Mt Kenya University

P.O. Box 342-01000 Thika
Email: info@mku.ac.ke
Web: www.mku.ac.ke

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE CODE: BIT 3203

COURSE TITLE: BUSINESS SYSTEMS SIMULATION AND MODELING

Instructional manual for BBIT – Distance Learning

COURSE OUTLINE

BIT 3203 BUSINESS SYSTEMS SIMULATION AND MODELING

Purpose of the course

To simulate and model a wide variety of real world problems in business using formal techniques from statistics, operations research as well as informal mathematical methods.

TOPICS - DETAILS

I. I. Introduction to Simulation And Modeling

A. Definition of terms; system, model, simulation
B. Application areas of simulation and modeling; soft science, engineering, business etc
C. Reasons for using simulation
D. Advantages and disadvantages of Simulation

II. System Modeling

A. Functional modeling: functional decomposition; functional modeling methods: functional flow block diagram, structured analysis and design technique, axiomatic design
B. Business function model: business reference model, operator function model, business process modeling: business model
C. Enterprise modeling: enterprise modeling basics, enterprise model, function modeling, data modeling, ontology engineering modeling, systems thinking

III. Case Study Of Simulation Model

A. Modeling behavior
B. Specification of models; Basic model characteristics
C. A railroad company case; Using the models for simulation
D. Organizing the models and the simulation setup

IV. Probability Theory

A. Definition of probability
B. Discrete Probability Distributions
C. Continuous Probability Distributions

V. Probability Distribution

A. Definition of probability Distribution
B. Types Of Probability Distribution; discrete probability distribution, cumulative density, continuous probability distribution
C. Properties of Probability Theory
D. Applications of probability Theory

VI. Random Number Generation

A. Application Of Random Numbers; gambling statistics, Monte Carlo etc
B. Methods For Creating Random Numbers
C. Generation Of Random Numbers By Physical Methods
D. Generation Of Random Numbers By Computational Methods
E. Generation From A Probability Distribution
F. Generation By Persons

VII. Simulation Languages

A. Definition and Types
B. Advantages Of Special Purpose Languages
C. Advantages Of General Purpose Languages
D. Advantages of Using Simulation Languages

VIII. Discrete Event Simulation

A. Introduction
B. Components of a Discrete-Event Simulation; Clock, Events List, Random Number, Generators, Statistics, Ending Condition
C. Application areas Of Discrete Event Simulation; Diagnosing process issues, Custom order environments, Lab test performance improvement ideas, Evaluating capital investment decisions, Stress test a system

IX. Statistical Inference

A. Introduction
B. Degree Of Models And Assumptions; fully parametric, non-parametric ,semi parametric
C. Importance of valid models/assumptions
D. Types Of Statistical Models; Approximate distributions, Randomization-based models
E. Model-based analysis of randomized experiments

X. Interpretation and Analysis of Simulation Results Using Models

A. Poison Distribution Models; Poison Simulation Areas, Proof of Poison Distributions, Generalization of the formular,2 dimensional Poisson process
B. Properties Of Poison Distribution Model; Evaluating The Poisson Distribution
C. Application Of Poison Models ;Bayesian Inference, Confidence Interval
D. Monte Carlo Models; Introduction to Monte Carlo models, Characteristics of Monte Carlo simulation
E. Monte Carlo Uses ; Random Numbers and "What If" Scenarios Testing
F. Applications; Design And Visuals, Finance and business, Telecommunications, Optimization
G. Inverse problems

Main course text

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition

Reference Books

Chung C & Chung C.A, (2003), Simulation and Modeling Hand book: Practical Approach CRC Press

Assessment: Examination - 70%: Coursework - 30%

Module compiler: John Kamau

CHAPTER ONE

INTRODUCTION TO SYSTEM MODELLING AND SIMULATION

Learning Objectives

By the end of this chapter the learner shall be able to;

Describe arrange of methods of arriving at the selling of a business

Explain the meaning of simulation, systems and models

Explain the advantages and disadvantages of simulation and modeling

Describe the types of systems

imulation can be used for the following reasons

Describe the application areas of simulation and modeling

1.1 Definition of terms

Simulation : A simulation is the manipulation of a model in such a way that it operates on time or space to compress it, thus enabling one to perceive the interactions that would not otherwise be apparent because of their separation in time or space. Simulation is the imitation of some real thing available, state of affairs, or process. The act of simulating something generally entails representing certain key characteristics or behaviors of a selected physical or abstract system. Simulation is used in many contexts, such as simulation of technology for performance optimization, safety engineering, testing, training, education, and video games. Training simulators include flight simulators for training aircraft pilots to provide them with a lifelike experience. Simulation is also used for scientific modeling of natural systems or human systems to gain insight into their functioning. Simulation can be used to show the eventual real effects of alternative conditions and courses of action. Simulation is also used when the real system cannot be engaged, because it may not be accessible, or it may be dangerous or unacceptable to engage, or it is being designed but not yet built, or it may simply not exist

System: This is a set of interacting or interdependent components forming an integrated whole. Most systems share common characteristics, including:

- Systems have structure, defined by components/elements and their composition;
- Systems have behavior, which involves inputs, processing and outputs of material, energy, information, or data;
- Systems have interconnectivity: the various parts of a system have functional as well as structural relationships to each other.
- Systems may have some functions or groups of functions

Types of systems

Systems are classified in different ways:

1. Physical or Abstract systems
2. Open or Closed systems
3. 'Man-made' Information systems
4. Formal Information systems
5. Informal Information systems
6. Computer Based Information systems

Physical systems are tangible entities that may be static or dynamic in operation. For Example, the physical parts of the computer center are the offices , desk and chairs that facilitate operation of the computer. They can be seen and counted as they are static. In contrast, a programmed computer is a dynamic demands or the priority of the information requested changes. Abstract systems are conceptual or non physical entities.

An open system has many interfaces with its environment. i.e. system that interacts freely with its environment, taking input and returning output. It permits interaction across its boundary; it receives inputs from and delivers outputs to the outside. A closed system does not interact with the environment; changes in the environment and adaptability are not issues for closed system.

Model: A model is a simplified representation of a system at some particular point in time or space intended to promote understanding of the real system.

Features of a Model

We need to briefly discuss the features of models, so we can understand why models might need to be simulated. Four general aspects of models are solvability, predictability, variability, and granularity.

Solvability denotes whether a model can be completely solved by analytic methods, such as linear programming. Even for such models, the level or skill and/or computational power required could be prohibitive. If the equations and computing horsepower are available, the model is said to be **solvable**. If the equations cannot be found or are too difficult to solve, a simulation model can predict the behavior of the system, and such a model is said to be **simulatable**.

Predictability refers to the degree that reproduced inputs to the model will results in reproduced results. In a **deterministic** model, relationship between the system elements are fixed, therefore results are reproduced exactly. In a **stochastic** model, some of the relationships have been described to vary in a random fashion. Almost all stochastic models will include some deterministic behavior.

Variability describes how a system behavior changes over time. In a system which is **static**, a set of equations will describe the state of the system at a particular point in time, such as a P&L statement at the end of a fiscal quarter. In a **dynamic** system, a future event is proposed and it is desired to see how the system reacts subsequently. Because the system relationships are complicated and interrelated, the behavior of the system will vary as time passes.

Granularity refers to the treatment of time in dynamic models. **Continuous** models consider time as a continuous flow, and break time into small increments to update all system relationship at each time point. Continuous models are almost always described

by a set of equations, such as the flight of a projectile or relationships between predator and prey populations. **Discrete-event** models have relationships which are primarily concerning with important occurrences in the system, such as the arrival of a loan application, the failure of a machine, or an AS/RS being filled to capacity.

1.2 *Application Areas of Simulation*

Simulation in science and engineering research: Earlier most experiments were carried out physically in the laboratories, but today a majority of experiments are simulated on computers. 'Computer Experiments' besides being much faster, cheaper, and easier, frequently better insight into the system than laboratory experiments do.

Simulation in soft sciences: Simulation can be expected to play even a more vital role in biology, sociology, economics, medicine, psychology etc. where experiments could be very expensive, dangerous, or even impossible. Thus Simulation has become an indispensable tool for a modern researcher in most social, biological and life sciences.

Simulation for business executives: There are many problems faced by management that cannot be solved by standard operations research tools like linear and dynamic programming, inventory and queuing theory. Therefore, instead of taking decisions solely on intuition and experience, now a business executive can use computer simulation to make better and more powerful decisions. Simulation has been used widely for inventory control, facility planning, production scheduling and the like.

The real meat of a simulation project is running your model(s) and trying to understand the results. To do so effectively, you need to plan ahead before doing the runs, since just trying different things to see what happens can be a very inefficient way of attempting to learn about your models (and hopefully the systems) behaviors. Careful planning of how you are going to experiment with your model(s) will generally repay

big dividends in terms of how effectively you learn about the system(s) and how you can exercise your model(s) further.

This modules looks at such *experimental-design* issues in the broad context of a simulation project. The term experimental design has specific connotations in its traditional interpretation, and I will mention some of these. But I will also try to cover the issues of planning your simulations in a broader context, which consider the special challenges and opportunities you have when conducting a computer-based simulation experiment rather than a physical experiment. This includes questions of the overall purpose of the project, what the output performance measures should be, how you use the underlying random numbers, measuring how changes in the inputs might affect the outputs, and searching for some your experiments. For instance, if there is just one system of interest to analyze and understand, then there still could be questions like run length, the number of runs, allocation of random numbers, and interpretation of results, but there are no questions of which model configurations to run. Likewise, if there are just a few model configurations of interest, and they have been given to you (or are obvious), then the problem of experimental-design is similar to the single configuration situation. However, if you are interested more generally in how changes in the inputs affect the outputs, then there clearly are questions of which configurations to run, as well as the questions mentioned in the previous paragraph. Likewise, if you.re searching for a configuration of inputs that Maximizes or minimizes some key output performance measure, you need to decide very carefully which configurations you will run (and which ones you wont).The reality is that often you cant be completely sure what your ultimate goals are until you get into a bit. Often, your goals may change as you go along, generally becoming more ambitious as you work with your models and learn about their behavior. The good news is that as your goals become more ambitious, what you learned from your previous experiments can help you decide how to proceed with your future experiments.

1.3 Reasons for using simulation

Simulation can be used for the following reasons

1. Simulation enables the study of and experimentation with the internal interactions of a complex system, or of a subsystem within a complex system.
2. Informational, organisational and environmental changes can be simulated and the effect of these alterations on the model's behaviour can be observed.
3. The knowledge gained in designing a simulation model may be of great value toward suggesting improvements in the system under investigation.
4. By changing simulation inputs and observing the resulting outputs, valuable insight may be obtained as to which variables are most important and how the variables interact.
5. Simulation can be used as a pedagogical device to reinforce analytic solution methodologies.
6. Simulation can be used to experiment with new designs or policies prior to implementation, so as to prepare for what may happen.
7. Simulation can be used to verify analytic solutions.
8. A simulation study can help in understanding how the system operates rather than how individuals think the system operates.
9. "What if " questions can be answered. This is particularly useful in the design of new systems.

1.4 Advantages and Disadvantages of simulation

Advantages of simulation

One of the primary advantages of simulators is that they are able to provide users with practical feedback when designing real world systems. This allows the designer to determine the correctness and efficiency of a design before the system is actually constructed. Consequently, the user may explore the merits of alternative designs without actually physically building the systems. By investigating the effects of specific

design decisions during the design phase rather than the construction phase, the overall cost of building the system diminishes significantly. As an example, consider the design and fabrication of integrated circuits. During the design phase, the designer is presented with a myriad of decisions regarding such things as the placement of components and the routing of the connecting wires. It would be very costly to actually fabricate all of the potential designs as a means of evaluating their respective performance. Through the use of a simulator, however, the user may investigate the relative superiority of each design without actually fabricating the circuits themselves. By mimicking the behaviour of the designs, the circuit simulator is able to provide the designer with information pertaining to the correctness and efficiency of alternate designs. After carefully weighing the ramifications of each design, the best circuit may then be fabricated.

Another benefit of simulators is that they permit system designers to study a problem at several different levels of abstraction. By approaching a system at a higher level of abstraction, the designer is better able to understand the behaviours and interactions of all the high level components within the system and is therefore better equipped to counteract the complexity of the overall system. This complexity may simply overwhelm the designer if the problem had been approached from a lower level. As the designer better understands the operation of the higher level components through the use of the simulator, the lower level components may then be designed and subsequently simulated for verification and performance evaluation. The entire system may be built based upon this ``top-down" technique. This approach is often referred to as *hierarchical decomposition* and is essential in any design tool and simulator which deals with the construction of complex systems. For example, with respect to circuits, it is often useful to think of a microprocessor in terms of its registers, arithmetic logic units, multiplexors and control units. A simulator which permits the construction, interconnection and subsequent simulation of these higher level entities is much more useful than a simulator which only lets the designer build and connect simple logic gates. Working at a higher level abstraction also facilitates *rapid prototyping* in which

preliminary systems are designed quickly for the purpose of studying the feasibility and practicality of the high-level design.

Thirdly, simulators can be used as an effective means for teaching or demonstrating concepts to students. This is particularly true of simulators that make intelligent use of computer graphics and animation. Such simulators dynamically show the behaviour and relationship of all the simulated system's components, thereby providing the user with a meaningful understanding of the system's nature. Consider again, for example, a circuit simulator. By showing the paths taken by signals as inputs are consumed by components and outputs are produced over their respective fan out, the student can actually see what is happening within the circuit and is therefore left with a better understanding for the dynamics of the circuit. Such a simulator should also permit students to speed up, slow down, stop or even reverse a simulation as a means of aiding understanding. This is particularly true when simulating circuits which contain feedback loops or other operations which are not immediately intuitive upon an initial investigation.

During the presentation of the design and implementation of the simulator in this report, it will be shown how the above positive attributes have been or can be incorporated both in the simulator engine and its user interface.

Disadvantages of simulation

Despite the advantages of simulation presented above, simulators, like most tools, do have their drawbacks. Many of these problems can be attributed to the computationally intensive processing required by some simulators. As a consequence, the results of the simulation may not be readily available after the simulation has started -- an event that may occur instantaneously in the real world may actually take hours to mimic in a simulated environment. The delays may be due to an exceedingly large number of entities being simulated or due to the complex interactions that occur between the

entities within the system being simulated. Consequently, these simulators are restricted by limited hardware platforms which cannot meet the computational demands of the simulator. However, as more powerful platforms and improved simulation techniques become available, this problem is becoming less of a concern.

One of the ways of combating the aforementioned complexity is to introduce simplifying assumptions or heuristics into the simulator engine. While this technique can dramatically reduce the simulation time, it may also give its users a false sense of security regarding the accuracy of the simulation results. For example, consider a circuit simulator which makes the simplifying assumption that a current passing through one wire does not adversely affect current flowing in an adjacent wire. Such an assumption may indeed reduce the time required for the circuit simulator to generate results. However, if the user places two wires of a circuit too close together during the design, the circuit, when fabricated may fail to operate correctly due to electromagnetic interference between the two wires. Even though the simulation may have shown no anomalies in a design, the circuit may still have flaws.

Another means of dealing with the computational complexity is to employ the hierarchical approach to design and simulation so as to permit the designer to operate at a higher level of design. However, this technique may introduce its own problems as well. By operating at too high an abstraction level, the designer may tend to oversimplify or even omit some of the lower level details of the system. If the level of abstraction is too high, then it may be impossible to actually build the device physically due to the lack of sufficiently detailed information within the design. Actual construction of the system will not be able to occur until the user provides low level information concerning the system's subcomponents. With respect to circuit design and fabrication, work is currently on going in the field of *silicon compilers* which are able to convert high level designs of circuits and translate them accurately and efficiently into low level designs suitable for fabrication.

Further reading

Chapter Review Questions

1. Explain the following terms
    a. System
    b. Model
    c. Simulation
2. Explain any three types of systems
3. Outline any three application areas of simulation and modeling
4. State any two advantages and disadvantages of simulation and modeling

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition

CHAPTER TWO

SYSTEM MODELLING

Learning Objectives

By the end of this chapter the learner shall be able to;

Describe the types of system models

Explain the importance of system modeling

Describe the system modeling process

Describe the application areas of simulation and modeling

2.1 Introduction

Systems modeling is the scientific application and of the use of developed or generic models to conceptualize and construct systems in business and IT environment. A common type of systems modeling is function modeling and operation research , with specific techniques such as the Functional Flow Block Diagram . System modeling provides a precise and abstract way of specifying the informational and time characteristics of a data processing problem, and to created a notation that should enable the system analyst to organize the problem around any piece of hardware.

2.2 Types of Systems Modeling

In business and IT development systems are modeled with different scoops and scales of complexity, such as:

- Functional modeling
- Business process modeling
- Enterprise modeling

2.2.1 Functional Modeling

A function model or functional model in systems engineering and software engineering is a structured representation of the functions (activities, actions, processes, operations) within the modeled system or subject area. A function model, also called an activity model or process model, provides a graphical representation of an enterprise's function within a defined scope. The purposes of the function model are to describe the functions and processes, assist with discovery of information needs, help identify opportunities, and establish a basis for determining product and service costs.

2.1.1.1 Functional Perspective

In systems engineering a function model is created with a functional modeling perspective. The functional perspective is one of the perspectives for example behavioural, organisational or informational.A functional modeling perspective concentrates on describing the dynamic process. The main concept in this modeling perspective is the process, this could be a function, transformation, activity, action, task etc. A well-known example of a modeling language employing this perspective is The perspective uses four symbols to describe a process, these being: This decomposed process is a DFD, data flow diagram.

a. Process: Illustrates transformation from input to output.
b. Store: Data-collection or some sort of material.
c. Flow: Movement of data or material in the process.
d. External Entity: External to the modeled system, but interacts with it.

2.1.1.2 Functional decomposition

This refers broadly to the process of resolving a functional relationship into its constituent parts in such a way that the original function can be reconstructed from those parts by function composition. In general, this process of decomposition is undertaken either for the purpose of gaining insight into the identity of the constituent

components, or for the purpose of obtaining a compressed representation of the global function, a task which is feasible only when the constituent processes possess a certain level of modularity. Functional decomposition has a prominent role in computer programming, where a major goal is to modularize processes to the greatest extent possible. For example, a school management system may be broken up into an inventory module, a patron information module, and a fee assessment module. Functional decomposition of engineering systems is a method for analyzing engineered systems. The basic idea is to try to divide a system in such a way that each block of the block diagram can be described without an "and" or "or" in the description.

2.1.1.1.3. *Functional modeling methods*

2.1.1.1.3.1. Functional Flow Block Diagram

The Functional flow block diagram (FFBD) is a multi-tier, time-sequenced, step-by-step flow diagram of the system's functional flow. The diagram is widely used in classical systems engineering. The Functional Flow Block Diagram is also referred to as Functional Flow Diagram, functional block diagram, and functional flow.Functional Flow Block Diagrams (FFBD) usually define the detailed, step-by-step operational and support sequences for systems, but they are also used effectively to define processes in developing and producing systems.. In the FFBD method, the functions are organized and depicted by their logical order of execution. Each function is shown with respect to its logical relationship to the execution and completion of other functions. A node labeled with the function name depicts each function. Arrows from left to right show the order of execution of the functions. Logic symbols represent sequential or parallel execution of functions.

2.1.1.1.3.2 Structured Analysis and Design Technique

SAD is a software engineering methodology for describing systems as a hierarchy of functions, a diagrammatic notation for constructing a sketch for a software application. It offers building blocks to represent entities and activities, and a variety of arrows to

relate boxes. These boxes and arrows have an associated informal semantics. SAD can be used as a functional analysis tool of a given process, using successive levels of details. The SADT method allows to define user needs for IT developments, which is used in industrial Information Systems, but also to explain and to present an activity's manufacturing processes, procedures.

The SAD supplies a specific functional view of any enterprise by describing the functions and their relationships in a company. These functions fulfill the objectives of a company, such as sales, order planning, product design, part manufacturing, and human resource management. The SAD can depict simple functional relationships and can reflect data and control flow relationships between different functions.

2.1.1.1.3.3. Axiomatic Design

Axiomatic design is a top down hierarchical functional decomposition process used as a solution synthesis framework for the analysis, development, re-engineering, and integration of products, information systems, business processes or software engineering solutions. Its structure is suited mathematically to analyze coupling between functions in order to optimize the architectural robustness of potential functional solution models.

2.1.1.1.3.4 Business function model

A Business Function Model (BFM) is a general description or category of operations performed routinely to carry out an organization's mission. It can show the critical business processes in the context of the business area functions. The processes in the business function model must be consistent with the processes in the value chain models. Processes are a group of related business activities performed to produce an end product or to provide a service. Unlike business functions that are performed on a continual basis, processes are characterized by the fact that they have a specific beginning and an end point marked by the delivery of a desired output. The figure on

the right depicts the relationship between the business processes, business functions, and the business area's business reference model.

2.1.1.1.3.5 Business reference model

This is a reference model, concentrating on the functional and organizational aspects of the core business of an enterprise, service organization or government agency. In enterprise engineering a business reference model is part of an Enterprise Architecture Framework or Architecture Framework, which defines how to organize the structure and views associated with an Enterprise Architecture. A reference model in general is a model of something that embodies the basic goal or idea of something and can then be looked at as a reference for various purposes. A business reference model is a means to describe the business operations of an organization, independent of the organizational structure that perform them. Other types of business reference model can also depict the relationship between the business processes, business functions, and the business area's business reference model. These reference model can be constructed in layers, and offer a foundation for the analysis of service components, technology, data, and performance.

2.1.1.1.3.6 Operator function model

The Operator Function Model (OFM) is proposed as an alternative to traditional task analysis techniques used by human factors engineers. An operator function model attempts to represent in mathematical form how an operator might decompose a complex system into simpler parts and coordinate control actions and system configurations so that acceptable overall system performance is achieved. The model represents basic issues of knowledge representation, information flow, and decision making in complex systems. A network structure can be thought of as a possible representation of an operator's internal model of the system plus a control structure which specifies how the model is used to solve the decision problems that comprise operator control functions.

2.1.2 Business Process Modeling (BPM)

(BPM) in systems engineering is the activity of representing processes of an enterprise, so that the current process may be analyzed and improved. BPM is typically performed by business analysts and managers who are seeking to improve process efficiency and quality. The process improvements identified by BPM may or may not require Information Technology involvement, although that is a common driver for the need to model a business process, by creating a process master. Change management programs are typically involved to put the improved business processes into practice. With advances in technology from large platform vendors, the vision of BPM models becoming fully executable (and capable of simulations and round-trip engineering) is coming closer to reality every day.The ability of the extranet to automate the trading tasks between you and your trading partners can lead to enhanced business relationships and help to integrate your business firmly within their supply chain.

2.1.2.1 Business Model

A business model is a framework for creating economic, social, and/or other forms of value. The term 'business model' is thus used for a broad range of informal and formal descriptions to represent core aspects of a business, including purpose, offerings, strategies, infrastructure, organizational structures, trading practices, and operational processes and policies.In the most basic sense, a business model is the method of doing business by which a company can sustain itself. That is, generate revenue. The business model spells-out how a company makes money by specifying where it is positioned in the value chain.A business process is a collection of related, structured activities or tasks that produce a specific service or product (serve a particular goal) for a particular customer or customers. There are three main types of business processes:

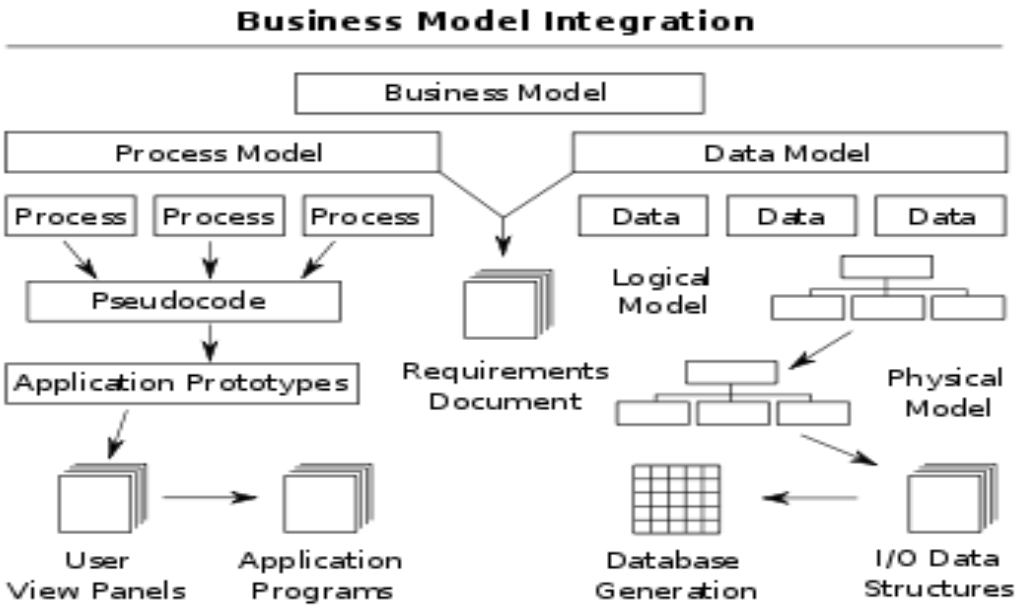
1. Management processes, the processes that govern the operation of a system. Typical management processes include "Corporate Governance" and "Strategic Management".

2. Operational processes, processes that constitute the core business and create the primary value stream. Typical operational processes are Purchasing, Manufacturing, Marketing, and Sales.
3. Supporting processes, which support the core processes. Examples include Accounting, Recruitment, Technical support.

A business process can be decomposed into several sub-processes, which have their own attributes, but also contribute to achieving the goal of the super-process. The analysis of business processes typically includes the mapping of processes and sub-processes down to activity level. A business process model is a model of one or more business processes, and defines the ways in which operations are carried out to accomplish the intended objectives of an organization. Such a model remains an abstraction and depends on the intended use of the model. It can describe the workflow or the integration between business processes. It can be constructed in multiple levels.

A workflow is a depiction of a sequence of operations, declared as work of a person, work of a simple or complex mechanism, work of a group of persons, work of an organization of staff, or machines. Workflow may be seen as any abstraction of real work, segregated in workshare, work split or whatever types of ordering. For control purposes, workflow may be a view on real work under a chosen aspect.The Artifact-centric business process model has emerged as a new promising approach for modeling business processes, as it provides a highly flexible solution to capture operational specifications of business processes. It particularly focuses on describing the data of business processes, known as “artifacts”, by characterizing business-relevant data objects, their lifecycles, and related services. The artifact-centric process modeling approach fosters the automation of the business operations and supports the flexibility of the workflow enactment and evolution.

1.1.1. 2.1.3 Enterprise Modeling



Enterprise modeling is the abstract representation, description and definition of the structure, processes, information and resources of an identifiable business, government body, or other large organization.It deals with the process of understanding an enterprise business and improving its performance through creation of enterprise models. This includes the modeling of the relevant business domain (usually relatively stable), business processes (usually more volatile), and Information technology.Enterprise modeling is the process of building models of whole or part of an enterprise with process models, data models, resource models and or new ontology etc. It is based on knowledge about the enterprise, previous models and/or reference models as well as domain ontologies using model representation languages. An enterprise in general is a unit of economic organization or activity. These activities are required to develop and deliver products and/or services to a customer. An enterprise includes a number of functions and operations such as purchasing, manufacturing, marketing, finance, engineering, and research and development. The enterprise of

interest are those corporate functions and operations necessary to manufacture current and potential future variants of a product.

The term "enterprise model" is used in industry to represent differing enterprise representations, with no real standardized definition.. Enterprise modeling constructs can focus upon manufacturing operations and/or business operations; however, a common thread in enterprise modeling is an inclusion of assessment of information technology. For example, the use of networked computers to trigger and receive replacement orders along a material supply chain is an example of how information technology is used to coordinate manufacturing operations within an enterprise.

2.1.3.1.1. Enterprise modeling basics

2.1.3.1 Enterprise model

An enterprise model is a representation of the structure, activities, processes, information, resources, people, behavior, goals, and constraints of a business, government, or other enterprises

2.1.3.2 Function modeling

Example of a function model of the process of "Maintain Reparable Spares" in IDEF0 notation.

Function modeling in systems engineering is a structured representation of the functions, activities or processes within the modelled system or subject area. A function model, also called an activity model or process model, produces a graphical representation of an enterprise's function within a defined scope. The purpose of the function model are to describe the functions and processes, assist with discovery of information needs, help identify opportunities, and establish a basis for determining product and service costs.

2.1.3.3 Data modeling

Data modeling is the process of creating a data model by applying formal data model descriptions using data modeling techniques. Data modeling is a technique for defining business requirements for a database. It is sometimes called database modeling because a data model is eventually implemented in a database.

2.1.3.4 Ontology engineering modeling

Ontology engineering or ontology building is a subfield of knowledge engineering that studies the methods and methodologies for building ontologies. In the domain of enterprise architecture, an ontology is an outline or a schema used to structure objects, their attributes and relationships in a consistent manner.As in enterprise modeling, an ontology can be composed of other ontologies. The purpose of ontologies in enterprise modeling is to formalize and establish the sharability, re-usability, assimilation and dissemination of information across all organizations and departments within an enterprise. Thus, an ontology enables integration of the various functions and processes which take place in an enterprise. Using ontologies in enterprise modeling offers several advantages. Ontologies ensure clarity, consistency, and structure to a model. They promote efficient model definition and analysis. Generic enterprise ontologies allow for reusability of and automation of components. Because ontologies are schemata or outlines, the use of ontologies does not ensure proper enterprise model definition, analysis, or clarity. Ontologies are limited by how they are defined and implemented. An ontology may or may not include the potential or capability to capture the all of the aspects of what is being modelled.

2.1.3.5 Systems thinking

The modeling of the enterprise and its environment could facilitate the creation of enhanced understanding of the business domain and processes of the extended enterprise, and especially of the relations—both those that "hold the enterprise together" and those that extend across the boundaries of the enterprise. Since

enterprise is a system, concepts used in system thinking can be successfully reused in modeling enterprises. This way a fast understanding can be achieved throughout the enterprise about how business functions are working and how they depend upon other functions in the organization.

Further reading

Chapter Review Questions

1. Differentiate between structured analysis and design and axiomatic design
2. Explain any three forms of system modeling
3. Explain any three components of DFD diagram

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition

CHAPTER THREE

CASE STUDY

Learning Objectives

By the end of this chapter the learner shall be able to;

Describe the case study scenario
Describe the components of case study model
Explain the characteristics of a model
Describe the application areas of case study model

3.1 Case study: Modeling Transport System

Modeling behavior

Most large company decisions involve cost/benefit analysis based on estimates of various parameters , such as how willing the customers are to buy a certain product. But estimation of these parameters may require very complex considerations and considerable experience with the specific domain in order to be correct .For example, a transportation service provider may wish to find out whether a reduction of the ticket prices would increase or decrease his income. Experimenting with changing the ticket prices directly may, however, be expensive and also difficult since it can harm the reputation of the service provider.

Instead, one can try to model the behaviour of the customers. The service provider might conjecture that families with low income travel by bus if it is cheaper than traveling by car, whereas families with high income will travel by bus if it is more convenient no matter what. The service provider can then find the number of low-income and high-income families and examine the prices for gasoline and the average travel durations to calculate an estimate.

The introduction of a model shifts the burden of determining parameters for the decision itself to determining the parameters of the model, which in some situations may be much easier and cheaper. It does also introduce the problem of ensuring that the model is valid in itself, however.

3.1.2 Specification of models

So how can one specify a model? The goal of this project is to help that.

One approach is to express the relations between the objects in terms of mathematical functions. For instance, the transportation service provider may assume that the fraction of low-income families which travel by bus depends linearly on the ticket price between the two endpoints, everyone traveling by bus and everyone traveling by car. A mathematical description is in so far good enough, but the specification of a model is only part of the process. Another part is that of retrieving data from the model. If the amount of data to retrieve is large, this retrieval is best done automatically which requires the model to be specified with a strictly formal language.

The model can then be run through a simulator that processes the description and outputs the needed data.

3.1.3 Basic model characteristics

Some fundamentally different options exist for constructing the model. One can choose a continuous time approach or model time with discrete events so that each state is calculated from the previous state. And when modeling many similar objects, one can either actually construct all these objects and simulate their individual behaviour, or only construct one and let it exhibit an average behaviour. The best choice is not obvious and may depend on the situation, but we have chosen to support discrete-event simulation of multiple objects. One important advantage of formulating the models as discrete events is that it makes it possible to express dependencies on past events directly. For instance, with the transportation model if someone in a family has borrowed the car for a couple of days, the car is not available which affects the decision process. With a discrete model, we can easily keep track of such situations, whereas a continuous model would have to concern itself with how they affect the average case behaviour. Also using as many objects as modeled instead of only average-case ones lets one focus on modeling a particular object of interest, splitting individual differences into different cases, instead of having to think about all at once to capture the average case behaviour. A more sophisticated analysis of the data generated by the model may

also be concerned with more than just the averages in which case we must have data from individual objects.

3.2 A railroad company case

Based on the observations in the previous sections, this section will develop the fundamentals of a formal modeling language for simulation in parallel with presenting a model of some of the customers of a railroad company. The model assumes that the persons can either go by train or by car and incorporates three kinds of persons who often travel by train: commuters, business men (e.g. marketing or sales persons) and students visiting their family. Commuters need to travel each workday and we will characterize them as wanting low price

and short travel times since traveling takes up much of their lives. Business persons do not care much about the price since the company pays the bill and presumably they do not mind a long journey provided they are able to work meanwhile. They will travel only on workdays but not every day. The students usually only travel during the week-ends, i.e. Friday to Monday (we will not consider holidays or vacations).They are concerned about price, but not as much about the duration of the journey.For specification of the types, we will borrow the class syntax from C++/Java:

```
type Commuter {
type Commuter {
//variabledefinitions
//functiondefinitions
void iterate(int iteration)
{
//processobject
}
}
```

In this and the following code snippets, reserved words are in bold face. The iterate method is called once for each iteration and is responsible for updating the state of the

object. We can then extend the usual C variable definition syntax with an extra qualifier **watched** which causes the member variable to be output after each iteration:

```
type Commuter {
watched bool train; //go by train?
//...
}
```

This simple structure realizes the discrete event model and at the same time allows us to easily define what output we want from the simulation.

3.2 .1 The three models

So sticking to the notation of the C family of languages, we can define a model ofa commuter:

```
type Commuter {
float income = 50000...500000; //randomly selected
watched bool train;

void iterate(int iteration)
{
int weekday = iteration % 7;
if( weekday < 5 ) {
float prob;

float price = price_car/(price_train + price_car);
float time = time_car/(time_train + time_car);

if( income < 150000 )
prob = 0.7_price + 0.3_time;
else if( income < 300000 )
prob = 0.3_price + 0.7_time;
else
```

```
prob = t ime ;

float random = 0 . 0 . . . 1 . 0 ;
t r a i n = random < prob ;
}
else
t r a i n = false ;
}
}
```

On line 2, the model defines an income in the range min_income to max_income; the binary operator ... returns a random number linearly distributed in the range denoted by the two arguments. Then it defines an output boolean variable for indicating whether the person went by train. The iterate method checks whether it is a workday (line 8); if so, the probability
of the person choosing the train is calculated by segmenting the commuters into three groups depending on income (line 14, 16 and 18) and setting weights for the dependency on price and journey duration. Finally a random value is generated and the outcome is chosen from that (line 22). The calculations depend on some variables that are not defined above, the cost of traveling by train or car and the respective journey durations. These depend on the particular person – we will later discuss how to incorporate them.A simple model of a business man is:

```
type Bus ines sMan {
watched bool t r a i n ;

void i t e r a t e ( int iteration )
{
int weekday = i t e r a t i o n % 7 ;

if ( weekday < 5 && 0 . 0 . . . 1 . 0 < t r a v e l _ p r o b ) {
```

```
if(time_car<0.5)//less than half an hour
train=false;
elseif(time_car>3)//more than three hours
train=true;
else{
float base_prob=(time_car□0.5)/(3□0.5);
float fraction=time_train/time_car;
float final_prob=base_prob^fraction;

train=0.0...1.0<final_prob;
}
}
else//do not travel today
train=false;
}
}
```

The model assumes that travel_prob has been defined. On line 14, the base probability is linearly interpolated between half an hour (always go by car) and three hours (always go by train). On line 16, the final probability is then skewed by raising it to the power of the relative journey durations between going by train and going by car (^ is the power operator); Figure 1.1 illustrates the effect.A model of a student is:

```
type Student{
 int max_weeks ;
 int departure_day,return_day;
 watched bool train;
 int last_visit=0;

 void Student()

{
```

```
//expensive travel implies staying away longer
max_weeks = price_train/100+(1...12);
//set off Friday, Saturday or Sunday
departure_day=0.6:4|0.3:5|0.1:6;
//return Saturday, Sunday or Monday
return_day=((departure_day+1)...7)%7;
}

void iterate(int iteration)
{
int weekday = iteration % 7;
float prob = price_car/(price_train+price_car);

train=false;

if( weekday==departure_day){
float return_prob=last_visit/max_weeks;
if(0.0...1.0<return_prob){
train=0.0...1.0<prob;
last_visit=0;

}
else
++last_visit;
}

if( weekday==return_day&&last_visit==0)
train=0.0...1.0<prob;
}
```

```
}
```

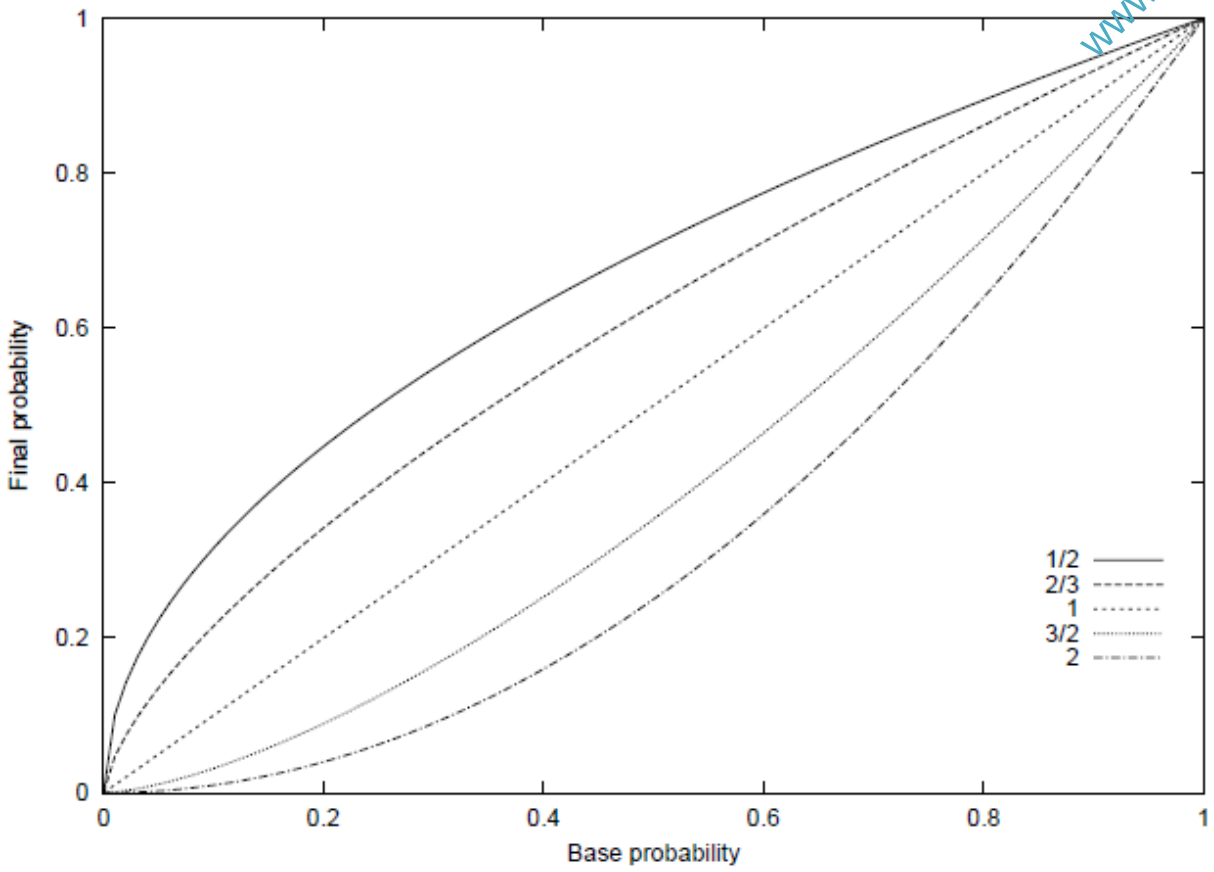


Figure 1. Illustration of pfinal = pt
base for some values of t where t = ttrain/tcar.
When k > 1, the probability is skewed towards zero; vice versa for k < 1. When k = 1, the final probability is simply the base probability.

For the student we introduce a constructor (line 7) which initializes each object with a maximum number of weeks (line 10) before returning home (based on the ticket prices, adding 100 to the ticket price will enlarge the possible maximum by one week) and chooses the week days for departure and return (line 12 and 14).The decision about whether to depart a specific week is then determined by max_weeks and the amount of time elapsed since last visit (line 25). Whether to go by train is probabilistically determined by the relative prices of traveling by car and

by train (line 20).The constructor uses the special notation *e*1 : *ea* j *e*2 : *eb* j . which makes a non-deterministic choice between the expressions *ea*, *eb*, . . . using *e*1, *e*2, . . . as weights. So the value of the whole expression is either *ea* with probability
*e*1/(*e*1 + *e*2 + . . .) or *eb* with probability *e*2/(*e*1 + *e*2 + . . .), etc.

Using the models for simulation

The goal of specifying the above models is to simulate their behaviour for a given period of time, e.g. a month. For this we need instances of each of the models. Thus the basic setup consists of instructing the simulator to realize a number of objects of the different types:

```
create 5 0 0 0 0 0 of Commuter ;
create 1 0 0 0 0 0 of Bus ines sMan ;
create 1 0 0 0 0 0 of S t u d e n t ;
```

To actually get a working simulation, we need to specify the times and prices for traveling by train and car in the models, though. In general, some of the parameters that we wish to control are specific for each instantiated object of a model and as such parameterise that model specification, and some are the same for all objects. For instance, if a model were to take the
weather into account, the controlling parameter would be the same for all objects. We can model this kind of parameters with global variables defined outside the types, e.g.:

```
float weather_factor = 17.3; //globalvariable
type WeatherDependent {
void iterate(int iteration)
{
if(weather_factor > 10.0)
//...
}
}
```

The missing variables for the three models we have developed above all fall into the category of parameterising the models, however. Since we have introduced a constructor for **type**s, we can conveniently pass the parameters through that. Theconstructor for BusinessMan would then be:

```
type Bus ines sMan {
f l o a t t ime _ t r a i n , t ime _ c a r , t r a v e l _ p r o b a b i l i t y ;
//...
void Bus ines sMan ( f l o a t t r a i n , f l o a t c a r , f l o a t t r a v e l )
{
t ime _ t r a i n = t r a i n ;
t ime _ c a r = c a r ;
t r a v e l _ p r o b a b i l i t y = t r a v e l ;
}
}
```

Hence, we need to change the create statements:

```
c r e a t e 2 0 0 0 0 of
Bus ines sMan ( 1 . 0 . . . 1 . 3 , 0 . 8 . . . 1 . 1 , 0 . 1 . . . 0 . 6 ) ;
c r e a t e 2 0 0 0 0 of
Bus ines sMan ( 2 . 0 . . . 2 . 5 , 1 . 8 . . . 2 . 3 , 0 . 1 . . . 0 . 6 ) ;
//...
```

These create statements could easily be generated from collected real-world data.

3.3 Organising the models and the simulation setup

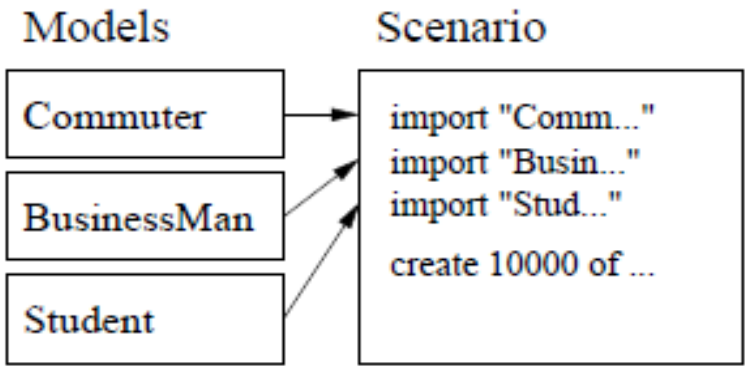
The simplest approach for physically organising the code for the models and the simulation setup is to put everything into one file. But this does not scale well:

I. The single file may become very large which makes it difficult to manage and navigate.

II. Reusing models for different scenarios is only possible by copying and pasting the code.

Instead we can break the file up into smaller files by adding a directive for importing definitions from other files:

import " o t h e r . model " ;This means that all definitions of types, functions and variables in "other.model" are parsed and imported into the symbol tables of the processing of the current file.Name clashes are considered errors. With the import directive, a reasonable way of organising the simulation would be to put the model code in separate ".model" files which are imported in a main ".scenario" file that contains the necessary create statements. The setup is illustratedin Figure 1.2. This way it is also easier to generate the ".scenario" files from an external source with real-world data.



Another problem is that global variables in a model need to be known prior to their use. For example, type WeatherDependent cannot use the global variable weather_factor if

weather_factor has not been defined before type WeatherDependent. We can remedy this by being careful with ordering the definitions and import directives:

```
float weather_factor = 15.0;
import "weather_dependent.model";
```

However, this makes the modularization much more fragile. Instead we will introduce declarations which are simply definitions with the body removed and **extern** in front:

```
extern float weather_factor;
extern float max(float x, float y);
```

Putting these in the model files corrects the problem, and also makes it explicit that the identifiers are global. Note that the purpose of a declaration is to make a name (with its type) known to the simulator, whereas a definition also has the actual data associated with it. Thus there is no limit on the number of declarations as long as they all agree on the type of the name they are declaring, but there must be exactly one definition. If a model that refers to a declared, but undefined identifier is created, it is an error.

Further reading

Chapter Review Questions

1. Explain the basic guidelines you would follow while choosing a model

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition

CHAPTER FOUR

Probability theory

Learning Objectives

By the end of this chapter the learner shall be able to;

Explain the probability theory

Describe the continuous probability theory

Describe the discrete probability theory

Calculate the probability theory in a model

4.1 Introduction

Probability theory is the branch of mathematics concerned with probability, the analysis of random phenomena. The central objects of probability theory are random variables, stochastic processes, and events. If an individual coin toss or the roll of dice is considered to be a random event, then if repeated many times the sequence of random events will exhibit certain patterns, which can be studied and predicted. As a mathematical foundation for statistics, probability theory is essential to many human activities that involve quantitative analysis of large sets of data. Methods of probability theory also apply to descriptions of complex systems given only partial knowledge of their state, as in statistical mechanics Consider an experiment that can produce a number of outcomes. The collection of all results is called the sample space of the experiment. The power set of the sample space is formed by considering all different collections of possible results. For example, rolling a die produces one of six possible results. One collection of possible results corresponds to getting an odd number. Thus, the subset {1,3,5} is an element of the power set of the sample space of die rolls. These collections are called events. In this case, {1,3,5} is the event that the die falls on some odd number. If the results that actually occur fall in a given event, that event is said to have occurred. Probability is a way of assigning every "event" a value between zero and one, with the requirement that the event made up of all possible results (in our example, the event {1,2,3,4,5,6}) be assigned a value of one. To qualify

as a probability distribution, the assignment of values must satisfy the requirement that if you look at a collection of mutually exclusive events (events that contain no common results, e.g., the events {1,6}, {3}, and {2,4} are all mutually exclusive), the probability that at least one of the events will occur is given by the sum of the probabilities of all the individual events.The probability that any one of the events {1,6}, {3}, or {2,4} will occur is 5/6. This is the same as saying that the probability of event {1,2,3,4,6} is 5/6. This event encompasses the possibility of any number except five being rolled. The mutually exclusive event {5} has a probability of 1/6, and the event {1, 2, 3, 4, 5, and 6} has a probability of 1 - absolute certainty. For convenience's sake, we ignore the possibility that the die, once rolled, will be obliterated before it can hit the table.

4.2 Discrete Probability Distributions

Discrete probability theory deals with events that occur in countable sample spaces. The major examples: Throwing dice, experiments with decks of cards, and random walk. Initially the probability of an event to occur was defined as number of cases favorable for the event, over the number of total outcomes possible in an equiprobable sample space: see Classical definition of probability. For example, if the event is "occurrence of an even number when a die is rolled", the probability is given by $\frac{3}{6} = \frac{1}{2}$, since 3 faces out of the 6 have even numbers and each face has the same probability of appearing. The modern definition starts with a finite or countable set called the sample space, which relates to the set of all possible outcomes in classical sense, denoted by Ω. It is then assumed that for each element $x \in \Omega$, an intrinsic "probability" value $f(x)$is attached, which satisfies the following properties:

1. $f(x) \in [0, 1] \text{ for all } x \in \Omega\,;$
2. $\sum_{x \in \Omega} f(x) = 1\,.$

That is, the probability function f(x) lies between zero and one for every value of x in the sample space Ω, and the sum of f(x) over all values x in the sample space Ω is equal to 1. An event is defined as any subset $E$of the sample space Ω. The probability of the event $E$is defined as

$$P(E) = \sum_{x \in E} f(x)\,.$$

So, the probability of the entire sample space is 1, and the probability of the null event is 0.

The function $f(x)$ mapping a point in the sample space to the "probability" value is called a probability mass function abbreviated as pmf. The modern definition does not try to answer how probability mass functions are obtained; instead it builds a theory that assumes their existence.

4.2.1 Continuous Probability Distributions

Continuous probability theory deals with events that occur in a continuous sample space.

Classical definition: The classical definition breaks down when confronted with the continuous case. If the outcome space of a random variable X is the set of real numbers ($\mathbb{R}$) or a subset thereof, then a function called the cumulative distribution function (or cdf) F exists, defined by $F(x) = P(X \leq x)$. That is, F(x) returns the probability that X will be less than or equal to x.

The cdf necessarily satisfies the following properties.

1. F is a monotonically non-decreasing, right-continuous function;
2. $\lim_{x \to -\infty} F(x) = 0$;
3. $\lim_{x \to \infty} F(x) = 1$.

If F is absolutely continuous, i.e., its derivative exists and integrating the derivative gives us the cdf back again, then the random variable X is said to have a probability density function or pdf or simply density $f(x) = \frac{dF(x)}{dx}$.

For a set $E \subseteq \mathbb{R}$, the probability of the random variable X being in E is

$$P(X \in E) = \int_{x \in E} dF(x)\,.$$

In case the probability density function exists, this can be written as

$$P(X \in E) = \int_{x \in E} f(x)\,dx\,.$$

Whereas the pdf exists only for continuous random variables, the cdf exists for all random variables (including discrete random variables) that take values in $\mathbb{R}$.

These concepts can be generalized for multidimensional cases on $\mathbb{R}^n$ and other continuous sample spaces. The measure-theoretic treatment of probability is that it unifies the discrete and the continuous cases, and makes the difference a question of which measure is used. Furthermore, it covers distributions that are neither discrete nor continuous nor mixtures of the two.

An example of such distributions could be a mix of discrete and continuous distributions —for example, a random variable that is 0 with probability 1/2, and takes a random value from a normal distribution with probability 1/2. It can still be studied to some extent by considering it to have a pdf of (δ[x] + φ(x)) / 2, where δ[x] is the Dirac delta function.

Other distributions may not even be a mix, for example, the Cantor distribution has no positive probability for any single point, neither does it have a density. The modern approach to probability theory solves these problems using measure theory to define the probability space:

Given any set Ω, (also called sample space) and a σ-algebra $\mathcal{F}$ on it, a measure P defined on $\mathcal{F}$ is called a probability measure if $P(\Omega) = 1.$

If $\mathcal{F}$ is the Borel σ-algebra on the set of real numbers, then there is a unique probability measure on $\mathcal{F}$ for any cdf, and vice versa. The measure corresponding to a cdf is said to be induced by the cdf. This measure coincides with the pmf for discrete variables, and pdf for continuous variables, making the measure-theoretic approach free of fallacies.

The probability of a set E in the σ-algebra $\mathcal{F}$ is defined as

$$P(E) = \int_{\omega \in E} \mu_F(d\omega)$$

where the integration is with respect to the measure μ_F induced by F.

Along with providing better understanding and unification of discrete and continuous probabilities, measure-theoretic treatment also allows us to work on probabilities outside $\mathbb{R}^n$, as in the theory of stochastic processes. For example to study Brownian motion, probability is defined on a space of functions.

Chapter Review Questions

1. Define the term probability theory
2. Explain any two forms of probability distributions
3. Explain any three conditions that must be satisfied by continuous probability distribution

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition.

CHAPTER FIVE

PROBABILITY DISTRIBUTION

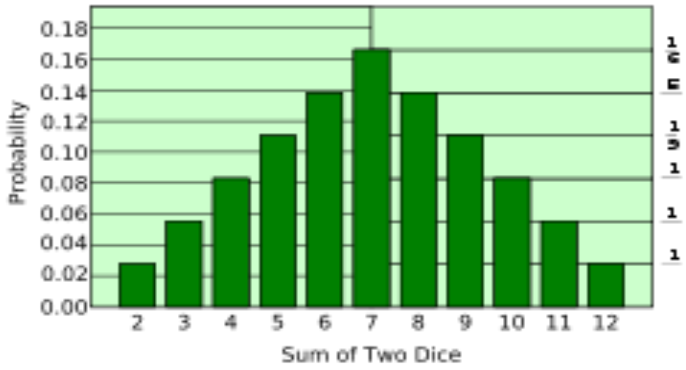
Learning Objectives

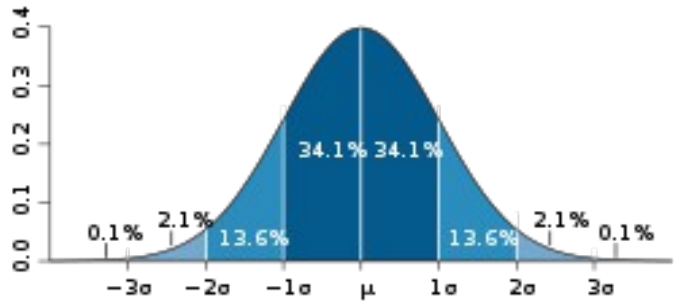
By the end of this chapter the learner shall be able to;

Explain the probability distribution process
Describe the discrete probability distribution theory
Describe the properties of probability distribution theory
Describe the application of probability distribution theory

5.1 Definition of Probability distribution

Probability distribution is a function that describes the probability of a random variable taking certain values. For a more precise definition one needs to distinguish between discrete and continuous random variables. In the discrete case, one can easily assign a probability to each possible value: when throwing a die, each of the six values 1 to 6 has the probability 1/6. In contrast, when a random variable takes values from a continuum, probabilities are nonzero only if they refer to finite intervals: in quality control one might demand that the probability of a "500 g" package containing between 500 g and 510 g should be no less than 98%.

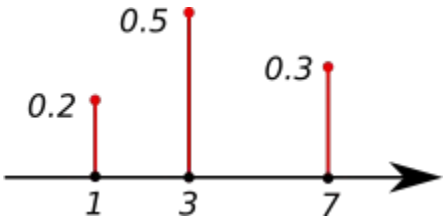




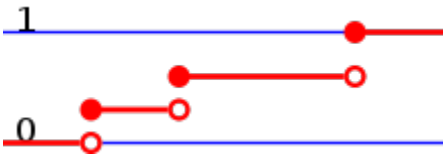
Normal distribution, also called Gaussian or "bell curve", the most important continuous random distribution. If total order is defined for the random variable, the cumulative distribution function gives the probability that the random variable is not larger than a given value; it is the integral of the non-cumulative distribution.

5.2 Discrete probability distribution

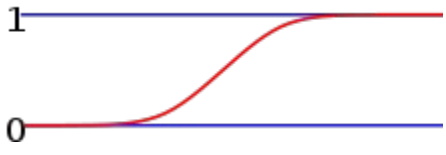
It can also be referred to as Probability mass function and Categorical distribution



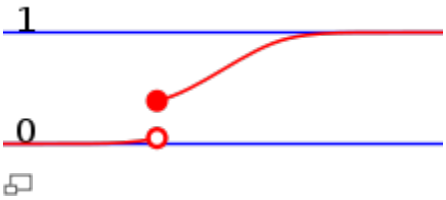
The probability mass function of a discrete probability distribution. The probabilities of the singletons {1}, {3}, and {7} are respectively 0.2, 0.5, 0.3. A set not containing any of these points has probability zero.



The cdf of a discrete probability distribution, ...



... of a continuous probability distribution, ...



... of a distribution which has both a continuous part and a discrete part.

A discrete probability distribution shall be understood as a probability distribution characterized by a probability mass function. Thus, the distribution of a random variable X is discrete, and X is then called a discrete random variable, if

$$\sum_{u} \Pr(X = u) = 1$$

as u runs through the set of all possible values of X. It follows that such a random variable can assume only a finite or count ably infinite number of values.In cases more frequently considered, this set of possible values is a topologically discrete set in the sense that all its points are isolated points. But there are discrete random variables for which this countable set is dense on the real line (for example, a distribution over rational numbers).Among the most well-known discrete probability distributions that are used for statistical modeling are the Poisson distribution, the Bernoulli distribution, the binomial distribution, the geometric distribution, and the negative binomial distribution. In addition, the discrete uniform distribution is commonly used in computer programs that make equal-probability random selections between a number of choices.

5.3 Cumulative density

Equivalently to the above, a discrete random variable can be defined as a random variable whose cumulative distribution function (cdf) increases only by jump discontinuities—that is, its cdf increases only where it "jumps" to a higher value, and is constant between those jumps. The points where jumps occur are precisely the values which the random variable may take. The number of such jumps may be finite or countably infinite. The set of locations of such jumps need not be topologically discrete; for example, the cdf might jump at each rational number. Consequently, a discrete

probability distribution is often represented as a generalized probability density function involving Dirac delta functions, which substantially unifies the treatment of continuous and discrete distributions. This is especially useful when dealing with probability distributions involving both a continuous and a discrete part.

Indicator-function representation

For a discrete random variable X, let u_0, u_1, ... be the values it can take with non-zero probability. Denote

$$\Omega_i = \{\omega : X(\omega) = u_i\}, \, i = 0, 1, 2, \ldots$$

These are disjoint sets, and by formula (1)

$$\Pr\left(\bigcup_i \Omega_i\right) = \sum_i \Pr(\Omega_i) = \sum_i \Pr(X = u_i) = 1.$$

It follows that the probability that X takes any value except for u_0, u_1, ... is zero, and thus one can write X as

$$X = \sum_i u_i 1_{\Omega_i}$$

except on a set of probability zero, where 1_A is the indicator function of A. This may serve as an alternative definition of discrete random variables.

5.4 Continuous probability distribution

A continuous probability distribution is a probability distribution that has a probability density function. It is also call such distribution absolutely continuous, since its cumulative distribution function is absolutely continuous with respect to the Lebesgue measure λ. If the distribution of X is continuous, then X is called a continuous random variable. There are many examples of continuous probability distributions: normal, uniform, chi-squared. Intuitively, a continuous random variable is the one which can

take a continuous range of values — as opposed to a discrete distribution, where the set of possible values for the random variable is at most countable. While for a discrete distribution an event with probability zero is impossible (e.g. rolling 3½ on a standard die is impossible, and has probability zero), this is not so in the case of a continuous random variable. For example, if one measures the width of an oak leaf, the result of 3½ cm is possible, however it has probability zero because there are uncountably many other potential values even between 3 cm and 4 cm. Each of these individual outcomes has probability zero, yet the probability that the outcome will fall into the interval (3 cm, 4 cm) is nonzero. This apparent paradox is resolved by the fact that the probability that X attains some value within an infinite set, such as an interval, cannot be found by naively adding the probabilities for individual values. Formally, each value has an infinitesimally small probability, which statistically is equivalent to zero.

Formally, if X is a continuous random variable, then it has a probability density function $f(x)$, and therefore its probability to fall into a given interval, say [a, b] is given by the integral

$$\Pr[a \leq X \leq b] = \int_a^b f(x)\,dx$$

In particular, the probability for X to take any single value a (that is a ≤ X ≤ a) is zero, because an integral with coinciding upper and lower limits is always equal to zero.

The definition states that a continuous probability distribution must possess a density, or equivalently, its cumulative distribution function be absolutely continuous. This requirement is stronger than simple continuity of the cdf, and there is a special class of distributions, singular distributions, which are neither continuous nor discrete nor their mixture. An example is given by the Cantor distribution. Such singular distributions however are never encountered in practice..By one convention, a probability distribution $\mu$is called continuous if its cumulative distribution function $F(x) = \mu(-\infty, x]$is continuous and, therefore, the probability measure of singletons $\mu\{x\} = 0$for all x

.Another convention reserves the term continuous probability distribution for absolutely continuous distributions. These distributions can be characterized by a probability density function: a non-negative Lebesgue integrable function f defined on the real numbers such that

$$F(x) = \mu(-\infty, x] = \int_{-\infty}^{x} f(t)\,dt.$$

Discrete distributions and some continuous distributions (like the Cantor distribution) do not admit such a density.

5.5 Properties of probability theory

- The probability density function of the sum of two independent random variables is the convolution of each of their density functions.
- The probability density function of the difference of two independent random variables is the cross-correlation of their density functions.
- Probability distributions are not a vector space – they are not closed under linear combinations, as these do not preserve non-negativity or total integral 1 – but they are closed under convex combination, thus forming a convex subset of the space of functions (or measures).

5.6 Applications of probability theory

The concept of the probability distribution and the random variables which they describe underlies the mathematical discipline of probability theory, and the science of statistics. There is spread or variability in almost any value that can be measured in a population (e.g. height of people, durability of a metal, sales growth, traffic flow, etc.); almost all measurements are made with some intrinsic error

Below is a framework built by O'Keefe and lVlcEachern (1998) for a Web purchasing model. As shown in Exhibit 4.2, each of the phases of the purchasing model can be supported by both Consumer Decision Support System (CDSS) facilities and Internet and Web facilities. The CDSS facilities support the specific decisions in the process.

Chapter Review Questions

1. Define term probability distribution
2. Explain any three application areas of probability distributions
3. Describe any two properties of probability distribution
4. Highlight differences between the following types of probability distributions

   a) Discrete distribution
   b) Continuous distribution
   c) Cumulative distribution

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition.

CHAPTER SIX

RANDOM NUMBER GENERATION

Learning Objectives

By the end of this chapter the learner shall be able to;

- Explain the types of random number generation techniques
- Describe the random number generation process
- Describe the methods of generating random numbers
- Describe the application of areas of random number generation process

6.1 Introduction to Random Number Generation

The ability to generate pseudorandom numbers is important for simulating events, estimating probabilities and other quantities, making randomized assignments or selections, and numerically testing symbolic results. Such applications may require uniformly distributed numbers, non uniformly distributed numbers, elements sampled with replacement, or elements sampled without replacement. Random number generation is also highly useful in estimating distributions for which closed form results are not known or known to be computationally difficult. Properties of random matrices provide one example. A **random number generator (RNG)**) is a computational or physical device designed to generate a sequence of numbers or symbols that lack any pattern, i.e. appear random. The many applications of randomness have led to the development of several different methods for generating random data. Many of these have existed since ancient times, including dice, coin flipping, the shuffling of playing cards, the use of yarrow stalks (by divination) in the IChing, and many other techniques.

6.2 Application of Random Numbers

Random number generators have applications in gambling, statistical sampling, computer simulation, cryptography, completely randomized design, and other areas where producing an unpredictable result is desirable. Random number generators are

very useful in developing Monte Carlo-method simulations, as debugging is facilitated by the ability to run the same sequence of random numbers again by starting from the same *random seed*. They are also used in cryptography - so long as the *seed* is secret. Sender and receiver can generate the same set of numbers automatically to use as keys. The generation of pseudo-random numbers is an important and common task in computer programming. While cryptography and certain numerical algorithms require a very high degree of *apparent* randomness, many other operations only need a modest amount of unpredictability. Some simple examples might be presenting a user with a "Random Quote of the Day", or determining which way a computer-controlled adversary might move in a computer game. Weaker forms of *randomness* also feature in hash algorithms and in creating amortized searching and sorting algorithms.

Some applications which appear at first sight to be suitable for randomization are in fact not quite so simple. For instance, a system that "randomly" selects music tracks for a background music system must only *appear* random, and may even have ways to control the selection of music: a true random system would have no restriction on the same item appearing two or three times in succession.

6.3 Methods for creating Random Numbers

There are several ways of creating random numbers as explained below;

a) Generation of random numbers by Physical methods

The earliest methods for generating random numbers dice, coin flipping, roulette wheels are still used today, mainly in games and gambling as they tend to be too slow for most applications in statistics and cryptography. A physical random number generator can be based on an essentially random atomic or subatomic physical phenomenon whose unpredictability can be traced to the laws of quantum mechanics. Sources of entropy include radioactive decay, thermal noise, shot noise, avalanche noise in diodes, clock drift, the timing of actual movements of a hard disk read/write head, and radio noise. However, physical phenomena and tools used to measure them

generally feature asymmetries and systematic biases that make their outcomes not uniformly random. A randomness extractor, such as a cryptographic hash function, can be used to approach a uniform distribution of bits from a non-uniformly random source, though at a lower bit rate. Another common entropy source is the behavior of human users of the system. While people are not considered good randomness generators upon request, they generate random behavior quite well in the context of playing mixed strategy games. Some security-related computer software requires the user to make a lengthy series of mouse movements or keyboard inputs to create sufficient entropy needed to generate random keys or to initialize pseudorandom number generators.

b) Generation of Random Numbers By Computational Methods

Pseudo-random number generators (PRNGs) are algorithms that can automatically create long runs of numbers with good random properties but eventually the sequence repeats (or the memory usage grows without bound). The string of values generated by such algorithms is generally determined by a fixed number called a **seed.** One of the most common PRNG is the linear congruential generator, which uses the recurrence

$$X_{n+1} = (aX_n + b) \bmod m$$

to generate numbers. The maximum number of numbers the formula can produce is the modulus, *m*. To avoid certain non-random properties of a single linear congruential generator, several such random number generators with slightly different values of the multiplier coefficient *a* can be used in parallel, with a "master" random number generator that selects from among the several different generators.

A simple pen-and-paper method for generating random numbers is the so-called middle square method suggested by John Von Neumann. While simple to implement, its output is of poor quality.Most computer programming languages include functions or library routines that purport to be random number generators. They are often designed to provide a random byte or word, or a floating point number uniformly distributed

between 0 and 1.Such library functions often have poor statistical properties and some will repeat patterns after only tens of thousands of trials. They are often initialized using a computer's real time clock as the seed, since such a clock generally measures in milliseconds, far beyond the person's precision. These functions may provide enough randomness for certain tasks (for example video games) but are unsuitable where high-quality randomness is required, such as in cryptographic applications, statistics or numerical analysis. Better pseudo-random number generators such as the Mersenne Twister are widely available.

c) Generation From A Probability Distribution

There are a couple of methods to generate a random number based on a probability density function. These methods involve transforming a uniform random number in some way. Because of this, these methods work equally well in generating both pseudo-random and true random numbers. One method, called the inversion method, involves integrating up to an area greater than or equal to the random number (which should be generated between 0 and 1 for proper distributions). A second method, called the acceptance-rejection method, involves choosing an x and y value and testing whether the function of x is greater than the y value. If it is, the x value is accepted. Otherwise, the x value is rejected and the algorithm tries again.

Generation by Persons

Random number generation may also be done by humans directly. However, most studies find that human subjects have some degree of nonrandomness when generating a random sequence of, e.g., digits or letters. They may alternate too much between choices compared to a good random generator.

Chapter Review Questions

1. Explain any three application areas of random number generations
2. State any two methods for generating random numbers
3. State any three advantages of generating random numbers using the following methods

    a) Probability distributions

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition.

CHAPTER SEVEN

SIMULATION LANGUAGES

Learning Objectives

By the end of this chapter the learner shall be able to;

Explain the types of simulation languages

Describe the advantages of special purpose and general purpose simulation languages

Describe the application areas of simulation languages

7.1 Introduction to simulation Languages

A programming language that is specialized to the implementation of simulation programs. Such languages are usually classified as either discrete event simulation languages or continuous simulation languages. Early effort in a simulation study is concerned with defining the system to be modeled and describing it in terms of logic flow diagrams and functional relationships. But eventually one is faced with the problem of describing the model in a language acceptable to the computer to be used. Most digital computers operate in a binary method of data representation, or in some multiple of binary such as octal or hexadecimal. Since these are awkward languages for users to communicate with, programming languages have evolved to make, easier to converse with the computer. Unfortunately, so many general and special purpose programming languages have been developed over the years, that it is a nearly impossible task to decide which language best fits or is even a near best fit to any particular application. Over 170 programming languages were in use in the United States in 1972 and today there are even more. Consequently, the usual procedure is to use a language known by the analyst, not because it is best, but because it is known. It should be stated that any general algorithmic language is capable of expressing the desired model; however, one of the specialized simulation languages may have very distinct advantages in terms of ease, efficiency and effectiveness of use. The major differences between special purpose simulation languages in general are:

a) the organization of time and activities,

b) the naming and structuring of entities within the model,

c) the testing of activities and conditions between elements,

d) the types of statistical tests possible on the data and (5) the ease of changing model structure

7.2 Advantages of General purpose Languages

Below are the advantages of general purpose languages;

a) Most modelers already know a general—purpose language, but this is often not the case with a simulation language.

b) General purpose languages are available on virtual1y every computer, but a particular simulation language may not be accessible on the computer that the analyst wants to use.

c) An efficiently written general purpose program may require less execution time than the corresponding program written in a simulation language. This is because a simulation language is designed to model a wide variety of systems with one set of building blocks, whereas general purpose program can be tailored to the particular application.

d) General—purpose languages allow greater programming flexibility than certain simulation languages. For example, complicated numerical calculations are not easy in GPSS.

7.3 Advantages of special purpose Languages

Below are the advantages of special purpose languages;

a) Simulation languages automatically provide most (if not all) of the features needed in programming a simulation model.

b) Simulation languages provide a natural framework for simulation modeling.

c) Simulations are generally easier to change when with in a simulation language.

d) Most simulation languages provide dynamic storage allocation during execution.

e) Most simulation languages provide better error detection.

f) Provide all of the statistical tools you need.

Chapter Review Questions

1. Differentiate between special purpose and general purpose simulation language

2. State any three benefits of simulation languages

3. State any two advantages of each of the following

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition.

CHAPTER EIGHT

DISCRETE EVENT SIMULATION

Learning Objectives

By the end of this chapter the learner shall be able to;

Explain the process of discrete event simulation

Describe the components of discrete event simulation

Describe the application of areas of discrete event simulation experiments

8.1 Introduction to Discrete event simulation

Discrete-event simulation represents the operation of a system as a chronological sequence of events. each event occurs at an instant in time and marks a change of state in the system . for example, if an elevator is simulated, an event could be "level 6 button pressed", with the resulting system state of "lift moving" and eventually (unless one chooses to simulate the failure of the lift) "lift at level 6".a common application in learning how to build discrete-event simulations is to model a queue such as customers arriving to the supermarket teller, such as customers arriving at a bank to be served by a teller. in this example, the system entities are customer in queue and tellers. the system events are customer-arrival and customer-departure. (the event of teller-begins-service can be part of the logic of the arrival and departure events.) The system states, which are changed by these events, are number-of-customers-in-the-queue (an integer from 0 to n) and teller-status (busy or idle). the random variables that need to be characterized to model this system stochastically are customer-inter arrival-time and teller-service-time. a number of mechanisms have been proposed for carrying out discrete-event simulation, among them are the event-based, activity-based, process-based and three-phase approaches the three-phase approach is used by a number of commercial simulation software packages, but from the user's point of view, the specifics of the underlying simulation method are generally hidden.

8.2 Components of a discrete event simulation

For successful discrete event simulation to be carried out, the following components are required for the activity.

a) Clock:

The simulation clock can be configured to minutes, seconds, microsecond or even hours. This is necessary since the simulation must keep track of the current simulation time, in whatever measurement units are suitable for the system being modeled. In discrete-event simulations, as opposed to real time simulations, time 'hops' because events are instantaneous – the clock skips to the next event start time as the simulation proceeds.

b) Events List:

The simulation process should always contain a list of event .the minimum list being at least one list of simulation events. This is sometimes called the *pending event set* because it lists events that are pending as a result of previously simulated event but have yet to be simulated themselves. An event is described by the time at which it occurs and a type, indicating the code that will be used to simulate that event. It is common for the event code to be parameterized, in which case, the event description also contains parameters to the event code. When events are instantaneous, activities that extend over time are modeled as sequences of events. Some simulation frameworks allow the time of an event to be specified as an interval, giving the start time and the end time of each event. Single-threaded simulation engines based on instantaneous events have just one current event. In contrast, multi-threaded simulation engines and simulation engines supporting an interval-based event model may have multiple current events. In both cases, there are significant problems with synchronization between current events. The pending event set is typically organized as a priority queue, sorted by event time. That is, regardless of the order in which events are added to the event set, they are removed in strictly chronological order. Several general-purpose priority queue algorithms have proven effective for discrete-event simulation, most notably, the splay tree. More recent alternatives include skip lists and

calendar queues. Typically, events are scheduled dynamically as the simulation proceeds. For example, in the bank example noted above, the event CUSTOMER-ARRIVAL at time t would, if the CUSTOMER_QUEUE was empty and TELLER was idle, include the creation of the subsequent event CUSTOMER-DEPARTURE to occur at time **t+s,** where s is a number generated from the SERVICE-TIME distribution.

c) Random-Number Generators

The simulation needs to generate random variables of various kinds, depending on the system model. This is accomplished by one or more pseudorandom number generators. The use of pseudorandom numbers as opposed to true random numbers is a benefit should a simulation need a rerun with exactly the same behavior. One of the problems with the random number distributions used in discrete-event simulation is that the steady-state distributions of event times may not be known in advance. As a result, the initial set of events placed into the pending event set will not have arrival times representative of the steady-state distribution. This problem is typically solved by bootstrapping the simulation model. Only a limited effort is made to assign realistic times to the initial set of pending events. These events, however, schedule additional events, and with time, the distribution of event times approaches its steady state. This is called *bootstrapping* the simulation model. In gathering statistics from the running model, it is important to either disregard events that occur before the steady state is reached or to run the simulation for long enough that the bootstrapping behavior is overwhelmed by steady-state behavior. (This use of the term *bootstrapping* can be contrasted with its use in both statistics and computing.)

d) Statistics

The simulation typically keeps track of the system's statistics, which quantify the aspects of interest. In the bank example, it is of interest to track the mean waiting times. the variance of the waiting times and probably the deviations.

e) Ending Condition

Because events are bootstrapped, theoretically a discrete-event simulation could run forever. So the simulation designer must decide when the simulation will end. Typical choices are "at time t" or "after processing n number of events" or, more generally, "when statistical measure X reaches the value x".

8.3 Application areas of discrete event simulation

The following are the areas where discrete event simulation could be applied;

a) Diagnosing process issues

Simulation approaches are particularly well equipped to help users diagnose issues in complex environments. The Theory of Constraints illustrates the importance of understanding bottlenecks in a system. Only process 'improvements' at the bottlenecks will actually improve the overall system. In many organizations bottlenecks become hidden by excess inventory, overproduction, variability in processes and variability in routing or sequencing. By accurately documenting the system inside a simulation model it is possible to gain a bird's eye view of the entire system.

A working model of a system allows management to understand performance drivers. A simulation can be built to include any number of performance indicators such as worker utilization, on-time delivery rate, scrap rate, cash cycles, and so on.

b) Custom order environments

Many systems show very different characteristics from day to day depending on the order mix. Many small orders may cause bottle-necks due to excess changeovers. Large custom orders may require extra processing at a point where the system has particularly low capacity. Simulation modeling allows management to understand what changes 'on average' would have the largest impact and greatest return-on-investment.

c) Lab test performance improvement ideas

Many systems improvement ideas are built on sound principles, proven methodologies (Lean, Six Sigma, TQM, etc.) yet fail to improve the overall system. A simulation model allows the user to understand and test a performance improvement idea in the context of the overall system.

d) Evaluating capital investment decisions

Simulation modeling is commonly used to model potential investments. Through modeling investments decision-makers can make informed decisions and evaluate potential alternatives. Often these decisions look at altering existing operations. Typically, a model of the current state is constructed. This 'current state' model is tested and validated against historical data. Once the model is operating correctly, the simulation is altered to reflect the proposed capital investments. This 'future state' model is then stress-tested to ensure the alterations perform as desired. Occasionally, organizations take on entirely new operations processes. These could be new Lean facilities, designed around new products or using new technology. In these cases only a 'future state' model is constructed. The testing and validation may require more analysis. There are companies and experts that specialize in simulation building who may be brought in to help.

e) Stress test a system

Models can be used to understand how a system will be able to weather extraordinary conditions. A simulation can help management understand: large increases in orders, significant swings in product mix, new client delivery demands (e.g. 1 week lead times), and economic events (e.g. a multinational with operations in South America and Asia sees significant swings in currencies).

Chapter Review Questions

1. State any two purpose of discrete event simulation
2. Explain the application of discrete event simulation in evaluating customer ordering
3. Describe any four components of discrete event simulation

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition.

CHAPTER NINE

STATISTICAL INFERENCING

Learning Objectives

By the end of this chapter the learner shall be able to;

Explain the concept of statistical inference
Describe the models of statistical inferences
Explain the model assumptions in statistical inference
Describe the types of statistical models

9.1 Introduction to statistical inference

Statistical inference is the process of drawing conclusions from data that are subject to random variation, for example, observational errors or sampling variation. More substantially, the terms statistical inference, statistical induction and inferential statistics are used to describe systems of procedures that can be used to draw conclusions from datasets arising from systems affected by random variation. Initial requirements of such a system of procedures for inference and induction are that the system should produce reasonable answers when applied to well-defined situations and that it should be general enough to be applied across a range of situations. The outcome of statistical inference may be an answer to the question "what should be done next?", where this might be a decision about making further experiments or surveys, or about drawing a conclusion before implementing some organizational or governmental policy.

For the most part, statistical inference makes propositions about populations, using data drawn from the population of interest via some form of random sampling. More generally, data about a random process is obtained from its observed behavior during a finite period of time. Given a parameter or hypothesis about which one wishes to make inference, statistical inference most often uses:

- a statistical model of the random process that is supposed to generate the data, and

- a particular realization of the random process; i.e., a set of data.

The conclusion of a statistical inference is a statistical proposition. Some common forms of statistical proposition are:

- an estimate; i.e., a particular value that best approximates some parameter of interest,
- a confidence interval (or set estimate); i.e., an interval constructed from the data in such a way that, under repeated sampling of datasets, such intervals would contain the true parameter value with the probability at the stated confidence level,
- a credible interval; i.e., a set of values containing, for example, 95% of posterior belief,
- rejection of a hypothesis
- clustering or classification of data points into groups

Any statistical inference requires some assumptions. A statistical model is a set of assumptions concerning the generation of the observed data and similar data. Descriptions of statistical models usually emphasize the role of population quantities of interest, about which we wish to draw inference.

9.2 Degree of Models/Assumptions

Statisticians distinguish between three levels of modeling assumptions;

- Fully parametric: The probability distributions describing the data-generation process are assumed to be fully described by a family of probability distributions involving only a finite number of unknown parameters. For example, one may assume that the distribution of population values is truly Normal, with unknown mean and variance, and that datasets are generated by 'simple' random sampling. The family of generalized linear models is a widely used and flexible class of parametric models.

- Non-parametric: The assumptions made about the process generating the data are much less than in parametric statistics and may be minimal. For example, every continuous probability distribution has a median, which may be estimated using the sample median or the Hodges-Lehmann-Sen estimator, which has good properties when the data arise from simple random sampling.

Semi-parametric: This term typically implies assumptions 'between' fully and non-parametric approaches. For example, one may assume that a population distribution have a finite mean. Furthermore, one may assume that the mean response level in the population depends in a truly linear manner on some covariate (a parametric assumption) but not make any parametric assumption describing the variance around that mean (i.e., about the presence or possible form of any heteroscedasticity). More generally, semi-parametric models can often be separated into 'structural' and 'random variation' components. One component is treated parametrically and the other non-parametrically. The well-known Cox model is a set of semi-parametric assumptions.

9.3 Importance of valid Models/Assumptions

Whatever level of assumption is made, correctly calibrated inference in general requires these assumptions to be correct; i.e., that the data-generating mechanisms really has been correctly specified. Incorrect assumptions of 'simple' random sampling can invalidate statistical inference. More complex semi- and fully parametric assumptions are also cause for concern. For example, incorrectly assuming the Cox model can in some cases lead to faulty conclusions. Incorrect assumptions of Normality in the population also invalidates some forms of regression-based inference. The use of any parametric model is viewed skeptically by most experts in sampling human populations: "most sampling statisticians, when they deal with confidence intervals at all, limit themselves to statements about [estimators] based on very large samples, where the central limit theorem ensures that these [estimators] will have distributions that are nearly normal." In particular, a normal distribution "would be a totally unrealistic and catastrophically unwise assumption to make if we were dealing with any kind of

economic population."Here, the central limit theorem states that the distribution of the sample mean "for very large samples" is approximately normally distributed, if the distribution is not heavy tailed.

9.4 Types of statistical Models

There are several types of statistical models that could be used in simulation as discussed below;

a) Approximate distributions

Approximation distribution results measure how close a limiting distribution approaches the statistic's sample distribution. Approximation provides a good approximation to the sample-mean's distribution when there are 10 (or more) independent samples. With infinite samples, limiting results like the central limit theorem describe the sample statistic's limiting distribution, if one exists. Limiting results are not statements about finite samples, and indeed are logically irrelevant to finite samples. However, the asymptotic theory of limiting distributions is often invoked for work in estimation and testing. For example, limiting results are often invoked to justify the generalized method of moments and the use of generalized estimating equations, which are popular in econometrics and biostatistics. The magnitude of the difference between the limiting distribution and the true distribution can be assessed using simulation:. The use of limiting results in this way works well in many applications, especially with low-dimensional models with log-concave likelihoods.

b) Randomization-based models

For a given dataset that was produced by a randomization design, the randomization distribution of a statistic is defined by evaluating the test statistic for all of the plans that could have been generated by the randomization design. In frequentist inference, randomization allows inferences to be based on the randomization distribution rather than a subjective model, and this is important especially in survey sampling and design of experiments. Statistical inference from randomized studies is also more

straightforward than many other situations. In Bayesian inference, randomization is also of importance in :

1. Survey sampling, use of sampling without replacement ensures the exchangeability of the sample with the population; in randomized experiments, randomization warrants a missing at random assumption for covariate information.

2. Objective randomization allows properly inductive procedures. Many statisticians prefer randomization-based analysis of data that was generated by well-defined randomization procedures.

Similarly, results from randomized experiments are recommended by leading statistical authorities as allowing inferences with greater reliability than do observational studies of the same phenomena. However, a good observational study may be better than a bad randomized experiment the statistical analysis of a randomized experiment may be based on the randomization scheme stated in the experimental protocol and does not need a subjective model. However, not all hypotheses can be tested by randomized experiments or random samples, which often require a large budget, a lot of expertise and time, and may have ethical problems.

c) Model-based analysis of randomized experiments

It is standard practice to refer to a statistical model, often a normal linear model, when analyzing data from randomized experiments. However, the randomization scheme guides the choice of a statistical model. It is not possible to choose an appropriate model without knowing the randomization scheme. Seriously misleading results can be obtained analyzing data from randomized experiments while ignoring the experimental protocol; common mistakes include forgetting the blocking used in an experiment and confusing repeated measurements on the same experimental unit with independent replicates of the treatment applied to different experimental units.

Chapter Review Questions

a. Explain any three statistical models that can be used in statistical inferencing

b. State any two importance of models

c. Explain the difference between randomized and approximation models in statistical inference

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition.

CHAPTER TEN

INTERPRETATION AND ANALYSIS OF SIMULATION RESULTS USING MODELS

Learning Objectives

By the end of this chapter the learner shall be able to;

Explain the concept of poison distribution model
Conduct analysis of proof of poison simulation
Explain the concept of poison distribution model
Describe the Monte Carlo simulation process

Explain the application of Monte Carlo simulation

10.1 Poison Distribution Models

In simulation a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time and/or space if these events occur with a known average rate and independently of the time since the last event. If the expected number of occurrences in a given interval is λ, then the probability that there are exactly k occurrences (k being a non-negative integer, k = 0, 1, 2, ...) is equal to

$$f(k;\lambda) = \frac{\lambda^k e^{-\lambda}}{k!},$$

where

- e is the base of the natural logarithm (e = 2.71828...)
- k is the number of occurrences of an event — the probability of which is given by the function
- k! is the factorial of k
- λ is a positive real number, equal to the expected number of occurrences during the given interval. For instance, if the events occur on average 4 times per minute, and one is interested in the probability of an event occurring k times in a 10 minute interval, one would use a Poisson distribution as the model with λ = 10×4 = 40.

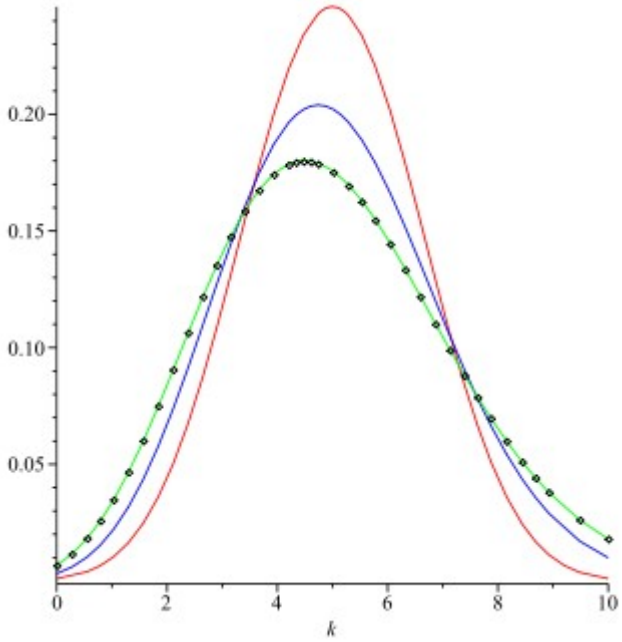
As a function of k, this is the probability mass function. The Poisson distribution can be derived as a limiting case of the binomial distribution. The Poisson distribution can be applied to systems with a large number of possible events, each of which is rare. The Poisson distribution is sometimes called a Poissonian.

10.2 Poison Simulation Areas

The Poisson distribution arises in connection with Poisson processes. It applies to various phenomena of discrete properties (that is, those that may happen 0, 1, 2, 3, times during a given period of time or in a given area) whenever the probability of the phenomenon happening is constant in time or space. Examples of events that may be modeled as a Poisson distribution include:

- The number of phone calls arriving at a call centre per minute.
- The number of goals in sports involving two competing teams.
- The number of deaths per year in a given age group.
- The number of jumps in a stock price in a given time interval.
- Under an assumption of homogeneity, the number of times a web server is accessed per minute.
- The number of mutations in a given stretch of DNA after a certain amount of radiation.
- The proportion of cells that will be infected at a given multiplicity of infection.

Poison Graph



Comparison of the Poisson distribution (black dots) and the binomial distribution with n=10 (red line), n=20 (blue line), n=1000 (green line). All distributions have a mean of 5. The horizontal axis shows the number of events k. Notice that as n gets larger, the Poisson distribution becomes an increasingly better approximation for the binomial distribution with the same mean.

In several of the above examples—such as, the number of mutations in a given sequence of DNA—the events being counted are actually the outcomes of discrete trials, and would more precisely be modelled using the binomial distribution, that is

$$X \sim \mathrm{B}(n, p).$$

In such cases n is very large and p is very small (and so the expectation np is of intermediate magnitude). Then the distribution may be approximated by the less cumbersome Poisson distribution

$$X \sim \mathrm{Pois}(np).$$

This is sometimes known as the law of rare events, since each of the n individual Bernoulli events rarely occurs. The name may be misleading because the total count of success events in a Poisson process need not be rare if the parameter np is not small. For example, the number of telephone calls to a busy switchboard in one hour follows a Poisson distribution with the events appearing frequent to the operator, but they are

rare from the point of view of the average member of the population who is very unlikely to make a call to that switchboard in that hour.

10.3 Proof of Poison Distributions

We will prove that, for fixed λ, if

$$X_n \sim \mathrm{B}(n, \lambda/n); \qquad Y \sim \mathrm{Pois}(\lambda).$$

then for each fixed k

$$\lim_{n\to\infty} P(X_n = k) = P(Y = k).$$

To see the connection with the above discussion, for any Binomial random variable with large n and small p set λ = np. Note that the expectation $E(X_n) = \lambda$ is fixed with respect to n.

First, recall from calculus

$$\lim_{n\to\infty} \left(1 - \frac{\lambda}{n}\right)^n = e^{-\lambda},$$

then since p = λ / n in this case, we have

$$\begin{aligned}
\lim_{n\to\infty} P(X_n = k) &= \lim_{n\to\infty} \binom{n}{k} p^k (1-p)^{n-k} \\
&= \lim_{n\to\infty} \frac{n!}{(n-k)!k!} \left(\frac{\lambda}{n}\right)^k \left(1 - \frac{\lambda}{n}\right)^{n-k} \\
&= \lim_{n\to\infty} \underbrace{\left[\frac{n!}{n^k\,(n-k)!}\right]}_{A_n} \left(\frac{\lambda^k}{k!}\right) \underbrace{\left(1 - \frac{\lambda}{n}\right)^n}_{\to \exp(-\lambda)} \underbrace{\left(1 - \frac{\lambda}{n}\right)^{-k}}_{\to 1} \\
&= \left[\lim_{n\to\infty} A_n\right] \left(\frac{\lambda^k}{k!}\right) \exp(-\lambda)
\end{aligned}$$

Next, note that

$$
\begin{aligned}
A_n &= \frac{n!}{n^k\,(n-k)!} \\
&= \frac{n\cdot(n-1)\cdots\big(n-(k-1)\big)}{n^k} \\
&= 1\cdot(1-\tfrac{1}{n})\cdots(1-\tfrac{k-1}{n}) \\
&\to 1\cdot 1\cdots 1 = 1,
\end{aligned}
$$

Where we have taken the limit of each of the terms independently, which is permitted since there is a fixed number of terms with respect to n (there are k of them). Consequently, we have shown that

$$\lim_{n\to\infty} P(X_n = k) = \frac{\lambda^k \exp(-\lambda)}{k!} = P(Y = k).$$

Generalization of the formular

We have shown that if

$$X_n \sim \mathrm{B}(n, p_n); \qquad Y \sim \mathrm{Pois}(\lambda),$$

Where $p_n = \lambda / n$, then $X_n \to Y$ in distribution. This holds in the more general situation that p_n is any sequence such that

$$\lim_{n\to\infty} np_n = \lambda.$$

2-dimensional Poisson process

$$P(N(D) = k) = \frac{(\lambda|D|)^k e^{-\lambda|D|}}{k!}$$

Where

- e is the base of the natural logarithm (e = 2.71828...)
- k is the number of occurrences of an event - the probability of which is given by the function
- k! is the factorial of k
- D is the 2-dimensional region
- |D| is the area of the region

N(D) is the number of points in the process in region D.

10.4 Properties of Poison Distribution models

- The expected value of a Poisson-distributed random variable is equal to λ and so is its variance. The higher moments of the Poisson distribution are Touchard polynomials in λ, whose coefficients have a combinatorial meaning. In fact, when the expected value of the Poisson distribution is 1, then Dobinski's formula says that the nth moment equals the number of partitions of a set of size n.
- The mode of a Poisson-distributed random variable with non-integer λ is equal to $\lfloor \lambda \rfloor$, which is the largest integer less than or equal to λ. This is also written as floor (λ). When λ is a positive integer, the modes are λ and λ − 1.
- Given one event (or any number) the expected number of other events is independent so still λ. If reproductive success follows a Poisson distribution with expected number of offspring λ, then for a given individual the expected number of (half)siblings (per parent) is also λ. If full siblings are rare total expected sibs are 2λ.
- Sums of Poisson-distributed random variables:

  If $X_i \sim \operatorname{Pois}(\lambda_i)$ follow a Poisson distribution with parameter λ_i and X_i are independent, then

  $$Y = \sum_{i=1}^{N} X_i \sim \operatorname{Pois}\left(\sum_{i=1}^{N} \lambda_i\right)$$

  also follows a Poisson distribution whose parameter is the sum of the component parameters. A converse is Raikov's theorem, which says that if the sum of two independent random variables is Poisson-distributed, then so is each of those two independent random variables.

- The sum of normalized square deviations is approximately distributed as chi-squared if the mean is of a moderate size (λ > 5 is suggested).[8] If

$X_1, \ldots, X_N$ are observations from independent Poisson distributions with means $\lambda_1, \ldots, \lambda_N$ then $\sum_{i=1}^{N} \frac{(X_i - \lambda_i)^2}{\lambda_i} \sim \chi^2$

- The moment-generating function of the Poisson distribution with expected value λ is

$$\mathrm{E}\left(e^{tX}\right) = \sum_{k=0}^{\infty} e^{tk} f(k;\lambda) = \sum_{k=0}^{\infty} e^{tk} \frac{\lambda^k e^{-\lambda}}{k!} = e^{\lambda(e^t-1)}.$$

- All of the cumulants of the Poisson distribution are equal to the expected value λ. The nth factorial moment of the Poisson distribution is λ^n.
- The Poisson distributions are infinitely divisible probability distributions.
- The directed Kullback-Leibler divergence between Pois(λ) and Pois(λ_0) is given by

$$D_{\mathrm{KL}}(\lambda \| \lambda_0) = \lambda_0 - \lambda + \lambda \log \frac{\lambda}{\lambda_0}.$$

- Upper bound for the tail probability of a Poisson random variable X~Pois(λ).[9]

  The proof uses a Chernoff bound argument.

$$P(X \geq x) \leq \frac{e^{-\lambda}(e\lambda)^x}{x^x}, \text{ for } x > \lambda$$

Similarly,

$$P(X \leq x) \leq \frac{e^{-\lambda}(e\lambda)^x}{x^x}, \text{ for } x < \lambda$$

10.5 Proof of Poison Distributions

Although the Poisson distribution is limited by

$$0 < f(k,\lambda) \leq f(\lfloor \lambda \rfloor, \lambda) < 1,$$

the numerator and denominator of f(k,λ) can reach extreme values for large values of k or λ.

If the Poisson distribution is evaluated on a computer with limited precision by first evaluating its numerator and denominator and then dividing the two, then a significant loss of precision may occur.

For example, with the common double precision a complete loss of precision occurs if f(150,150) is evaluated in this manner.

A more robust evaluation method is:

$$\begin{aligned} f(k,\lambda) &= \exp\left(\ln\left(f(k,\lambda)\right)\right) \\ &= \exp\left(\ln\left(\frac{\lambda^k \exp\left(-\lambda\right)}{k!}\right)\right) \\ &= \exp\left(k\ln\left(\lambda\right) - \lambda - \sum_{i=1}^{k}\ln\left(i\right)\right). \end{aligned}$$

Generating Poisson-distributed random variables

A simple algorithm to generate random Poisson-distributed numbers (pseudo-random number sampling)

Algorithm Poisson random number (Knuth):

init:
  Let L ← $e^{-\lambda}$, k ← 0 and p ← 1.
do:
  k ← k + 1.
  Generate uniform random number u in [0,1] and let p ← p × u.
while p > L.
return k − 1.

While simple, the complexity is linear in λ. There are many other algorithms to overcome this. Some are given in Ahrens & Dieter, see References below. Also, for large values of λ, there may be numerical stability issues because of the term $e^{-\lambda}$. One solution for large values of λ is Rejection sampling, another is to use a Gaussian approximation to the Poisson. Inverse transform sampling is simple and efficient for

small values of λ, and requires only one uniform random number u per sample. Cumulative probabilities are examined in turn until one exceeds u.

10.6 Proof of Poison Distributions

Given a sample of n measured values k_i we wish to estimate the value of the parameter λ of the Poisson population from which the sample was drawn. To calculate the maximum likelihood value, we form the log-likelihood function

$$\begin{aligned} L(\lambda) &= \ln \prod_{i=1}^{n} f(k_i \mid \lambda) \\ &= \sum_{i=1}^{n} \ln\left(\frac{e^{-\lambda}\lambda^{k_i}}{k_i!}\right) \\ &= -n\lambda + \left(\sum_{i=1}^{n} k_i\right) \ln(\lambda) - \sum_{i=1}^{n} \ln(k_i!). \end{aligned}$$

Take the derivative of L with respect to λ and equate it to zero:

$$\frac{\mathrm{d}}{\mathrm{d}\lambda} L(\lambda) = 0 \iff -n + \left(\sum_{i=1}^{n} k_i\right) \frac{1}{\lambda} = 0.$$

Solving for λ yields a stationary point, which if the second derivative is negative is the maximum-likelihood estimate of λ:

$$\widehat{\lambda}_{\mathrm{MLE}} = \frac{1}{n} \sum_{i=1}^{n} k_i.$$

Checking the second derivative, it is found that it is negative for all λ and k_i greater than zero, therefore this stationary point is indeed a maximum of the initial likelihood function:

$$\frac{\partial^2 L}{\partial \lambda^2} = -\lambda^{-2} \sum_{i=1}^{n} k_i$$

Since each observation has expectation λ so does this sample mean. Therefore it is an unbiased estimator of λ. It is also an efficient estimator, i.e. its estimation variance achieves the Cramér–Rao lower bound (CRLB). Hence it is MVUE. Also it can be proved that the sample mean is complete and sufficient statistic for λ.

a) Bayesian Inference

In Bayesian inference, the conjugate prior for the rate parameter λ of the Poisson distribution is the Gamma distribution. Let

$$\lambda \sim \mathrm{Gamma}(\alpha, \beta)$$

denote that λ is distributed according to the Gamma density g parameterized in terms of a shape parameter α and an inverse scale parameter β:

$$g(\lambda \mid \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} \qquad \text{for } \lambda > 0.$$

Then, given the same sample of n measured values k_i as before, and a prior of Gamma(α, β), the posterior distribution is

$$\lambda \sim \mathrm{Gamma}(\alpha + \sum_{i=1}^{n} k_i, \beta + n).$$

The posterior mean E[λ] approaches the maximum likelihood estimate $\widehat{\lambda}_{\mathrm{MLE}}$ in the limit as $\alpha \to 0,\ \beta \to 0$.

The posterior predictive distribution of additional data is a Gamma-Poisson (i.e. negative binomial) distribution.

b) Confidence Interval

A simple and rapid method to calculate an approximate confidence interval for the estimation of λ is proposed is proposed in confidence interval evaluation. This method provides a good approximation of the confidence interval limits, for samples containing at least 15 – 20 elements. Denoting by N the number of sampled points or events and

by L the length of sample line (or the time interval), the upper and lower limits of the 95% confidence interval are given by:

$$\lambda_{low} = \frac{(1 - \frac{1.96}{\sqrt{N-1}})N}{L}$$
$$\lambda_{upp} = \frac{(1 + \frac{1.96}{\sqrt{N-1}})N}{L}$$

10.7 Proof of Poison Distributions

a) Introduction

Monte Carlo methods (or Monte Carlo experiments) are a class of computational algorithms that rely on repeated random sampling to compute their results. Monte Carlo methods are often used in computer simulations of physical and mathematical systems. These methods are most suited to calculation by a computer and tend to be used when it is infeasible to compute an exact result with a deterministic algorithm. This method is also used to complement the theoretical derivations. Monte Carlo methods are especially useful for simulating systems with many coupled degrees of freedom, such as fluids, disordered materials, strongly coupled solids, and cellular structures . They are used to model phenomena with significant uncertainty in inputs, such as the calculation of risk in business. They are widely used in mathematics, for example to evaluate multidimensional definite integrals with complicated boundary conditions. When Monte Carlo simulations have been applied in space exploration and oil exploration, their predictions of failures, cost overruns and schedule overruns are routinely better than human intuition or alternative "soft" method.

Monte Carlo methods vary, but tend to follow a particular pattern:

1. Define a domain of possible inputs.
2. Generate inputs randomly from a probability distribution over the domain.
3. Perform a deterministic computation on the inputs.
4. Aggregate the results.

For example, consider a circle inscribed in a unit square. Given that the circle and the square have a ratio of areas that is π/4, the value of π can be approximated using a Monte Carlo method:

1. Draw a square on the ground, then inscribe a circle within it.
2. Uniformly scatter some objects of uniform size (grains of rice or sand) over the square.
3. Count the number of objects inside the circle and the total number of objects.
4. The ratio of the two counts is an estimate of the ratio of the two areas, which is π/4. Multiply the result by 4 to estimate π.

In this procedure the domain of inputs is the square that circumscribes our circle. We generate random inputs by scattering grains over the square then perform a computation on each input (test whether it falls within the circle). Finally, we aggregate the results to obtain our final result, the approximation of π.To get an accurate approximation for π this procedure should have two other common properties of Monte Carlo methods. First, the inputs should truly be random. If grains are purposefully dropped into only the center of the circle, they will not be uniformly distributed, and so our approximation will be poor. Second, there should be a large number of inputs. The approximation will generally be poor if only a few grains are randomly dropped into the whole square. On average, the approximation improves as more grains are dropped.

b) Monte carlo uses in random numbers

Monte Carlo simulation methods do not always require truly random numbers to be useful — while for some applications, such as primarily testing, unpredictability is vital.Many of the most useful techniques use deterministic, pseudorandom sequences, making it easy to test and re-run simulations. The only quality usually necessary to make good simulations is for the pseudo-random sequence to appear "random enough" in a certain sense.

What this means depends on the application, but typically they should pass a series of statistical tests. Testing that the numbers are uniformly distributed or follow another

desired distribution when a large enough number of elements of the sequence are considered is one of the simplest, and most common ones.

The following are lists the characteristics of a high quality Monte Carlo simulation:

- the (pseudo-random) number generator has certain characteristics (e. g., a long "period" before the sequence repeats)
- the (pseudo-random) number generator produces values that pass tests for randomness
- there are enough samples to ensure accurate results
- the proper sampling technique is used
- the algorithm used is valid for what is being modeled
- it simulates the phenomenon in question.

c) Monte Carlo Simulation In "What If" Scenarios Testing

There are ways of using probabilities that are definitely not Monte Carlo simulations—for example, deterministic modeling using single-point estimates. Each uncertain variable within a model is assigned a "best guess" estimate. Scenarios (such as best, worst, or most likely case) for each input variable are chosen and the results recorded by contrast, Monte Carlo simulations sample probability distribution for each variable to produce hundreds or thousands of possible outcomes. The results are analyzed to get probabilities of different outcomes occurring. For example, a comparison of a spreadsheet cost construction model run using traditional "what if" scenarios, and then run again with Monte Carlo simulation and Triangular probability distributions shows that the Monte Carlo analysis has a narrower range than the "what if" analysis. This is because the "what if" analysis gives equal weight to all scenarios.

d) Monte Carlo Applications

Monte Carlo methods are especially useful for simulating phenomena with significant uncertainty in inputs and systems with a large number of coupled degrees of freedom. Areas of application include:

Design And Visuals

Monte Carlo methods have also proven efficient in solving coupled integral differential equations of radiation fields and energy transport, and thus these methods have been used in global illumination computations which produce photo-realistic images of virtual 3D models, with applications in video games, architecture, design, computer generated films, and cinematic special effects.

Finance and business

Monte Carlo methods in finance are often used to calculate the value of companies, to evaluate investments in projects at a business unit or corporate level, or to evaluate financial derivatives. They can be used to model project schedules, where simulations aggregate estimates for worst-case, best-case, and most likely durations for each task to determine outcomes for the overall project.

Telecommunications

When planning a wireless network, design must be proved to work for a wide variety of scenarios that depend mainly on the number of users, their locations and the services they want to use. Monte Carlo methods are typically used to generate these users and their states. The network performance is then evaluated and, if results are not satisfactory, the network design goes through an optimization process.

Optimization

Another powerful and very popular application for random numbers in numerical simulation is in numerical optimization. The problem is to minimize (or maximize) functions of some vector that often has a large number of dimensions. Many problems can be phrased in this way: for example, a computer chess program could be seen as trying to find the set of, say, 10 moves that produces the best evaluation function at the end. In the traveling salesman problem the goal is to minimize distance traveled.

There are also applications to engineering design, such as multidisciplinary design optimization.Most Monte Carlo optimization methods are based on random walks. Essentially, the program moves randomly on a multi-dimensional surface, preferring moves that reduce the function, but sometimes moving "uphill".

Inverse problems

Probabilistic formulation of inverse problems leads to the definition of a probability distribution in the model space. This probability distribution combines a priori information with new information obtained by measuring some observable parameters (data). As, in the general case, the theory linking data with model parameters is nonlinear, the a posteriori probability in the model space may not be easy to describe (it may be multimodal, some moments may not be defined, etc.).When analyzing an inverse problem, obtaining a maximum likelihood model is usually not sufficient, as we normally also wish to have information on the resolution power of the data. In the general case we may have a large number of model parameters, and an inspection of the marginal probability densities of interest may be impractical, or even useless. But it is possible to pseudorandomly generate a large collection of models according to the posterior probability distribution and to analyze and display the models in such a way that information on the relative likelihoods of model properties is conveyed to the spectator. This can be accomplished by means of an efficient Monte Carlo method, even in cases where no explicit formula for the a priori distribution is available.

The best-known importance sampling method, the Metropolis algorithm, can be generalized, and this gives a method that allows analysis of (possibly highly nonlinear) inverse problems with complex a priori information and data with an arbitrary noise distribution.

Chapter Review Questions

1. Using suitable examples differentiate between generalization and 2 dimensional poison process
2. Outline any two properties of poison distribution models
3. Explain any three characteristics of Monte Carlo models simulation

Zeigler P., Praehofer H., and Kim T.(2000),Theory of Modeling and Simulation, Second Edition

SAMPLE EXAM QUESTIONS

Mt Kenya University

UNIVERSITY EXAMINATION 2010/2011

SCHOOL OF PURE AND APPLIED SCIENCES

DEPARTMENT OF INFORMATION TECHNOLOGY

EXAMINATION FOR BACHELOR OF BUSINESS INFORMATION TECHNOLOGY

BBIT 3203: BUSINESS SYSTEMS SIMULATION AND MODELLING

INSTRUCTIONS: ANSWER QUESTION ONE AND ANY OTHER TWO QUESTIONS

August 2011 **TIME 2HRS**

Question one

(a) Define the following terms

(i) Model (2marks)

(ii) Simulation (2 marks)

(b) Name four real world problems in business where simulation is applied and their solution methods (4marks)

(c) Differentiate between stochastic model and deterministic model of system (8marks)

(d) Explain seven stages in the model development process (14mark)

Question two

(a) Explain any four advantages of simulation (8marks)

(b) Name any two programming languages used in simulation (2marks)

(c) Differentiate between discrete event models and continuous models (8 marks)

(d) State two types of simulation solutions to problems (2marks)

Question three

(a) Explain any four disadvantages of simulation (8marks)

(b) A company generated revenue worth 100 millions last year and incurred cost to the time of 75 million. How much profit did the company make before taxes (3marks)

(c) A company tenders for two contract A and B. The probability that it will obtain A is 0.2 and contract B is 0.3. what will is the probability that it will obtain either contract A or contract B (4marks)

(d) Outline five characteristics of a good random generator (5marks)

Question four

(a) What is meant by the term" Abstraction"? (2marks)

(b) With the aid of diagram show the four basic structure of queuing (8marks)

(c) What is a system? (2marks)

(d) A box of 16 components contains 4 defective components. If 3 components are drown from the box what is the probability that they are all good.

(i) If there is replacement

(ii) If there is no replacement (8marks)

Question five

(a) State and explain seven systems components? (14marks)

(b) Describe three types of models (6marks)