Chapter 1. Basic Simulation Modelling

Basic Concepts

Continuous system: airplane moving through the air
Discrete system: number of customers in the bank
(Traffic flow can be discrete if the characteristics and movement of individual cars are important)
Static Simulation Model:(Monte Carlo models)
Representation of system at a particular time, or may be used to represent a system in which time simply plays no role
Dynamic Simulation Model:(Conveyor system in a factory)
System evolves over time
Deterministic Simulation Model: (Complicated system of differential equations)
No probabilistic components, output is determined once input and relationship have been specified
Stochastic Simulation Model:(Queueing and inventory systems)
Random output, only an estimate of the true characteristics of the model
Discrete event simulation models: discrete, dynamic, stochastic model

1.4 Simulation of a single-server queueing system

Example see Page 15

1.4.1 Problem Statement

- $q(n)$:= expected average number of customers in the queue, where this is taken over the time period needed to observe the n delays
- $Q(t)$:= number of customers in queue at time t
- $T(n)$:= time required to observe our n delays in queue
- p_i := expected proportion of the time that $Q(t) = i$

- T_i := total time during the simulation that the queue is of length i

weighted average: $q(n) = \sum_{i=0}^{\infty} ip_i$
observed average: $\hat{q}(n) = \sum_{i=0}^{\infty} i\hat{p}_i$ where $\hat{p}_i = \frac{T_i}{T(n)}$

Then $\hat{q}(n) = \sum_{i=0}^{\infty} \frac{iT_i}{T(n)}$, this summation is the area under the $Q(t)$ curve between the beginning and the end of the simulation
Which can be written as $\hat{q}(n) = \frac{\int_0^{T(n)} Q(t)dt}{T(n)}$

Expected Utilization of the server u_n := expected proportion of time during the simulation (from 0 to $T(n)$) that the server is busy, therefore is between 0 and 1

Busy Function:

$$f(x) = \begin{cases} 0 & \text{if the server is busy at time t} \\ 1 & \text{if the server is idle at time t} \end{cases}$$

Then the $\hat{u}(n)$ can be expressed as the proportion of time that $B(t) = 1$
$\hat{u}(n) = \frac{\int_0^{T(n)} B(t)dt}{T(n)}$, which is the continous average of $B(t)$

To recap, the three measures of performance are:
Average delay in queue $\hat{d}(n)$
Time-average number of customers in queue $\hat{q}(n)$
Proportion of time the server is busy $\hat{u}(n)$

$\hat{d}(n)$ is discrete-time statistic: defined relative to the collection of random variables D_i that have a discrete time index
$\hat{q}(n)$ and $\hat{u}(n)$ are continous-time statistic: defined on the collection of random variables $Q(t)$ and $B(t)$ that have a continous time index

The event for this system: arrival and departure of a customer
The state variables necessary to estimate $d(n)$,$q(n)$,$u(n)$:

- the status of the server (0,1)

- number of customers in the queue
- the time of arrival of each customer currently in the queue
- time of the last event

1.4.2 Intuitive Explanation:
Example see Page 18

Some Comments:

- The key element in the dynamics of a simulation is the interaction between the simulation clock and the event list
- While processing an event, no "simulated" time passes
- In some simulations it can happen that two or more entries in the event list ties

1.4.3 Program Organization and Logic:
Flowchart of arrival routine and departure routine see Page 30, 31

1.4.4 FORTRAN Program:
Page 32

1.4.4 & 1.4.5 FORTRAN & C Program:

Page 32-49

1.4.6 Output and Discussion:

Why not estimate the expected average waiting time in the system of a customer, $w(n)$
Rather than the expected average delay in queue, $d(n)$, where the waiting time of a customer is defined as the time interval from the instant the customer arrives to the instant the customer completes service and departs

(1)For many queueing systems we believe that the customer's delay in queue while waiting for other customers to be served is the most troublesome part of a customer's wait in the system

(2)If the queue represents part of a manufacturing system where the 'customers' are actually parts waiting for service at a machine, then the delay in queue represents a loss, whereas the time spent in service is 'necessary'

The usual estimator of $w(n)$ would be:
$\hat{h}(n) = \frac{\sum_{i=1}^{n} W_i}{n} = \frac{\sum_{i=1}^{n} D_i}{n} + \frac{\sum_{i=1}^{n} S_i}{n} = \hat{d}(n) + \bar{S}(n)$
Where $W_i = D_i + S_i$ is the waiting time in the system of the ith customer and $\bar{S}(n)$ is the average of the n customers' service times
The service time distribution and expected/mean service time E(S) would be known for simulation
$\tilde{w}(n) = \hat{d}(n) + E(S)$

$\tilde{w}(n)$ is more efficient and preferable than $\hat{w}(n)$

1.4.8 Determining the events and variables:
Different examples of event graph see Page 57

1.5 Simulation of an inventory system

1.5.1 Problem Statement

A company that sells a single product would like to decide how many items it should have in inventory for each of the next n months
The time between demands are IID exponential random variables with a mean of 0.1 month
The size of demands, D, are IID random variables, with

$$D = \begin{cases} 1 & \text{w.p. } \frac{1}{6} \\ 2 & \text{w.p. } \frac{1}{3} \\ 3 & \text{w.p. } \frac{1}{3} \\ 4 & \text{w.p. } \frac{1}{6} \end{cases}$$

(Note: w.p. = 'with probability')

At the beginning of each month, the company reviews the inventory (how much he still stores) level and decides how many itmes to order from its supplier.
Order Z items: incurs a cost of $K + iZ$, where K is the setup cost
When an order is placed, the time required for it arrive (deliver lag/lead time) is r.v. that distributed uniformly between 0.5 and 1 month

The company uses a stationary (s, S) policy to decide how much to order

$$D = \begin{cases} S - I & \text{if } I < s \\ 0 & \text{if } I \geq s \end{cases}$$

I is the inventory level at the beginning of the month

Most real inventory systems also have two additional types of costs, *holding* and *shortage* costs.

(*holding cost* includes costs as rental, insurance, taxes etc.)
holding cost: h
shortage cost: π

Let $I(t)$ be the inventory level at time t, which could be positive, negative or zero
Let $I^+(t) = \max(I(t), 0)$, the number of items physically on hand in the inventory at time t
Let $I^-(t) = \max(-I(t), 0)$, the backlog at time t (Note: $I^-(t) \geq 0$ as well)

The time-average (per month) number of items held in **inventory** for the n-month period is
$\bar{I}^+ = \frac{\int_0^n I^+(t)dt}{n}$, thus the average holding cost per month is $h\bar{I}^+$

The time-average (per month) number of items in **backlog** is
$\bar{I}^+ = \frac{\int_0^n I^-(t)dt}{n}$, thus the average backlog cost per month is $\pi\bar{I}^-$

The state variables for a simulation model of this inventory system:

- inventory level $I(t)$
- amount of an outstanding order from the company to the supplier
- time of the last event (needed to compute areas under $\bar{I}^+$ and $\bar{I}^-$)

1.5.1 Problem Statement

Event graph: P63
Flowchart (Order arrival, Demand, Inventory Evaluation, Update the continous-time statistical accumulators): P64-66

1.6 Alternative approaches to modeling and coding simulations

Previous two examples are executed in order of the events' simulated time of occurence(i.e.the simulation is *sequential*), and all work is done on a single

computer

Parallel and Distributed Simulation
Simulation across the Internet and Web-Based Simulation

1.7 Steps in a sound simulation study

(∗)Page 83-86

1.8 Other types of simulation

Continuous Simulation:
Modeling over time of a system by a representation in which the state variables change continuously with respect to time

Combined Discrete-Continuous Simulation:
With aspects of both discrete-event and continuous simulation

- A discrete event may cause a discrete change in the value of a continuous state variable
- A discrete event may cause the relationship governing a continuous state variable to change at a particular time
- A continuous state variable achieving a threshold value may cause a discrete event to occur or to be scheduled

Monte-Carlo Simulation:
A scheme employing random numbers, which is used for solving certain stochastic or deterministic problems where the passage of time plays no substantive role (generally static rather than dynamic)

1.9 Advantages, disadvantages, and pitfalls of simulation
Page 91-93

APPENDIX 1A: Fixed-Increment Time Advance
The simulation clock is advanced in increments of exactly Δt time units for some appropriate choice of Δt
If one or more events were scheduled to have occurred during this interval, these events are considered to occur at the end of the interval and the system state (and statistical counters) are updated accordingly
(Primary appears for systems where it can be reasonably be assumed that all events actually occur at one of the times $n\Delta t$: data in economic systems are often available only on an annual basis, the simulation clock: increments of 1 year)

APPENDIX 1B: Queueing systems
Characterize by three componenets: *arrival process, service mechanism, and queue discipline*
Measures of performance:

- D_i = delay in queue of ith customer
- $W_i = D_i + S_i$ = waiting time in system of ith customer
- $Q(t)$ = number of customers in queue at time t
- $L(t)$ = number of customers in system at time t [$Q(t)$ plus number of customers being served at time t]
- *steady-state average delay*: $d = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} D_i}{n}$
- *steady-state average waiting time*: $w = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} W_i}{n}$
- *steady-state time-average number in queue*: $Q = \lim_{T\to\infty} \frac{\int_0^T Q(t)dt}{T}$

- *steady-state time-average number in system*: $L = \lim_{T\to\infty} \frac{\int_0^T L(t)dt}{T}$
- *conservation equations*: $Q = \lambda d$, $L = \lambda w$
- $w = d + E(s)$

Chapter 2. Modeling Complex Systems

2.1: Introduction
List processing and similib(C-based simulation "language") utility functions can help us modeling more complexy systems

2.2 List Processing In Simulation
Two approaches to sorting lists of records in a computer — sequential and linked allocation (latter is preferable for complex simulations)

Sequential-allocation: records in a list are put into physically adjacent storage locations *Sequential-allocation*: pointers/links giving the logical relationship of the record to other records

- Faster process time
- Speed up event-list processing considerably
- Less computer memory required for storage
- Provides a general framework that allows one to store and manipulate many lists simultaneously with ease

Example2.1 Page 109 (List for queueing simulation)
Example2.2 Page 111 (List for inventory simulation)

Notes: Physical row is the place where it is stored, Number in the box has the time and type of event, and Pointer to other events (Needs to be verified)

2.3 A Simple Simulation Language: simlib

The heart of simlib is a collection of doubly linked lists, all residing together in dynamic memory, with space allocated as new records are filled into thelists, and space freed as records are removed from the lists. (There's a maximum of 25 lists, and the records in each list can have up to 10 attributes, with all data stored as type float)

Description of *simlibdefs.h* in C see Page 114-122

2.4 Single-Server Queueing Simulation With simlib

Code example see Page 125-127

We do not have to check for overflow of the queue here since simlib is automatically allocating storage dynamically for the lists as it is needed.

Question(P127): Why do still need to increment number of customers delayed when the server is idle?
Answer: Even the server is idle, we'll say the customer experiences a delay of 0

2.5 Time-Shared Computer Model

Example see Page 129
Where simlib package simplify coding the model considerably

Chapter 3. Simultaion Software

3.1: Introduction

- Generating random numbers/variates
- Advancing simulated time
- Determining the next event from the event list and passing control to the appropriate block of code
- Adding records to, or deleting records from a list
- Collecting output statistics and reporting the results
- Detecting error conditions

3.2: Simulation Packages vs Programming Languages

3.3: Classification of Simulation Software

Prototype customer-process routine for a single-server queueing system (Page 206)

3.4: Desirable Software Features

Chapter 4. Probability Review

4.2: Random variables and properties

Covariance: C_{ij} or $Cov(X_i, X_j) = E[(X_i - \mu_i)(X_j - \mu_j)] = E(X_i X_j) - \mu_i \mu_j$
If $C_{ij} = 0$, then X_i and X_j are said to be uncorrelated
If $C_{ij} \geq 0$, then positively correlated

Correlation $\rho_{ij} = \frac{C_{i,j}}{\sqrt{\sigma_i^2 \sigma_j^2}}$ (something like normalize?)

4.3: Simulation Output Data and Stochastic Processes

Stochastic process: a collection of similar random variables ordered over time
State space: Set of all possible values that these variables can take
Discrete-time: X_1, X_2, ...
Continuous-time: $X(t), t \geq 0$

Example 4.19: $M/M/1$ queue
IID interarrival times A_1, A_2...
IID service times S_1, S_2...
Stochastic process of delays in queue: D_1, D_2...
Then:
$D_1 = 0, D_{i+1} = max(D_i + S_i - A_{i+1}, 0)$

Covariance-stationary: for a discrete-time stochastic process X_1, X_2 ..., if

(i)**Mean**: $\mu_i = \mu$ for $i = 1, 2...$ and $-\infty < \mu < \infty$
(ii)**Variance**: $\sigma_i^2 = \sigma^2$ for $i = 1, 2...$ and $\sigma^2 < \infty$
(The mean and variance are stationary over time, where μ, σ^2 are the common mean and variance)
(iii)**Covariance**: $C_{i,i+j} = Cov(X_i, X_{i+j})$ is independent of i for $j = 1, 2, ...$
(The covariance between X_i, X_{i+j} depends only on the seperation/lag j)

(iv)**Correlation**: $\rho_j = \rho_{i,i+j} = \frac{C_{i,i+j}}{\sqrt{\sigma_i^2 \sigma_{i+j}^2}} = \frac{C_j}{\sigma^2} = \frac{C_j}{C_0}$

If $X_1, X_2, ...$ is a stochastic process beginning at time 0 in a simulation, then it is queite likely not to be covariance-stationary
However, for some simulations $X_{k+1}, X_{k+2}, ...$ will be approximately covariance-stationary if k is large enough, where k is the length of the warmup period.

4.4: Estimation of means, variances, and correlations

4.5: Confidence intervals and hypothesis tests for the mean

4.6: The strong law of large numbers

4.7: The danger of replacing a probability distribution by its mean

APPENDIX 4A: Comments on convariance-stationary processes
Consider a process $\{D_i, i \geq 1\}$ for the $M/M/1$ queue when no customers are present at time 0, then $D_1 = 0$, $E(D_1) = 0$ and $E(D_i) \geq 0$ for $i = 2, 3, ...$, therefore $\{D_i, i \geq 1\}$ is not covariance-stationary.
However, when k is sufficiently large ('warm up' for some amount of time),

$\{D_i, i \geq 1\}$ is covariance-stationary.

Chapter 5. Building valid, credible, and approriately detailed simulation models

Page 264 - 291 for reference

Chapter 6. Selecting Input Probability Distributions

6.1: Introduction

Choice of probability distributions can evidently have a large impact on the simulation output and potentially on the quality of the decisions made with the simulation results.

Collect data to specify a distribution for input random variable:

1. Data values themselves are dirrectly in the simulation (*trace-driven simulation*), but simlutaion can only reproduce what has happened historically and seldom enough for all desired simulation, but it can be used for *model validation* to compare with the model output

2. Data values themselves are used to define an emperical distribution function in some way, which is generally preferable

3. Standard techniques of statistical inference are used to "fit" a theoretical distribution form, to the data and to perform hypothesis tests to determine the goodness of fit, which is generally preferable when the oberseved data reasonably fits a theoretical distribution

***6.2: Useful Probability Distributions

Continuous Distributions: See Page 299-318

- Uniform: $U(a, b)$
- Exponential: $expo(\beta)$
- Gamma: $gamma(\alpha, \beta)$
- Weibull: $Weibull(\alpha, \beta)$
- Normal: $N(\mu, \sigma^2)$
- Lognormal: $LN(\mu, \sigma^2)$
- Beta: $beta(\alpha_1, \alpha_2)$
- Pearson type V: $PT5(\alpha, \beta)$
- Pearson type VI: $PT6(\alpha_1, \alpha_2, \beta)$
- Log-logistic: $LL(\alpha, \beta)$
- Johnson S_b: $JSB(\alpha_1, \alpha_2, a, b)$
- Johnson S_U: $JSU(\alpha_1, \alpha_2, \gamma, \beta)$
- Triangular: $triang(a, b, c)$

Discrete Distributions: See Page 319-326

- Bernoulli: $Bernoulli(p)$
- Discrete uniform: $DU(i, j)$
- Binomial: $bin(t, p)$
- Geometric: $geom(p)$
- Negative binomial: $negbin(s, p)$
- Poisson: $Poisson(\lambda)$

6.4: ACTIVITY I: Hypothesizing Families of Distributions

In practice we seldom have enough theoretical prior information to select a single distribution, we'll hypothesize families of distributions that might be representative of a simulation input random variable

Useful **summary statistics**: some functions that are useful to suggest an appropriate distribution family
(P334 table 6.5: Min/Max, Mean, Median, Variance, Coefficient of variation, Lexis ratio, Skewness etc.)
lexis ratio: coefficient of variation for discrete distribution
skewness: measure of the symmetry of a distribution

A **histogram** is an estimate (except for rescaling) of the density function
(There are other more sophisticated ways)

The **The quantile summary** is a synopsis of the sample that is useful for determining whether the underlying probability density function or probability mass function is symmetric or skewed to the right/left (applicable to both discrete/continuous)

6.5: ACTIVITY II: Estimation of Parameters

After one(or more) candidate families of distributions have been hypothesized in ACTIVITY I, we need to specify the values of their parameters in order to have completely specified distributions for simulation

When data are used directly in this way to specify a numerical value for an unknown parameter, we say that we are *estimating* that parameter from the data.

Estimator: numerical function of the data
There are many ways to specify and the form of an estimator and evaluate the quality.
Here we consider on type: *Maximum-likelihood estimators (MLEs)* for 3 reasons:

- MLEs have serveral desirable properties oftern not enjoyed by alternative methods of estimation (e.g. least-squared estimators, unbiased estimators)
- the use of MLEs turns out to be important in justifying the chi-square goodness-of-fit test
- the central idea of maximum-likehood estiamtion has a strong intuitive appeal

Suppose we've hypothesized a discrete distribution for our data that has one unknown parameter θ
Let $p_\theta(x)$ denote the probability mass function
Given that we've observed $X_1, X_2...$, we define *likelihood function* $L(\theta)$
$L(\theta) = p_\theta(X_1)p_\theta(X_2)...p_\theta(X_n)$ which is also a joint probability mass function since data are independent
This gives us the probability of obtaining our observed data if θ is the value of the unknown parameter
Then the MLE of the unknown value of θ, denoted by $\hat{\theta}$, is the value that maximizes $L(\theta)$
We can say that $\hat{\theta}$ best explains the data
Examples: Page 344-345

Some properties of MLEs:

- For most distributions, the MLE is unique
- Although MLEs nees not be unbiased, the asymptotic distribution of $\hat{\theta}$ has mean eqaul to θ
- MLEs are invariant
- MLEs are asymptotically normally distributed
- MLEs are strongly consistent

6.6: ACTIVITY III: Determining How Representative the Fitted Distributions are
Heuristic Procedures:

- Density/Histogram Overplots and Frequency Comparisons
- Distribution Function Differences Plots
- Probability Plots(*quantile-quantile plot, probability-probability plot*)

Goodness-of-fit Tests: is a statistical hypothesis test that is used to assess formally whether the observations are an independent sample from a particular distribution with distribution function $\hat{F}$
Null hypothesis H_0: the X_i's are IID random variables with distribution function $\hat{F}$

- Chi-square Test
- Kolmogorov-Smirnov Tests
- Anderson-Darling Tests
- Poisson-Process Tests

6.11: Selecting a Distribution in the Absence of Data
The first step: identify an interval $[a, b]$ in which it is felt that X will lie with probability close to 1
2 approach to place a density function on $[a, b]$
Triangular approach: Asked for their subjective estimate of the most likely time c to perform the task. Given a, b, c, the random variable X is then considered to ahve a triangular distribution on the interval $[a, b]$ with mode c
Beta approach: Assume that the random variable X has a beta distribution on this interval with sape parameters α_1, α_2
(If we assume X is equally likely to take on any value between a and b, choose $\alpha_1 = \alpha_2 = 1$)
(An generally more realistic way is to assume X is skewed to the right: $\alpha_2 > \alpha_1 > 1, \tilde{\alpha}_1 = \frac{(\mu-a)(2c-a-b)}{(c-\mu)(b-a)}, \tilde{\alpha}_2 = \frac{(b-\mu)\tilde{\alpha}_1}{\mu-a}$, $\mu > c$) ($\mu < c$: skewed to the left)

6.12: Models of Arrival Processes
Poisson Processes, Nonstationary Poisson Processes, Batch Arrivals(people arrive in groups)

6.13: Accessing the Homogeneity of Different Data Sets

Chapter 7. Random-Number Generators

7.1: Introduction
How random values can be conveniently and efficiently generated from a desired probability distribution for use in executingsimulation models

"Generating random variables" is not strictly correct since a random variable is defined in probability theory as a function satisfying certain conditions

We'll use more precise terminology "Generating random variates"

This entire chapter is devoted to methods of generating random variates from the unifrom distribution on the interval $[0, 1]$

A 'good' arithmetic random-number generator should posses serveral properties:

- The number produced should distributed uniformly on $[0, 1]$, no correlation with each other
- Fast and avoid the need for a lot of storage
- Able to reproduce a given stream of random numbers exactly

- There should be provision in the generator fo easily producing separate 'streams' of random numbers (i.e. generating interarrival times / service times)

7.2: Linear Congruential Generators
A sequence of integers $Z_1, Z_2...$ is defined by the recursive formula:

$$Z_i = (aZ_{i-1} + c)(\text{mod } m) = [a^i Z_0 + \frac{c(a^i-1)}{a-1}] \text{ (mod } m)$$

Where m (the modulus), a (the multiplier), c (the increment), and Z_0 (the seed or starting value), in addition to nonnegativity, the integers m, a, c and Z_0 should satisfy $0 < m, a < m, c < m, Z_0 < m$

LCG have a period of at most m, if it is exactly m, then it is called full period

Theorem 7.1: The LCG has full period iff the following three conditions hold:
(a) The only positive integer that (exactly) divides both m and c is 1
(b) If q is a prime number that devides m, then q devides $a - 1$
(c) If 4 devides m, then 4 divides $a - 1$

Mixed Generators: A choice of m that is good in all these respects is $m = 2^b$, where b is number of bits

Multiplicative Generators: the addition of c is not needed, but cannot have full period since Thm7.1(a), in this case the period is at most 2^{b-2}
The best m is now the largest prime number less than 2^b, the period is $m-1$ if a is a primitive element modulo m ... (Page 410)

7.3: Other Kinds of Generators
More General Congruences

Compostie Generators: use two or more separate generators and combine them in some wat to generate the final random numbers

7.4: Testing Random-Number Generators
Empirical Tests: use the generator to generate some U_i's, and to see how closely they resemble IID $U(0,1)$ random variates, we'll discuss four such tests:

- Whether U_i uniformly distributed between 0 and 1, which is a special case of chi-square test (all parameters known)
- Serial test: generalization of the chi-square test to higher dimensions
- Runs test: a test of independence, examine the U_i sequence for unbroken subsequences of maximal length within which the U_i's increase monotonically (such sbusequence is called a run up)
- Assess whether the generated U_i's exhibit discernible correlation: compute an estimate of the correlation at lags $j = 1, 2, ...l$ for some value of l

Empirical tests are only local (within that segment of a cycle)

Theoretical Tests:

Some General Observations on Testing: No amount of testing that can absolutely convince everyone. So the random-number generator should be tested in a way that is consistent with it's intended use

Chapter 8. Generating Random Variates

8.1: Introduction

In this chapter we assume that a distribution has already been specified somehow, and we address the issue of how we can generate random variates with this distribution in order to run the simultation model

The basic ingredient needed for every method of generating random variates from any distribution/random process is a source of IID $U(0,1)$ random variates. For this reason it is essential that a statistically reliable $U(0,1)$ random-number generator be available.

Issues to consider when selecting an alogrithm for generating random variates: exactness, efficient, complexity, robustness)

8.2: General Approaches to Generating Random Variables

8.2.1 Inverse Transform:
The algorithm for generating a random variable X having distribution function F is:

X continuous
(i) Generate $U \sim U(0,1)$
(ii) Return $X = F^{-1}(U)$

Example: Let X have the exponential distribution with mean β
The distribution function is:

$$F(x) = \begin{cases} 1 - e^{-\frac{x}{\beta}} & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

so to find F^{-1}, we set $u = F(x)$ and solve for x to obtain
$F^{-1}(u) = -\beta \ln(1-u)$
Thus to generate the desired r.v, we first generate $U \sim U(0,1)$ and then let $X = -\beta \ln U$. (We use U here instead of $1-U$ because both have the same $U(0,1)$ distribution)

X discrete
(i) Generate $U \sim U(0,1)$
(ii) Determine the smallest positive integer I such that $U \leq F(x_I)$, and return $X = x_I$

General form (for both discrete and continuous): $X = min\{x : F(x) \geq U\}$
Disadvantages:
Sometimes unable to write a formula for F^{-1} in closed form for the desired distribution
For a given distribution the inverse transform method may not be the fastest way to generate the corresponding random variate
Advantages:
(i) Fcilitate variance-reduction techniques that rely on inducing correlation between random variates (?), the inverse-transform method induces the strongest correlation between the generated random variates which we hope will propagate through the simulation model to induce the strongest possible correlation in the output, thereby contributing to the success of the variance-reduction technique

(ii) Ease of generating from truncated distributions

$$f^*(x) = \begin{cases} \frac{f(x)}{F(b)-F(a)} & \text{if } a \leq x \leq b \\ 0 & \text{otherwise} \end{cases}$$

$$F^*(x) = \begin{cases} 0 & \text{if } x \leq a \\ \frac{F(x)-F(a)}{F(b)-F(a)} & \text{if } a \leq x \leq b \\ 1 & \text{if } b \leq x \end{cases}$$

Then an algorithm for genearting an X having F^* is:
1. Generate $U \sim U(0,1)$
2. Let $V = F(a) + [F(b) - F(a)]U$
3. Return $X = F^{-1}(V)$

(iii) Useful for generating order statistics, usually we can generate n IID variates $Y_1, Y_2, ..., Y_n$ with distribution function F, then sort them into increasing order, and finally set X to the ith value of the Y_j's after sorting. Instead we can use the following algorithm:
1. Generate $V \sim beta(i, n-i+1)$
2. Return $X = F^{-1}(V)$

8.2.2 Composition:
Applies when the distribution function F from which we wish to generate can be expressed as a convex combination of other distribution functions $F_1, F_2, ...$
Page 449-451

8.2.3 Convolution:
A desired random variable X can be expressed as a sum of other random variables that are IID and can be generated more readily than direct generation of X.
We assume that IID r.v. $Y_1, Y_2, ..., Y_m$, s.t. $Y_1 + Y_2 + ... + Y_m$ has the same distribution as X, then we write $X = Y_1 + Y_2 + ... + Y_m$
The distribution of X is called the *m-fold convolution* (from stochastic process)

Note:
Composition: distribution of X is a weighted sum of other distribution functions
Convolution: r.v. X can be represented as a sum of other random variables

The three general approaches for generating random variates (inverse transform, composition and convolution) might be called *direct* since they deal directly with the distribution or random variable desired.
The *acceptance-rejection method* is less direct in its approach and can be useful when the direct methods fail or are inefficient.

8.2.4 Acceptance-Rejection:
We specify a function t that *majorizes* the density f $(t(x) \geq f(x), \forall x)$.
$c = \int_{-\infty}^{\infty} t(x)dx \geq \int_{-\infty}^{\infty} f(x)dx = 1$
Note: t in general is not a density, but $r(x) = t(x)/c$ clearly is a density
We must be able to generate a random variate Y having density r:
1. Generate Y having density r
2. Generate $U \sim U(0, 1)$, independent of Y
3. If $U \leq f(Y)/t(Y)$, return $X = Y$. Otherwise go back to step 1 and try

again
This algorithm continues looping back to step 1 until finally we have a (Y, U) pair in steps 1 and 2 for which $U \leq f(Y)t(Y)$

8.2.5 Special Properties:
Representing X in terms of other random variables that are more easily generated. (for example, convolution)
Examples see page 459-478

Rest of the Chapter: Generating Continuous/Discrete variables, Arrival Process etc.

Chapter 9. Output Data Analysis for a Single System

9.1 Introduction:
Output data analyses have not been conducted appropriate:

- Users often have the unfortunate impression that simulation is just an exercise in programming
- output porcesses of virtually all simulations are nonstationary and autocorrected, classical statistical techniques based on IID observations are not directly applciable(?)
- Cost of computer time

Example: Let $Y_1, Y_2...$ be an output stochastic process from a single simulation run. The Y_i's (production at ith hour) are random variables, in general, neither independent nor identically distributed.
$y_{11}, y_{12}, ..., y_{1m}$: realization of the random variables $Y_1, Y_2..., Y_m$ resulting from making a simulation run of length m observations using the random numbers $u_{11}, u_{12}, ...$ (u_{ji}: the ith random number used in the jth run)

Repeat this on different set of random numbers: $u_{11}, u_{12}, ...$, we obtain different realization $y_{21}, y_{22}, ..., y_{2m}$
Then
$y_{11}, y_{12}, ..., y_{1i}, ...y_{1m}$
$y_{21}, y_{22}, ..., y_{1i}, ...y_{2m}$
...
$y_{n1}, y_{n2}, ..., y_{ni}, ...y_{nm}$
Note: observations from a particular row are clearly not IID, but $y_{1i}, y_{1i}, ...y_{ni}$ are IID observations of Y_i (indenpedent across runs)

$\bar{y_i}(n) = \sum_{j=1}^{n} y_{ji}/n$ is a unbiased estimate of $E(Y_i)$

Problems:
1.$\hat{\theta}$ not an unbiased estimator of θ, that is, $E(\hat{\theta}) \neq \theta$
2.$\hat{Var}(\hat{\theta})$ not an unbiased estimator of $Var(\hat{\theta})$

9.2 Transient and Steady-state Behavior of a Stochastic Process:
Output stochastic process: $Y_1, Y_2, ...$
$F_i(y|I) = P(Y_i \leq y|I)$ for $i = 1, 2, 3...$ is the transient distribution of the output process at time i for initial conditions I

For a fixed y and I, $F_1(y|I), F_2(y|I)...$ are just a sequence of numbers

If $F_i(y|I) \to F(y)$ as $i \to \infty$ for all y and any initial conditions I, then $F(y)$ is called the steady-state distribution of the output process $Y_1, Y_2, ...$
(Strictly speaking $F(y)$ is only obtained in the limit as $i \to \infty$)
(See Figure 9.1)

Note: the steady-state distribution $F(y)$ does not depend on the initial conditions I, but the rate of convergence of the transient distributions $F_i(y|Y)$ to $F(y)$ does
(Distribution of Y_i is convergeing to the distribution of Y as i gets large)

9.3 Types of Simulations with Regard to Output Analysis:
Terminating simulation: for which there's a event E that specifies the length of each run/replication *initial conditions for a terminating simulation generally affect the desired measures of performance*
Example: bank closes each evening, company producing 100 airplanes etc.

Nonterminating simulation: no nature event E to specify the length of a run (a measure of performance for such simulation is said to be a steady-state parameter if it is a characteristic of the steady-state distribution of some output stochastic process $Y_1, Y_2, ...$ such as the mean $v = E(Y)$ or $P(Y \leq y)$)

A simulation might be either terminating or nonterminating, depending on the objectives of the simulation study
Consider a stochastic process for a nonterminating simulation that does not have steady-state distribution, we divide the time axis into equal-length, contiguous time intervals called *cycles*.
Let Y_i^C be a r.v. defined on the ith cycle, and assume $Y_1^C, Y_2^C, ...$ are comparable
Suppose that $Y_1^C, Y_2^C, ...$ has a steady-state distribution F^C and that $Y^C \sim F^C$, then a measure of performance is said to be a steady-state cycle parameter if it is a characteristic of Y^C (such as mean $v^C = E(Y^C)$)

9.4 Statistical Analysis for Terminating Simulations:
Suppose we make n independent replications of a terminating simulation (terminated by event E and begun with the "same" initial conditions, independence accomplished by using different random number for each replication)
Let X_j: random variable defined on the jth replication

9.4.1 Estimating Means:
Suppose we want to obtain a point estimate and condidence interval for the mean $\mu = E(X)$
$\bar{X}(n)$ is an unbiased point estimator for μ, and an approximate $100(1-\alpha)$ precent confidence interval for μ is given by:
$\bar{X}(n) \pm t_{n-1,1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}$
We call this confidence interval *fixed-sample-size procedure*, where the correctness depends on the assumption that X_j are normal random variables (rarely satisfied in practice)

????Example: 500 independent simulation experiments for M/M/1 queue with $\rho = 0.9$, for each experiment considered $n = 5, 10, 20, 40$ a for each n use the formula above to construct an approximate 90 percent confidence

interval for:
$d(25|s=0) = E(\frac{\sum_{i=1}^{25} D_i}{25}|s=0) = 2.12$ where s is number of customers at time 0
$\hat{p}$: proportion of the 500 confidence intervals that covered the true $d(25|s=0)$, a 90 percent confidence interval for the true coverage p (????)
The 90 percent confidence interval for the true coverage is computed from:
$\hat{p} \pm z_{0.95}\sqrt{\frac{\hat{p}(1-\hat{p})}{500}}$
For detail, see page 508

Obtaining a Specified Precision
One disadvantage of fixed-sample-size procedure is that we have no control over the confidence-interval half-length (precision of $\hat{X}(n)$)

Absolute error: $\beta = |\bar{X} - \mu|$, if we make replications of a simulation until the half-length of the $100(1-\alpha)$ percent confidence interval is less than or equal to β, then:
$1 - \alpha \approx P(\bar{X} - halflength \leq \mu \leq \bar{X} + halflength) \leq P(|\bar{X} - \mu| \leq \beta)$
Then $\bar{X}$ has an absolute error of at most β with a probability of approximately $1 - \alpha$

$n_\alpha^*(\beta)$: an approxiamte expression for the total number of replications to obtain an absolute error of β

Relative error: $\gamma = |\bar{X} - \mu|/|\mu|$, then: $1 - \alpha \approx P(|\bar{X} - \mu|/|\bar{X}| \leq halflength/|\bar{X}|) \leq P(|\bar{X} - \mu| \leq \gamma|\bar{X}|) = ... = P(|\bar{X} - \mu|/|\mu| \leq \gamma/(1-\gamma))$
Then $\bar{X}$ has a relative error of at most $\gamma/(1-\gamma)$ with a probability of approximately $1 - \alpha$

$n_r^*(\gamma)$: an approxiamte expression for the total number of replications to obtain a relative error of β

Sequential procedure: new replications are added one at a time, for obtaining an estimate of μ with a specified relative error that takes only as many replications as are actually needed (with a relative error of γ and a confidence level of $100(1-\alpha)$ precent)

Choose an initial number of replications $n_0 \geq 2$ and let $\delta(n, \alpha) = ...$ be the

usual confidence-interval half-length.
Then,
(0) Make n_0 replications of the simulation and set $n = n_0$
(1) Compute $\bar{X}(n)$ and $\delta(n, \alpha)$ from $X_1, X_2, ..., X_n$
(2) If $\delta(n, \alpha)/|\bar{X}(n)| \leq \gamma'$, use $\bar{X}$ as the point estimate for μ and stop. Equivalently, $I(\alpha, \gamma) = [\bar{X}(n) - \delta(n, \alpha), \bar{X}(n) + \delta(n, \alpha)]$ is an approximiate $100(1 - \alpha)$ precent confidence interval.
Otherwise replace n by $n + 1$ and make an additional replication of the simulation, goto step 1

Recommended Use of the Procedures (see page 515)

9.4.2 Estimating Other Measures of Performance:

9.4.3 Choosing Initial Conditions:
Terminating simulation depend explicitly on the state of the system at time 0
(Example: estimate delay at bank, but we start counting from midnight, will cause an underestimate result)
First approach: start the simulation early ("warmup period"), and only uses the delays between the busiest period. The disvantage is that "warmup period"(few hours of simulation) are not used directly in the estimate, and no guarantee that the conditions at noon will be representative. *Second approach*: collect data on the number of customers present in the bank at noon for several different days, simulate with number of customers randomly choosen from the distribution collected and calculated.

9.5 Statistical Analysis:

9.5.1 The Problem of the Initial Transient:
Suppose we want to estimate the steady-state mean $v = E(Y)$, which is also generally defined by

$$v = \lim_{x \to \infty} E(Y_i)$$

The most serious consequence of the problem of the initial transient: $E[\bar{Y}(m)] \neq v$ for any m (what does this actually mean?)
Most common technique: called *warming up the model* or *initial-data deletion*
Delelting some number of obeservations from the begining of a run and to use only the remaining observations to estimate v
For example, given obeservations $Y_1, Y_2, ..., Y_m$, suggested to use

$$\bar{Y}(m,l) = \frac{\sum_{i=l+1}^{m} Y_i}{m-l}$$

rather than $\bar{Y}(m)$ as an estimator of v
How do we chooose the warmup period l? We pick l, m such that $E[\bar{Y}(m,l)] \approx v$, detail algorithm see page 520

9.5.2 Replication/Deletion Approach for Means:
Six fundamental approaches for addressing the problem to estimate steady-state mean $v = E(Y)$, we'll concentrate on one of these, **replication/deletion approach**
(gives reasonably good performance, easiest approach to understand and implement, applies to all type output parameters, easily used to estimate serveral different parameters, can be used to compare different system configurations)
It uses one set of n replications to determine the warmup period l, and then uses only the last $m' - l$ obervations from a different set of n' replications to perform the actual analyses.
Detail see page 526

9.8 Time Plots of Important Variables:

Chapter 10. Comparing Alternative System

10.1 Introduction:
Discuss statistical analyses of the output from several different simulation models that might represent competing system designs or alternative operating policies.

Example: Zippy costs twice as much to purchase and opertate as Klunky
We could either puchase: (a) one Zippy (b) two Klunky
After performing 100 indenpendent experiments, it turns out the best system (with smaller average delay) is actually two-Klunky installation (but the observed average delays overlap substantially) (The proportion of experiments favoring one-Zippy is also shown, which refers to the proportion of a wrong recommmendation, and it is considerably high 0.34 even for $n = 20$ replications)

10.2 Confidence Intervals for the Difference Between the Expected Responses of Two Systems:
Compare two systems on the basis of some performance measure, or expected response.
Effect this comparison by forming a condifidence interval for the difference in the two expectations, rather than by doing a hypothesis test to see whether the observed difference is significantly different from zero.
(A confidence interval quantifies how much the expectations differ, if at all)

For $i = 1, 2$ let $X_{i1}, X_{i2}, ..., X_{in_i}$ be a sample of n_i IID observations from system i
$\mu_i = E(X_{ij})$: expected response of interest, and we want to construct a confidence interval for $\xi = \mu_1 - \mu_2$

10.2.1 A Paired-t Confidence Interval:
If $n_1 = n_2$, or we're willing to discard some observations, then we can pair X_{1j}, X_{2j} to define $Z_j = X_{1j} - X_{2j}$ for $j = 1, 2, ..., n$
Then Z_j are IID r.v, $E(Z_j) = \xi$

$$\bar{Z}(n) = \frac{\sum_{j=1}^{n} Z_j}{n}$$

$$\hat{Var}[\bar{Z}(n)] = \frac{\sum_{j=1}^{n} [Z_j - \bar{Z}(n)]^2}{n(n-1)}$$

Therefore the approx $100(1-\alpha)$ percent confidence interval:

$$\bar{Z}(n) \pm t_{n-1,1-\alpha/2}\sqrt{\hat{Var}[\bar{Z}(n)]}$$

(Note: it Z_j's are normally distributed then this is exact, otherwise CLT implies that this covers ξ with probability near $1-\alpha$ for large n)
(Note: We don't have to assume X_{1j}, X_{2j} are independent or $Var(X_{1j}) = Var(X_{2j})$)
(Note: X_{ij} are r.v over an entire replication, such as average of 100 delays on jth replciation, but not a delay of a individual customer)

10.2.2 A Modified Two-Sample-t Confidence Interval:
Unliked paired-t, we do not pair up observations from the two systems (also do not require $n_1 = n_2$), but require X_{1j}'s indenpendent of the X_{2j}'s
To apply classic Two-Sample-t : must have $Var(X_{1j}) = Var(X_{2j})$ (not realistic)
We'll use a old but reliable approximate solution:

$$\bar{X}_i(n_i) = \frac{\sum_{j=1}^{n_i} X_{ij}}{n_i}$$

$$S_i^2(n_i) = \frac{\sum_{j=1}^{n_i}[X_{ij} - \bar{X}_i(n_i)]^2}{n_i - 1}$$

for $i = 1, 2$
Compute the estimated degrees of freedom: $f = ...$
and use

$$\bar{X}_1(n_1) - \bar{X}_2(n_2) \pm t_{\hat{f},1-\alpha/2}\sqrt{\frac{S_1^2(n_1)}{n_1} + \frac{S_2^2(n_2)}{n_2}}$$

as an approx $100(1-\alpha)$ precent confidence interval for ξ, which is usually called **welch confidence interval**

10.2.3 Contrasting the Two Methods:
Using common random numbers for simluating the two systems can often lead to a considerable reduction in $Var(Z_j)$ and to a much smaller confidence

interval, this implies that $n_1 = n_2$, and X_{1j}, X_{2j} will not be independent (paired-t approach)
If $n_1 \neq n_2$ (welch approach)

10.2.4 Comparisons Based on Steady-State Measures of Performance:
Previous are all terminating simulations, which we can simply replicating the simulation some number of times
But for non-terminating ones, we can no longer simply replicate the models, since initialization effects may bias the output(difficult to effect a valid comparison based on steady-state performance measures)

(????)
Example 10.5 (Page 561): how the replication/deletion approach for steady-state analysis can be adapted to constructing a confidence interval for the difference between two steady-state means

10.3 Confidence Intervals for Comparing More Than Two Systems:

We'll make serveral confidence-interval statements simultaneously, so their individual levels will have to be adjusted upward so that the overall confidence level of all intervals' covering their respective targets is at the desired level $1 - \alpha$ (We'll use Bonferroni inequality to ensure)

10.3.1 Comparisons with a Standard:
Suppose one of the model variants is a 'standard' systems (perhaps representing the existing system or policy). If we call the standard system 1 and the other variants systems $2, 3, ..., k$, the goal is to construct $k-1$ confidence intervals for the $k-1$ differences $\mu_2 - \mu_1, \mu_3 - \mu_1, ..., \mu_k - \mu_1$ with overall confidence level $1 - \alpha$

Thus we're making $c = k - 1$ individual intervals, so they should each be constructed at level $1 - \frac{\alpha}{k-1}$
With a confidence level of at least $1 - \alpha$ for all $i = 2, 3, ..., k$, system i differs

from the standard if the inverval for $\mu_i - \mu_1$ misses 0, or it is not significantly different from the standard if this interval contains 0
Example of comparison: Page 563
The Bonferroni inequality is quite general, so it doesn't matter how the individual confidence intervals are formed (need not result from the same number of replications of each model, nor independent)
The above approach could also be used for steady-state comparisons by using a technique for constructing individual confidence intervals for steady-state differences.

10.3.2 All Pairwise Comparisons:
Compare each system with every other system to detect and quantify any significant pairwise differences.
One approach: form confidence intervals for $\mu_{i_2} - \mu_{i_1}$ for all i_1, i_2 between 1 and k with $i_1 < i_2$ (there will be $\frac{k(k-1)}{2}$ individual intervals, therefore each must be made at level $1 - \frac{\alpha}{\frac{k(k-1)}{2}}$ in order to have a confidence level of at least $1 - \alpha$)

10.3.3 Multiple Comparisons with the Best (MCB):
Forms simultaneous confidence intervals for the differences between the means of each of the k alternatives and that of the best of the other alternatives: form k simultaneous confidence intervals on $\mu_i - max_{l \neq i} \mu_l$ (assuming that bigger means better)

MCB usually results in confidence intervals that are smaller than those resulting from application of the Bonferroni inequality (good or bad?)

10.4 Ranking and Selection:

10.4.1 Selecting the Best of k systems:
A procedure to select one of the k systems as being the best one, in some

sense, and to control the probability that the selected system really is the best one

Let X_{ij}: r.v. of interest from the jth replication of the ith system and let $\mu_i = E(X_{ij})$ (X_{ij} are assumed to be independent of each other)

Let μ_{i_l}: lth smallest of the μ_i's, s.t. $\mu_{i_1} \leq \mu_{i_2} \leq ... \leq \mu_{i_k}$
(Our goal is to select the smallest expected response μ_{i_1})

Inherent randomness of X_{ij} implies that we can never be absolutely sure of the correct selection, but we can prespecify the probability of making a correct selection.

Denote correct selection as **CS**, then we want $P(CS) \geq P^*$ provided $\mu_{i_2} - \mu_{i_1} \geq d^*$, where the minimal CS probability $P^* \geq 1/k$ and the "indifference" amount $d^* \geq 0$ are both specified by the analyst.

Statistical procedure for this problem involves **"two-stage"** sampling from each of the k systems.

We make a fixed number of replications of each system, then use the resulting variance estimates to determine how many more replications from each systems are necessary in a second stage of sampling in order to reach a decision.
(It must be assumed that X_{ij} are normally distributed, but we need not assume $\sigma_i^2 = Var(X_{ij})$ are known, nor do we need to assume that σ_i^2 are same for different i's)

We make $n_0 \geq 2$ replications of each of the k systems and define the first-stage sample means and variances for $i = 1, 2, ..., k$:

$$\bar{X}_i^{(1)}(n_0) = \frac{\sum_{j=1}^{n_0} X_{ij}}{n_0}$$

and

$$S_i^{(1)}(n_0) = \frac{\sum_{j=1}^{n_0} [X_{ij} - \bar{X}_i^{(1)}(n_0)]^2}{n_0 - 1}$$

Then we compute total sammple size N_i needed for system i

$$N_i = max...$$

Next we make $N_i - n_0$ more replciations of system i and obtain the second-stage sample means:

$$\bar{X}_i^{(2)}(N_i - n_0) = \frac{\sum_{j=n_0+1}^{N_i} X_{ij}}{N_i - n_0}$$

Then define the weights

$$W_{i1} = ...$$

and $W_{i2} = 1 - W_{i1}$

Finally define the weigthed sample means:

$$\tilde{X}_i(N_i) = W_{i1}\bar{X}_i^{(1)}(n_0) + W_{i2}\bar{X}_i^{(2)}(N_i - n_0)$$

And we select the system with the smallest $\tilde{X}_i(N_i)$
Choices of P^*, d^* depend on goals, and specifying them might be tempered by the computing cost of obtaining a large N_i with a large P^* or small d^*

But n_0 is more hard to choose, and based on experiments and various statements in the literature, it should be at least 20?? (Can't be too small or too large, then we'll have a poor estimate or 'overshoot' the necessary numbers of replications for some of the systems which is wasteful)

10.4.2 Selecting a Subset of Size m Containing the Best of k Systems:
Picking a subset of m of the k systems so that this *selected subset contains the best system*, again with a specified probability

10.4.3 Selecting the m Best of k systems:

Selecting the m Best of k systems