symbols in detail
4. Discuss with neat diagram about the model of the Telephone System Study of GPSS
5. What do you meant by Simscript Languages

Unit – VII

1. Discuss Poisson Arrival Patterns
2. Explain in detail about the Analysis of Simulation Outputs
3. What do you meant by Regenerative Techniques Explain
4. Discuss Time Series Analysis in detail
5. Elaborate Spectral Analysis.