Kui Wu

Lecture Notes of CSC 446/546

Computer Science Dept., University of Victoria

– Discrete-Event System Simulation –

Victoria

British Columbia
Canada V8W 3P6

Contents

1

Overview

1.1 Course Logistics and Workload

Refer to the course outline.

1.2 How Do We Study a System?

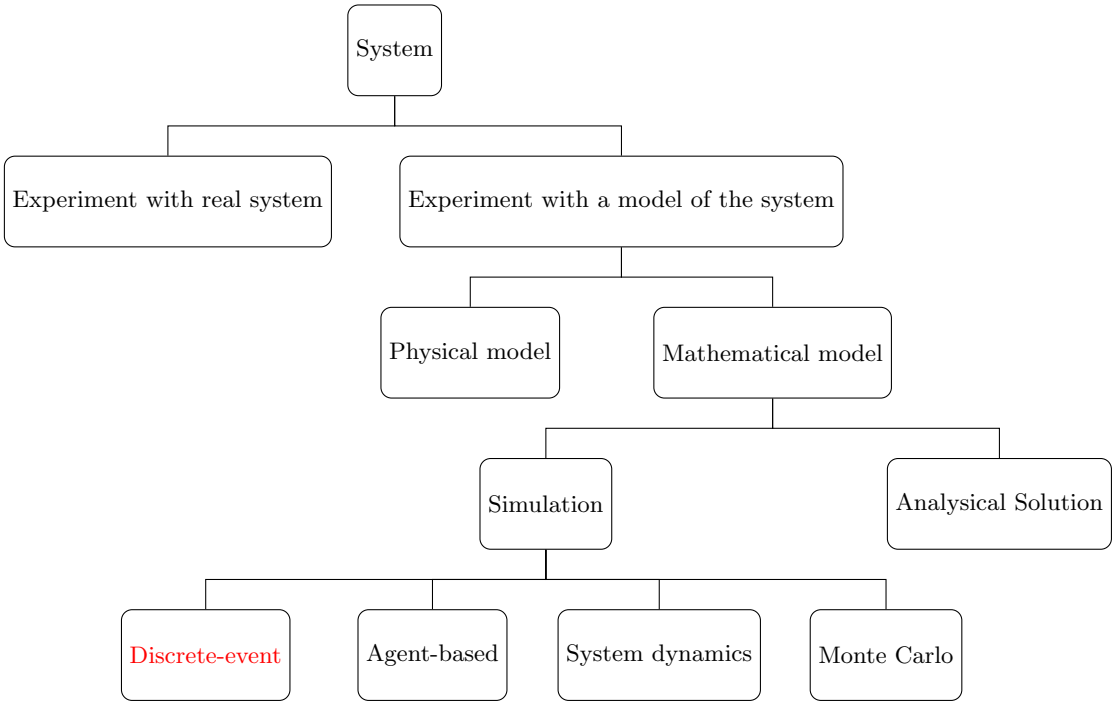


Fig. 1.1. Ways to study a system (revised based on [1]).

Fig. 1.1 shows different ways to study a system. In this course, we will focus on discrete, dynamic, and stochastic simulation. *Typically*, the term *discrete-event simulation* is used to describe this type of simulation[1].

[1] Although discrete-event simulation can include deterministic models, we exclude deterministic simulation from our scope of study.

1.3 Why Simulation?

A "simulation" is the software imitation of the operation of a real-world process or system over time. There are many advantages of using simulation, including, for example:

- Simulation enables the study of, and experimentation with, the internal interactions of a complex system or of a subsystem within a complex system. Many modern systems (factory, wafer fabrication plant, service organization, etc.) are so complex that its internal interactions can be treated only through simulation.
- Informational, organizational, and environmental changes can be simulated, and the effect of these alterations on the model's behavior can be observed.
- The knowledge gained during the design of a simulation model could be of great value towards suggesting improvement in the system under investigation.
- Simulation can be used to experiment with new designs or policies before implementation, so as to prepare for what might happen. It can effectively answer "what-if" questions.
- Simulation can be used to verify analytic solutions.
- Simulation models designed for training make learning possible without the cost and disruption of on-the-job instruction.
- Animation shows a system in simulated operation so that the plan can be visualized.

1.4 Disadvantages and Pitfalls

Simulation, however, has its drawbacks, including, for example:

- Each run of simulation produces only *estimates* of a model's true characteristics for a particular set of input parameters.
- Simulation models are often expensive and time-consuming to develop. While building a simulation is cheaper than building a real system, the cost of simulation should not be underestimated.
- If a model is not a valid representation of a system under study, the simulation results would provide little useful, sometimes even harmful, information about the actual system.

Simulation must be used with caution. Below are common pitfalls in a simulation study:

- Inappropriate level of model abstraction.
- Failure to have people with a knowledge of simulation methodology and statistics on the modeling team. (This explains why we need to understand probability and statistics.)
- Failure to collect appropriate system data.
- Using incorrect distribution as input to the simulation. (We will address this problem in *input modelling*.)
- Analyzing the output data from one simulation run. (We will address this problem in *output analysis*.)
- Failure to have a warmup period, if the steady-state behavior of a system is of interest. (We will address this problem in *output analysis*.)

1.5 Applications of Simulation

Simulation plays a crucial role in many critical areas, including for example:

- Healthcare and Medicine: Simulation is widely used for training medical professionals in procedures, emergency response, and patient safety. It allows for practice without risk to real patients and is essential for developing both technical and teamwork skills.
- Aviation and Aerospace: Pilots and crew use simulators to practice flying, emergency procedures, and teamwork, significantly improving safety and preparedness.
- Emergency Response and Defense: Simulations help train first responders, military personnel, and disaster teams to handle complex, high-pressure situations, such as natural disasters or mass casualty incidents, without real-world risks.

- Manufacturing and Engineering: Simulations are used to optimize production processes, test new designs, and improve safety, reducing the need for costly prototypes and minimizing errors before implementation.
- Business and Project Management: Simulation-based training enhances decision-making, negotiation, and change management skills, allowing teams to explore scenarios and outcomes in a risk-free environment.
- Robotics and Automation: Robotics simulators allow for the development and testing of algorithms and applications without the need for expensive or unavailable hardware, accelerating innovation.
- Sports and Biomechanics: Computer simulations are used to predict outcomes, analyze performance, and improve training techniques for athletes.
- AI/ML: Simulation data can be used to train AI models faster and more accurately, helping overcome the challenge of limited or costly real-world data collection. Simulation can significantly enhance reinforcement learning (RL) by addressing key challenges in training and deployment.

1.6 Supplementary Reading Materials

Textbook (Law 2024): Chapter 1.1, 1.2, 1.7. Note that my lectures do not strictly follow the textbook.

2

Simulation Examples

2.1 General Steps in a Sound Simulation Study

The general steps of a sound simulation study are shown in Figure 2.1. Roughly speaking, the whole process includes four phases:

- Phase 1: Pre-modeling. Step 1 in Figure 2.1.
- Phase 2: Model Building. Steps 2-6 in Figure 2.1.
- Phase 3: Model Runs. Steps 7-8 in Figure 2.1.
- Phase 4: Implementation. Steps 9-10 in Figure 2.1. Note that "implementation" here does not refer to the programming implementation of the simulation. Instead, "implementation" here means applying the conclusions drawn from the simulation study in real-world practice, e.g., for the decision-making process.

2.2 General Structure of Discrete Event-Driven Simulation

The general structure for a discrete event-driven simulation is shown in Figure 2.2. The simulation examples below follow this general structure.

2.3 Single Server Queueing

We make the following assumptions:

- Queue: It is first-in-first-out (FIFO) queue.
- Server: There is only one server serving the customers.
- Work-conserving: If the queue is not empty, the server cannot be idle.
- Arrival Process: customers' interval arrival time follows a given distribution, e.g., uniform, exponential.
- Service Process: customer service time follows a given distribution, e.g., exponential.

The key concepts in simulating the above single-serve queueing system include:

- System state: The system state can be represented by (# of customers in the system, status of the server), where "# of customers in the system" denotes the total number of customers in the system, including those waiting in the queue and the one who is being served, and "status of the server" is a boolean variable with 1 denoting busy and 0 idle.
- Event: A set of circumstances that causes an instantaneous change in the system's state. Obviously, a customer arrival and a customer departure are two events. At a minimum, an event should include the event type and the time, i.e., when the event occurs. (**Q: do we need a "service event"?**)

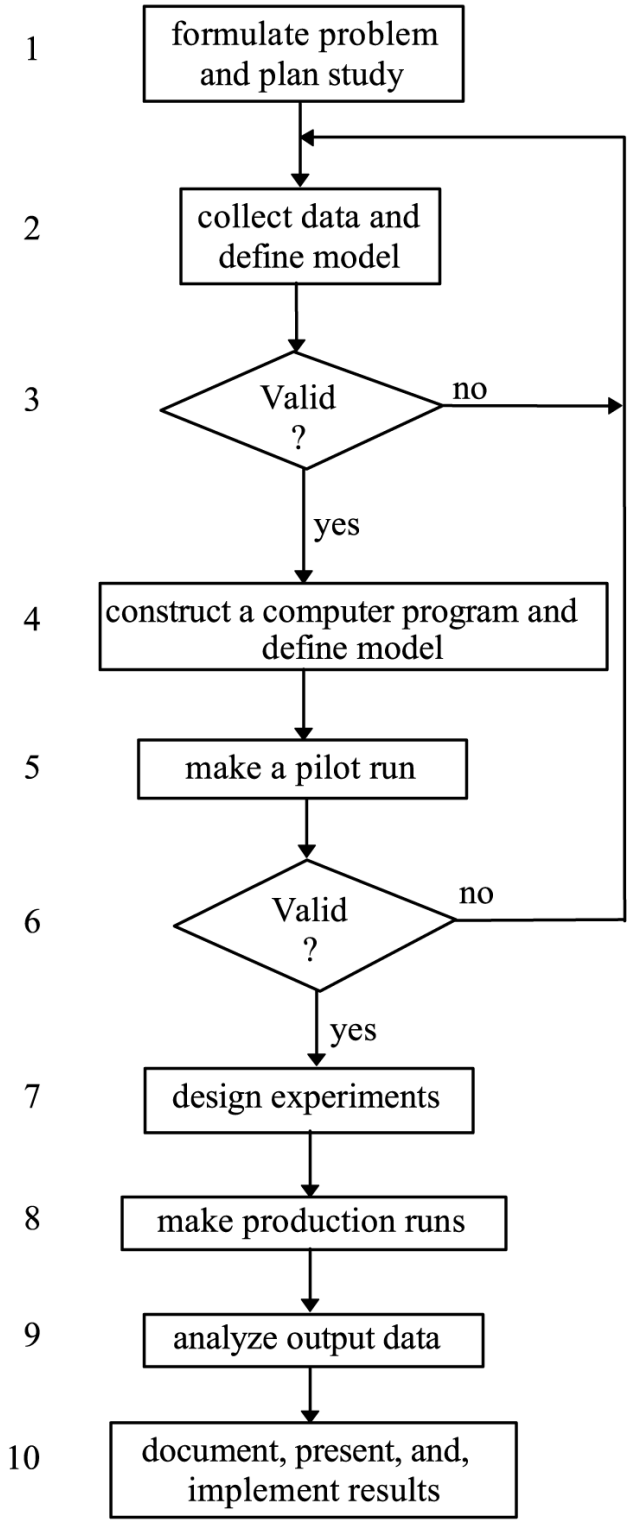


Fig. 2.1. Steps in a simulation study (source: Law (2024)).

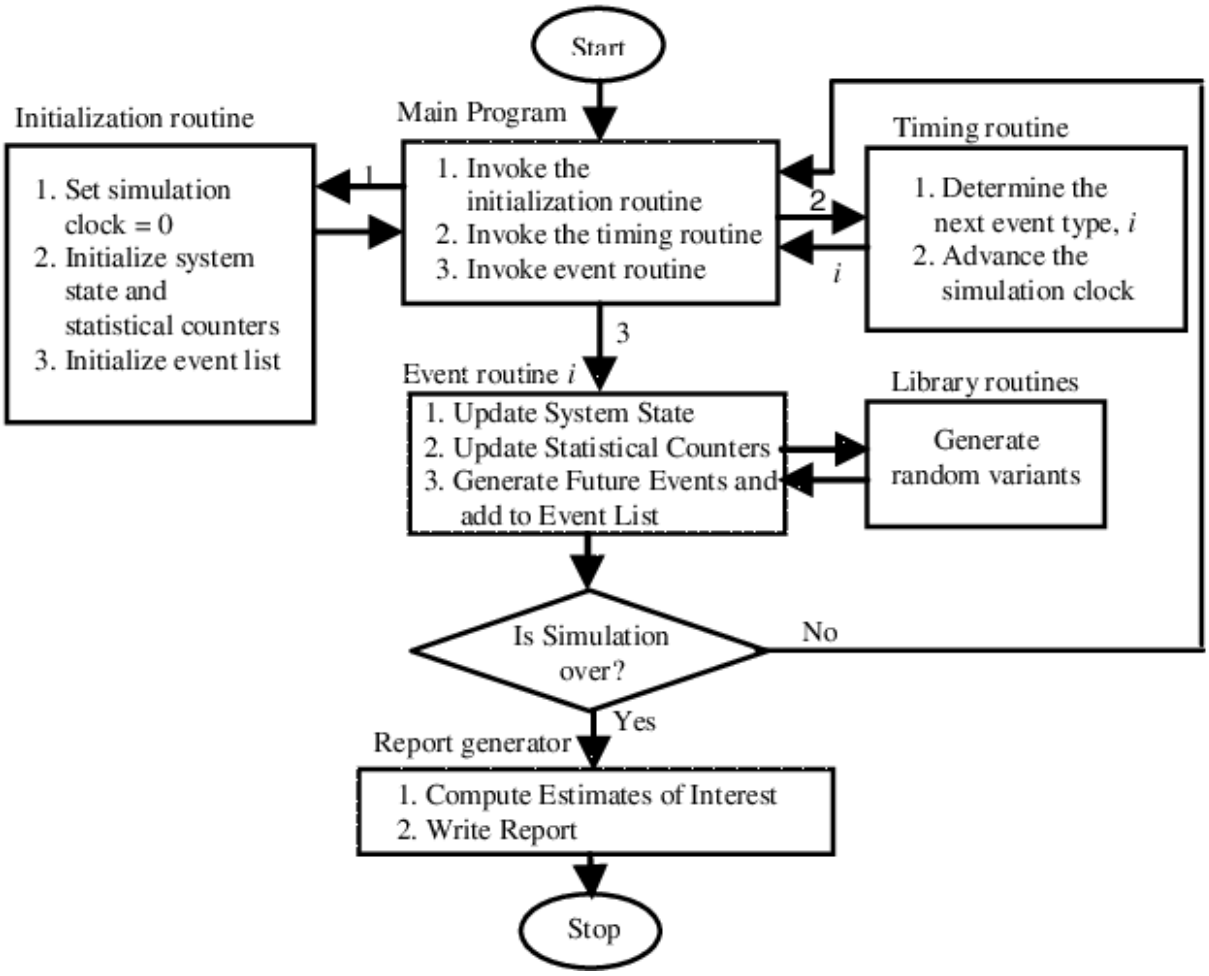


Fig. 2.2. General structure of discrete event-driven simulation (source: Law (2024)).

- Event list: A data structure that stores the events. Note that events should be ordered in their time. A complex simulation may need to store thousands of events, which should be inserted, ordered, and removed from the event list. Implementing the event list is critical to the speed of the simulation. Note that the event list is often called the Future Event List (**FEL**), because the simulation clock is always moving forward.
- Simulation Clock: This is a virtual clock that tracks the progress of the simulation. It does not matter whether or not to set the simulation clock to integers, but the simulation clock should always move forward!

The main structure of the simulation is as follows:

```
Initialization();

// Loop until first "TotalCustomers" have departed
while(NumberOfDepartures < TotalCustomers ) {
  Event evt = (Event)FutureEventList.getMin();  // get imminent event
  FutureEventList.dequeue();                    // be rid of it
  Clock = evt.get_time();                       // advance simulation time
  if( evt.get_type() == arrival ) ProcessArrival(evt);
```

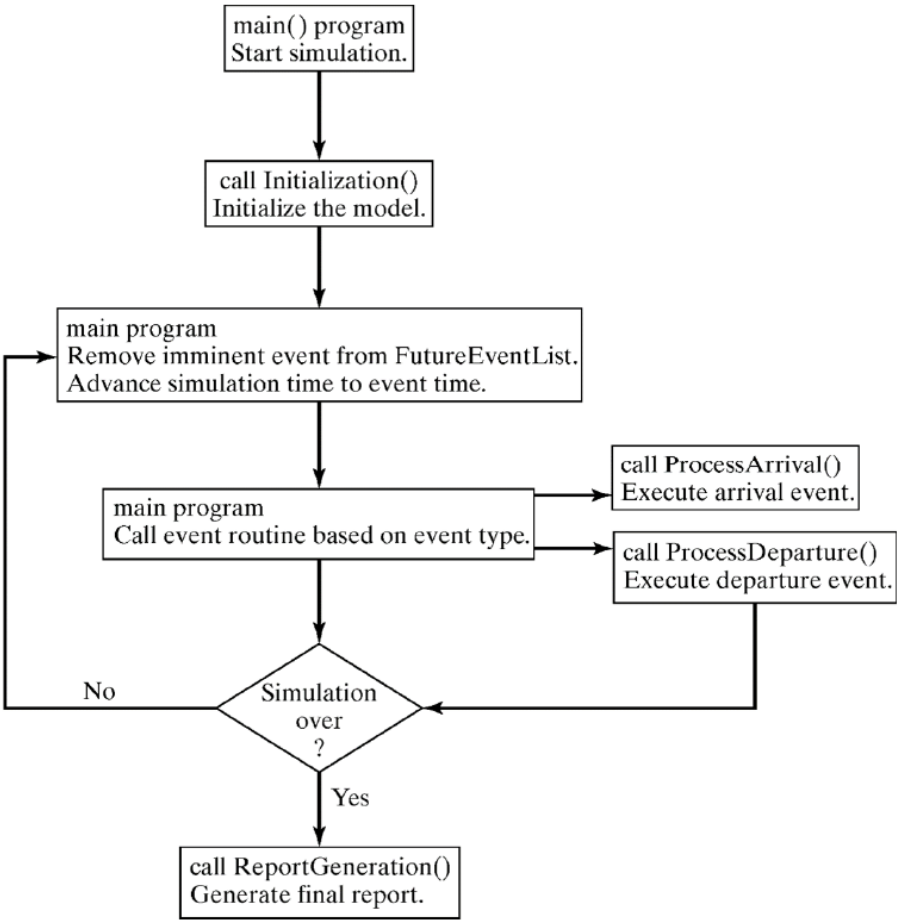


Fig. 2.3. The main structure of single-serve queueing.

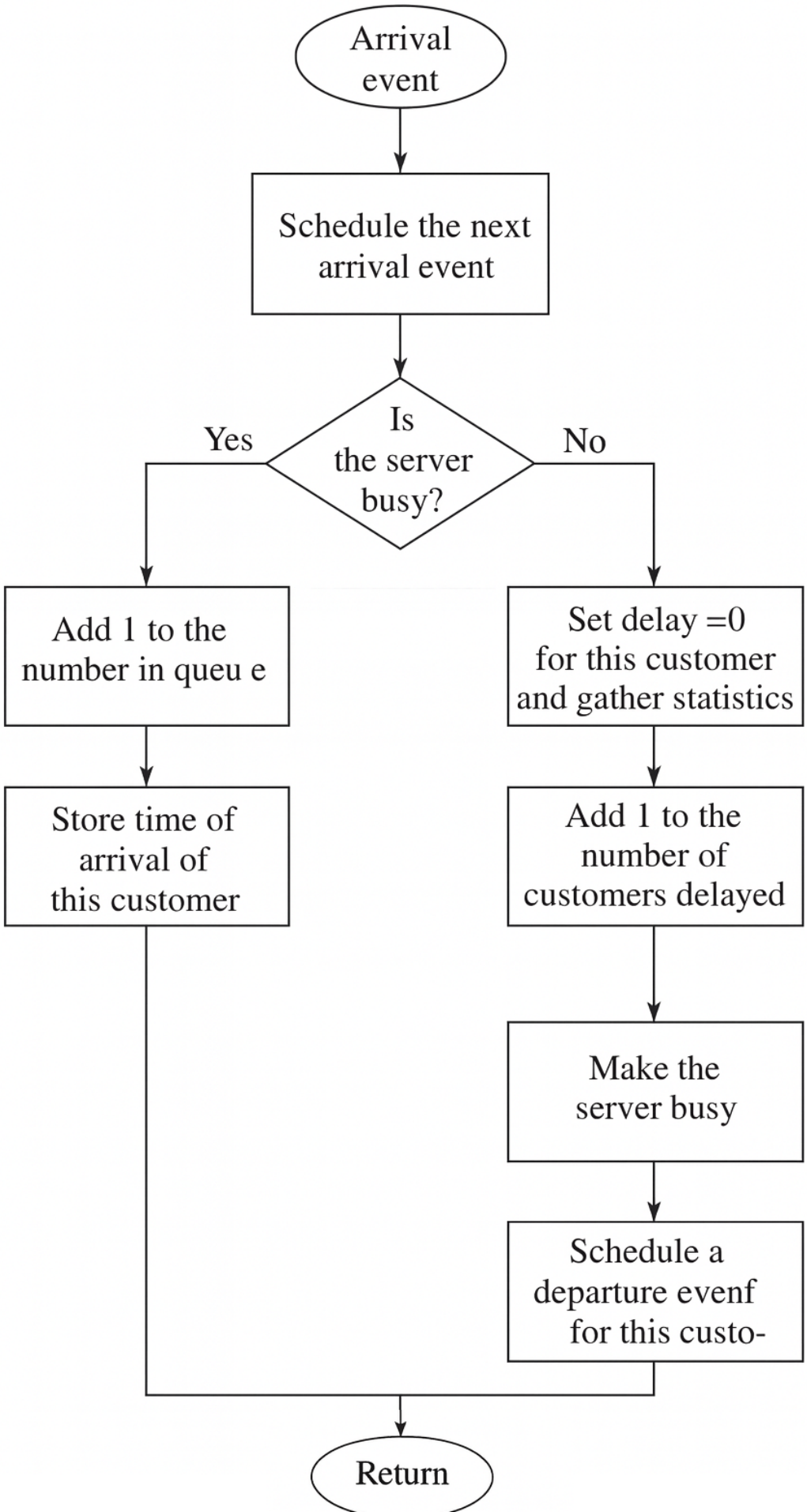
```
    else  ProcessDeparture(evt);
    }
  ReportGeneration();
}
```

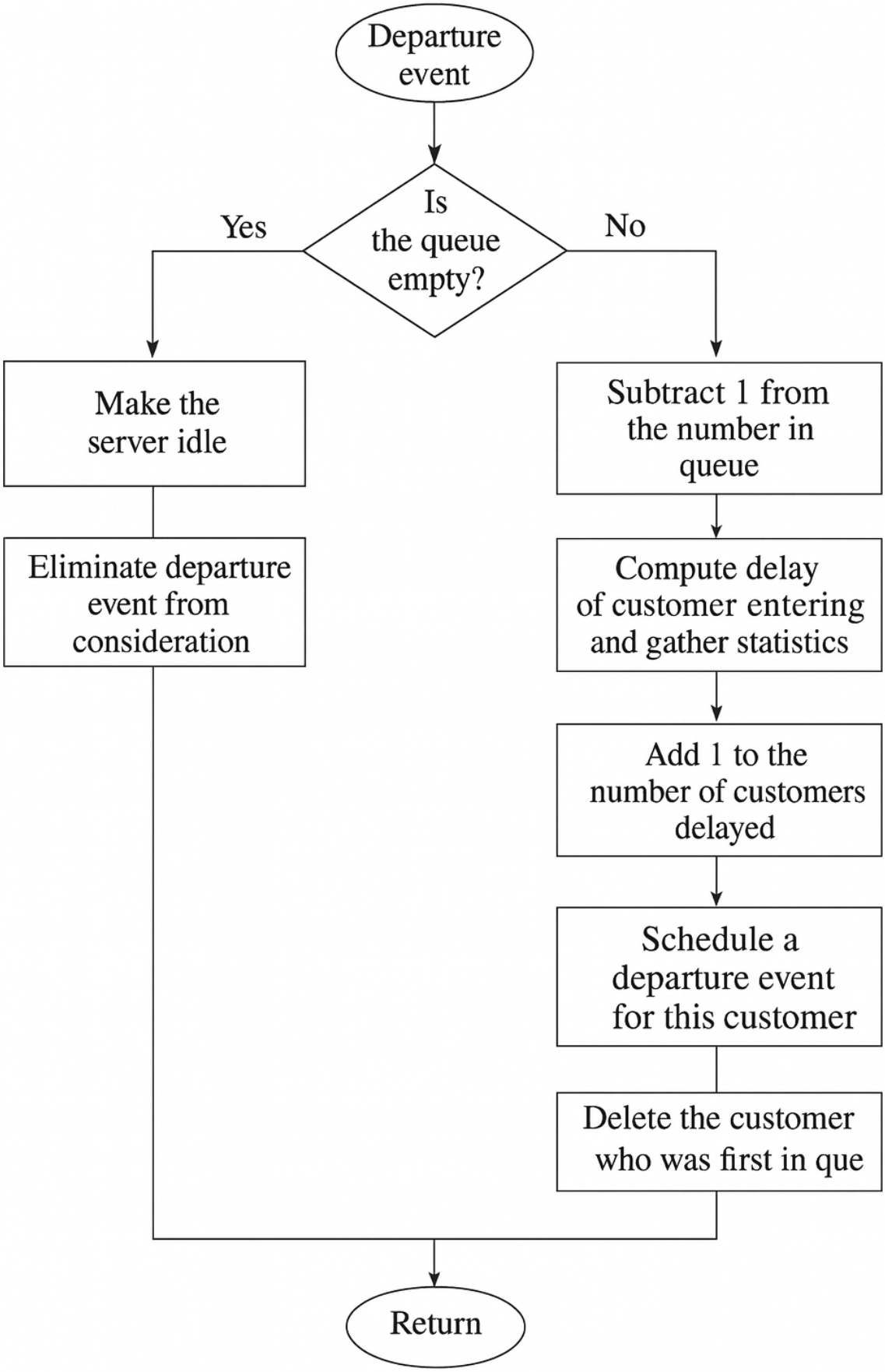
The workflows for *ProcessArrival* and *ProcessDeparture* are as follows:

Example 2.1. We use an example run to illustrate the behaviour of the simulation. Assume that the customers' arrival times and service duration are as follows.

Table 2.1. An example run of the simulation

Customer No.	Arrival Time	Service Duration	Service Start Time	Service End Time
1	0	2	0	2
2	2	1	2	3
3	6	3	6	9
4	7	2	9	11
5	9	1	11	12
6	15	4	15	19


Arrival event
Schedule the next arrival event
Is the server busy?
Yes
No
Add 1 to the number in queu e
Store time of arrival of this customer
Set delay =0 for this customer and gather statistics
Add 1 to the number of customers delayed
Make the server busy
Schedule a departure evenf for this custo-
Return


Departure event
Is the queue empty?
Yes
No
Make the server idle
Eliminate departure event from consideration
Subtract 1 from the number in queue
Compute delay of customer entering and gather statistics
Add 1 to the number of customers delayed
Schedule a departure event for this customer
Delete the customer who was first in que
Return

Based on the above table, we can list all the events generated during this simulation run. Since we have 6 customers and each customer generates an arrival event and a departure event, we have 12 events in total. Below is the list of events in chronological order:

Table 2.2. The list of events in chronological order

Event Type	Customer No.	Event Time
Arrival	1	0
Departure	1	2
Arrival	2	2
Departure	2	3
Arrival	3	6
Arrival	4	7
Departure	3	9
Arrival	5	9
Departure	4	11
Departure	5	12
Arrival	6	15
Departure	6	19

Figure 2.6 shows the dynamic changes regarding the number of customers in the system:

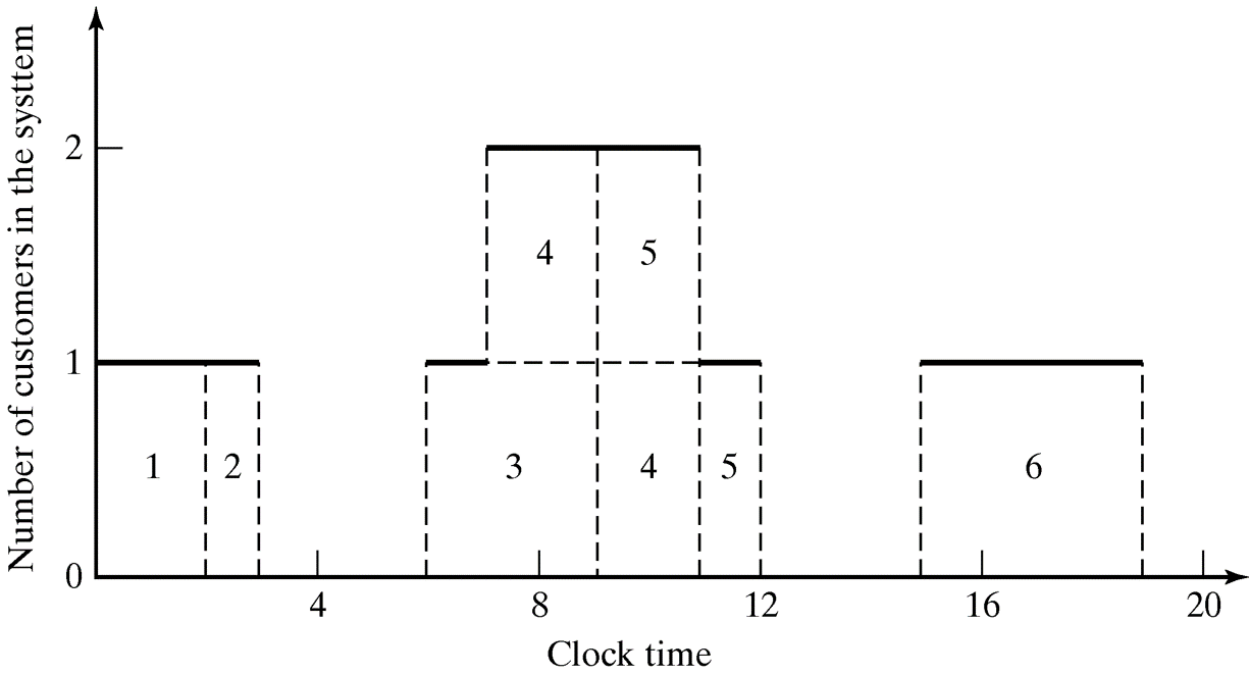


Fig. 2.6. The number of customers in the system over time.

2.4 Performance Metrics for Queueing Systems

- Average waiting time in the queue
- Probability for waiting

- Probability of idle server
- Average service time
- Average time between arrivals
- Average time customer spends in the system
- server utilization

All the above performance metrics are calculated using the data from a simulation run.

The following metric is theoretical and is calculated using system parameters instead of the simulation run data.

- Theoretical server utilization (also called long-term server utilization): $\rho = \frac{\lambda}{\mu}$, where λ denotes the average arrival rate, i.e., the average number of customer arrivals per unit time, and μ denotes the average service rate, i.e., the average number of customers that the server can serve per unit time.

Rule of thumb: A theoretical result is derived using mathematical formulas, whereas an empirical result must be computed based on simulation output data. If the simulation is correctly implemented and runs for a sufficiently long duration, the two results should closely align.

2.5 Other Simulation Examples

Please follow Example 2.1 to analyze the simulation of a single-queue, multiple-server system by yourself.

For example, we assume that we have two servers, A and B, serving a single FIFO queue. Assume that server A is faster (on average) than server B.

Policy: A simplifying rule is that server A serves customers if both servers A and B are idle. The solution would be different if the decision were made at random or by any other rule.

2.6 Simulation with Event Time Relationship

So far, we have provided a general framework for performing *any* discrete-event simulation. Nevertheless, in some cases, we can build simpler simulations if we can determine the relationship between events.

Below is an example that uses event time relationships to simulate a single-queue, single-server queueing system.

```
# Time Service Begin
 Time_Service_Begin[i] = max(arrival_time[i],Time_Service_Ends[i-1])

 # Time customer waiting in queue
 Time_Customer_Waiting_in_Queue[i] = Time_Service_Begin[i]-arrival_time[i]

 # Time service ends
 Time_Service_Ends[i] = Time_Service_Begin[i] + service_time[i]

 # Time Customer Spend in the system
 Time_Customer_Spend_in_System[i] = Time_Service_Ends[i] - arrival_time[i]

 # Time when the system remains idle
 if (arrival_time[i]>Time_Service_Ends[i-1]):
   System_idle[i] = arrival_time[i]-Time_Service_Ends[i-1]
 else:
   System_idle[i] = 0
```

2.7 Monte Carlo Simulation Examples

A bomber attempting to destroy an ammunition depot, as shown in Figure 2.7 (This bomber has conventional rather than laser-guided weapons). The bomber flies in the horizontal direction and carries 10 bombs. The aiming point is (0,0). If a bomb falls anywhere on the target, a hit is scored; otherwise, the bomb is a miss.

The point of impact is assumed to be normally distributed around the aiming point with a standard deviation of 400 meters in the direction of flight and 200 meters in the perpendicular direction. Our goal is to simulate the operation and make statements about the number of bombs on target.

Recall that the standardized normal variate, Z, having mean 0 and standard deviation 1 is distributed as $Z = (X - \mu)/\sigma$, where X is a normal random variable, μ is the mean of distribution X and σ is the standard deviation of X. Then $X = Z\sigma_X$ and $Y = Z\sigma_Y$ where (X, Y) are the simulated co-ordinates of the bomb after it has fallen. Since $\sigma_X = 400$ and $\sigma_Y = 200$, we have $X = 400Z_i$ and $Y = 200Z_j$. Note that the i and j subscripts have been added to indicate that the values of Z should be independent (Why???).

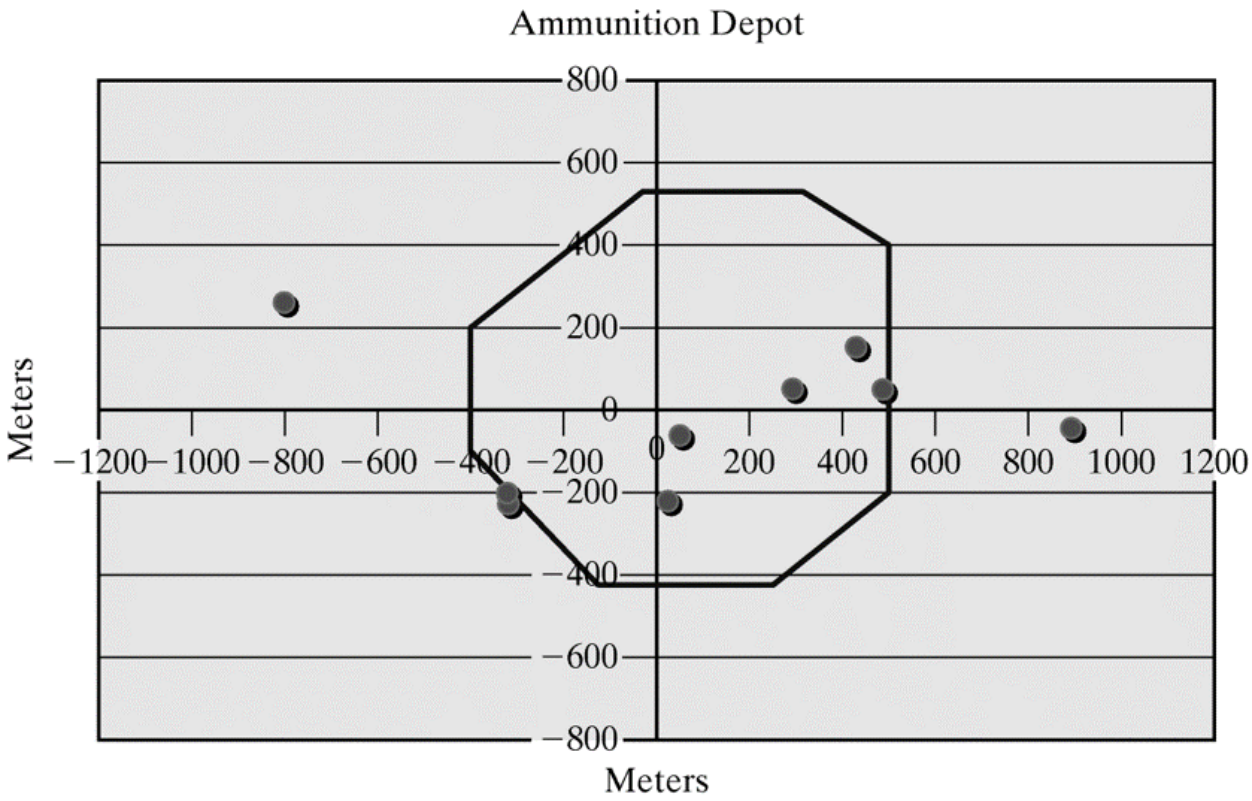


Fig. 2.7. Monte Carlo simulation for a bombing mission[2]

2.8 Supplementary Reading Materials

Textbook (Law 2024): Chapter 1.4, 1.6. Note that my lectures do not strictly follow the textbook.

3

Statistical Models in Simulation

3.1 Brief Review

3.1.1 Probability

Simply put, the probability is a measure of chance. Human beings, by nature, are not good at probabilistic thinking and often estimate the chance by intuition. To be slightly formal, we need to define probability space so that we understand the exact meaning of "chance".

Definition 3.1. ***Probability space*** *is defined as the triple* $\{\Omega, \mathcal{F}, P\}$. Ω *is the sample space, defined as a set of all possible outcomes of an experiment (or all possible sample points).* $\mathcal{F}$ *is the event space, i.e., the set of events with an event being a set of outcomes in the sample space.* $P(w)$ *is the probability measure associated with* $w(w \in \mathcal{F})$. *In addition,* $P(w)$ *must satisfy the following properties:*

1. $0 \leq P(w) \leq 1$;
2. $P(\Omega) = 1$ *and* $Prob(\emptyset) = 0$;
3. *For any sequence of events* $E_1, E_2, \ldots$, *that are mutually exclusive, i.e.,* $\forall n, m, E_n E_m = \emptyset$ *when* $n \neq m$, *then*

$$P(\bigcup_{n=1}^{\infty} E_n) = \sum_{n=1}^{\infty} P(E_n).$$

Example 3.2. Assume that we throw a fair dice and count the points of the up face. The probability space in this example is: $\Omega = \{1, 2, 3, 4, 5, 6\}$. We can define an event A as "the points of the up face are an even number". Then $P(A) = P(\text{the up face is } 2) + P(\text{the up face is } 4) + P(\text{the up face is } 6) = \frac{1}{2}$.

Remark 3.3. When you calculate probability, you must always be aware of the sample space. Human beings tend to forget about the sample space and jump to irrational conclusions. It is suggested that you always ask yourself "**out of how many**" when you calculate your chance. Several examples: the Monty Hall problem, the birthday paradox, the medical test fallacy (base rate neglect).

3.1.2 Random Variables

A random variable is a real-valued function defined on a set of possible outcomes, the sample space Ω. That is, the random variable is a function that maps from its domain, the sample space Ω, to its range, the real numbers or a subset of the real numbers.

Note that a random variable is a function. In this sense, it is right to say that "*let X denote* ...", where ... is the statement of the function, i.e., the statement must be a value. It is right to say that "*Let X denote the sum of two fair dice.*" But it is not right to say "*let X denote a random variable that the sum of two fair dice is* 5," because "*the sum of two fair dice is* 5" is an event instead of a function! If you really want to use a random variable to mean some event that happens, you could use a special random variable, indicator random variable, which returns 1 when the event happens and 0 otherwise.

Example 3.4. Let E be the event that the sum of two fair dice is an odd number. Let I_E be an indicator random variable such as $I_E = 1$ if E is true and $I_E = 0$ otherwise.

A random variable X is either a discrete random variable or a continuous random variable. X is a discrete random variable if the number of possible values of X is finite or countably infinite. X is a continuous random variable if its range space is an interval or a collection of intervals. Put in another way, we can enumerate the values of a discrete random variable, but we cannot enumerate the values of a continuous random variable.

Remark 3.5. In this course, it is safe and totally fine to assume that a random variable X is either a discrete random variable or a continuous random variable. Note that this classification is rather artificial. For example, assume that X is a discrete random variable and Y is a continuous random variable. We can define a third random variable Z by tossing a coin.

$$Z = \begin{cases} X, & \text{if the coin is heads} \\ Y, & \text{otherwise,} \end{cases}$$

which does not fall in the above classification. Fortunately, in this course we do not need to deal with such random variables, which require the knowledge of measure theory.

Remark 3.6. **Random variable vs. random variate:** A *random variable* is a function that maps elements from a sample space Ω to real numbers (or a subset thereof). In contrast, a *random variate* refers to a specific numerical outcomea single realizationof a random variable. Therefore, when we refer to the process of producing numerical samples, we speak of *random variate generation*, not *random variable generation*.

3.1.3 PMF, PDF, and CDF

Probability Mass Function (PMF): For a **discrete** random variable X, let's use $P(x_i)$ to denote the probability that $X = x_i$, i.e., $P(x_i) = P(X = x_i)$. The collection of pairs $(x_i, P(x_i)), i = 1, 2, \ldots$ is called PMF of X.

Probability Density Function (PDF): For a **continuous** random variable X, let's assume

$$P(a \leq X \leq b) = \int_a^b f(x)dx.$$

Then $f(x)$ is called the PDF of X.

Cumulative Distribution Function (CDF): CDF of X, $F_X(x) = P(X \leq x)$. When the context is clear, we often omit the subscript from F_X, i.e., $F(x) = P(X \leq x)$.

Based on the definition of CDF, we have $F(x) = \sum_{x_i \leq x} P(x_i)$ if X is a discrete random variable, and $F(x) = \int_{-\infty}^{x} f(t)dt$ if X is a continuous random variable.

Question 3.7. **Discrete random variable**: Consider the experiment of tossing a single die. Assume that the die is loaded so that the probability that a given face lands up is proportional to the number of spots showing. Define X as the number of spots on the up face of the die after a toss. What is the PMF of X? What is the CDF of X?

Question 3.8. **Continuous random variable**: Assume that we track the lifetime of a device. Let X denote the life of the device (in the unit of years). Assume that the PDF of X is

$$f(x) = \begin{cases} \frac{1}{2}e^{-x/2}, & x \geq 0 \\ 0, & x < 0, \end{cases}$$

What is the probability that $X = 2$ years? What is the probability that X is between 2 and 3 years?

3.1.4 Expectation

The expectation of a random variable X is denoted by $E(X) = \mu$.

$$E(X) = \begin{cases} \sum_{\text{all i}} x_i P(X = x_i) & \text{If } X \text{ is discrete} \\ \int_{-\infty}^{\infty} x f(x) dx & \text{If } X \text{ is continuous} \end{cases}$$

The expected value of X is also known as the mean, the average, or the first moment of X. Correspondingly, we can define the n-th moment of X for $n > 1$.

$$E(X^n) = \begin{cases} \sum_{\text{all i}} x_i^n P(X = x_i) & \text{If } X \text{ is discrete} \\ \int_{-\infty}^{\infty} x^n f(x) dx & \text{If } X \text{ is continuous} \end{cases}$$

In this class, we only care about the second moment of X because the second moment relates to the dispersion we will discuss below.

Question 3.9. **Discrete random variable**: Consider the experiment of tossing a single die. Assume that the die is loaded so that the probability that a given face lands up is proportional to the number of spots showing. Define X as the number of spots on the up face of the die after a toss. What is the expectation of X?

Question 3.10. **Continuous random variable**: Assume that we track the lifetime of a device. Let X denote the life of the device (in the unit of years). Assume that the PDF of X is

$$f(x) = \begin{cases} \frac{1}{2} e^{-x/2}, & x \geq 0 \\ 0, & x < 0, \end{cases}$$

What is the expectation of X?

3.1.5 Measure of Dispersion

In this class, we only consider two measures of dispersion, **variance** and **standard deviation**. Note that there are other types of measures for dispersion using higher moments. Also, note that there is no such thing called "standard variance".

The variance of X is denoted by $V(X)$, or $var(X)$, or σ^2, and is calculated by:

$$V(X) = E((X - E(X))^2) = E((X - \mu)^2) = E(X^2) - \mu^2.$$

Question 3.11. Can you figure out the last step in the above equation?

The standard deviation of X is denoted by σ, i.e.,

$$\sigma = \sqrt{V(X)}.$$

Note that the variance of X is always non-negative, and the unit of the standard deviation of X is the same as the mean of X.

Question 3.12. **Continuous random variable**: Assume that we track the lifetime of a device. Let X denote the life of the device (in the unit of years). Assume that the PDF of X is

$$f(x) = \begin{cases} \frac{1}{2} e^{-x/2}, & x \geq 0 \\ 0, & x < 0, \end{cases}$$

What are the variance and the standard deviation of X? What conclusion can you draw for an exponential random variable?

Remark 3.13. It is not hard to calculate that the mean value of an exponentially distributed random variable with parameter λ is $\frac{1}{\lambda}$. You must be very careful that "an exponential random variable with the parameter of λ" and "an exponential random variable with a mean of λ" are different!

3.1.6 Mode

In the discrete random variable case, the mode is the value of the random variable that occurs most frequently.

In the continuous random variable case, the mode is the value at which the PDF is maximized.

Clearly, mode might not be unique. If the modal value occurs at two values of the random variable, we call the random variable bi-modal.

Question 3.14. **Discrete random variable**: Consider the experiment of tossing a single die. Assume that the die is loaded so that the probability that a given face lands up is proportional to the number of spots showing. Define X as the number of spots on the up face of the die after a toss. What is the mode of X?

Question 3.15. **Continuous random variable**: Assume that we track the lifetime of a device. Let X denote the life of the device (in the unit of years). Assume that the PDF of X is

$$f(x) = \begin{cases} \frac{1}{2}e^{-x/2}, & x \geq 0 \\ 0, & x < 0, \end{cases}$$

What is the mode of X?

3.2 Important Discrete Distribution

3.2.1 Bernoulli Trials and Bernoulli's Distribution

Bernoulli Trials: Consider an experiment consisting of n trials; each can be a success with probability p or a failure with probability $q = 1 - p$. Denote $X_j = 1$ if the j-th trail is a success, otherwise $X_j = 0$, i.e.,

$$P(X_j) = \begin{cases} p, & X_j = 1 \\ 1 - p, & X_j = 0 \end{cases}$$

X_j follows the Bernoulli distribution. The Bernoulli distribution, named after Swiss mathematician Jacob Bernoulli, is the discrete probability distribution of a random variable which takes the value 1 with probability p and the value 0 with probability $1 - p$.

Note that $E(X_j) = p$ and $V(X_j) = pq$.

3.2.2 Binomial Distribution

In n Bernoulli trials, the number of successes, X, has a binomial distribution:

$$P(X = x) = \begin{cases} \binom{n}{x} p^x q^{n-x}, & x = 0, 1, 2, \ldots, n \\ 0, & \text{otherwise} \end{cases}$$

An easy way is to consider a binomial random variable as the sum of n **independent** Bernoulli random variables $X = X_1 + X_2 + \ldots + X_n$. Therefore, $E(x) = np$ and $V(X) = npq$.

3.2.3 Geometric Distribution and Negative Binomial Distribution

The number of Bernoulli trials, X, to achieve the **first** success is geometric distribution. That is, if X follows geometric distribution with parameter p, then

$$P(X = x) = q^{x-1}p, x = 1, \ldots$$

Clearly, $E(X) = 1/p$ and $V(X) = q/p^2$.

The number of Bernoulli trials, X, until the k^{th} success, is the negative binomial distribution. That is, if X follows negative binomial distribution with parameter p, then

$$P(X = x) = \binom{x-1}{k-1} q^{x-k} p^k, x = k, k+1, \ldots$$

An easy way is to consider a negative binomial random variable as the sum of k **independent** geometric random variables $X = X_1 + X_2 + \ldots + X_k$. Therefore, $E(x) = k/p$ and $V(X) = kq/p^2$.

Remark 3.16. There is another form of geometric distribution, which models the number of failures until the first success. In this case,

$$Prob(X = x) = q^x p, x = 0, 1, 2, \ldots$$

3.2.4 Poisson Distribution

A non-negative discrete random variable X follows a Poisson distribution with rate $\lambda > 0$ if its pmf follows

$$P(X = k) = \begin{cases} \frac{e^{-\lambda}\lambda^k}{k!}, & k = 0, 1, 2, \ldots, \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

Note that $E(X) = V(X) = \lambda$.

Refer to Remark 3.26 (bullet 1) for more explanation. Mathematical evidence is provided in Section 3.2.5.

Question 3.17. 1. Why does (3.1) defines a probability density function?
2. What is the practical meaning that $P(X = 0)$?
3. (*) Why can the Poisson distribution be used to model the number of calls in a telephone network? (Read Section 2 of "Concentration Inequalities and Martingale Inequalities: A Survey" by Chung and Lu)

Remark 3.18. Read Section 2 of "Concentration Inequalities and Martingale Inequalities: A Survey". What is the practical meaning of the binomial distribution? Understand when we should use the Poisson distribution to approximate the binomial distribution and when we should use the normal distribution to approximate the binomial distribution.

3.2.5 Relationship Between Exponential Random Variables and Poisson Arrivals: Memoryless Property

Poisson arrival is a random process, in which the number of arrivals in a unit of time follows a Poisson distribution with parameter λ. This is equivalent to saying that the interval times of Poisson arrivals follow an exponential distribution with parameter λ (or with mean $\frac{1}{\lambda}$).

The above fact is based on (1) a Poisson arrival has the memoryless property, and (2) the exponential random variable is the **only** random variable that has the memoryless property.

Definition 3.19. *A random variable X is said to be memoryless if*

$$P(X > s + t|X > t) = P(X > s), \forall s, t \geq 0.$$

For instance, if we think of X as being the lifetime of some instrument, the probability that it lives for at least $s + t$ hours given that it has worked for t hours is the same as the initial probability that it lives for at least s hours. That is, it does not remember that it has been in use for a time duration of length t.

Note that the memoryless property seems to be counter-intuitive. But this property significantly simplifies mathematical analysis and is broadly accepted as a good approximation of real-world phenomena.

It is easy to validate that exponential distributions have the memoryless property:

$$\begin{aligned} P(X > s + t|X > t) &= \frac{P(X>s+t, X>t)}{P(X>t)} \\ &= \frac{P(X>s+t)}{P(X>t)} \\ &= \frac{e^{-\lambda(s+t)}}{e^{-\lambda t}} \\ &= e^{-\lambda s} \\ &= P(X > s). \end{aligned}$$

The proof of the uniqueness is omitted.

Example 3.20. Suppose buses arrive at a stop such that the time between arrivals follows an exponential distribution with an average of 10 minutes. That is,

$$T \sim \text{Exponential}(\lambda = \tfrac{1}{10}).$$

Now, suppose you have already been waiting for 5 minutes. What is the probability that you will have to wait at least 7 more minutes?

By the memoryless property:

$$P(T > 5 + 7 \mid T > 5) = P(T > 7).$$

So, the probability that you have to wait at least 7 more minutes (given you already waited 5) is the same as if you just started waiting now.

Using the survival function of the exponential distribution:

$$P(T > t) = e^{-\lambda t},$$

we compute:

$$P(T > 7) = e^{-\frac{1}{10} \cdot 7} = e^{-0.7} \approx 0.4966.$$

3.3 Important Continuous Distribution

We introduce the following continuous distributions:

- Uniform Distribution: **the principle of indifference**[1].
- Exponential Distribution: Memoryless, purely random.
- Normal Distribution: approximation of Binomial distribution when p is constant; fairly constant but with some random variability.
- Truncated normal distribution: similar to normal distribution but with restricted values.
- Weibull Distribution: suitable for modelling Time to Failure (TTF) where failures are due to the most serious of a large number of defects in a system of components. (Refer to Weibull Distribution)
- Lognormal Distribution: heavy-tailed distribution (the prob. that extreme values happen is not negligible) (Refer to Lognormal Distribution)

3.4 Poisson Process

3.4.1 Definition

Poisson process is one of the most important models in queueing theory.

Based on our previous discussion on the relationship between Poisson distribution and exponential distribution, we know that a Poisson process can be defined in at least two (equivalent) ways:

1. The number of arrivals $N(t)$ in a finite time interval of length t follows the Poisson (λt) distribution, i.e.,

$$P(N(t) = n) = \frac{(\lambda t)^n}{n!} e^{-\lambda t},$$

   if we assume the rate parameter of the Poisson process is λ.
2. The interarrival times, T, are independent and follow the exponential distribution of parameter λ, i.e.,

$$P(T > t) = e^{-\lambda t}.$$

[1] This principle states that if there is no reason to believe one outcome is more likely than another, then all possible outcomes should be assigned equal probabilities.

Actually, there is a third equivalent definition of Poisson process:

3. **Poisson process is a pure birth process**: In an infinitesimal time interval dt the probability that only **one** arrival occurs is λdt. This probability is independent of arrivals outside the interval.

We omit the mathematical proofs of the equivalence of the above three definitions of Poisson process.

3.4.2 Properties of the Poisson process

The Poisson process has several interesting and useful properties:

1. Splitting: If a Poisson process with intensity λ is randomly split into two subprocesses with probabilities p_1 and p_2, where $p_1 + p_2 = 1$, then the resulting processes are independent Poisson processes with intensities $p_1\lambda$ and $p_2\lambda$, respectively.
2. Pooling: The superposition of two independent Poisson processes with intensities λ_1 and λ_2, respectively, is a Poisson process with intensity $\lambda_1 + \lambda_2$.
3. Conditioning on the number of arrivals: Assume that in the interval $(0, t)$ the number of arrivals is $N(t) = n$. These n arrivals are independently and uniformly distributed in the interval.
   - This property can be used to generate a Poisson process in the interval $(0, t)$ as follows: (Step 1) Draw the number number of arrivals n from the Poisson (λt) distribution. (Step 2) for each arrival, draw its position in the interval $(0, t)$ from the uniform distribution, independently of the others.
4. Possion Arrivals See Time Average (PASTA): This property is also referred to as Random Observer Property (ROP), i.e., the probability of the state as seen by an outside random observer is the same as the probability of the state seen by an arriving customer. In other words, an arrival from a Poisson process observes the system as if it was arriving at a random moment in time. Therefore, the expected value of any parameter of the queue at the instant of a Poisson arrival is simply the **long-run average value** of that parameter. This parameter could be:
   - the expected number of customers in the queue
   - or the probability the server is busy
   - or the expected number waiting and not being served
   - or the expected waiting time for this arrival

Example 3.21. (An example of PASTA): Assume that:

- Cars are passing a point on a road according to a Poisson process.
- The mean interval between the cars is 10 minutes.
- A hitchhiker arrives at the roadside point at a random instant of time.

What is the mean waiting time until the next car? (5 minutes or 10 minutes).

3.4.3 Non-stationary Poisson Process (NSPP)

A Poisson process without the stationary increments, characterized by $\lambda(t)$, the arrival rate at time t, is called a non-stationary Poisson process (NSPP).

The expected number of arrivals by time t, $\Lambda(t)$ can be calculated by:

$$\Lambda(t) = \int_0^t \lambda(s)ds.$$

This function $\Lambda(t)$ is also called a compensator.

Relationship between stationary Poisson process with rate $\lambda = 1$ and NSPP with rate $\lambda(t)$:

Let $t_1, t_2, \ldots$ be the arrival times of stationary Poisson process $N_1(t)$ with rate $\lambda = 1$. Let $T_1, T_2, \ldots$ be the arrival times of NSPP $N(t)$ with rate $\lambda(t)$. We have the following fundamental relationship:

$$t_i = \Lambda(T_i)$$
$$T_i = \Lambda^{-1}(t_i)$$

Formally, the above relationship is described in the theorem.

Theorem 3.22. ***Time-Change Theorem*** *Let $N_1(t)$ denote the number of arrivals by time t in a stationary Poisson process with rate $\lambda = 1$. Define the* compensator*:*

$$\Lambda(t) = \int_0^t \lambda(s)\, ds.$$

Then the number of arrivals by time t, denoted by $N(t)$, in the nonstationary Poisson process with rate $\lambda(t)$ can be constructed from $N_1(t)$ as:

$$N(t) \stackrel{d}{=} N_1(\Lambda(t)).$$

Therefore, the number of arrivals in the interval $[a, b]$ satisfies:

$$N(b) - N(a) \sim Poisson(\Lambda(b) - \Lambda(a)).$$

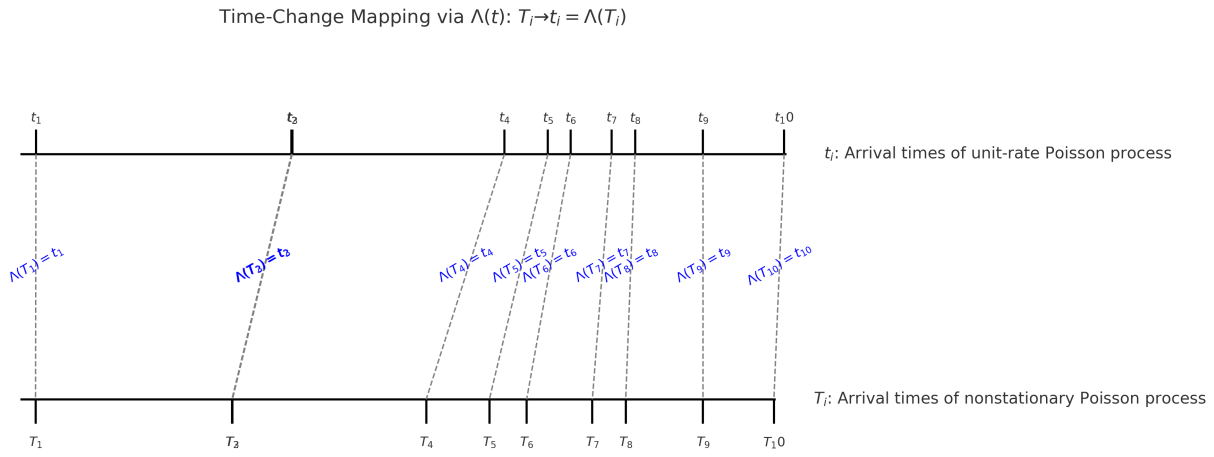


Fig. 3.1. Illustration of time-change theorem.

Question 3.23. Based on the above relationship, can you design an algorithm to generate an NSPP with rate $\lambda(t)$? (Hint: finding $\Lambda(T_i)$ and its inverse function. An algorithm could be found at Chapter 8.6.2 of the textbook (Law 2024)).

Example 3.24. Suppose arrivals to a rural Post Office follow a nonstationary Poisson process, with rates of 2 customers per hour from 8 am until 12 pm, and then 0.5 per hour until 4 pm. Assume that $t = 0$ corresponds to 8 am. What is the probability distribution of the number of arrivals between 11 am and 2 pm?

Ans:

The nonstationary Poisson process (NSPP) has the time-varying rate function:

$$\lambda(t) = \begin{cases} 2, & 0 \le t < 4 \quad \text{(from 8am to 12pm)} \\ 0.5, & 4 \le t \le 8 \quad \text{(from 12pm to 4pm)} \end{cases}$$

where time t is measured in **hours** from 8:00 am, so that:

$$\text{11:00 am} \to t = 3, \qquad \text{2:00 pm} \to t = 6.$$

For $t \in [0, 4]$:

$$\Lambda(t) = \int_0^t 2\, ds = 2t.$$

For $t \in [4, 8]$:

$$\Lambda(t) = \Lambda(4) + \int_4^t 0.5\, ds = 8 + 0.5(t-4) = 0.5t + 6.$$

Evaluate $\Lambda(3)$ and $\Lambda(6)$:

$$\Lambda(3) = 2 \cdot 3 = 6, \qquad \Lambda(6) = 0.5 \cdot 6 + 6 = 9.$$

So,

$$\Lambda(6) - \Lambda(3) = 9 - 6 = 3.$$

The number of arrivals between 11:00 am and 2:00 pm, i.e., over $t \in [3, 6]$, is distributed as:

$$N(6) - N(3) \sim \text{Poisson}(3).$$

Hence, the probability mass function is:

$$P(N = k) = e^{-3}\frac{3^k}{k!}, \quad \text{for } k = 0, 1, 2, \ldots$$

Remark 3.25. At this point, you should have developed a solid understanding of the practical meanings of commonly used discrete and continuous random variables. In simulation, a critical task is to generate a sequence of random variates that follow a specified probability distribution. This capability enables us to model stochastic behaviors accurately. Later, we will introduce several techniques for random variate generation, such as the inverse-transform and acceptance-rejection methods, and discuss their applications in simulation studies.

3.4.4 Algorithms for Generating Poisson Process and NSPP

Generating PP: Since the interarrival times $A_i = t_i - t_{i-1}$ are independent and identically distributed (IID) exponential random variables with rate parameter λ, we use the following algorithm to generate the stationary Poisson process with arrival rate $\lambda > 0$.
Repeat the following steps:

1. Generate $U \sim U(0, 1)$ independent of any previous random variates.
2. Return $t_i = t_{i-1} - \frac{1}{\lambda} \ln U$.

Step 2 is to generate exponential random variates. We will discuss this technique in a later chapter.

Generating NSPP: Based on the time-change theorem, we can easily design the following algorithm to generate NSPP with rate $\lambda(t)$.
Repeat the following steps:

1. Generate $U \sim U(0, 1)$ independent of any previous random variates.
2. Set $t'_i = t'_{i-1} - \ln U$.
3. Return $t_i = \Lambda^{-1}(t'_i)$.

3.5 Useful Statistical Models

3.5.1 Models in Queueing Systems

In a queueing system, we need to model

1. Arrival process: we can either use the interarrival time or the number of arrivals in unit time to model the arrival process.
2. Service time.

Distributions used for modelling time (i.e., continuous random variable) include:

- Exponential distribution: completely random.
- Normal distribution: fairly constant but with some deviation from the average. Note that we can only use positive values because time is positive.
- Truncated normal distribution: similar to normal distribution but with restricted values.
- Gamma and Weibull distribution: more general than exponential since they involve the location of the modes of the pdf and the shapes of tails. (Their definition and shape can be found in wikipedia).
- Uniform distribution: mathematically simple, but use it with caution (may not match real-world data).

Distributions used for modelling the number of arrivals (i.e., discrete random variable that takes non-negative integer values) per time unit include:

- Bernoulli distribution
- Binomial distribution
- Geometric and negative binomial distribution
- Poisson distribution: completely random

3.5.2 Models in Inventory and Supply-chain Systems

We need to model (at least) three random variables:

1. The number of units demanded per order or per time unit (discrete random variable)
2. The time between demands (continuous random variable)
3. the lead time, i.e., the amount of time between when a purchase order is placed to replenish products in stock (warehouse) and when the order is received. The typical distribution for lead time is the Gamma distribution.

3.5.3 Models for Reliability and Maintainability

For simulation used to evaluate the reliability of a system, the representative metrics include:

1. TTR (Time To Repair): the amount of time used for repairing
2. TBF (Time Between Failures): the amount of time between two failures
3. TTF (Time To Failure): the amount of time from installing/turning on the device to the first failure.

All are continuous random variables. Using TTF as an example:

- Exponential: failures are random (rarely the case for real-world products)
- Gamma: for standby redundancy where each component has an exponential TTF
- Weibull: failure is due to the most serious of a large number of defects in a system of components
- Normal: failures are due to wear

3.5.4 Models with Limited Data

For cases with limited data, useful distributions include: Uniform, triangular, beta, Bernoulli, binomial, and hyper-exponential.

Remark 3.26. In this section, we intentionally ignore the mathematical definitions of different random variables but rather focus on their practical meaning. You need to remember the following important facts:

1. If the arrival process is purely random, we use Poisson distribution with arrival rate λ to model the number of arrivals per time unit. This arrival process is also called the Poisson process. Equivalently, the interarrival time of a Poisson process with the rate λ can be modelled with an exponential random variable with (rate) parameter λ. That is, the mean of this exponential random variable is $\frac{1}{\lambda}$.
2. Gamma distribution, denoted by $\Gamma(n, \lambda)$, is the sum of n independent exponential random variables, whose (rate) parameters all equal λ. Due to this, $\Gamma(n, \lambda)$ could be considered the time we have to wait for n independent events to occur in a Poisson process. Clearly, the exponential distribution is a special case of Gamma distribution when $n = 1$.
3. Weibull distribution $W(\lambda, k)$ (where λ is the scale parameter and k is the shape parameter) describes the time we have to wait for one event to occur if that event becomes more or less likely with time.
   - A value of $k < 1$ indicates that the failure rate decreases over time.
   - A value of $k = 1$ indicates that the failure rate is constant over time.
   - A value of $k > 1$ indicates that the failure rate increases over time.

3.6 Supplementary Reading Materials

Textbook (Law 2024): Chapter 4.1, 4.2, 6.2, 6.12. Note that my lectures do not strictly follow the textbook.

4

Modelling Queueing Systems and Rough-Cut Analysis

4.1 The Purpose

The purpose of introducing queueing models is to understand the well-known theoretical results. Those theoretical results are very useful for a rough estimation of system performance before a simulation is performed. They also help simulation developers to identify potential bugs if a significant difference is observed between the theoretical estimation and the simulation results.

This part is **not** to develop queueing theory. So, we accept the mathematical results from queueing theory as given without providing a formal proof.

4.2 Characteristics of Queueing Systems

Conceptually, we need to model three components for a queueing system: arrival process, queueing behavior/queueing discipline, and service process.

Queue behavior refers to the actions of customers while in a queue waiting for service to begin, for example:

- Balk: leave when they see that the line is too long.
- Renege: leave after being in the line when its moving too slowly.
- Jockey: move from one line to a shorter line.

For simplicity, we normally assume that customers wait in the same queue until they receive service.

Queue discipline refers to the logical ordering of customers in a queue that determines which customer is chosen for service when a server becomes free, for example:

- First-in-first-out (FIFO)
- Last-in-first-out (LIFO)
- Service in random order (SIRO)
- Shortest processing time first (SPT)
- Service according to priority (PR) (e.g., type, class, priority).

4.3 Queueing Notations

4.3.1 Model Notation

A queueing model can be denoted by $A/B/c/N/K$:

- A represents the inter-arrival-time distribution,
- B represents the service-time distribution,
- c represents the number of parallel servers,
- N represents the system capacity,

- K represents the size of the calling population.

Common symbols for A and B include: M (exponential), D (constant or deterministic), E_k (for Erlang order k), PH (Phase-type), H (hyper-exponential), G (arbitrary or general), and GI (general independent).

We often omit N and K when they are infinite.

4.3.2 Performance Metrics

Table 4.1. Main performance measures of queueing systems

λ	Customer arrival rate (given)
μ	Service rate of one server (given)
P_n	Steady-state probability of having n customers in system
ρ	Long-run server utilization
L	Long-run time-average number of customers in system
L_q	Long-run time-average number of customers in queue
W	Long-run average time spent in system per customer
W_q	Long-run average time spent in queue per customer

Note that we use "long-term" and "long-run" interchangeably. People often omit the phrase "time-average."

4.4 Long-term Measure

4.4.1 Little's Law

- **Time-average number in System** and **Long-term time-average number in System**
- **Average system time** and **Long-term average system time**
- **Server utilization** and **Long-term server utilization** (also called **theoretical server utilization**)

Theorem 4.1. ***(Little's Law)*** $L = \lambda w$.

Remark 4.2. Note that:

- Little's law is for the long-term measure, i.e., $T \to \infty$
- Little's law is universal, i.e., it holds for **all queueing systems or subsystems**, irrelevant to the internal structure of the queueing system!

4.4.2 Server Utilization

Server utilization ($\hat{\rho}$) is defined as the proportion of time that a server is busy. When $T \to \infty$ and if the system exhibits long-run stability, $\hat{\rho} \to \rho$. We call ρ the long-term (or theoretical) server utilization.

For G/G/1, the theoretical server utilization can be **calculated** as $\rho = \frac{\lambda}{\mu}$.

For G/G/c, the theoretical server utilization can be **calculated** as $\rho = \frac{\lambda}{c\mu}$.

Question 4.3. For a G/G/1 system, what is the condition for the system to be stable? For a G/G/c system, what is the condition for the system to be stable?

4.5 Steady-State Behavior of Infinite-Population Markovian Models

The infinite-population Markovian model refers to arrivals that follow a Poisson process. Hence, we here care about M/M(or G)/c queueing systems.

Table 4.2. Steady-state parameters of the M/M/1 queue

Symbol	Formula / Expression
λ	Given
μ	Given
ρ	$\frac{\lambda}{\mu}$
P_0	$1-\rho$
P_n	$(1-\rho)\rho^n$
L	$\frac{\rho}{1-\rho}$
L_q	$\frac{\rho^2}{1-\rho}$
W	$\frac{1}{\mu-\lambda} = \frac{1}{\mu(1-\rho)}$
W_q	$\frac{\lambda}{\mu(\mu-\lambda)} = \frac{\rho}{\mu(1-\rho)}$

Table 4.3. Steady-state parameters of the M/M/c queue.

Symbol	Formula / Expression
λ	Given
μ	Given
c	Given (number of servers)
ρ	$\frac{\lambda}{c\mu}$
P_0	$\left[\sum_{n=0}^{c-1} \frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n + \frac{1}{c!}\left(\frac{\lambda}{\mu}\right)^c \cdot \frac{c\mu}{c\mu-\lambda}\right]^{-1}$
P_{wait}	$\frac{1}{c!}\left(\frac{\lambda}{\mu}\right)^c \cdot \frac{c\mu}{c\mu-\lambda} \cdot P_0$
L_q	$\frac{\rho}{1-\rho} P_{\text{wait}}$
L	$L_q + \frac{\lambda}{\mu}$
W_q	$\frac{L_q}{\lambda}$
W	$W_q + \frac{1}{\mu} = \frac{L}{\lambda}$

Note that P_{wait} refers to the probability that a customer waits in the queue, or in other words, the probability that the long-term queue length is no smaller than c.

Table 4.4. Steady-state parameters of the M/G/1 queue

Symbol	Formula / Expression
λ	Given
μ	Given
ρ	$\frac{\lambda}{\mu}$
L_q	$\frac{\rho^2(1+\sigma^2\mu^2)}{2(1-\rho)}$
L	$L_q + \rho$
W_q	$\frac{L_q}{\lambda}$
W	$\frac{L}{\lambda}$

Note that σ^2 denotes variance of service times. Note that the mean of service times is $\frac{1}{\mu}$.

Table 4.5. Steady-state parameters of the M/G/∞ queue.

Symbol	Formula / Expression
λ	Given
μ	Given
ρ	$\frac{\lambda}{\mu}$
P_n	$\frac{\rho^n}{n!}e^{-\rho} \quad n = 0, 1, 2, ...$
L	$\frac{\lambda}{\mu}$
W	$\frac{1}{\mu}$
L_q	0 (no queueing; infinite servers)
W_q	0

Note: there are at least three situations in which it is appropriate to treat the number of servers as infinite: (1) when each customer is its own server, i.e., in a self-service system; (2) when service capacity far exceeds service demands (so-called ample-server system; and (3) when we want to know how many servers are required so that customers will rarely be delayed.

Table 4.6. Steady-state parameters of the M/M/c/ N/ ∞ queue.

Symbol	Formula / Expression
$\lambda,\ \mu,\ c,\ N$	Given (arrival rate, service rate, #servers, system capacity)
a	$a = \dfrac{\lambda}{\mu}$
ρ	$\rho = \dfrac{\lambda}{c\mu}$ (offered load per server)
P_0	$\left[1 + \sum_{n=1}^{c} \dfrac{a^n}{n!} + \dfrac{a^c}{c!} \sum_{n=c+1}^{N} \left(\dfrac{a}{c}\right)^{n-c}\right]^{-1}$
P_n	$P_n = \begin{cases} P_0 \dfrac{a^n}{n!}, & 0 \le n \le c-1, \\ P_0 \dfrac{a^n}{c!\, c^{n-c}}, & c \le n \le N, \end{cases}$
P_{block}	P_N (blocking probability)
λ_{eff}	$\lambda(1 - P_N)$ (throughput)
P_{wait}	$\sum_{n=c}^{N-1} P_n$ (delay probability)
L_q	$\sum_{n=c}^{N} (n-c)\, P_n$
L	$\sum_{n=0}^{N} n\, P_n$
W_q	$\dfrac{L_q}{\lambda_{\text{eff}}}$
W	$\dfrac{L}{\lambda_{\text{eff}}}$

Note: The difference between arrival rate λ and effective arrival rate λ_{eff}: arrival rate refers to the number of arrivals per time unit, and effective arrival rate refers to the number of arrivals, who arrive and enter the system, per time unit. We care about the difference only when there is a limit to the number of customers in the waiting line or system (here "system" includes both the waiting line and servers).

4.6 Rough-cut Analysis for Network of Queues

Note that in general, a network of queues does not render tractable theoretical analysis as in single queue systems. There is a special research area, called network calculus, that focuses on the analysis of a network of queues. The mathematical tool is quite different from traditional queueing theory. This is beyond the scope of this course. Nevertheless, we can perform rough-cut modelling to derive useful results with the knowledge of a single queue system. The results from the rough-cut analysis can indicate whether there are significant bugs in our simulation implementation.

Example: Assume an open network of M/M/1 queues models a web service pipeline:

$$\textbf{Web} \to \textbf{App} \to \begin{cases} \text{DB} & \text{with prob } p = 0.3 \\ \text{Cache} & \text{with prob } 1 - p = 0.7 \end{cases} \to \text{depart}$$

Assume that the arrival rate $\lambda = 8$ requests/sec. Assume that the ervice rates: $\mu_{\text{Web}} = 12$, $\mu_{\text{App}} = 10$, $\mu_{\text{DB}} = 5$, $\mu_{\text{Cache}} = 20$ (all in requests/sec).

Traffic Rates

$$\begin{aligned} \lambda_{\text{Web}} &= \lambda = 8 \\ \lambda_{\text{App}} &= \lambda = 8 \\ \lambda_{\text{DB}} &= p\lambda = 0.3 \times 8 = 2.4 \\ \lambda_{\text{Cache}} &= (1-p)\lambda = 0.7 \times 8 = 5.6 \end{aligned}$$

Station-wise Parameters (M/M/1 Rough-Cut)

Table 4.7. Per-station M/M/1 estimates

Station	λ	μ	ρ	L	L_q	W (s)
Web	8.0	12.0	0.667	2.000	1.333	0.250
App	8.0	10.0	0.800	4.000	3.200	0.500
DB	2.4	5.0	0.480	0.923	0.443	0.385
Cache	5.6	20.0	0.280	0.389	0.109	0.069

End-to-End Estimates

Expected response time:

$$E[T] \approx W_{\text{Web}} + W_{\text{App}} + pW_{\text{DB}} + (1-p)W_{\text{Cache}} = 0.25 + 0.5 + 0.3 \times 0.385 + 0.7 \times 0.069 \approx \boxed{0.914 \text{ s}}$$

Expected number in system:

$$L = \lambda E[T] = 8 \times 0.914 = \boxed{7.31}$$

What-If Analysis

(A) Traffic surge near saturation: $\lambda = 9.5$/s (keep $p = 0.3$).

$$\begin{aligned} &W_{\text{Web}} = \tfrac{1}{12-9.5} = 0.400, \quad W_{\text{App}} = \tfrac{1}{10-9.5} = 2.000, \\ &W_{\text{DB}} = \tfrac{1}{5-0.3\times 9.5} = \tfrac{1}{2.15} \approx 0.4651, \quad W_{\text{Cache}} = \tfrac{1}{20-0.7\times 9.5} = \tfrac{1}{13.35} \approx 0.0750. \\ &\Rightarrow \; E[T] \approx 0.400 + 2.000 + 0.3 \cdot 0.4651 + 0.7 \cdot 0.0750 \approx \boxed{2.592 \text{ s}}, \\ &\quad\; L \approx 9.5 \times 2.592 \approx \boxed{24.62}. \end{aligned}$$

(B) Scale-up + better caching: $\mu_{App} = 16/s$, $p = 0.1$ *(keep* $\lambda = 8/s$*).*

$$
\begin{aligned}
&W_{\text{App}} = \tfrac{1}{16-8} = 0.125, \quad W_{\text{DB}} = \tfrac{1}{5-0.1\times 8} = \tfrac{1}{4.2} \approx 0.2381, \\
&W_{\text{Cache}} = \tfrac{1}{20-0.9\times 8} = \tfrac{1}{12.8} = 0.078125 \quad \text{(Web unchanged at 0.25).} \\
&\Rightarrow\ E[T] \approx 0.25 + 0.125 + 0.1 \cdot 0.2381 + 0.9 \cdot 0.078125 \approx \boxed{0.469 \text{ s}}, \\
&\quad L \approx 8 \times 0.469 \approx \boxed{3.75}.
\end{aligned}
$$

Table 4.8. End-to-end metrics under what-if scenarios.

Scenario	λ	p	μ_{App}	$E[T]$ (s)	L
Baseline	8.0	0.3	10	0.914	7.31
(A) Traffic surge	9.5	0.3	10	2.592	24.62
(B) Scale-up + caching	8.0	0.1	16	0.469	3.75

4.7 Supplementary Reading Materials

Textbook (Law 2024): Appendix 1B. Note that my lectures do not strictly follow the textbook.
Prof. Stoica (2017): Queuing Theory - Little's Law, Sections 1-3.

5

Random Number Generation

Our next goal is to generate random variates that follow a given distribution. As the first step, we focus on generating random numbers that are uniformly distributed in $(0, 1)$, denoted by $U(0, 1)$. This will be our basis for generating other types of random variates in the next chapter.

Learning points:

1. Two properties of random numbers: uniformity and independence
2. Two well-known methods for generating pseudo-random numbers: Linear Congruential Method (LCM) and Combined Linear Congruential Generators (CLCG).
3. Two types of tests for pseudo-random numbers: frequency test for uniformity and runs test for independence. Understand the Chi-square test in the former category. Understand the runs-up test with normal approximation in the latter category.

5.1 Two Properties

What do we mean by "*random*" numbers?

1. Uniformity: If the interval $(0, 1)$ is divided into n classes or subintervals of equal length, the expected number of observations in each interval is $\frac{N}{n}$, where N is the total number of observations.
2. Independence: The probability of observing a value in a particular interval is independent of the previous values drawn. In other words, based on the numbers that we have generated so far, we cannot infer any property of the next number.

When we design algorithms to generate random numbers, we want the algorithm to generate numbers that closely approximate the ideal statistical properties of uniformity and independence. Besides that, we want the algorithm to have the following desirable features:

1. The algorithm should be fast: Individual computations are inexpensive, but a simulation may require many millions of random numbers.
2. Portable to different computers: ideally to different programming languages. This ensures the program produces the same results.
3. Have sufficiently long cycle: The cycle length, or period, represents the length of the random number sequence before previous numbers begin to repeat in an earlier order.
4. Replicable: Given the starting point, it should be possible to generate the same set of random numbers, completely independent of the system that is being simulated.

5.2 Two algorithms

5.2.1 Linear Congruential Method (LCM)

A single LCG generates a sequence of integers $\{X_n\}$ as

$$X_{n+1} = (aX_n + c) \bmod m, \tag{5.1}$$

where a is the multiplier, c is the increment, m is the modulus, and X_0 is the seed. The corresponding uniform random number in $(0, 1)$ is

$$U_n = \frac{X_n}{m}. \tag{5.2}$$

Note that when $c \neq 0$, LCM is also called the mixed congruential method; otherwise, it is also called the multiplicative congruential method.

Question: What are the potential issues of LCM?

5.2.2 Combined Linear Congruential Generator (CLCG)

Idea: Combine two or more multiplicative congruential generators to produce a generator with better statistical properties.

A CLCG uses k independent LCGs, each defined as

$$X_{i,n+1} = (a_i X_{i,n}) \bmod m_i, \quad i = 1, 2, \ldots, k, \tag{5.3}$$

where each component generator has its own parameters (a_i, m_i) and seed $X_{i,0}$. The combined output is defined by

$$U_n = \left(\sum_{i=1}^{k} \frac{X_{i,n}}{m_i} \right) \bmod 1. \tag{5.4}$$

We slightly abuse the notation mod in the above equation by using it to indicate that we require a fraction number in the range $(0, 1)$. To achieve this, L'Ecuyer suggested the following form:

$$X_n = \left(\sum_{i=1}^{k} (-1)^{i-1} X_{i,n} \right) \bmod (m_1 - 1). \tag{5.5}$$

$$U_n = \begin{cases} \dfrac{X_n}{m_1}, & \text{if } X_n > 0, \\ \dfrac{m_1 - 1}{m_1}, & \text{if } X_n = 0. \end{cases} \tag{5.6}$$

For the case of two generators ($k = 2$), the form is:

$$X_{1,n} = (a_1 X_{1,n-1}) \bmod m_1, \tag{5.7}$$

$$X_{2,n} = (a_2 X_{2,n-1}) \bmod m_2, \tag{5.8}$$

$$X_n = (X_{1,n} - X_{2,n}) \bmod (m_1 - 1), \tag{5.9}$$

$$U_n = \begin{cases} \dfrac{X_n}{m_1}, & \text{if } X_n > 0, \\ \dfrac{m_1 - 1}{m_1}, & \text{if } X_n = 0. \end{cases} \tag{5.10}$$

Period: If the parameters (a_i, m_i) are properly chosen, the overall period is approximately

$$\text{Period} \approx \text{lcm}(m_1 - 1, m_2 - 1, \ldots, m_k - 1), \tag{5.11}$$

where lcm stands for Least Common Multiple. This period can be very large (e.g., $> 2^{60}$).

Example 5.1. **L'Ecuyer Two-Component CLCG (1988)**

$$X_{1,n} = (40014X_{1,n-1}) \bmod 2147483563, \tag{5.12}$$
$$X_{2,n} = (40692X_{2,n-1}) \bmod 2147483399, \tag{5.13}$$
$$X_n = (X_{1,n} - X_{2,n}) \bmod (2147483563 - 1), \tag{5.14}$$
$$U_n = \begin{cases} \dfrac{X_n}{2147483563}, & X_n > 0, \\ \dfrac{2147483562}{2147483563}, & X_n = 0. \end{cases} \tag{5.15}$$

This generator achieves excellent statistical performance with a period of approximately 2^{61}.

5.3 Empirical Tests

The question that we want to answer in this section is: How do we know the generated random numbers are good enough?

5.3.1 Test for Uniformity

Brief Introduction to Chi-square test
In hypothesis testing, we begin with a statement called the **null hypothesis**, denoted by H_0. The null hypothesis represents the default assumption that there is no significant difference between the observed data and what is expected under a specified theoretical model. The alternative hypothesis, denoted by H_1, represents the opposite claim, suggesting that a significant difference exists.

The **Chi-square test** is a statistical method used to determine whether the observed frequencies in a set of categorical data differ significantly from the expected frequencies under H_0. It measures the degree of deviation between observation and expectation. The Chi-square test is widely used in applications such as testing the uniformity of random numbers and evaluating the goodness of fit for theoretical probability distributions.

Let the data be divided into k categories with observed frequencies O_i and expected frequencies E_i $(i = 1, 2, \ldots, k)$. The Chi-square statistic is computed as

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}.$$

Under the null hypothesis, the statistic χ^2 follows a Chi-square distribution with $k - 1$ degrees of freedom. For a chosen significance level α, the null hypothesis H_0 is **rejected** if the computed χ^2 value exceeds the critical value from the Chi-square distribution table. Otherwise, the null hypothesis is **not rejected.**

Remark 5.2. The **level of significance**, denoted by α, is the probability of rejecting the null hypothesis H_0 when it is actually true. It represents the maximum risk the researcher is willing to take of making a *Type I error*—that is, concluding that a significant difference exists when in fact it does not.

Common choices for α are 0.05, 0.01, or 0.10, depending on how strict the test needs to be. A smaller value of α indicates a more stringent criterion for rejecting H_0, thereby reducing the chance of a Type I error but increasing the chance of a *Type II error* (failing to reject a false H_0).

Test Uniformity with Chi-square test
Next, we show how to use the Chi-square test to evaluate whether a set of random numbers generated in the interval $(0, 1)$ follows a *uniform distribution*. The idea is to divide the interval $(0, 1)$ into k equal subintervals (or bins) and count how many of the n generated random numbers fall into each subinterval. If the numbers are truly uniform, each subinterval should contain approximately n/k observations.

Formally, let O_i be the observed frequency in the i-th subinterval, and $E_i = n/k$ be the expected frequency. The **Chi-square statistic** is then computed as

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}.$$

This statistic follows a Chi-square distribution with $k-1$ degrees of freedom under the null hypothesis that the numbers are uniformly distributed. The hypothesis is accepted if χ^2 lies within the acceptance region defined by the chosen significance level α; otherwise, the uniformity assumption is rejected.

Example 5.3. Suppose we generate $n = 100$ random numbers uniformly in the interval $(0, 1)$. To test whether these numbers are indeed uniformly distributed, we divide the interval $(0, 1)$ into $k = 10$ equal subintervals:

$$(0, 0.1],\ (0.1, 0.2],\ (0.2, 0.3],\ \ldots,\ (0.9, 1.0].$$

Step 1: Hypotheses.

- Null hypothesis H_0: The numbers are uniformly distributed in $(0, 1)$.
- Alternative hypothesis H_1: The numbers are not uniformly distributed in $(0, 1)$.

Step 2: Observed and Expected Frequencies.
Assume the following observed frequencies (O_i) for the ten subintervals:

Interval	$(0, 0.1]$	$(0.1, 0.2]$	$(0.2, 0.3]$	$(0.3, 0.4]$	$(0.4, 0.5]$	$(0.5, 0.6]$	$(0.6, 0.7]$	$(0.7, 0.8]$	$(0.8, 0.9]$	$(0.9, 1.0]$
O_i	8	12	11	9	10	7	8	12	13	10

Since the total number of samples is $n = 100$, the expected frequency for each interval is

$$E_i = \frac{n}{k} = \frac{100}{10} = 10.$$

Step 3: Compute the Chi-Square Statistic.

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i} = \frac{(8-10)^2}{10} + \frac{(12-10)^2}{10} + \cdots + \frac{(10-10)^2}{10}.$$

Calculating each term gives

$$\chi^2 = 0.4 + 0.4 + 0.1 + 0.1 + 0.0 + 0.9 + 0.4 + 0.4 + 0.9 + 0.0 = 3.6.$$

Step 4: Decision.
The degrees of freedom are $k - 1 = 9$. From the Chi-square distribution table, the critical value for $\alpha = 0.05$ and 9 degrees of freedom is

$$\chi^2_{0.05,9} = 16.92.$$

Since the computed value $3.6 < 16.92$, H_0 **is not rejected.**
Conclusion: There is no statistical evidence to reject the assumption that the generated numbers are uniformly distributed in $(0, 1)$ at the 5% significance level.

5.3.2 Test for Independence

Runs-up Test for Independence of Random Numbers
We can use the **runs-up test** to examine the *independence* of random numbers generated in the interval $(0, 1)$. Independence means that the occurrence of one number does not influence the occurrence of the next; in other words, there should be no detectable pattern or trend among the numbers. In our context, the test examines whether the numbers show an increasing ("upward") or decreasing ("downward") tendency.

A *run-up* is defined as a consecutive sequence of numbers in which each number is larger than the previous one. For example, in the sequence

$$0.15,\ 0.42,\ 0.37,\ 0.56,\ 0.71,\ 0.48,$$

the runs-up are:

$$(0.15,\ 0.42),\ (0.37,\ 0.56,\ 0.71),\ (0.48),$$

so there are three runs-up in total.

Under the null hypothesis H_0 that the numbers are independent and uniformly distributed in $(0, 1)$, the number of runs-up follows a distribution with a known mean and variance that depend on the sample size n. For large n, the distribution of the number of runs-up R can be approximated by the normal distribution:

$$Z = \frac{R - E[R]}{\sqrt{\text{Var}[R]}} \sim N(0, 1),$$

where $E[R]$ and $\text{Var}[R]$ are the expected value and variance of the number of runs-up, respectively. In our context, $E[R]$ and $\text{Var}[R]$ are calculated, respectively, by[1]:

$$E[R] = \frac{2n - 1}{3}$$
$$\text{Var}[R] = \frac{16n - 29}{90}$$

If the absolute value of Z exceeds the critical value $Z_{1-\alpha/2}$ from the standard normal table (e.g., $z_{1-0.025} = 1.96$ for $\alpha = 0.05$), the hypothesis of independence is rejected. The standard normal distribution table could be found at the end of this lecture note.

Example 5.4. Consider the following sequence of 10 random numbers uniformly distributed in $(0, 1)$:

$$0.31,\ 0.58,\ 0.75,\ 0.62,\ 0.45,\ 0.81,\ 0.90,\ 0.39,\ 0.52,\ 0.66.$$

Step 1: Identify Runs-Up. The runs-up in the sequence are:

$$(0.31, 0.58, 0.75),\ (0.62),\ (0.45, 0.81, 0.90),\ (0.39, 0.52, 0.66),$$

so the total number of runs-up is $R = 4$.

Step 2: Compute Expected Number of Runs-Up and Variance.

$$E[R] = \frac{2n - 1}{3} = \frac{19}{3} \approx 6.33, \qquad \text{Var}[R] = \frac{16n - 29}{90} = \frac{131}{90} \approx 1.46.$$

Step 3: Compute Test Statistic.

$$Z = \frac{R - E[R]}{\sqrt{\text{Var}[R]}} = \frac{4 - 6.33}{\sqrt{1.46}} = \frac{-2.33}{1.21} \approx -1.93.$$

Step 4: Decision. For a two-tailed test at the 5% significance level, the critical values are ± 1.96. Since $|Z| = 1.93 < 1.96$, we **fail to reject** H_0.

Conclusion. There is no statistical evidence to reject the assumption that the generated numbers are independent at the 5% significance level.

5.3.3 Final remarks

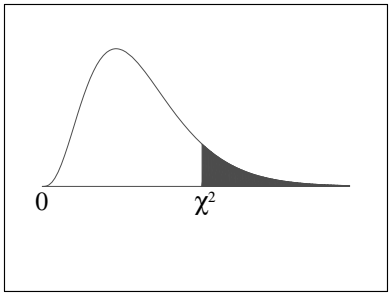
- Even generators that have been used for years, some of which are still in use, are found to be inadequate.
- This chapter provides only the basics.
- There are many other statistical tests for uniformity (e.g., the Kolmogorov-Smirnov test) and independence (e.g., the autocorrelation test), which are out of the focus of this course.
- Even if generated numbers pass all the tests, some underlying patterns might have gone undetected. Therefore, it is more accurate to say that the null hypothesis is "**not rejected**" rather than "**accepted**". This phrase "the null hypothesis is not rejected" acknowledges the possibility that the null hypothesis may still be false, even though we were unable to reject it based on the available data.

5.4 Supplementary Reading Materials

Textbook (Law 2024): Chapter 7.1 (background), 7.2 (basic algorithms), 7.4.1 (empirical test). Note that my lectures do not strictly follow the textbook.

[1] Refer to https://real-statistics.com/non-parametric-tests/one-sample-runs-test/runs-up-down-test/

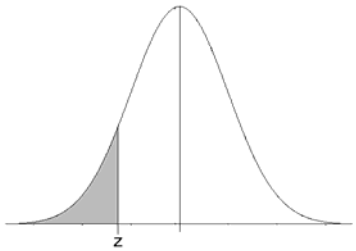
Chi-Square Distribution Table



The shaded area is equal to α for $\chi^2 = \chi^2_\alpha$.

df	$\chi^2_{.995}$	$\chi^2_{.990}$	$\chi^2_{.975}$	$\chi^2_{.950}$	$\chi^2_{.900}$	$\chi^2_{.100}$	$\chi^2_{.050}$	$\chi^2_{.025}$	$\chi^2_{.010}$	$\chi^2_{.005}$
1	0.000	0.000	0.001	0.004	0.016	2.706	3.841	5.024	6.635	7.879
2	0.010	0.020	0.051	0.103	0.211	4.605	5.991	7.378	9.210	10.597
3	0.072	0.115	0.216	0.352	0.584	6.251	7.815	9.348	11.345	12.838
4	0.207	0.297	0.484	0.711	1.064	7.779	9.488	11.143	13.277	14.860
5	0.412	0.554	0.831	1.145	1.610	9.236	11.070	12.833	15.086	16.750
6	0.676	0.872	1.237	1.635	2.204	10.645	12.592	14.449	16.812	18.548
7	0.989	1.239	1.690	2.167	2.833	12.017	14.067	16.013	18.475	20.278
8	1.344	1.646	2.180	2.733	3.490	13.362	15.507	17.535	20.090	21.955
9	1.735	2.088	2.700	3.325	4.168	14.684	16.919	19.023	21.666	23.589
10	2.156	2.558	3.247	3.940	4.865	15.987	18.307	20.483	23.209	25.188
11	2.603	3.053	3.816	4.575	5.578	17.275	19.675	21.920	24.725	26.757
12	3.074	3.571	4.404	5.226	6.304	18.549	21.026	23.337	26.217	28.300
13	3.565	4.107	5.009	5.892	7.042	19.812	22.362	24.736	27.688	29.819
14	4.075	4.660	5.629	6.571	7.790	21.064	23.685	26.119	29.141	31.319
15	4.601	5.229	6.262	7.261	8.547	22.307	24.996	27.488	30.578	32.801
16	5.142	5.812	6.908	7.962	9.312	23.542	26.296	28.845	32.000	34.267
17	5.697	6.408	7.564	8.672	10.085	24.769	27.587	30.191	33.409	35.718
18	6.265	7.015	8.231	9.390	10.865	25.989	28.869	31.526	34.805	37.156
19	6.844	7.633	8.907	10.117	11.651	27.204	30.144	32.852	36.191	38.582
20	7.434	8.260	9.591	10.851	12.443	28.412	31.410	34.170	37.566	39.997
21	8.034	8.897	10.283	11.591	13.240	29.615	32.671	35.479	38.932	41.401
22	8.643	9.542	10.982	12.338	14.041	30.813	33.924	36.781	40.289	42.796
23	9.260	10.196	11.689	13.091	14.848	32.007	35.172	38.076	41.638	44.181
24	9.886	10.856	12.401	13.848	15.659	33.196	36.415	39.364	42.980	45.559
25	10.520	11.524	13.120	14.611	16.473	34.382	37.652	40.646	44.314	46.928
26	11.160	12.198	13.844	15.379	17.292	35.563	38.885	41.923	45.642	48.290
27	11.808	12.879	14.573	16.151	18.114	36.741	40.113	43.195	46.963	49.645
28	12.461	13.565	15.308	16.928	18.939	37.916	41.337	44.461	48.278	50.993
29	13.121	14.256	16.047	17.708	19.768	39.087	42.557	45.722	49.588	52.336
30	13.787	14.953	16.791	18.493	20.599	40.256	43.773	46.979	50.892	53.672
40	20.707	22.164	24.433	26.509	29.051	51.805	55.758	59.342	63.691	66.766
50	27.991	29.707	32.357	34.764	37.689	63.167	67.505	71.420	76.154	79.490
60	35.534	37.485	40.482	43.188	46.459	74.397	79.082	83.298	88.379	91.952
70	43.275	45.442	48.758	51.739	55.329	85.527	90.531	95.023	100.425	104.215
80	51.172	53.540	57.153	60.391	64.278	96.578	101.879	106.629	112.329	116.321
90	59.196	61.754	65.647	69.126	73.291	107.565	113.145	118.136	124.116	128.299
100	67.328	70.065	74.222	77.929	82.358	118.498	124.342	129.561	135.807	140.169

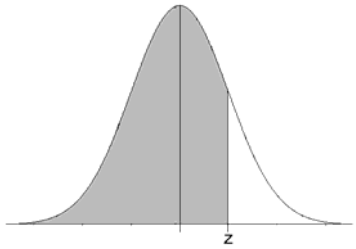
Standard Normal Cumulative Probability Table



Cumulative probabilities for NEGATIVE z-values are shown in the following table:

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
-3.4	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0002
-3.3	0.0005	0.0005	0.0005	0.0004	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
-3.2	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006	0.0006	0.0005	0.0005	0.0005
-3.1	0.0010	0.0009	0.0009	0.0009	0.0008	0.0008	0.0008	0.0008	0.0007	0.0007
-3.0	0.0013	0.0013	0.0013	0.0012	0.0012	0.0011	0.0011	0.0011	0.0010	0.0010
-2.9	0.0019	0.0018	0.0018	0.0017	0.0016	0.0016	0.0015	0.0015	0.0014	0.0014
-2.8	0.0026	0.0025	0.0024	0.0023	0.0023	0.0022	0.0021	0.0021	0.0020	0.0019
-2.7	0.0035	0.0034	0.0033	0.0032	0.0031	0.0030	0.0029	0.0028	0.0027	0.0026
-2.6	0.0047	0.0045	0.0044	0.0043	0.0041	0.0040	0.0039	0.0038	0.0037	0.0036
-2.5	0.0062	0.0060	0.0059	0.0057	0.0055	0.0054	0.0052	0.0051	0.0049	0.0048
-2.4	0.0082	0.0080	0.0078	0.0075	0.0073	0.0071	0.0069	0.0068	0.0066	0.0064
-2.3	0.0107	0.0104	0.0102	0.0099	0.0096	0.0094	0.0091	0.0089	0.0087	0.0084
-2.2	0.0139	0.0136	0.0132	0.0129	0.0125	0.0122	0.0119	0.0116	0.0113	0.0110
-2.1	0.0179	0.0174	0.0170	0.0166	0.0162	0.0158	0.0154	0.0150	0.0146	0.0143
-2.0	0.0228	0.0222	0.0217	0.0212	0.0207	0.0202	0.0197	0.0192	0.0188	0.0183
-1.9	0.0287	0.0281	0.0274	0.0268	0.0262	0.0256	0.0250	0.0244	0.0239	0.0233
-1.8	0.0359	0.0351	0.0344	0.0336	0.0329	0.0322	0.0314	0.0307	0.0301	0.0294
-1.7	0.0446	0.0436	0.0427	0.0418	0.0409	0.0401	0.0392	0.0384	0.0375	0.0367
-1.6	0.0548	0.0537	0.0526	0.0516	0.0505	0.0495	0.0485	0.0475	0.0465	0.0455
-1.5	0.0668	0.0655	0.0643	0.0630	0.0618	0.0606	0.0594	0.0582	0.0571	0.0559
-1.4	0.0808	0.0793	0.0778	0.0764	0.0749	0.0735	0.0721	0.0708	0.0694	0.0681
-1.3	0.0968	0.0951	0.0934	0.0918	0.0901	0.0885	0.0869	0.0853	0.0838	0.0823
-1.2	0.1151	0.1131	0.1112	0.1093	0.1075	0.1056	0.1038	0.1020	0.1003	0.0985
-1.1	0.1357	0.1335	0.1314	0.1292	0.1271	0.1251	0.1230	0.1210	0.1190	0.1170
-1.0	0.1587	0.1562	0.1539	0.1515	0.1492	0.1469	0.1446	0.1423	0.1401	0.1379
-0.9	0.1841	0.1814	0.1788	0.1762	0.1736	0.1711	0.1685	0.1660	0.1635	0.1611
-0.8	0.2119	0.2090	0.2061	0.2033	0.2005	0.1977	0.1949	0.1922	0.1894	0.1867
-0.7	0.2420	0.2389	0.2358	0.2327	0.2296	0.2266	0.2236	0.2206	0.2177	0.2148
-0.6	0.2743	0.2709	0.2676	0.2643	0.2611	0.2578	0.2546	0.2514	0.2483	0.2451
-0.5	0.3085	0.3050	0.3015	0.2981	0.2946	0.2912	0.2877	0.2843	0.2810	0.2776
-0.4	0.3446	0.3409	0.3372	0.3336	0.3300	0.3264	0.3228	0.3192	0.3156	0.3121
-0.3	0.3821	0.3783	0.3745	0.3707	0.3669	0.3632	0.3594	0.3557	0.3520	0.3483
-0.2	0.4207	0.4168	0.4129	0.4090	0.4052	0.4013	0.3974	0.3936	0.3897	0.3859
-0.1	0.4602	0.4562	0.4522	0.4483	0.4443	0.4404	0.4364	0.4325	0.4286	0.4247
0.0	0.5000	0.4960	0.4920	0.4880	0.4840	0.4801	0.4761	0.4721	0.4681	0.4641

Standard Normal Cumulative Probability Table



Cumulative probabilities for POSITIVE z-values are shown in the following table:

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998

6

Random Variate Generation

6.1 The Goal

Our goal is to generate random variates that follow a given distribution. This goal is achieved with tools that we introduced in the previous chapter, which generate random numbers uniformly distributed in $[0, 1]$.

6.2 Inverse-transform Technique

The foundation of the inverse-transform technique is based on one simple fact:

> Assume that F_X is the CDF of a random variable X. Define a new random variable $Y = F_X(X)$. Then Y is uniformly distributed in $[0, 1]$.

To see why this is true:

$$P(Y \le y) \;=\; P\big(F_X(X) \le y\big) \;=\; P\big(X \le F_X^{-1}(y)\big) \;=\; F_X\big(F_X^{-1}(y)\big) \;=\; y,$$

that is, $P(Y \le y) = y$, indicating that Y is uniformly distributed. Since probability values must fall in $[0, 1]$, Y is uniformly distributed in $[0, 1]$.

Based on the above fact, once we can find F_X^{-1}, generating random variates X is straightforward. The key idea is illustrated in Fig. 6.1.

Example 6.1. Use the inverse-transform technique to generate random variates following an exponential distribution with rate parameter λ. The probability distribution function (PDF) of an exponential random variable with rate parameter $\lambda > 0$ is given by

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x \ge 0, \\ 0, & x < 0. \end{cases}$$

Step 1: Derive the Cumulative Distribution Function (CDF)

The cumulative distribution function is obtained by integrating the PDF:

$$F(x) = \int_0^x \lambda e^{-\lambda t}\, dt = 1 - e^{-\lambda x}, \quad x \ge 0.$$

Step 2: Apply the Inverse-Transform Principle

Let U be a uniform random variable in $(0, 1)$. According to the inverse-transform method,

$$F(X) = U.$$

Substituting the expression for $F(X)$ gives

$$1 - e^{-\lambda X} = U.$$

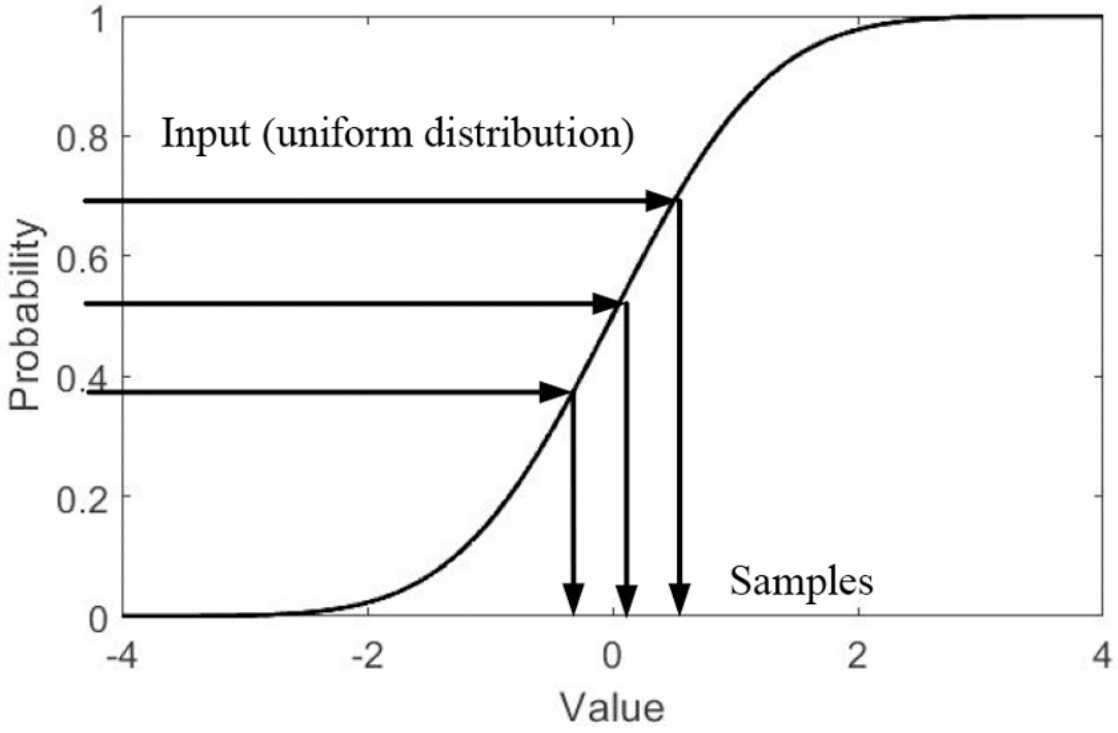


Fig. 6.1. The key idea of inverse-transform sampling.

Step 3: Solve for X

Rearranging terms,

$$e^{-\lambda X} = 1 - U,$$
$$-\lambda X = \ln(1-U),$$
$$X = -\frac{1}{\lambda}\ln(1-U).$$

Step 4: Simplify the Expression

Since U and $1-U$ are identically distributed as Uniform$(0,1)$, we can simplify:

$$\boxed{X = -\frac{1}{\lambda}\ln(U)}.$$

Algorithm Summary

1. Generate a uniform random number $U \sim$ Uniform$(0,1)$.
2. Compute
$$X = -\frac{1}{\lambda}\ln(U).$$
3. Repeat 1 and 2.

This is exactly the same method that we have introduced in Section 3.4.4, when we generate Poisson arrivals and NSPP.

Problem 6.2. Use the inverse-transform technique to generate random variates following:

- Uniformly distribution in $[a, b]$

- Triangular Distribution
- Empirical continuous distribution
- Discrete distribution:
  - Empirical discrete
  - Discrete uniform
  - Geometric

Remark 6.3. Since the CDF of a discrete random variable is not continuous and has discontinuities (jumps) at the points where the random variable takes specific values, we need to be careful when generating discrete random variates. In general, we first generate $U \sim \text{Uniform}(0,1)$; if $F(x_{i-1}) < U \leq F(x_i)$, set $X = x_i$.

Example 6.4. Assume that the value of a discrete random variable is 0, 1, or 2, and that the pmf of this random variable is given by:

$$P(0) = 0.5$$
$$P(1) = 0.3$$
$$P(2) = 0.2$$

How to generate discrete random variates following the above distribution?

Step 1. Derive the CDF

The CDF, $F(x) = P(X \leq x)$, is given by:

$$F(x) = \begin{cases} 0 & x < 0 \\ 0.5 & 0 \leq x < 1 \\ 0.8 & 1 \leq x < 2 \\ 1 & 2 \leq x \end{cases}$$

which is illustrated in Fig. 6.2.

Step 2. Inverse-Transform Method

The generation method is as follows:

$$X = \begin{cases} 0 & U \leq 0.5 \\ 1 & 0.5 < U \leq 0.8 \\ 2 & 0.8 < U \leq 1 \end{cases}$$

where $U \sim \text{Uniform}(0,1)$.

Example 6.5. **Generating geometric distributed random variates**: Let X be a discrete random variable representing the number of trials until the first success, with parameter $p \in (0,1]$. The probability mass function (PMF) is

$$\Pr(X = k) = p(1-p)^{k-1}, \qquad k = 1, 2, 3, \ldots$$

Step 1. Derive the CDF

The cumulative distribution function (CDF) is

$$F(k) = \Pr(X \leq k) = \sum_{i=1}^{k} p(1-p)^{i-1}.$$

Using the formula for a finite geometric series,

$$F(k) = p\frac{1-(1-p)^k}{1-(1-p)} = 1-(1-p)^k, \qquad k = 1, 2, \ldots$$

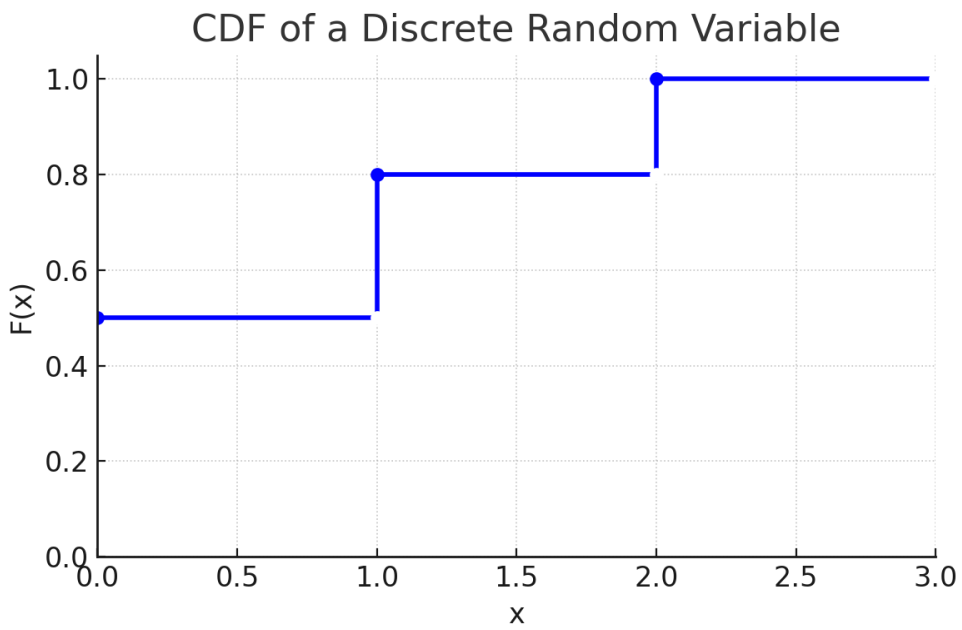


Fig. 6.2. The cdf in Example 6.4

Step 2. Inverse-Transform Method

By the inverse-transform principle on a discrete random variable, $X = k$ if and only if

$$F(k-1) < U \leq F(k),$$

where we define $F(0) = 0$ in the context of geometric distribution.

2.a. Solve for k

Since $F(k) = 1 - (1-p)^k$,

$$U \leq F(k) \quad \Longleftrightarrow \quad U \leq 1 - (1-p)^k \quad \Longleftrightarrow \quad (1-p)^k \leq 1 - U.$$

Taking the natural logarithm of both sides (note that $0 < 1 - p < 1$ implies $\ln(1-p) < 0$), we obtain

$$k \ln(1-p) \leq \ln(1-U).$$

Dividing by $\ln(1-p)$ reverses the inequality direction:

$$k \geq \frac{\ln(1-U)}{\ln(1-p)}.$$

Since k must be an integer, the smallest integer satisfying the above inequality is

$$\boxed{X = \left\lceil \frac{\ln(1-U)}{\ln(1-p)} \right\rceil}.$$

2.b. Final Formula

Hence, the inverse-transform method for generating geometric random variates with support $\{1, 2, 3, \ldots\}$ is

$$X = \left\lceil \frac{\ln(1-U)}{\ln(1-p)} \right\rceil, \qquad U \sim \text{Uniform}(0, 1).$$

2.c. An Implementation Note

For numerical stability, it is recommended to use the function python `log1p` to compute logarithms whenever the generated U is close to zero:

$$\ln(1-U) = \texttt{log1p}(-U).$$

Also, a similar trick is applied if the parameter p is too small.

This implementation detail ensures the correct discrete CDF behavior for the geometric distribution representing the number of trials until the first success.

6.3 Acceptance-Rejection Technique

This method is particularly useful when the inverse CDF does not exist in closed form. The main idea is to generate random numbers and only accept those that meet a condition. This condition is set based on a *majorizing function*, which is sometimes called an *envelope* function.

Before we formally define *majorizing function*, let's use the following example to get some intuition behind this function.

Example 6.6. Generate random variates, X, uniformly distributed in $(0, \frac{1}{4}]$.

- **Step 1:** Generate $U \sim \text{Uniform}(0, 1)$.
- **Step 2.a:** If $U \leq \frac{1}{4}$, accept $X = U$, and go to **Step 1**.
- **Step 2.b** otherwise, reject U, and go to **Step 1**.

Q: Can you prove why the above method is correct? Hint: For $0 < a < b \leq \frac{1}{4}$, calculate $P(a < U \leq b | 0 < U \leq \frac{1}{4})$.

The above example shows that instead of sampling on $X \sim \text{Uniform}(0, \frac{1}{4}]$ directly, we sample on an easier one $U \sim \text{Uniform}(0, 1)$ (easier because we already have methods to generate U. In this case, the easier U is an envelope of X. Keeping this example in mind, we next formally define the majorizing function and explain how to use this function in the acceptance-rejection algorithm.

1. Definition of Majorizing Function

Suppose we wish to generate random samples from a probability density function (PDF) $f(x)$, but direct sampling from $f(x)$ is difficult. We choose another PDF $g(x)$, called the *proposal distribution*, from which random variates can be generated easily.

We then find a constant $c > 0$ such that

$$f(x) \leq c\, g(x), \qquad \forall x.$$

The function

$$M(x) = c\, g(x)$$

is called a *majorizing function* (or *envelope function*), because it "majorizes" (i.e., upper-bounds) the target density $f(x)$.

2. Acceptance–Rejection Algorithm

Given $f(x)$, $g(x)$, and c, the algorithm proceeds as follows:

1. Generate a random variate X from the proposal distribution $g(x)$.
2. Generate $U \sim \text{Uniform}(0, 1)$.
3. Accept X as a sample from $f(x)$ if
$$U \leq \frac{f(X)}{c g(X)}.$$
Otherwise, reject X and repeat.

The ratio $f(X)/(cg(X)) \leq 1$ represents the probability of acceptance for a given sample X.

Q: could you see how the algorithm in Example 6.6 matches the above acceptance-rejection algorithm? Hint: what is $f(x)$ in Example 6.6? Note that in Example 6.6, the proposal distribution $g(x)$ is Uniform$(0, 1)$ and $c = 4$. Draw a figure similar to Figure 6.3 explain the situation.

3. Intuition Behind the Majorizing Function

The acceptance-rejection method can be interpreted geometrically. Consider the two curves $y = f(x)$ and $y = cg(x)$ on the same coordinate axes.

- The function $f(x)$ represents the shape of the target distribution.
- The function $cg(x)$ is an "inflated" version of $g(x)$ that lies entirely above $f(x)$.

We conceptually generate random points (X, Y) uniformly distributed under the curve $y = cg(x)$. A point (X, Y) is accepted if it lies below the target curve $y = f(x)$. The x-coordinate of each accepted point follows the desired distribution $f(x)$.

Thus, the majorizing function $M(x) = cg(x)$ defines a region that fully covers the target PDF, ensuring that the accepted points correspond to the correct distribution.

4. Efficiency of the Method

The acceptance probability is

$$P(\text{accept}) = \frac{1}{c}.$$

A smaller value of c (i.e., a tighter majorizing function) yields a higher acceptance rate and better efficiency. Conversely, if c is large, the envelope is loose, leading to more rejections and reduced efficiency.

5. Summary

Symbol	Meaning	Description
$f(x)$	Target PDF	Distribution we want to sample from
$g(x)$	Proposal PDF	Easy-to-sample distribution
$M(x) = cg(x)$	Majorizing function	Upper bound on $f(x)$
c	Scaling constant	Ensures $f(x) \leq cg(x)$
Acceptance condition	$U \leq \frac{f(X)}{cg(X)}$	Criterion for acceptance
Efficiency	$1/c$	Average proportion of accepted samples

6. Geometric Illustration (Conceptual)

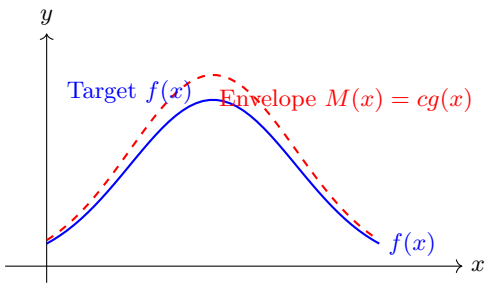


Fig. 6.3. Geometric illustration of the acceptance-rejection method.

In the figure above, $M(x)$ lies entirely above $f(x)$. Random points are uniformly generated below $M(x)$, and only those falling below $f(x)$ are accepted. The x-coordinates of accepted points follow the desired PDF $f(x)$.

7. Summary

- The majorizing function $M(x) = cg(x)$ provides a *ceiling* over $f(x)$.
- It allows uniform sampling under a known shape (via $g(x)$).
- The acceptance condition $U \leq f(X)/(cg(X))$ ensures the correct distributional weighting.
- The smaller the constant c, the tighter the fit and the higher the sampling efficiency.

Example 6.7. **Generating Beta(4,3) Random Variates**

The probability density function (PDF) of a Beta$(4, 3)$ random variable X on the unit interval $[0, 1]$ is

$$f(x) = 60\, x^3(1-x)^2, \qquad 0 \leq x \leq 1,$$

where the Beta function $B(4, 3)$ is given by

$$B(4,3) = \frac{\Gamma(4)\Gamma(3)}{\Gamma(7)} = \frac{3!\,2!}{6!} = \frac{1}{60}.$$

Clearly, it is hard to use the inverse transform method because $f(x)$ is a sixth-degree polynomial.

Choice of the Proposal Distribution

We choose a simple proposal distribution $g(x)$ that is easy to sample from. A convenient choice is the uniform distribution on $[0, 1]$:

$$g(x) = 1, \qquad 0 \leq x \leq 1.$$

The goal is to find a constant $c > 0$ such that

$$f(x) \leq c\, g(x), \qquad \forall x \in [0, 1].$$

Thus, $M(x) = cg(x)$ serves as the *majorizing function* (or envelope). In order to find the Scaling Constant c, we must determine

$$c = \max_{x\in[0,1]} f(x).$$

The maximum of $f(x)$ occurs at the mode of the Beta distribution, which for $\alpha > 1$ and $\beta > 1$ is (by solving $\frac{df}{dx} = 0$)

$$x^* = \frac{\alpha - 1}{\alpha + \beta - 2}.$$

For Beta$(4, 3)$,

$$x^* = \frac{4-1}{4+3-2} = \frac{3}{5} = 0.6.$$

Hence,

$$f_{\max} = f(0.6) = 60 \times (0.6)^3 \times (1-0.6)^2 = 60 \times 0.216 \times 0.16 = 2.0736.$$

Therefore, the smallest valid constant is

$$c = f_{\max} = 2.0736.$$

Apply Acceptance–Rejection Algorithm

We now have:

$$f(x) = 60x^3(1-x)^2, \qquad g(x) = 1, \qquad c = 2.0736.$$

Algorithm Steps:

1. Generate $X \sim$ Uniform$(0, 1)$.
2. Generate $U \sim$ Uniform$(0, 1)$ independently.
3. Accept X as a Beta(4,3) random variate if

$$U \leq \frac{f(X)}{cg(X)} = \frac{60X^3(1-X)^2}{2.0736}.$$

Otherwise, reject X and repeat.

6.4 Special Properties

They are variate-generation techniques that are based on features of a particular family of probability distributions, rather than general-purpose techniques like the inverse-transform or acceptance-rejection techniques introduced above.

Example 6.8. **Box-Muller Transform for generating standard Normal variates**:

1. Generate two independent random uniform variates R_1 and R_2 on the interval $(0, 1)$.
2. Compute the following:

$$Z_1 = sqrt(-2 * ln(R_1)) * cos(2 * \pi * R_2) \tag{6.1}$$
$$Z_2 = sqrt(-2 * ln(R_1)) * sin(2 * \pi * R_2) \tag{6.2}$$

3. return Z_1 and Z_2 as independent standard normal variates.

Note:

- To generate normal variates with a specified mean μ and standard deviation σ, simply multiply Z_1 (and Z_2) by σ and add μ, since $X = \mu + \sigma Z$ follows $N(\mu, \sigma^2)$.
- To generate lognormal variates $Y = e^X$, where X follows $N(\mu, \sigma^2)$. Why? By definition, a random variable Y follows a lognormal distribution if its logarithm $X = ln(Y)$ follows a normal distribution.

Example 6.9. **Generating Poisson Random Variates**
Theoretical Basis: A Poisson process with rate parameter λ can be viewed as a counting process $\{N(t), t \geq 0\}$ where

$$\Pr(N(t) = k) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}, \qquad k = 0, 1, 2, \ldots$$

and the interarrival times between events are independent and identically distributed exponential random variables with rate λ:

$$T_i \sim \text{Exp}(\lambda), \qquad f_T(t) = \lambda e^{-\lambda t}, \quad t > 0.$$

Let

$$S_k = \sum_{i=1}^{k} T_i$$

denote the time of the kth event. Then, for a fixed time window t_0 (often $t_0 = 1$ for simplicity),

$$N(t_0) = \max\{k : S_k \leq t_0\}$$

follows a Poisson distribution with mean λt_0:

$$N(t_0) \sim \text{Poisson}(\lambda t_0).$$

Algorithm using Exponential Waiting Time
To generate one realization of $X \sim \text{Poisson}(\lambda)$:

1. Initialize $S = 0$ and $k = 0$.
2. Repeat:
   a) Generate $U \sim \text{Uniform}(0, 1)$.
   b) Compute an exponential interarrival time:

$$T = -\frac{1}{\lambda} \ln(U).$$

   c) Update the cumulative time:

$$S = S + T, \quad k = k + 1.$$

3. Continue until $S > 1$.

4. Return $X = k - 1$.

This method simulates the occurrence of random arrivals in a Poisson process during the interval $(0, 1]$ and counts how many arrivals occur before the time limit.
Intuitive Explanation: Each exponential random variable T_i represents the waiting time until the next event. By summing these waiting times, we trace the occurrence times of events in the Poisson process. Once the total waiting time S_k exceeds the observation period (e.g., $t_0 = 1$), we know that exactly $k - 1$ events occurred within that interval.
Numerical Example: Let $\lambda = 3$, and let the observation interval be $t_0 = 1$.

1. Generate successive exponential interarrival times:

$$\begin{aligned} T_1 &= -\frac{1}{3}\ln(0.8) = 0.0743, \\ T_2 &= -\frac{1}{3}\ln(0.6) = 0.1703, \\ T_3 &= -\frac{1}{3}\ln(0.4) = 0.3054, \\ T_4 &= -\frac{1}{3}\ln(0.2) = 0.5365, \end{aligned}$$

2. Compute cumulative arrival times:

$$\begin{aligned} S_1 &= 0.0743, \\ S_2 &= 0.2446, \\ S_3 &= 0.5500, \\ S_4 &= 1.0865, \end{aligned}$$

3. Since $S_4 = 1.0865 > 1$, the process stops at the 4th arrival.
4. Hence, the number of arrivals within $(0, 1]$ is:

$$X = 4 - 1 = 3.$$

If we need more samples, we repeat the above steps.

Remark 6.10. The above method directly uses the definition of a Poisson process: a process with exponentially distributed interarrival times. It is exact and provides physical insight into the relationship between the exponential and Poisson distributions. For small or moderate λ, this is both simple and efficient. For large λ, however, more efficient methods (e.g., acceptance-rejection method) are usually preferred.

6.5 Supplementary Reading Materials

Textbook (Law 2024): Chapter 8.1 (background), 8.2.1, 8.2.4, 8.2.6 (General approaches).

Note that my lectures do not strictly follow the textbook.

7

Input Modelling

Goal: Any simulation needs data as input. As such, this chapter aims to develop a useful model for input data. After such a model is built, we can then use the techniques introduced in the previous chapter to generate more synthetic data for a simulation study.

7.1 Univariate Models

Learning points:

1. The four steps for developing an input model: (1) **Step 1**: data collection, (2) **Step 2**: identify distribution, (3) **Step 3**: estimate the parameters of the distribution, and (4) **Step 4**: goodness-of-fit test.
2. Methods used in Step 2:
   a) Histogram
   b) Physical meaning of distribution and the application context
   c) Q-Q plot.
3. Methods used in Step 3: well-known estimators for distributions (Table 7.2).
4. Methods used in Step 4: Chi-Square Test and Kolmogorov-Smirnov Test.
   - Understand p-value and the concept of "best fit"

7.1.1 Step 1: Data Collection

When building input models for a simulation study, it is critical to collect data that accurately represents the real-world system or process being simulated. Below are some general guidelines.

Understand the Simulation Purpose and Input Needs

- Clarify which system performance measures the simulation aims to estimate (e.g., waiting time, queue length, utilization).
- Identify the inputs that drive system behavior and impact the system performance, such as interarrival times, service times, failure times, or travel times.
- Determine the required level of data granularity (e.g., individual transaction times versus hourly averages).

Ensure Realism and Representativeness

- Collect data under normal and steady operating conditions to reflect routine system performance.
- Capture variability by including data from different time periods or conditions (e.g., peak and off-peak hours, weekdays and weekends).
- Ensure the dataset covers all relevant operating modes of the system.

Establish a Rigorous Measurement Plan

- Use accurate and reliable measurement tools (e.g., automatic logs, sensors, or digital timestamps) to minimize human error.

- Synchronize all data sources to a common time reference when multiple devices or systems are used.
- Choose an observation period long enough to capture typical cycles or seasonal variations.

Data Cleansing

- Clean the raw data by removing errors, duplicates, and missing entries.
- Garbage-in-garbage-out.

Verify and Cross-Validate

- Consult domain experts to ensure the collected data accurately represents system behaviour.
- Cross-validate results with alternative data sources or time periods for consistency.

Documentation and Archiving

- Maintain a detailed log of data collection procedures, instruments, sampling frequency, and assumptions.
- Store both raw and processed data for future verification and reproducibility.

7.1.2 Step 2: Identify Distribution

Method 1: Histogram

A **histogram** is a graphical representation of the frequency distribution of data. The range of the data is divided into equal intervals (called *bins*), and the height of each bar represents the number (or proportion) of observations falling within that interval. Unlike bar charts for categorical data, histogram bars touch each other to indicate continuity.

The number of bins depends on the number of observations and the dispersion of the data. If the interval is too wide, the histogram will be coarse or blocky and its shape and other details will not show well. If the intervals are too narrow, the histograms will be ragged and will not smooth the data.

Suggestion: The number of intervals set to the square root of the sample size *usually* works well in practice.

Example 7.1. Suppose we record the service times (in minutes) for 20 customers:

$$1.2,\ 1.8,\ 2.0,\ 2.3,\ 2.5,\ 3.2,\ 3.0,\ 3.1,\ 3.3,\ 3.5,\ 4.6,\ 4.8,\ 4.0,\ 4.1,\ 4.3,\ 4.5,\ 4.7,\ 5.4,\ 5.1,\ 5.2$$

We group the data into 1-minute intervals so that the number of intervals roughly equals the square root of the sample size:

Interval	**Frequency**
$[1, 2)$	2
$[2, 3)$	3
$[3, 4)$	5
$[4, 5)$	7
$[5, 6)$	3

Method 2: Physical Meaning of Distribution

Using the physical basis of the distribution as a guide, we can "propose" a distribution that matches the real-world application context.

To recall:

- Binomial: # of successes in n Bernoulli trials.
- Poisson: # of independent events that occur in a fixed amount of time or space.
- Normal: distribution of a process that is the sum of a number of component processes.
- Weibull: time to failure for components.
- Exponential: time between independent events, or a process time that is memoryless.
- Uniform: models complete uncertainty. All outcomes are equally likely (Principle of indifference).
- Triangular: a process for which only the minimum, most likely, and maximum values are known. Improvement over uniform.

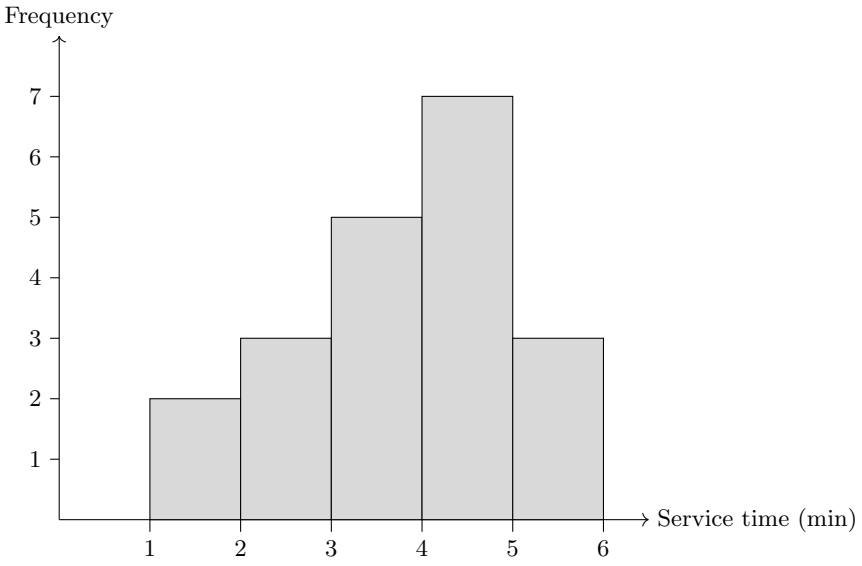


Fig. 7.1. An example histogram.

Remark 7.2. Do not ignore the physical characteristics of the process. Ask yourself "is the process naturally discrete or continuous valued? Is it bounded or is there no natural bound?" Keep in mind that there is no "true" distribution for any stochastic input process. Our goal is to obtain a good approximation input model that yields useful results from the simulation experiment.

Method 3: Q-Q Plot

Q-Q plot is a reliable way for identifying the type of distribution.

- Definition of q-quantile: If X is random variable with cdf F, then the q-quantile of X is the value r such that
$$F(r) = P(X \leq r) = q.$$
- Figure 7.2 helps you understand why Q-Q plot can be used to identify the distribution type. **If the estimated distribution type is proper, the Q-Q plot should be close to a straight line.**

Table 7.1. Interpretation of Q–Q Plot Patterns

Case	**Expected Q–Q Plot Shape**	**Interpretation**
Data perfectly follow theoretical distribution	Straight line with slope = 1, intercept = 0	Perfect match; same mean and variance
Same family but different parameters	Straight line with different slope/intercept	Same shape, but different mean or variance
Data more heavy-tailed than theoretical model	Concave upward curve	Empirical distribution has heavier tails
Data more light-tailed than theoretical model	Concave downward curve	Empirical distribution has lighter tails
Skewness mismatch	S-shaped curve	Different skewness between data and theoretical model

Q: Why a Q–Q plot is (approximately) a straight line if the estimated distribution type is proper?

Ans: Let $X_{(i)}$ be the ith order statistic of a sample of size n, and let $p_i = \frac{i-1/2}{n}$. If the data truly follow the suggested distribution F with quantile function F^{-1}, then the ith sample quantile matches the theoretical quantile at the same cumulative probability:

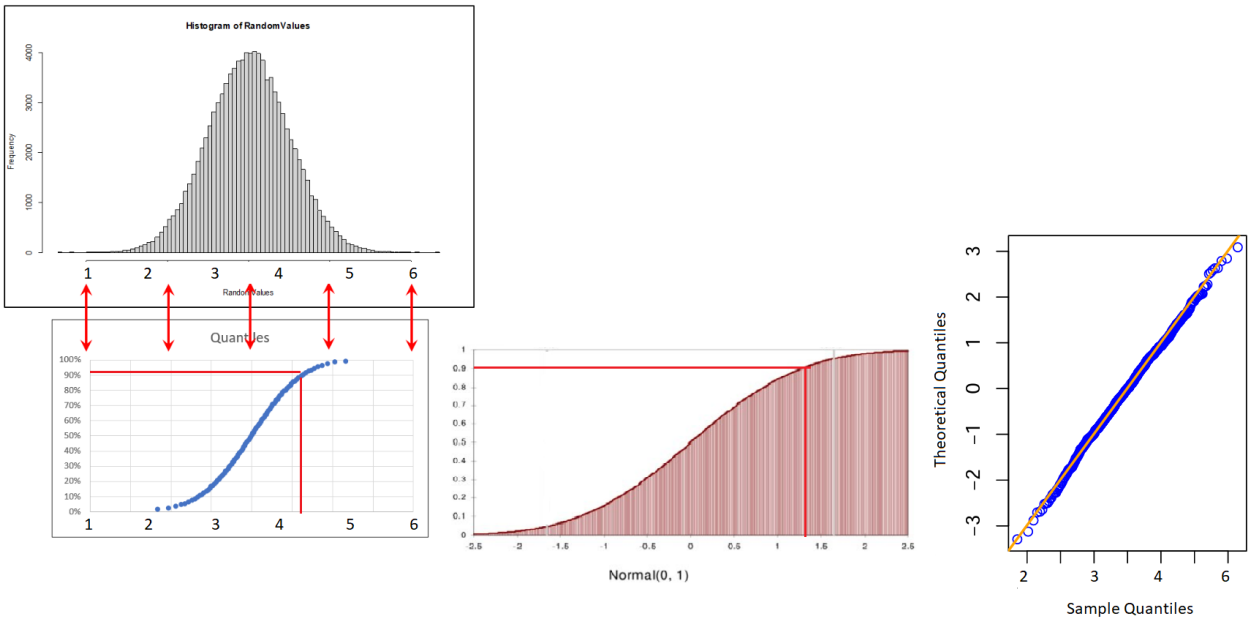


Fig. 7.2. Illustration of Q-Q plot.

$$X_{(i)} \approx F^{-1}(p_i).$$

Hence points $\left(F^{-1}(p_i),\, X_{(i)}\right)$ lie near the line $y = x$. For a location–scale family, $X = a + bZ$ with $Z \sim F$, we have

$$\text{Quantile}(X; p) = a + b\, F^{-1}(p),$$

so the Q–Q points lie near the straight line $y = a + bx$ (not necessarily $y = x$), as shown in Figure 7.2.

Q: Why use $(i-\frac{1}{2})/n$ instead of i/n as the cumulative probability when computing the empirical quantiles?

Ans: Let the ordered sample be

$$x_{(1)} \le x_{(2)} \le \cdots \le x_{(n)}.$$

The empirical cumulative distribution function (ECDF) is defined as

$$F_n(x) = \frac{1}{n} \sum_{j=1}^{n} I(x_{(j)} \le x),$$

which increases by $1/n$ at each observed value. However, when we compare the empirical distribution to a continuous theoretical CDF (e.g., in Q-Q plots), we must decide what cumulative probability each ordered value $x_{(i)}$ represents.

The issue with using i/n:

If we assign $x_{(i)}$ the cumulative probability i/n, we implicitly assume that $x_{(i)}$ corresponds to the *upper edge* of the i-th quantile segment. For example,

$$x_{(1)} \leftrightarrow p = \frac{1}{n}, \qquad x_{(n)} \leftrightarrow p = 1.$$

This choice leads to several issues:

- The maximum sample value corresponds to $p = 1$, but in a continuous distribution the probability of exactly hitting the maximum value is zero.

- The minimum sample value corresponds to $p = 1/n$, implying no probability mass below the minimum, which underestimates the lower tail.

As a result, the ECDF based on i/n tends to be *biased high* in the lower tail and *biased low* in the upper tail when compared with the true continuous CDF.

To fix the above problem, using $(i - \frac{1}{2})/n$ shifts each probability by half of one step, effectively centering each quantile probability on the midpoint of its step interval rather than at its edge. This adjustment has several advantages:

- It symmetrically positions the empirical probabilities, reducing bias at both extremes.
- It provides a better approximation to the expected value of the order statistics, $E[F(X_{(i)})]$.
- It avoids extreme probabilities 0 and 1, which correspond to infinite quantiles for some continuous distributions (e.g., normal).

More Intuitive interpretation:

Each observation represents a slice of probability mass of width $1/n$. Using i/n places the probability entirely below $x_{(i)}$, whereas $(i - \frac{1}{2})/n$ places the probability at the midpoint of that slice, aligning it more naturally with the underlying continuous distribution. Therefore, $(i - \frac{1}{2})/n$ gives a more balanced and **visually smoother** representation of the empirical quantiles when comparing with the theoretical quantiles. In short, **the Q-Q plot uses $(i - \frac{1}{2})/n$ to place each data point in the middle of its cumulative probability interval, giving a visually smoother and less biased representation of the datas quantiles.**

7.1.3 Step 3: Parameter Estimation

Estimators for common distributions used in simulation are listed in Table 7.2. Note that

- The denominator in the **sample variance** calculation is $n - 1$ rather than n, where n is the number of samples. If we have collected data from every member of the population that we are interested in, we can get an exact value for population variance, which uses N in the denominator, where N is the size of the population.
- The sample variance is an unbiased estimator of the **population variance**. The mathematical proof is omitted.
- When the sample size is big, people might use n in the denominator when estimating the variance. Strictly speaking, it is incorrect because this calculation will underestimate the population variance. Nevertheless, people normally do not care too much about the minor difference. When the sample size is small, we must use the sample variance to obtain an unbiased estimation of the population variance.

7.1.4 Step 4: Goodness-of-fit Test

We introduce two statistical test methods for the goodness-of-fit test: the Chi-square test and the Kolmogorov-Smirnov (K-S) test.

Method 1: Chi-square Test

We introduced the Chi-square test in Chapter 5.3.1 for testing the uniformity of random numbers, in which case we used equal probabilities for all bins. Actually, the equal probabilities requirement in the Chi-square test can be relaxed, as long as the expected frequency in each bin is high enough.

Principles for Choosing the Number of Intervals

Let n be the total number of observations and k be the number of intervals (or classes). The following guidelines are commonly used:

1. The expected frequency in each interval should not be too small. A common rule of thumb is:

$$E_i = np_i \geq 5 \quad \text{for all } i = 1, 2, \ldots, k,$$

where p_i is the theoretical probability that a random observation falls into the i-th interval.

Table 7.2. Estimators for Common Distributions Used in Simulation

Distribution	**Parameters**	**PDF / PMF**	**Estimator(s)**
Uniform (a, b)	a, b	$f(x) = \frac{1}{b-a},\ a \le x \le b$	$\hat{a} = \min(X_i),\ \hat{b} = \max(X_i)$
Poisson (λ)	λ	$P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!}$	$\hat{\lambda} = \bar{X} = \frac{1}{n}\sum_1^n X_i$
Exponential (λ)	λ	$f(x) = \lambda e^{-\lambda x},\ x \ge 0$	$\hat{\lambda} = \frac{1}{\bar{X}}$
Normal (μ, σ^2)	μ, σ^2	$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\hat{\mu} = \bar{X},\ \hat{\sigma}^2 = s^2 = \frac{1}{n-1}\sum_1^n (X_i - \bar{X})^2$
Lognormal (μ, σ^2)	μ, σ^2	$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}}$	$\hat{\mu} = \overline{\ln X} = \frac{1}{n}\sum_1^n \ln X_i,\ \hat{\sigma}^2 = \frac{1}{n-1}\sum (\ln X_i - \overline{\ln X})^2$
Geometric (p)	p	$P(X = k) = (1-p)^{k-1}p$	$\hat{p} = \frac{1}{\bar{X}}$
Gamma (α, β)	α, β	$f(x) = \frac{x^{\alpha-1}e^{-x/\beta}}{\Gamma(\alpha)\beta^\alpha},\ x > 0$	$\hat{\alpha} = \frac{\bar{X}^2}{s^2},\ \hat{\beta} = \frac{s^2}{\bar{X}}$, s^2 is sample variance

2. If some intervals have $E_i < 5$, merge adjacent intervals until this condition is satisfied.
3. A recommendation[1] for the number of intervals for continuous data can be found in Table 7.3.

Table 7.3. Recommendations for Chi-square Test of Continuous Data

Criterion	Recommendation
Expected frequency per interval	$E_i \ge 5$
Merging rule	Merge adjacent intervals if $E_i < 5$
Sample size smaller than 20	do not use Chi-square test
Sample size n between 20 and 50	Number of intervals typically $5 \le k \le 10$
Sample size n between 50 and 100	Number of intervals typically $10 \le k \le 20$
Sample size $n > 100$	Number of intervals typically $\sqrt{n} \le k \le \frac{n}{5}$

Example 7.3. **Chi-square Goodness-of-Fit Test for a Poisson Model**
Setup:

Assume that we observe $n = 120$ counts (e.g., arrivals per minute). Suppose the observed frequencies are

$$O_0 = 12,\ O_1 = 28,\ O_2 = 30,\ O_3 = 24,\ O_4 = 15,\ O_{5+} = 11,$$

where $O_k (k = 0, 1, 2, 3, 4, 5+)$ denote the frequency for k arrivals, respectively.

We want to test

$$H_0 :\ X \sim \text{Poisson}(\lambda) \quad \text{vs.} \quad H_1 :\ \text{not Poisson.}$$

From the raw data, the sample mean is $\bar{X} = 2.4$, so the MLE is $\hat{\lambda} = \bar{X} = 2.4$.

Binning and expected counts:

To keep expected frequencies ≥ 5, we use the bins

$$\{0\},\ \{1\},\ \{2\},\ \{3\},\ \{4\},\ \{5+\}.$$

For $k = 0, 1, 2, 3, 4$, $p_k = \Pr_{\hat{\lambda}}(X = k) = e^{-\hat{\lambda}}\hat{\lambda}^k/k!$, and for the tail $p_{5+} = 1 - \sum_{k=0}^4 p_k$. With $\hat{\lambda} = 2.4$, the probabilities are

$$p_0 = 0.0907, \quad p_1 = 0.2177, \quad p_2 = 0.2613, \quad p_3 = 0.2090,$$
$$p_4 = 0.1254, \quad p_{5+} = 0.0959,$$

[1] Data from the book "Discrete-Event System Simulation" (fifth edition) by Jerry Banks et al.

hence expected counts $E_i = n\,p_i$ are

$$E_0 = 10.886,\ E_1 = 26.127,\ E_2 = 31.352,\ E_3 = 25.082,\ E_4 = 15.049,\ E_{5+} = 11.504.$$

Test statistic:

The Chi-square statistic is

$$\chi^2 = \sum_i \frac{(O_i - E_i)^2}{E_i} = \sum_{i\in\{0,1,2,3,4,5+\}} \frac{(O_i - E_i)^2}{E_i} = 0.3755.$$

Degrees of freedom:

With $k = 6$ bins and $m = 1$ estimated parameter (λ),

$$\mathrm{df} = k - 1 - m = 6 - 1 - 1 = 4.$$

Test result:

At significance level $\alpha = 0.05$, the critical value is $\chi^2_{0.95,4} = 9.49$. Since $0.3755 < 9.49$, we **fail to reject** H_0**.**

Method 2: Kolmogorov-Smirnov (K-S) test

The main idea of the K-S test is to compare the empirical CDF with the hypothesized distribution's CDF. If the two distributions differ significantly, we reject the null hypothesis; otherwise, we do not. The geometric meaning of the K-S test is shown in Fig. 7.3.

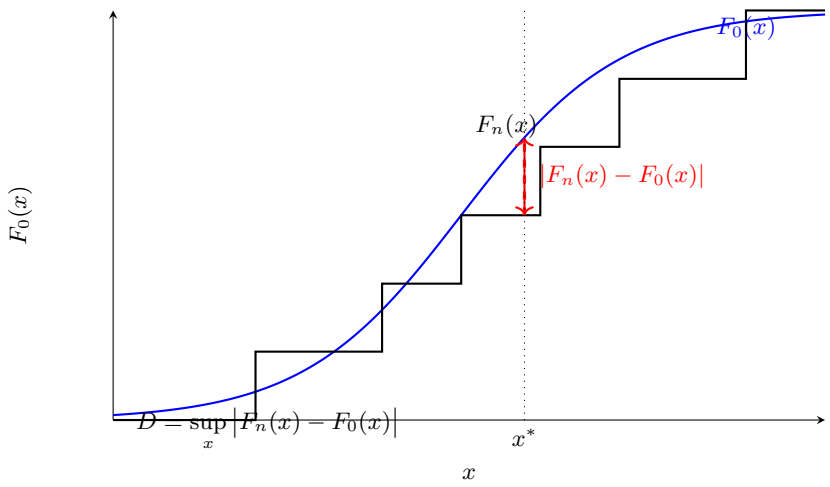


Fig. 7.3. Geometric meaning of the K-S test.

Given i.i.d. observations $X_1, \ldots, X_n$ with empirical CDF

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{X_i \le x\},$$

we test

$$H_0 : F_n(x) \quad \text{follows} \quad F_0(x) \quad \text{vs.} \quad H_1 : F(x) \quad \text{does not follow} \quad F_0(x),$$

where F_0 is a fully specified (continuous) CDF. The K-S statistic is

$$D_n = \sup_x |F_n(x) - F_0(x)|, \qquad D_n^+ = \sup_x \big(F_n(x) - F_0(x)\big), \quad D_n^- = \sup_x \big(F_0(x) - F_n(x)\big),$$

with $D_n = \max(D_n^+, D_n^-)$. For large $n (n > 35)$, a level-α test rejects when

$$D_n > \frac{c_\alpha}{\sqrt{n}}, \quad \text{where } c_{0.10} \approx 1.22,\ c_{0.05} \approx 1.36,\ c_{0.01} \approx 1.63.$$

Table 7.4 lists the critical values for the K-S test when n is small [2]

Table 7.4. Kolmogorov–Smirnov Critical Values

Sample Size (n)	$\boldsymbol{\alpha = 0.10}$	$\boldsymbol{\alpha = 0.05}$	$\boldsymbol{\alpha = 0.01}$
1	0.950	0.975	0.995
2	0.776	0.842	0.929
3	0.636	0.708	0.828
4	0.564	0.624	0.733
5	0.510	0.565	0.669
6	0.470	0.521	0.618
7	0.438	0.486	0.577
8	0.411	0.457	0.543
9	0.388	0.432	0.514
10	0.368	0.410	0.490
11	0.352	0.391	0.468
12	0.338	0.375	0.450
13	0.325	0.361	0.433
14	0.314	0.349	0.418
15	0.304	0.338	0.404
16	0.295	0.328	0.392
17	0.286	0.318	0.381
18	0.278	0.309	0.371
19	0.272	0.301	0.363
20	0.264	0.294	0.356
25	0.240	0.270	0.320
30	0.220	0.240	0.290
35	0.210	0.230	0.270

Notes. (i) F_0 must be fully specified (no parameters estimated) for the standard critical values to be exact. (ii) For discrete F_0, the usual K–S p-values are conservative.

Example 7.4. Use the K-S test to test whether data are Uniform$(0, 1)$. Assume that we have $n = 10$ sorted observations:

$$0.05,\ 0.12,\ 0.18,\ 0.27,\ 0.33,\ 0.44,\ 0.59,\ 0.73,\ 0.85,\ 0.93.$$

The hypothesized distribution $F_0(x) = x$ on $[0, 1]$.

Compute D_n^+ and D_n^-.

Let $x_{(i)}$ denote the i-th order statistic. Then

$$D_n^+ = \max_{1 \le i \le n} \left(\frac{i}{n} - F_0\big(x_{(i)}\big) \right), \quad D_n^- = \max_{1 \le i \le n} \left(F_0\big(x_{(i)}\big) - \frac{i-1}{n} \right).$$

[2] Source: F.J. Massey, "The Kolmogorov–Smirnov Test for Goodness of Fit," *The Journal of the American Statistical Association*, Vol. 46, 1951.

i	$x_{(i)}$	$F_0(x_{(i)}) = x_{(i)}$	$i/n - F_0(x_{(i)})$	$F_0(x_{(i)}) - (i-1)/n$
1	0.05	0.05	$0.10 - 0.05 = 0.05$	$0.05 - 0.00 = 0.05$
2	0.12	0.12	$0.20 - 0.12 = 0.08$	$0.12 - 0.10 = 0.02$
3	0.18	0.18	$0.30 - 0.18 = 0.12$	$0.18 - 0.20 = -0.02$
4	0.27	0.27	$0.40 - 0.27 = 0.13$	$0.27 - 0.30 = -0.03$
5	0.33	0.33	$0.50 - 0.33 = 0.17$	$0.33 - 0.40 = -0.07$
6	0.44	0.44	$0.60 - 0.44 = 0.16$	$0.44 - 0.50 = -0.06$
7	0.59	0.59	$0.70 - 0.59 = 0.11$	$0.59 - 0.60 = -0.01$
8	0.73	0.73	$0.80 - 0.73 = 0.07$	$0.73 - 0.70 = 0.03$
9	0.85	0.85	$0.90 - 0.85 = 0.05$	$0.85 - 0.80 = 0.05$
10	0.93	0.93	$1.00 - 0.93 = 0.07$	$0.93 - 0.90 = 0.03$

Thus,

$$D_n^+ = \max(0.05, 0.08, 0.12, 0.13, 0.17, 0.16, 0.11, 0.07, 0.05, 0.07) = 0.17,$$
$$D_n^- = \max(0.05, 0.02, 0, 0, 0, 0, 0, 0.03, 0.05, 0.03) = 0.05.$$

Note that negative values are ignored. Hence $D_n = \max(D_n^+, D_n^-) = 0.17$.

Test Result.

At level $\alpha = 0.05$ with $n = 10$, the critical value is 0.410. Since $0.17 < 0.431$, we **do not reject** H_0.

Example 7.5. **K–S Test for an Exponential Distribution**

Data and Hypotheses.

Assume that we observe the following $n = 10$ i.i.d. samples:

$$x_{(i)} = (0.12,\ 0.21,\ 0.36,\ 0.40,\ 0.53,\ 0.62,\ 0.85,\ 1.10,\ 1.25,\ 1.50).$$

We test

$$H_0 :\ X \sim \text{Exp}(\lambda = 1) \quad \text{vs.} \quad H_1 :\ X \not\sim \text{Exp}(1),$$

so the null CDF is $F_0(x) = 1 - e^{-x}$ for $x \geq 0$.

Compute Empirical CDF and Discrepancies.

Let $F_n(x)$ be the empirical CDF. For each ordered observation $x_{(i)}$,

$$D_n^+ = \max_{1 \leq i \leq n} \left\{ \frac{i}{n} - F_0\big(x_{(i)}\big) \right\}, \qquad D_n^- = \max_{1 \leq i \leq n} \left\{ F_0\big(x_{(i)}\big) - \frac{i-1}{n} \right\},$$

and $D_n = \max\{D_n^+, D_n^-\}$.

i	$x_{(i)}$	$F_0(x_{(i)})$	i/n	$(i-1)/n$	$i/n - F_0$	$F_0 - (i-1)/n$
1	0.12	0.1131	0.10	0.00	−0.0131	**0.1131**
2	0.21	0.1894	0.20	0.10	0.0106	0.0894
3	0.36	0.3023	0.30	0.20	−0.0023	0.1023
4	0.40	0.3297	0.40	0.30	0.0703	0.0297
5	0.53	0.4114	0.50	0.40	0.0886	0.0114
6	0.62	0.4621	0.60	0.50	0.1379	−0.0379
7	0.85	0.5726	0.70	0.60	0.1274	−0.0274
8	1.10	0.6671	0.80	0.70	0.1329	−0.0329
9	1.25	0.7135	0.90	0.80	0.1865	−0.0865
10	1.50	0.7769	1.00	0.90	**0.2231**	−0.1231

From the table,

$$D_n^+ = 0.2231 \quad (\text{at } i = 10), \qquad D_n^- = 0.1131 \quad (\text{at } i = 1), \qquad \Rightarrow\ D_n = 0.2231.$$

Test Result

For $n = 10$, typical K–S critical values are

$$D_{0.10} = 0.368, \quad D_{0.05} = 0.410, \quad D_{0.01} = 0.490,$$

as listed in Table 7.4. Since $D_n = 0.2231 < 0.368$, we *do not reject* H_0 at the 10%, 5%, or 1% levels.

Remark 7.6. The K-S test can also be applied to test whether two datasets have the same distribution. Given two samples of sizes n and m with empirical CDFs F_n and G_m, test $H_0 : F = G$ using

$$D_{n,m} = \sup_x |F_n(x) - G_m(x)|.$$

For a significance level α, we reject the null hypothesis if $D_{n,m} > D_{n,m,\alpha}$, where $D_{n,m,\alpha}$ is the critical value. For m and n sufficiently large,

$$D_{n,m,\alpha} = c(\alpha)\sqrt{\frac{nm}{n+m}}$$

where $c(\alpha)$ the inverse of the Kolmogorov distribution at α. The detailed calculation of critical values could be found at `https://real-statistics.com/non-parametric-tests/goodness-of-fit-tests/two-sample-kolmogorov-smirnov-test/`

Example 7.7. **Two-Sample Kolmogorov–Smirnov Test**

Data and Hypotheses.

Assume that we have two independent samples:

$$A : (1.2,\ 2.1,\ 2.4,\ 2.9,\ 3.0,\ 3.8,\ 4.2,\ 4.8) \quad (n = 8),$$
$$B : (0.9,\ 1.5,\ 2.2,\ 2.7,\ 3.6,\ 4.0,\ 4.9) \quad (m = 7).$$

We test

$$H_0 :\ F_A(x) = F_B(x) \quad \text{vs.} \quad H_1 :\ F_A(x) \neq F_B(x),$$

using the two-sided K–S statistic $D_{n,m} = \sup_x |F_n(x) - G_m(x)|$, where F_n and G_m are the empirical CDFs of samples A and B.

Compute the Empirical CDFs.

Let $\mathcal{X}$ be the sorted set of all observed values from the two samples. At each $x \in \mathcal{X}$, compute

$$F_n(x) = \frac{\#\{a_i \le x\}}{n}, \qquad G_m(x) = \frac{\#\{b_j \le x\}}{m}, \qquad \Delta(x) = \big|F_n(x) - G_m(x)\big|.$$

x	$F_n(x)$	$G_m(x)$	$\lvert F_n - G_m\rvert$
0.9	0.0000	0.1429	0.1429
1.2	0.1250	0.1429	0.0179
1.5	0.1250	0.2857	0.1607
2.1	0.2500	0.2857	0.0357
2.2	0.2500	0.4286	0.1786
2.4	0.3750	0.4286	0.0536
2.7	0.3750	0.5714	**0.1964**
2.9	0.5000	0.5714	0.0714
3.0	0.6250	0.5714	0.0536
3.6	0.6250	0.7143	0.0893
3.8	0.7500	0.7143	0.0357
4.0	0.7500	0.8571	0.1071
4.2	0.8750	0.8571	0.0179
4.8	1.0000	0.8571	0.1429
4.9	1.0000	1.0000	0.0000

Hence,

$$D_{n,m} = \max_{x \in \mathcal{X}} \Delta(x) = \mathbf{0.1964} \quad \text{(attained at } x = 2.7\text{)}.$$

Test Result.

A common large-sample critical value for the two-sample K–S test is

$$D_{n,m,\alpha} \approx c(\alpha)\sqrt{\frac{n+m}{nm}}, \quad \text{with } c(0.10) = 1.22,\ c(0.05) = 1.36,\ c(0.01) = 1.63.$$

For $n = 8, m = 7$,

$$\sqrt{\frac{n+m}{nm}} = \sqrt{\frac{15}{56}} \approx 0.5176.$$

Thus

$$D_{0.05} \approx 1.36 \times 0.5176 \approx 0.704.$$

Since $D_{n,m} = 0.1964 < 0.704$, we *do not reject* H_0 at the 5% level (and likewise at 10% or 1%).

Q: Why do we use $(i - \frac{1}{2})/n$ in the Q-Q plot but use $\frac{i}{n}$ in the K-S test?

Ans: The difference is due to the different purposes of the Q-Q plot (distribution estimation) and the K-S test (goodness-of-fit test). Recall that the empirical cumulative distribution function (ECDF) is defined as

$$F_n(x) = \frac{1}{n}\sum_{j=1}^{n} I(x_{(j)} \le x),$$

which increases by $1/n$ at each observed value.

The K-S test uses $\frac{i}{n}$ since it needs the exact, formal ECDF to ensure that D_n has the correct theoretical distribution under the null hypothesis. The Q-Q plot uses $(i - \frac{1}{2})/n$ to place each data point in the middle of its cumulative probability interval, in order to get a visually smoother, less biased representation of the data's quantiles (as explained in Q-Q plot in Section 7.1.2).

7.1.5 The *p*-value and "Best Fit"

Recall that we previously set the level of significance α to some traditional values, such as $0.1, 0.05$ or 0.01, when we perform a statistical test. Also recall that the practical meaning of α is the probability of rejecting H_0 when H_0 is actually true. This means that when we decrease the level of significance, the threshold value for the test statistic increases (e.g., see the distribution tables in Chapter 5), and the chance of rejecting H_0 is smaller.

Now, imagine that we have a program that performs the statistical test using gradually decreasing levels of significance. The p-value is the significance level at which one would *just reject* H_0 for the given value of the test statistic. "*Just reject*" indicates the turning point where the statistical result changes from "not reject" to "reject" when we gradually reduce the significance level. Thus, a large p-value tends to indicate a good fit because we would have to accept a large chance of error in order to reject.

Summary Notes:

- A large p-value in a goodness-of-fit test implies that the observed data are well explained by the model, i.e., the model fits well.
- A large p-value does not mean the model is perfect; it only means there is no statistically significant evidence against it.
- When comparing multiple models, the one with the largest p-value is often labelled as the best-fit model.

So far, we have mainly focused on univariate models. In other words, we assume that input random variables are independent of any other variables within the context of the problem. When this is not the case, it is critical that input models account for dependence.

7.2 Multivariate and Time-Series Input Models

Learning points:

- The concepts of covariance and correlation
- The algorithm for generating bivariate normal random variables
- The algorithm for generating AR and ARMA time series
- The Normal-to-Anything Transform (NORTA)

7.2.1 Covariance and Correltation

Let X_1 and X_2 be two random variables defined on the same probability space.

Covariance

The **covariance** between X_1 and X_2 measures the degree to which they vary together. It is defined as

$$\mathrm{Cov}(X_1, X_2) = E\left[(X_1 - E[X_1])(X_2 - E[X_2])\right].$$

An equivalent expression is

$$\mathrm{Cov}(X_1, X_2) = E[X_1 X_2] - E[X_1]E[X_2].$$

Based on the definition, the covariance is expressed in units equal to the product of the units of X_1 and X_2.

Interpretation:

- If $\mathrm{Cov}(X_1, X_2) > 0$, then X_1 and X_2 tend to increase or decrease together (positive relationship).
- If $\mathrm{Cov}(X_1, X_2) < 0$, when X_1 increases, X_2 tends to decrease (negative relationship).
- If $\mathrm{Cov}(X_1, X_2) = 0$, X_1 and X_2 are uncorrelated (no linear relationship).

Correlation

The covariance can take any value between $-\infty$ and ∞. The correlation standardizes the covariance to be between -1 and 1. The **correlation** (also called the **Pearson correlation coefficient**) between X_1 and X_2.is defined as

$$\rho_{X_1,X_2} = \frac{\mathrm{Cov}(X_1, X_2)}{\sigma_{X_1}\sigma_{X_2}},$$

where $\sigma_{X_1} = \sqrt{\mathrm{Var}(X_1)}$ and $\sigma_{X_2} = \sqrt{\mathrm{Var}(X_2)}$. We normally omit the subscript and simply use ρ to denote correlation.

Properties:

- $-1 \le \rho \le 1$
- $\rho = 1$: perfect positive linear relationship
- $\rho = -1$: perfect negative linear relationship
- $\rho = 0$: no linear relationship

The correlation coefficient is unit-free, making it suitable for comparing relationships between variables measured on different scales.

Remark 7.8.
- Zero covariance/correlation does not imply independence!. Zero covariance only means no linear relationship: A covariance of zero indicates the variables are not related in a straight-line pattern, but they could still be related in a non-linear way. For instance, consider two random variables, X and Y, where X is a number being $-1, 0$, or 1 with equal probability, and Y is equal to X^2. In this case, knowing the value of Y changes the probability of X, so they are dependent. However, the covariance between X and Y is zero.
- Independence implies zero covariance/correlation: If two variables are independent, their covariance/correlation must be zero because independence means they do not affect each other's probability distribution.

7.2.2 Generating Bivariate Normal

Definition of Multivariate Normal Distribution

A random vector

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$$

is said to follow an n-**dimensional multivariate normal distribution** if any linear combination of its components is normally distributed. We write

$$\mathbf{X} \sim \mathcal{N}_n(\boldsymbol{\mu}, \Sigma),$$

where

- $\boldsymbol{\mu} = E[\mathbf{X}] = (\mu_1, \mu_2, \ldots, \mu_n)^\top$ is the mean vector,
- $\Sigma = \mathrm{Cov}(\mathbf{X}) = E[(\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^\top]$ is the **covariance matrix**.

Covariance and Correlation Matrices

The covariance matrix $\Sigma = [\sigma_{ij}]$ is defined elementwise as

$$\sigma_{ij} = \mathrm{Cov}(X_i, X_j) = E[(X_i - \mu_i)(X_j - \mu_j)].$$

The diagonal entries are the variances of each component:

$$\sigma_{ii} = \mathrm{Var}(X_i).$$

The **correlation matrix**, denoted by R, standardizes the covariance matrix by the variances of each variable:

$$R = [\rho_{ij}], \qquad \rho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}.$$

Example 7.9. **Example: Bivariate Normal Case**: For a bivariate normal random vector

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim \mathcal{N}_2 \left(\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \; \Sigma = \begin{bmatrix} \sigma_1^2 & \rho\,\sigma_1\sigma_2 \\ \rho\,\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} \right),$$

the corresponding correlation matrix is

$$R = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}.$$

The following simple algorithm can be used to generate bivariate normal random variables:

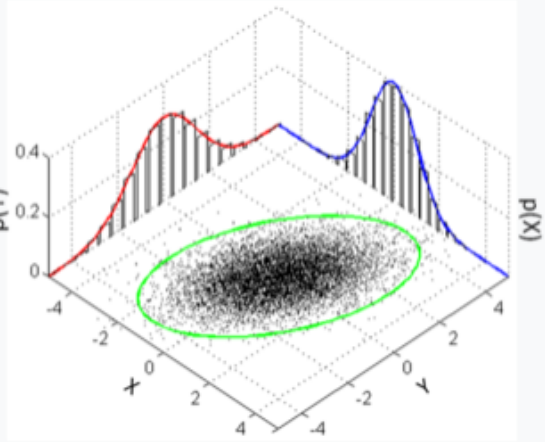
- Step 1: Generate independent standard normal random variables, Z_1 and Z_2 (using the method introduced in Section 6.4).
- Step 2: Set $X_1 = \mu_1 + \sigma_1 Z_1$.
- Step 3: Set $X_2 = \mu_2 + \sigma_2(\rho Z_1 + \sqrt{1-\rho^2} Z_2)$.

7.2.3 AR and ARMA Models

Let $\{X_t\}$ be a stationary time series with mean $E[X_t] = \mu$ (often taken as 0 after centering) and white-noise innovations $\{\varepsilon_t\}$, where $\varepsilon_t \overset{\text{i.i.d.}}{\sim} (0, \sigma_\varepsilon^2)$.

Multivariate normal

Probability density function



Many sample points from a multivariate normal distribution with $\boldsymbol{\mu} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 3/5 \\ 3/5 & 2 \end{bmatrix}$, shown along with the 3-sigma ellipse, the two marginal distributions, and the two 1-d histograms.

Notation	$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
Parameters	$\boldsymbol{\mu} \in \mathbf{R}^k$ — location $\boldsymbol{\Sigma} \in \mathbf{R}^{k \times k}$ — covariance (positive semi-definite matrix)

$$\boldsymbol{\mu} = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{pmatrix}.$$

Fig. 7.4. Illustration of bivariate normal distribution (figure from Wikipedia).

Autoregressive Model AR(p)

An **AR**(p) model expresses the current value as a linear combination of its past p values plus noise:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t.$$

where $-1 < \phi_i < 1$, for $i = 1, \ldots, p$.

Example 7.10. **Generating AR(1)**: The following algorithm generates a stationary AR(1) time series, given values of the parameters $\phi_1, \mu, \sigma_\varepsilon^2$.

- Step 1: Generate X_1 from the normal distribution with mean μ and variance $\sigma_\varepsilon^2/(1-\phi_1^2)$.
- Step 2: Generate ε_t from the normal distribution with mean 0 and variance σ_ε^2.
- Step 3: Set $X_t = \mu + \phi_1(X_{t-1} - \mu) + \varepsilon_t$.
- Step 4: Set $t = t + 1$ and go to Step 2.

Autoregressive Moving-Average Model ARMA(p, q)

An **ARMA(p, q)** model combines both structures:

$$X_t - \phi_1 X_{t-1} - \cdots - \phi_p X_{t-p} \;=\; \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}.$$

Special Cases

- **AR(1):** $X_t = \phi X_{t-1} + \varepsilon_t$.
- **ARMA(1,1):** $X_t = \phi X_{t-1} + \varepsilon_t + \theta \varepsilon_{t-1}$.

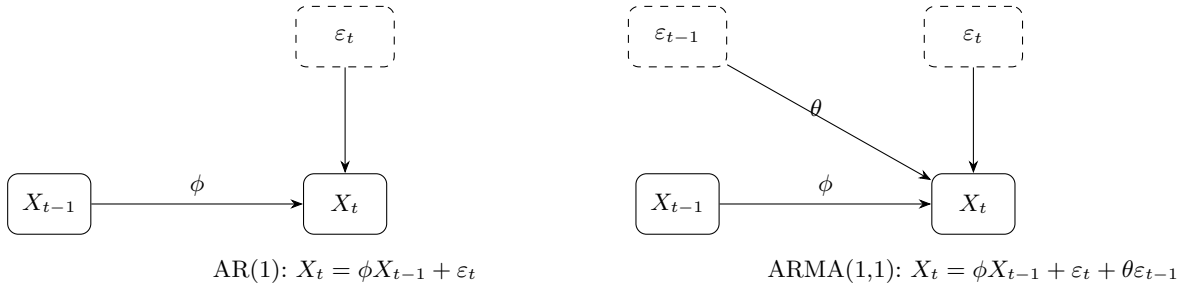


Fig. 7.5. Dependency structures for AR(1) and ARMA(1,1). Solid boxes denote X_t states; dashed boxes denote innovations ε_t.

7.2.4 The Normal-to-Anything Transformation (NORTA)

The Normal-to-Anything (NORTA) method is a widely used technique for generating multivariate random vectors with arbitrary marginal distributions and a specified correlation structure. It is especially useful in simulation studies where the input variables are non-normal but correlated.

Main Idea

The NORTA method relies on transforming a multivariate normal random vector into a target random vector with the desired marginal distributions and correlations.

Let

$$\mathbf{Z} = (Z_1, Z_2, \ldots, Z_n)^\top$$

be a multivariate normal random vector with mean vector $\mathbf{0}$, covariance matrix Σ_Z, and correlation matrix R_Z. The NORTA procedure constructs a new random vector

$$\mathbf{X} = (X_1, X_2, \ldots, X_n)^\top$$

such that each component X_i has a prescribed marginal distribution $F_i(x)$ and the pairwise correlations between X_i and X_j *approximately* match the desired target correlations.

The NORTA procedure can be summarized as:

$$\boxed{Z_i \xrightarrow{\Phi} U_i \xrightarrow{F_i^{-1}} X_i, \quad i = 1, 2, \ldots, n,}$$

where $\Phi(\cdot)$ is the standard normal CDF and $F_i^{-1}(\cdot)$ is the inverse CDF of the target distribution. This "Normal-to-Anything" transformation provides a flexible way to generate multivariate input data with arbitrary marginals and controlled dependence.

Transformation Steps

The transformation is performed as follows:

1. **Generate correlated normal variables:** Simulate $\mathbf{Z} \sim N(\mathbf{0}, R_Z)$, where R_Z is the correlation matrix chosen to induce the desired dependence among the components.
2. **Transform to uniform variables:** Use the standard normal cumulative distribution function (CDF) $\Phi(\cdot)$ to obtain uniform random variables:
$$U_i = \Phi(Z_i), \qquad i = 1, 2, \ldots, n.$$
Then, $\mathbf{U} = (U_1, \ldots, U_n)$ follows a joint uniform distribution on $(0,1)^n$ with dependence induced by R_Z.
3. **Transform to target marginals:** Apply the inverse CDF (quantile function) of each target distribution:
$$X_i = F_i^{-1}(U_i), \qquad i = 1, 2, \ldots, n.$$
The resulting vector $\mathbf{X}$ has marginal distributions F_i.

Remark 7.11. (NORTA): The main motivation of NORTA is to generate multivariate data following a given correlation. However, the correlation between the standard normal random variables (Z_1, Z_2) is distorted when it passes through the NORTA transformation. To be more specific, if (Z_1, Z_2) have correlation ρ, then in general $X_1 = F_1^{-1}(\Phi(Z_1)$ and $X_2 = F_2^{-1}(\Phi(Z_2)$ will have a correlation $\rho_X \neq \rho$. The difference is often small, but not always. Therefore, we need a fine-tuning algorithm to adjust the correlation.

Remark 7.12. The set of possible correlations between two random variables with fixed marginals is *not always* the full interval $[-1, 1]$. It depends on how the marginals can be "coupled that is, the structure of their joint dependence.

Example 7.13. **A Numerical Example of NORTA for Two Variables**. Assume that we want to construct a bivariate random vector (X_1, X_2) with:

- Marginals:
$$X_1 \sim \text{Exp}(1), \qquad X_2 \sim \text{Lognormal}(\mu = 0,\ \sigma = 0.5),$$
- Target Pearson correlation: $\rho_X = 0.50$.

Following NORTA, we perform:

1. **Generate correlated normal variables:** Generate $(Z_1, Z_2) \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} 1 & \rho_Z \\ \rho_Z & 1 \end{bmatrix}\right)$, using $\rho_Z = 0.50$
2. **Transform to uniform variables:** Set $U_i = \Phi(Z_i)$, $i = 1, 2$, where Φ is the standard normal CDF; then $U_i \sim \text{Uniform}(0, 1)$.
3. **Transform to target marginals:**

$$X_1 = F_1^{-1}(U_1) = -\ln(1-U_1) \quad (\text{Exponential}(1)), \qquad X_2 = F_2^{-1}(U_2) = \text{Lognormal}^{-1}(U_2;\ \mu = 0,\ \sigma = 0.5).$$

Nevertheless, as we have mentioned in Remark 7.11, we need to verify whether or not $\rho_X \neq \rho$, and if the gap is large, we need to fine-tune the correlation. Assume that $\rho_X = 0.47$, and we think this gap may be too large for our application.

Fine-Tuning, If Needed

We fine-tune using the following search method. The Pearson correlation after the transforms is a *monotone* function of ρ_Z:
$$\rho_X \ = \ g(\rho_Z) \quad (\text{no closed form in general}).$$
Based on the monotonicity, we then solve $g(\rho_Z) = 0.5$ numerically using bisection with simulated samples.

Fine-Tuning with Bisection

Not every $\rho_X \in [-1, 1]$ is attainable with given marginals. The achievable Pearson correlation interval may be a strict subset of $[-1, 1]$. A practical check is to evaluate $g(\rho_Z)$ at $\rho_Z \approx \pm 0.99$ to estimate $(g_{\min}, g_{\max})$ and confirm $\rho_X \in [g_{\min}, g_{\max}]$.

Bisection-based Fine-tuning for This Example

Using $\rho_Z = 0.5$, we have obtained $\rho_X = 0.47$. Assume that the upper ρ_Z is set to 0.9 (Note that data shows that $\widehat{g}_n(0.9) \approx 0.86$).

Bisection (illustrative).

With CRN and large n:

trial ρ_Z	$\widehat{g}_n(\rho_Z)$	decision
0.50	0.47	raise ρ_Z
0.70	0.55	lower ρ_Z
0.60	0.53	lower ρ_Z
0.55	0.51	narrow
0.542	0.500	accept (within tolerances)

Final estimate:

$$\boxed{\rho_Z^\star \approx 0.541\text{–}0.542, \qquad \widehat{g}_n(\rho_Z^\star) \approx 0.50.}$$

Algorithm: Bisection with Common Random Numbers (CRN)

1. **Bracket.** Choose (ℓ, u) with $-0.99 \le \ell < u \le 0.99$ s.t. $\widehat{g}_n(\ell) \le \rho_X \le \widehat{g}_n(u)$. If not, widen or reset (ℓ, u); if still infeasible, the target ρ_X may be unattainable.
2. **CRN.** Fix a single stream of base normals $(W_1^{(m)}, W_2^{(m)})$. For each trial ρ, construct correlated normals via
$$Z_1^{(m)} = W_1^{(m)}, \qquad Z_2^{(m)} = \rho\, W_1^{(m)} + \sqrt{1-\rho^2}\, W_2^{(m)}.$$
Using the *same* $(W_1^{(m)}, W_2^{(m)})$ at all ρ greatly reduces variance in the *difference* $\widehat{g}_n(\rho) - \widehat{g}_n(\rho')$ and stabilizes bisection.
3. **Iterate.** For $t = 1, 2, \ldots$:
   a) $m = (\ell + u)/2$.
   b) Compute $\widehat{g}_n(m)$ using the CRN construction above.
   c) If $\widehat{g}_n(m) < \rho_X$, set $\ell \leftarrow m$; else $u \leftarrow m$.
   d) Stop when $|u - \ell| < \varepsilon_\rho$ *and* $|\widehat{g}_n(m) - \rho_X| < \varepsilon_g$.
4. **(Optional) Stagewise n.** Start with modest n (e.g. $n = 5 \times 10^4$), then increase n at the last 1–2 iterations to tighten the final error.

Algorithms: Sample Covariance and Correlation Estimation

Let $\{(X_{1i}, X_{2i})\}_{i=1}^n$ be i.i.d. data pairs drawn from the joint distribution of (X_1, X_2).

1. Sample Means

The sample means of X_1 and X_2 are given by

$$\bar{X}_1 = \frac{1}{n}\sum_{i=1}^n X_{1i}, \qquad \bar{X}_2 = \frac{1}{n}\sum_{i=1}^n X_{2i}.$$

2. Sample Variances

The sample variances of X_1 and X_2 are

$$s_1^2 = \frac{1}{n-1}\sum_{i=1}^n (X_{1i} - \bar{X}_1)^2, \qquad s_2^2 = \frac{1}{n-1}\sum_{i=1}^n (X_{2i} - \bar{X}_2)^2,$$

where the factor $\frac{1}{n-1}$ makes s_1^2, and s_2^2 unbiased estimators.

3. Sample Covariance

Similarly, the sample covariance estimates the population covariance $\mathrm{Cov}(X_1, X_2) = E[(X_1 - \mu_1)(X_2 - \mu_2)]$ by

$$s_{12} = \frac{1}{n-1}\sum_{i=1}^{n}(X_{1i} - \bar{X}_1)(X_{2i} - \bar{X}_2).$$

4. Sample Correlation Coefficient

The sample correlation is calculated as

$$\hat{\rho} = \frac{s_{12}}{s_1 s_2},$$

which is the estimator of the correlation between X_1 and X_2.

Example 7.14. Consider the following pairs:

i	1	2	3	4	5	6	7	8	9	10
X_{1i}	2	3	4	5	6	7	8	9	10	11
X_{2i}	1	3	5	4	6	8	7	9	11	11

Compute the basic sums:

$$\sum X_{1i} = 65, \quad \sum X_{2i} = 65, \quad \sum X_{1i}X_{2i} = 511, \quad \sum X_{1i}^2 = 505, \quad \sum X_{2i}^2 = 523.$$

Hence

$$\bar{X}_1 = \bar{X}_2 = \frac{65}{10} = 6.5.$$

$$\sum(X_{1i} - \bar{X}_1)^2 \;=\; 82.5, \qquad \sum(X_{2i} - \bar{X}_2)^2 \;=\; 100.5, \qquad \sum(X_{1i} - \bar{X}_1)(X_{2i} - \bar{X}_2) \;=\; 88.5.$$

Divide by $n - 1 = 9$ to obtain the sample covariance and variances:

$$s_{12} = \frac{88.5}{9} = \frac{59}{6} \approx 9.8333, \qquad s_1^2 = \frac{82.5}{9} = \frac{55}{6} \approx 9.1667, \qquad s_2^2 = \frac{100.5}{9} = \frac{67}{6} \approx 11.1667.$$

Therefore, the sample standard deviations are:

$$s_1 = \sqrt{\frac{55}{6}} \approx 3.0277, \qquad s_2 = \sqrt{\frac{67}{6}} \approx 3.3417.$$

Finally, the sample correlation is

$$\hat{\rho} = \frac{s_{12}}{s_1 s_2} = \frac{\frac{59}{6}}{\sqrt{\frac{55}{6} \cdot \frac{67}{6}}} = \frac{59}{\sqrt{55 \cdot 67}} = \frac{59}{\sqrt{3685}} \approx 0.9719.$$

7.3 Supplementary Reading Materials

Textbook (Law 2024): Chapter 6.4, 6.5, 6.6, 6.10, 8.5.5. Note that my lectures do not strictly follow the textbook.

8

Output Analysis 1: Estimation of Absolute Performance

In this and the next chapters, we focus on the analysis of output data. This chapter is for absolute performance, i.e., the performance of one system, and the next chapter is for relative performance, i.e., the comparison of two or more systems.

A Note on Statistical Table and Notations

Different ways of presenting statistical tables correspond to different notational conventions, even though the numerical values in the tables remain the same. For example, in this lecture note we denote the critical value for a $(1-\alpha)$ confidence interval from the t-distribution as $t_{\alpha/2,\,R-1}$. However, if a table follows the left-tail (cumulative) presentation shown in Fig. 8.1, the same critical value would be written as $t_{1-\alpha/2,\,R-1}$. The same issue arises with other tables, such as the standard normal table.

Because research papers and textbooks may adopt different conventions, it is important to read their notations carefully. This lecture note (perhaps unintentionally) includes both table formats, which serve as a helpful reminder to pay attention to these subtleties.

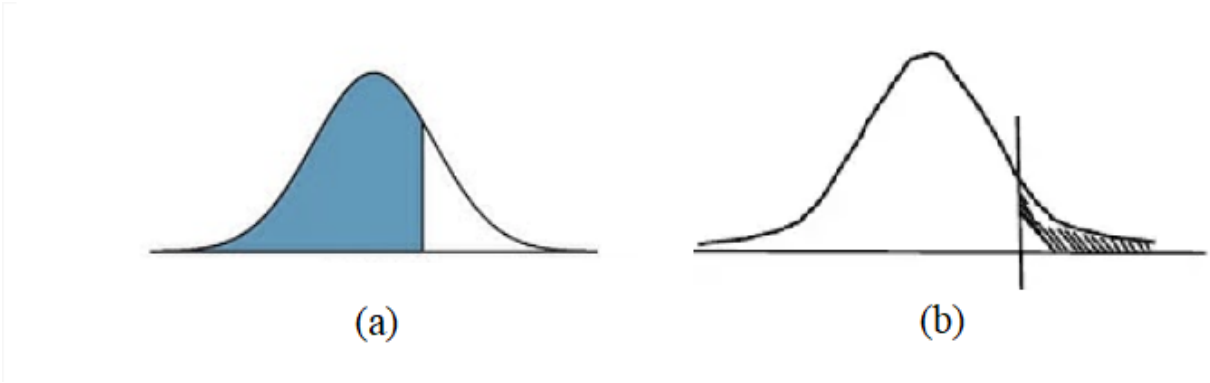


Fig. 8.1. Different ways of presenting a statistical table (standard normal).

As a side note, the t distribution can be approximated by the standard normal (z) distribution:

$$t_{\alpha,\,\nu} \approx z_{1-\alpha} \qquad \text{for large degrees of freedom } \nu,$$

if following the different table presentations used in this lecture note.

8.1 Type of Simulations

From the viewpoint of output analysis, the type of simulations could be either terminating or non-terminating simulations. A terminating simulation is one that runs for some duration of time T_E, where E is a specified event that stops the simulation. The initial conditions of the system at time 0 must be specified, and the stopping time T_E (or the stopping event) must also be well defined. A non-termination simulation is one that runs continuously or at least over a very long period of time (simulation clock).

In the terminating simulation, we care more about the transient performance, i.e., the performance during state transition; Hence, the impact of initial and stopping conditions should be investigated. In the latter, we focus on long-term stationary performance, i.e., the performance not impacted by the initial condition of the system. Hence, we need to find a method to eliminate the impact of the initial condition on the stationary performance.

8.2 Stochastic Nature of Output Data

Similar to input data, output data are also stochastic. Thus, we also need to estimate the "true" value (i.e., true performance), using estimators.

In input data analysis, we estimate the distribution and parameters. In output data analysis, we usually estimate the mean performance and the concentration of the performance.

8.3 Absolute Measures

We use both a point estimator, which estimates a performance parameter, and an interval estimator, which measures the error of the point estimator.

8.3.1 Point Estimator

Point estimator for discrete data: $\{Y_1, Y_2, \ldots, Y_n\}$, is defined by:

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} Y_i.$$

Point estimator for "continuous-time" data: $\{Y(t), 0 \le t \le T\}$, is defined by:

$$\hat{\theta} = \frac{1}{T}\int_0^T Y(t)dt.$$

8.3.2 Interval Estimators

What is the difference between confidence intervals and prediction intervals?

Confidence intervals and prediction intervals are both types of interval estimates used in statistics, but they have different purposes and interpretations.

A confidence interval is an estimate of the range of values within which the true value of a **performance parameter**, such as average system delay, is likely to fall with a certain level of confidence. The level of confidence is typically expressed as a percentage, such as 95%, and represents the probability that the interval will contain the true parameter value if the estimation process were repeated many times. A confidence interval is based on a sample of data and provides an estimate of the precision of the estimate of the **performance parameter**.

On the other hand, a prediction interval is an estimate of the range of values within which an **individual observation or future observation** is likely to fall with a certain level of confidence. Unlike a confidence interval, which estimates the range of values for a performance parameter, a prediction interval estimates the range of values for an individual observation (e.g., the delay of the next customer). A prediction interval

takes into account both the uncertainty associated with estimating the performance parameter from the sample data, as well as the variability of individual observations around the estimated mean.

In summary, the main difference between confidence intervals and prediction intervals is that a confidence interval is used to estimate the range of values for a performance parameter (e.g., the average delay), while a prediction interval is used to estimate the range of values for an individual observation (e.g., the delay of next customer). Therefore, in theory, the length of the confidence interval approaches zero when $n \to \infty$, but the length of the prediction interval will not go to zero when $n \to \infty$.

How to calculate confidence intervals and how to calculate prediction intervals?

First, calculate sample variance:

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$$

- Confidence interval (with $(1-\alpha)$-confident):

$$\bar{Y} \pm t_{\alpha/2,n-1}\frac{S}{\sqrt{n}}.$$

- Prediction interval (with $(1-\alpha)$-confident) :

$$\bar{Y} \pm t_{\alpha/2,n-1}S\sqrt{1+\frac{1}{n}}$$

Note that $t_{\alpha/2,n-1}$ denotes the $100(1-\alpha)$ percentage point of a t distribution with $n-1$ degree of freedom. For instance, $\alpha = 0.5$ means 95% confidence level. A table for t distribution critical values could be found in Fig. 8.2.

Example 8.1. Consider a sample of size $n = 8$:

$$x = \{12,\ 13,\ 15,\ 11,\ 14,\ 10,\ 9,\ 16\}.$$

Calculate Sample Mean and Standard Deviation

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{100}{8} = 12.5,$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} = \sqrt{6.0} \approx 2.4495.$$

Calculate the Critical Value

For a 95% confidence level and degrees of freedom $n-1 = 7$,

$$t_{0.025,7} = 2.3646.$$

Calculate the Confidence Interval

The formula for the confidence interval (CI) is

$$\bar{x} \pm t_{0.025,7}\,\frac{s}{\sqrt{n}}.$$

The half-width of the interval is

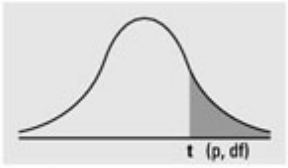
$$2.3646 \times \frac{2.4495}{\sqrt{8}} = 2.3646 \times 0.8660 \approx 2.0478.$$

Thus, the 95% CI is

$$(12.5 - 2.0478,\ 12.5 + 2.0478) = (10.4522,\ 14.5478).$$

$$\boxed{95\%\text{ CI: } (10.452,\ 14.548)}$$

Numbers in each row of the table are values on a *t*-distribution with (*df*) degrees of freedom for selected right-tail (greater-than) probabilities (*p*).



df/p	0.40	0.25	0.10	0.05	0.025	0.01	0.005	0.0005
1	0.324920	1.000000	3.077684	6.313752	12.70620	31.82052	63.65674	636.6192
2	0.288675	0.816497	1.885618	2.919986	4.30265	6.96456	9.92484	31.5991
3	0.276671	0.764892	1.637744	2.353363	3.18245	4.54070	5.84091	12.9240
4	0.270722	0.740697	1.533206	2.131847	2.77645	3.74695	4.60409	8.6103
5	0.267181	0.726687	1.475884	2.015048	2.57058	3.36493	4.03214	6.8688
6	0.264835	0.717558	1.439756	1.943180	2.44691	3.14267	3.70743	5.9588
7	0.263167	0.711142	1.414924	1.894579	2.36462	2.99795	3.49948	5.4079
8	0.261921	0.706387	1.396815	1.859548	2.30600	2.89646	3.35539	5.0413
9	0.260955	0.702722	1.383029	1.833113	2.26216	2.82144	3.24984	4.7809
10	0.260185	0.699812	1.372184	1.812461	2.22814	2.76377	3.16927	4.5869
11	0.259556	0.697445	1.363430	1.795885	2.20099	2.71808	3.10581	4.4370
12	0.259033	0.695483	1.356217	1.782288	2.17881	2.68100	3.05454	43178
13	0.258591	0.693829	1.350171	1.770933	2.16037	2.65031	3.01228	4.2208
14	0.258213	0.692417	1.345030	1.761310	2.14479	2.62449	2.97684	4.1405
15	0.257885	0.691197	1.340606	1.753050	2.13145	2.60248	2.94671	4.0728
16	0.257599	0.690132	1.336757	1.745884	2.11991	2.58349	2.92078	4.0150
17	0.257347	0.689195	1.333379	1.739607	2.10982	2.56693	2.89823	3.9651
18	0.257123	0.688364	1.330391	1.734064	2.10092	2.55238	2.87844	3.9216
19	0.256923	0.687621	1.327728	1.729133	2.09302	2.53948	2.86093	3.8834
20	0.256743	0.686954	1.325341	1.724718	2.08596	2.52798	2.84534	3.8495
21	0.256580	0.686352	1.323188	1.720743	2.07961	2.51765	2.83136	3.8193
22	0.256432	0.685805	1.321237	1.717144	2.07387	2.50832	2.81876	3.7921
23	0.256297	0.685306	1.319460	1.713872	2.06866	2.49987	2.80734	3.7676
24	0.256173	0.684850	1.317836	1.710882	2.06390	2.49216	2.79694	3.7454
25	0.256060	0.684430	1.316345	1.708141	2.05954	2.48511	2.78744	3.7251
26	0.255955	0.684043	1.314972	1.705618	2.05553	2.47863	2.77871	3.7066
27	0.255858	0.683685	1.313703	1.703288	2.05183	2.47266	2.77068	3.6896
28	0.255768	0.683353	1.312527	1.701131	2.04841	2.46714	2.76326	3.6739
29	0.255684	0.683044	1.311434	1.699127	2.04523	2.46202	2.75639	3.6594
30	0.255605	0.682756	1.310415	1.697261	2.04227	2.45726	2.75000	3.6460
z	0.253347	0.674490	1.281552	1.644854	1.95996	2.32635	2.57583	3.2905
CI	———	———	80%	90%	95%	98%	99%	99.9%

Fig. 8.2. Critical values for t distribution.

What is 95% Prediction Interval for One Future Observation?

The formula for the prediction interval (PI) is

$$\bar{x} \pm t_{0.025,7}\, s \sqrt{1 + \frac{1}{n}}.$$

The half-width of the interval is

$$2.3646 \times 2.4495 \times \sqrt{1 + \frac{1}{8}} = 2.3646 \times 2.4495 \times 1.0607 \approx 6.1435.$$

Therefore, the 95% PI is

$$(12.5 - 6.1435,\ 12.5 + 6.1435) = (6.3565,\ 18.6435).$$

$$\boxed{95\%\text{ PI: } (6.357,\ 18.643)}$$

8.4 Output Analysis for Terminating Simulations

Learning point 1: Understand the difference between *within-replication data* and *across-replication* data.

Within-replication data refers to data that are generated within a single simulation replication, where a replication is a single run of the simulation under a specific set of input parameters. Within-replication data can include measures of the simulation output, such as summary statistics, distributions, or other relevant metrics. Within-replication data are used to assess the variability of the simulation output within a single replication and to estimate the precision of the estimates generated by the simulation model.

Across-replication data, on the other hand, refers to data that are generated across multiple simulation replications, where multiple replications are multiple runs of the simulation each run using a different random number stream and independently chosen initial conditions. Across-replication data are used to assess the variability of the simulation output across different replications and to estimate the uncertainty associated with the simulation model.

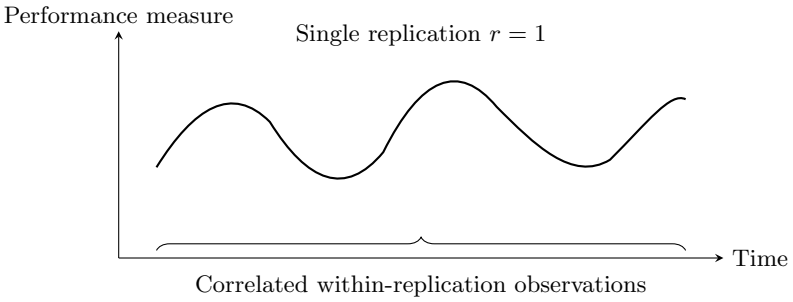


Fig. 8.3. Within-replication time-series data: many correlated observations from a single replication.

Table 8.1. Within-replication vs. across-replication data in simulation

Aspect	Within-replication data	Across-replication data
Definition	Data collected *within a single replication* of the simulation run.	Data collected *across multiple independent replications* of the simulation.
Purpose	Analyze time-series behavior, transient performance, and per-event/per-customer observations.	Estimate steady-state performance measures and quantify statistical uncertainty.
Typical examples	Waiting times of all customers in one run, queue length over time, server utilization trajectories, event logs.	Average waiting time per replication, average queue length per replication, replication-level throughput.
Randomness source	Uses a single random seed for that replication; observations over time are dependent.	Each replication uses an independent random seed; replication outputs are (approximately) independent samples.
Statistical properties	Observations are usually correlated (not i.i.d.).	Replication outputs are treated as i.i.d. and are suitable for confidence-interval estimation.
Main use	Warm-up detection, time-series plots, debugging model logic, distinguishing transient vs. steady-state behavior.	Point estimation, confidence intervals, comparison of alternative system designs, variance analysis.
Typical data size	Large (many observations per replication).	Small (often one summary statistic per replication).
Analysis tools	Time-series plots, moving averages, autocorrelation analysis, batch means (within a replication).	Sample mean and variance, confidence intervals, hypothesis tests, ranking-and-selection procedures.

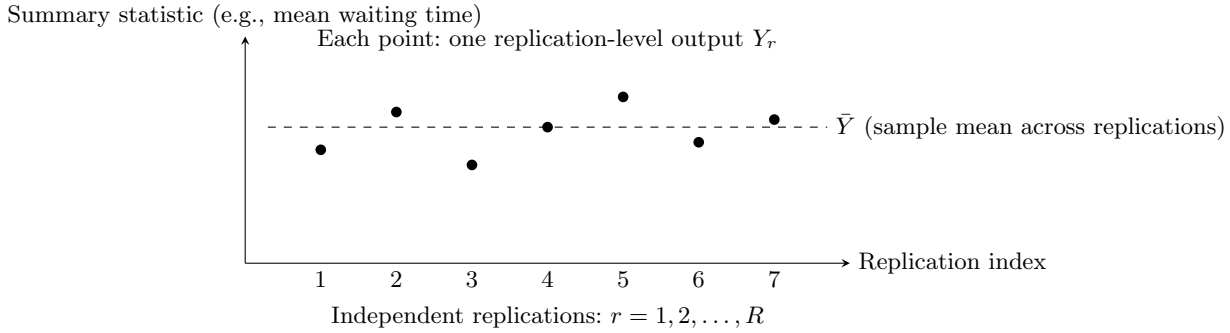


Fig. 8.4. Across-replication data: one summary statistic per replication.

Table 8.2. Within and Across-Replication Data

Replication	Within-Rep. Data	Across-Rep. Data
1	$Y_{11}, Y_{12}, \ldots, Y_{1n_1}$	$\bar{Y}_{1\cdot}, S_1, H_1$
2	$Y_{21}, Y_{22}, \ldots, Y_{2n_2}$	$\bar{Y}_{2\cdot}, S_2, H_2$
3	$Y_{31}, Y_{32}, \ldots, Y_{3n_3}$	$\bar{Y}_{3\cdot}, S_3, H_3$
⋮	⋮	⋮
R	$Y_{R1}, Y_{R2}, \ldots, Y_{Rn_R}$	$\bar{Y}_{R\cdot}, S_R, H_R$
		$\bar{Y}_{\cdot\cdot}, S, H$

Learning point 2: How to determine the number of runs?

We already know the half-length H of a $(1-\alpha)\%$ confidence interval is calculated as:

$$\boxed{H = t_{\alpha/2,n-1}\frac{S}{\sqrt{n}},}$$

which can also be approximated by z distribution when the degree of freedom is large:

$$\boxed{H = z_{1-\alpha/2}\frac{S}{\sqrt{n}}.}$$

Our goal is to determine how many replications are required to ensure that the half-width H of the confidence interval satisfies $H \leq \epsilon$. This creates an immediate dilemma: we cannot know H without first performing multiple simulation runs.

A standard approach to resolving this issue is to begin with an initial set of R_0 replications and compute H based on the resulting cross-replication data. If the estimated half-width satisfies $H \leq \epsilon$, we stop. Otherwise, we use this initial estimate to determine how many additional replications are needed to achieve the desired precision.

Assume that after R_0 replications,

$$H = t_{\alpha/2,R_0-1}\frac{S_0}{\sqrt{R_0}} > \epsilon,$$

where S_0 is the sample standard deviation of the R_0 data points. We use this S_0 as the initial estimation of the population standard deviation. By setting our goal,

$$H = t_{\alpha/2,R-1}\frac{S_0}{\sqrt{R}} \leq \epsilon,$$

we have

$$R \geq \left(\frac{t_{\alpha/2,R-1}S_0}{\epsilon}\right)^2.$$

The smallest R value that meets the above condition is the number of runs needed to achieve the desired precision. Alternatively, we can approximate

$$R \geq \left(\frac{z_{1-\alpha/2}S_0}{\epsilon}\right)^2.$$

Nevertheless, in practice, we normally just pre-set the α value and the number of runs, e.g., 10, and calculate the confidence interval based on the simulation results. This is mainly due to cost considerations, particularly when a simulation run takes a long time, e.g., 1 day.

Learning point 3: Estimation for Quantiles.

In some cases, we not only want to know the mean value and its confidence interval, but also the quantile of a given portion. For instance, if we have obtained an answer like "the average waiting time of customers in a queue" is 300 ± 30 seconds with 95% confidence, we may also want to know " what is the waiting time for 85% customers in the queue, i.e., 85% quantile"? In addition, if we have found that 85% of customers' waiting time in the queue is 310 seconds, what is the confidence interval for this $p = 85\%$?

Estimating Quantiles and Their Confidence Intervals

Suppose we run a simulation with R independent replications. In replication r, we collect within-replication observations

$$X_{r1}, X_{r2}, \ldots, X_{rn_r}, \qquad r = 1, \ldots, R.$$

We are interested in estimating the q-quantile of the underlying steady-state distribution, denoted by θ_q, which satisfies

$$P(X \leq \theta_q) = q.$$

Step 1. Estimating Quantiles Using Within-Replication Data

For each replication r, sort the data:

$$X_{r(1)} \le X_{r(2)} \le \cdots \le X_{r(n_r)}.$$

The within-replication estimate of the q-quantile is

$$\hat{\theta}_{q,r} = X_{r(\lceil qn_r \rceil)}.$$

Since the replications are independent, the R values

$$\hat{\theta}_{q,1},\ \hat{\theta}_{q,2},\ \ldots,\ \hat{\theta}_{q,R}$$

constitute i.i.d. estimates of θ_q.

The overall point estimator is the across-replications sample mean:

$$\bar{\theta}_q = \frac{1}{R}\sum_{r=1}^{R} \hat{\theta}_{q,r}.$$

Step 2. Confidence Interval for the q-Quantile

Compute the sample variance across replications:

$$S^2_{\theta_q} = \frac{1}{R-1}\sum_{r=1}^{R}\left(\hat{\theta}_{q,r} - \bar{\theta}_q\right)^2.$$

A $(1-\alpha)100\%$ confidence interval for θ_q is obtained as follows:

$$\boxed{\bar{\theta}_q \ \pm\ t_{\alpha/2,R-1}\,\frac{S_{\theta_q}}{\sqrt{R}}}.$$

Example 8.2. **Numerical Example:** Assume that we run $R = 5$ independent replications of a simulation. In each replication, we collect $n_r = 1000$ observations (e.g., customer waiting times), and we are interested in the $q = 0.9$ quantile (the 90th percentile).

Within-replication quantiles:

For each replication r, we sort the n_r observations and compute the within-replication 0.9-quantile:

$$\hat{\theta}_{0.9,r} = X_{r(\lceil 0.9\, n_r \rceil)}.$$

Suppose the resulting estimates are

$$\hat{\theta}_{0.9,1} = 12.5, \quad \hat{\theta}_{0.9,2} = 11.9, \quad \hat{\theta}_{0.9,3} = 13.1, \quad \hat{\theta}_{0.9,4} = 12.8, \quad \hat{\theta}_{0.9,5} = 12.0.$$

Point estimate of the quantile:

The across-replications sample mean is

$$\bar{\theta}_{0.9} = \frac{1}{5}\,(12.5 + 11.9 + 13.1 + 12.8 + 12.0) = 12.46.$$

Thus, our point estimate of the 90th percentile is $\bar{\theta}_{0.9} \approx 12.46$.

Sample variance across replications:

Compute the sample variance of the replication-level quantile estimates:

$$S^2_{\theta_{0.9}} = \frac{1}{R-1}\sum_{r=1}^{5}\left(\hat{\theta}_{0.9,r} - \bar{\theta}_{0.9}\right)^2 \approx 0.263,$$

so that

$$S_{\theta_{0.9}} \approx 0.51.$$

Confidence interval:

For a 95% confidence interval, we take $\alpha = 0.05$ and use the t-quantile with $R-1=4$ degrees of freedom,

$$t_{0.025,4} \approx 2.776.$$

The half-width of the confidence interval is

$$\mathrm{H} = t_{0.025,4}\frac{S_{\theta_{0.9}}}{\sqrt{R}} \approx 2.776 \times \frac{0.51}{\sqrt{5}} \approx 0.64.$$

Therefore, the 95% confidence interval for $\theta_{0.9}$ is

$$\boxed{\left[\bar{\theta}_{0.9} - \mathrm{H},\ \bar{\theta}_{0.9} + \mathrm{H}\right] = [\,12.46 - 0.64,\ 12.46 + 0.64\,] = [\,11.82,\ 13.10\,].}$$

Now, let's come to answer the second question raised before in Learning point 3: if we have found that $\hat{p}$ percent (e.g., 85%) of customers' waiting time in the queue is 310 seconds, what is the confidence interval for this probability?

Given independent replications $Y_1, \ldots, Y_R$, we use the following formula to estimate the confidence interval for the probability:

$$\hat{p} \pm z_{1-\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{(R-1)}}$$

where the value of $z_{1-\alpha/2}$ could be found in the standard normal cumulative probability table listed in Chapter 5.

Warning:

It is possible that the computed upper confidence bound exceeds 1, or that the computed lower bound becomes negative. Since a true probability must satisfy $0 \le p \le 1$, the final reported confidence interval should be *truncated* to lie within the feasible region. This truncation is standard practice in simulation analysis: confidence interval formulas based on asymptotic normality may produce bounds outside $[0, 1]$, but the final reported values should respect the fact that probabilities cannot exceed 1 or fall below 0. The same principle applies to other performance measures:

- Compute the CI as usual.
- Truncate the bounds to the feasible space.

A more difficult question is estimating a probability (or quantile) from *summary data*. When all we have is the sample mean and the confidence-interval halfwidth, one approach is to use a normal approximation for the probabilities (or quantiles) we desire. To be more specific,

$$Pr\{\bar{Y}_{i\cdot} \le c\} \approx Pr\{Z \le \frac{c - \bar{Y}_{\cdot\cdot}}{S}\}$$

$$\hat{\theta} \approx \bar{Y}_{\cdot\cdot} + z_p S$$

Remark 8.3. Using summary data to estimate a probability (or quantile) has important implications in data science, e.g., privacy-related research. For instance, given summary data regarding the salary (mean, standard deviation), we may estimate the percentage of persons whose salary is higher than a specified value.

Example 8.4. **Inferring a Quantile from Summary Data:** Suppose that from a simulation experiment we only have the following summary information about a performance measure Y based on $R = 25$ independent replications:

- sample mean: $\bar{Y}_{\cdot\cdot} = 50.0$,
- halfwidth of a 95% confidence interval: $H = 4.0$,

The reported 95% confidence interval is

$$\bar{Y}_{\cdot\cdot} \pm H = 50.0 \pm 4.0,$$

Since $H = t_{0.025,R-1}\frac{S}{\sqrt{R}}$, we can solve for the sample standard deviation S:

$$S = \frac{H}{t_{0.025,R-1}}\sqrt{R} = \frac{4.0}{2.064} \times \sqrt{25} = 1.94 \times 5 = 9.70.$$

Note that $\frac{S}{\sqrt{R}} = \frac{H}{t_{0.025,R-1}}$ is also referred to as ***standard error*** of the sample mean in the literature. In other words,

$$\boxed{CI = \textit{sample estimate} \pm (\textit{critical value} \times \textit{standard error}).}$$

because $H = t_{\alpha/2,R-1}\frac{S}{\sqrt{R}}$ where $t_{\alpha/2,R-1}$ is referred as critical value.

Thus, even though we do not have raw data, the CI halfwidth allows us to reconstruct an estimate of S, which can then be used to approximate quantiles under a normality assumption.

Approximating a p-Quantile

If Y is approximately normal, then the p-quantile of its distribution satisfies

$$\hat{\theta}_p \approx \bar{Y}_{\cdot\cdot} + z_p\, S.$$

For example, if $p = 0.90$, we use $z_{0.90} \approx 1.28$. Then:

$$\hat{\theta}_{0.90} \approx 50.0 + (1.28)(9.70) = 50.0 + 12.42 = 62.42.$$

Therefore, based only on the sample mean and its reported confidence interval, the estimated 90th percentile is

$$\boxed{\hat{\theta}_{0.90} \approx 62.42.}$$

Warning: This approach's accuracy depends on how well the underlying distribution is approximated by a normal distribution. If you assume a different underlying distribution, e.g., uniform, you need to use a different approach for tail estimation.

8.5 Output Analysis for Steady-State Simulation

8.5.1 Reducing Initialization Bias for Steady-State Simulation

For steady-state simulation, we care about the long-term performance, that is, the performance when time $\rightarrow \infty$. Nevertheless, we all understand that it is nearly impossible to run a simulation forever, and we must stop at some point. The question is: how long should we run the simulation? The answer to this question is addressed from a different and more practical angle: **how can we alleviate the bias caused by the initial condition of the simulation?**

Different strategies have been proposed to address the initialization bias, including for example:

- Warm-up period: Allow the simulation to run for a warm-up period, during which the system moves from the initial state to a state closer to the steady-state. After this warm-up period, start collecting data for analysis. This can help in reducing the initialization bias as the system is allowed to stabilize before data collection.
- Analytical methods: Use analytical methods, such as queuing theory, to **estimate** the steady-state behavior of the system or a similar system. These estimates can then be used to guide the choice of initial conditions for the simulation, reducing initialization bias.
- Multiple initial conditions: Run multiple simulations with different initial conditions to observe the behavior of the system under varying starting points. Averaging the results across these simulations can help reduce the effect of initialization bias. (**Cutoff based on multiple replications**)
- Cumulative averaging and truncation: calculating the cumulative average of a performance metric over time. Identify the cutoff point after which the system is stable. Discard the output data before this cutoff point. ((**Cutoff based on one replication**)

Remember that the appropriate method to control initialization bias will depend on the nature of the system being simulated and the specific requirements of your simulation study. Combining multiple approaches may sometimes be necessary to achieve accurate and reliable steady-state simulation results.

8.5.2 Replication Method vs. Batch Mean Method

The replication method is similar to that used in terminating simulation, except that we delete data during the warm-up period in each replication.

One disadvantage of the replication method is that some data must be deleted on each replication, and thus it wastes simulation time. One way to avoid this is to use the batch mean method, which divides the output data from one replication, after deleting the data during the warmup period, into a few large batches. Then the batches are treated as the output as if they were generated by multiple independent runs.

The batch mean method deletes fewer data, but the replication method usually creates more reliable results.

8.5.3 Final Summary for Steady-State Simulation Analysis

While we have introduced different ways of running and analyzing steady-state simulation, the following steps are commonly used in practice:

1. Use a method to alleviate the problem of initialization bias, e.g., collect data only after the warm-up period. You may also need to save the data and related figures (e.g., Fig. 8.5) in case you are required to justify the proper selection of the warm-up duration.
2. Use the replication method to generate data. Use the across-replication data to calculate the final result and the confidence interval.

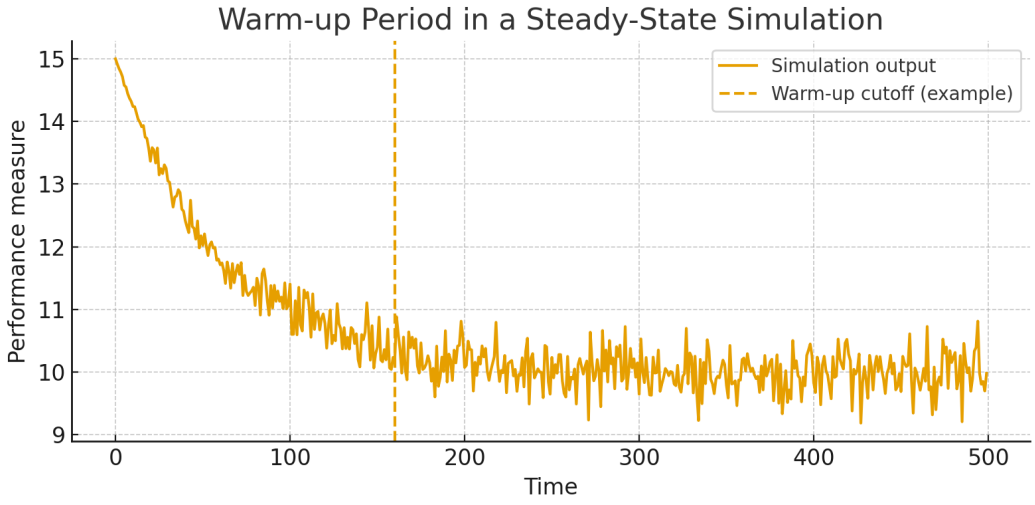


Fig. 8.5. An example figure justifying the selection of the warm-up period.

9

Estimation of Relative Performance

9.1 Comparison of Two System Designs

"Two system designs" refers to two possible configurations of a system, e.g., two possible queueing policies or two possible sets of servers in a queueing system.

For each system design, we follow the methods introduced in the previous chapter to run the simulation and calculate statistical results:

- For system design 1: we run the simulation R_1 times, record performance data $Y_{11}, Y_{21}, \ldots, Y_{R_1 1}$, based on which we obtain the final estimation: $\bar{Y}_{\cdot 1}$.
- For system design 2: we run the simulation R_2 times, record performance data $Y_{12}, Y_{22}, \ldots, Y_{R_2 2}$, based on which we obtain the final estimation: $\bar{Y}_{\cdot 2}$.

When comparing the performance of two system designs, we not only need to calculate $\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}$, but also need to calculate how confident we are about $\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}$, i.e., we need to know:

$$(\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}) \pm t_{\alpha/2,v} s.e.(\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}).$$

Next, we discuss how to calculate $s.e.(\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2})$ and how to determine the degree of freedom v. This depends on how we run the simulations: run "system design 1" and "system design 2" independently or run them with common random numbers (CRN).

9.1.1 Independent Sampling

Use the following steps to calculate $s.e.(\bar{Y}_{.1} - \bar{Y}_{.2})$:

1. Step 1: For each system $i (i = 1, 2)$, calculate the sample variance:

$$S_i^2 = \frac{1}{R_i - 1} \sum_{r=1}^{R_i} (Y_{ri} - \bar{Y}.i)^2$$

2. Step 2: Calculate $s.e.(\bar{Y}_{.1} - \bar{Y}_{.2})$:

$$s.e.(\bar{Y}_{.1} - \bar{Y}_{.2}) = \sqrt{\frac{S_1^2}{R_1} + \frac{S_2^2}{R_2}}$$

Use the following formula to determine the degree of freedom v:

$$v = \frac{(S_1^2/R_1 + S_2^2/R_2)^2}{[(S_1^2/R_1)^2/(R_1 - 1)] + [(S_2^2/R_2)^2/(R_2 - 1)]}$$

9.1.2 Run with CRN

If we run with CRN, R_1 and R_2 must be the same. Denote this value as R. Use the following steps to calculate $s.e.(\bar{Y}_{.1} - \bar{Y}_{.2})$:

1. Step 1: Calculate the differences:

$$D_r = Y_{r1} - Y_{r2},$$

where $r = 1, 2, \ldots, R$.

2. Step 2: Calculate the sample mean:

$$\bar{D} = \frac{1}{R} \sum_{r=1}^{R} D_r.$$

3. Step 3: Calculate the sample variance:

$$S_D^2 = \frac{1}{R - 1} \sum_{r=1}^{R} (D_r - \bar{D})^2.$$

4. Step 4: Calculate $s.e.(\bar{Y}_{.1} - \bar{Y}_{.2})$:

$$s.e.(\bar{Y}_{.1} - \bar{Y}_{.2}) = s.e.(\bar{D}) = \frac{S_D}{\sqrt{R}}.$$

The degree of freedom $v = R - 1$.

Important Note 1: CRN results in a smaller confidence interval than independent sampling. Using CRN when comparing the performance of two system designs is a common practice.

Important Note 2: Although the two methods use different formulas, they are *mathematically consistent* because both are based on the variance of the difference between two performance estimators. The difference lies in whether the covariance term is zero (independent sampling) or positive (CRN).

Why are the two methods mathematically consistent?

The standard error in the independent sampling case is

$$\text{s.e.}(\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}) = \sqrt{\frac{S_1^2}{R_1} + \frac{S_2^2}{R_2}}.$$

This follows from

$$\text{Var}(\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}) = \text{Var}(\bar{Y}_{\cdot 1}) + \text{Var}(\bar{Y}_{\cdot 2}),$$

because independent sampling implies

$$\text{Cov}(\bar{Y}_{\cdot 1}, \bar{Y}_{\cdot 2}) = 0.$$

Under CRN, we form paired differences

$$D_r = Y_{r1} - Y_{r2}, \qquad r = 1, \ldots, R.$$

The standard error is then

$$\text{s.e.}(\bar{D}) = \frac{S_D}{\sqrt{R}},$$

where S_D^2 estimates $\text{Var}(D_r)$.

Expanding the variance,

$$\text{Var}(D_r) = \text{Var}(Y_{r1}) + \text{Var}(Y_{r2}) - 2\ \text{Cov}(Y_{r1}, Y_{r2}).$$

CRN induces a positive covariance:

$$\text{Cov}(Y_{r1}, Y_{r2}) > 0,$$

so

$$\text{Var}(D_r) < \text{Var}(Y_{r1}) + \text{Var}(Y_{r2}),$$

which implies

$$\text{s.e.}_{\text{CRN}} < \text{s.e.}_{\text{independent}}.$$

Consistency:

Both methods rely on the same underlying variance identity:

$$\text{Var}(X - Y) = \text{Var}(X) + \text{Var}(Y) - 2\ \text{Cov}(X, Y).$$

- Independent sampling assumes Cov = 0.
- CRN estimates the full expression, including covariance, through the sample variance of the paired differences.

Thus the two standard-error formulas are mathematically consistent, but CRN typically yields a smaller standard error because it exploits positive correlation between the paired systems.

Example 9.1. **An Example: Comparing Two System Designs**

1. Independent Sampling

Suppose we perform $R_1 = R_2 = 5$ independent replications for each design. The observed performance measure (e.g., average waiting time per replication) is:

$$\text{Design 1: } Y_{11} = 9.8,\ Y_{21} = 10.1,\ Y_{31} = 9.9,\ Y_{41} = 10.3,\ Y_{51} = 9.7,$$

$$\text{Design 2: } Y_{12} = 10.4,\ Y_{22} = 10.5,\ Y_{32} = 10.2,\ Y_{42} = 10.7,\ Y_{52} = 10.3.$$

Sample Means.

$$\bar{Y}_{\cdot 1} = \frac{9.8 + 10.1 + 9.9 + 10.3 + 9.7}{5} = 9.96,$$
$$\bar{Y}_{\cdot 2} = \frac{10.4 + 10.5 + 10.2 + 10.7 + 10.3}{5} = 10.42.$$

Sample Variances.

$$S_1^2 \approx 0.058, \qquad S_2^2 \approx 0.037.$$

Difference of Means.

$$\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2} = 9.96 - 10.42 = -0.46.$$

Standard Error.

$$\text{s.e.} = \sqrt{\frac{S_1^2}{R_1} + \frac{S_2^2}{R_2}} = \sqrt{\frac{0.058}{5} + \frac{0.037}{5}} \approx 0.138.$$

Degree of Freedom.

$$\nu \approx \frac{\left(S_1^2/R_1 + S_2^2/R_2\right)^2}{\frac{(S_1^2/R_1)^2}{R_1 - 1} + \frac{(S_2^2/R_2)^2}{R_2 - 1}} \approx 7.6 \approx 8.$$

95% Confidence Interval.

Using $t_{0.025,8} = 2.31$,

$$H = 2.31 \times 0.138 \approx 0.32,$$
$$(\bar{Y}_{\cdot 1} - \bar{Y}_{\cdot 2}) \pm H = -0.46 \pm 0.32 = [-0.78,\ -0.14].$$

Conclusion: The entire interval is negative, so Design 1 has a smaller mean waiting time than Design 2.

2. Common Random Numbers (CRN)

Now, suppose each pair of replications uses the same random-number streams, as shown in Fig. 9.1. The observed paired outputs are:

$$\text{Design 1: } 9.8,\ 10.1,\ 9.9,\ 10.3,\ 9.7,$$
$$\text{Design 2: } 10.1,\ 10.3,\ 10.0,\ 10.4,\ 9.9.$$

Form Paired Differences.

$$D_r = Y_{r1} - Y_{r2},$$
$$D_1 = -0.3,\ D_2 = -0.2,\ D_3 = -0.1,\ D_4 = -0.1,\ D_5 = -0.2.$$

Mean Difference.

$$\bar{D} = \frac{-0.3 - 0.2 - 0.1 - 0.1 - 0.2}{5} = -0.18.$$

Variance of Differences.

$$S_D^2 = \frac{1}{R-1} \sum_{r=1}^{R} (D_r - \bar{D})^2 \approx 0.007.$$

Standard Error.

$$\text{s.e.} = \sqrt{\frac{S_D^2}{R}} = \sqrt{\frac{0.007}{5}} \approx 0.037.$$

95% Confidence Interval.

For CRN, degrees of freedom are $R - 1 = 4$. Using $t_{0.025,4} = 2.776$,

$$H = 2.776 \times 0.037 \approx 0.10,$$

$$\bar{D} \pm H = -0.18 \pm 0.10 = [-0.28,\ -0.08].$$

Conclusion: The interval is narrower than in the independent-sampling case, demonstrating the variance-reduction benefit of CRN. Design 1 again outperforms Design 2.

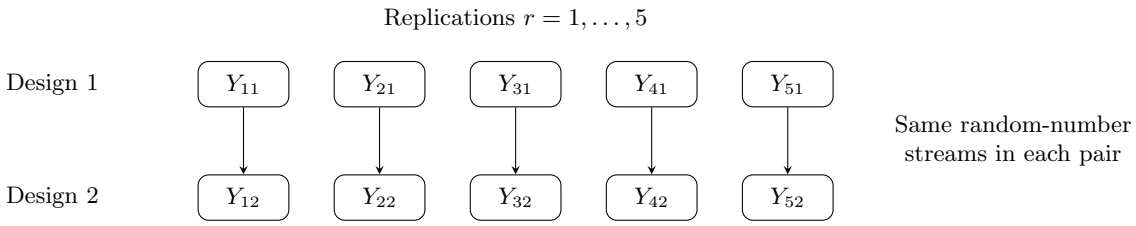


Fig. 9.1. Illustration of common random numbers (CRN): each replication of Design 1 is paired with the corresponding replication of Design 2 using the same underlying random-number streams.

9.2 Comparison of Multiple System Designs

Assume that we want to compare K alternative system designs by comparing a specific performance metric θ_i of system design $i (i = 1, 2, \ldots, K)$. The possible goals of the simulation study include:

1. Comparison of each performance measure θ_i to a benchmark θ_1 (e.g., θ_1 could represent the mean performance of existing system).
2. All pair of comparisons, $\theta_i - \theta_j$ for $i \neq j$.
3. Selection of the best θ_i (i.e., the largest or the smallest).

9.2.1 Bonferroni Approach to Multiple Comparison

The Bonferroni approach, also known as the Bonferroni correction or Bonferroni adjustment, is a method used in multiple hypothesis testing to control the family-wise error rate (FWER). The **Bonferroni inequality** states that, for any finite set of events $A_1, A_2, \ldots, A_n$:

$$P(A_1 \cup A_2 \cup \ldots \cup A_n) \leq P(A_1) + P(A_2) + \ldots + P(A_n)$$

In other words, the probability of the union of multiple events is less than or equal to the sum of their individual probabilities. This inequality holds even when the events are not independent. The Bonferroni inequality is often used in multiple hypothesis testing as the basis for the Bonferroni correction, which adjusts the significance level to account for multiple comparisons and control the family-wise error rate.

Remember that the practical meaning of confidence interval $\bar{Y} \pm H$ is that with probability no larger than α the performance of θ does not fall in the range $\bar{Y} \pm H$. If we use S_i **to denote the statement that the performance of** θ_i **does not fall in the range** $\bar{Y} \pm H$, then the Bonferroni inequality means:

$$P(\text{one or more of the } C \text{ confidence intervals does not contain the parameter being estimated}) \leq \sum_{i=1}^{C} \alpha_i$$

Denote $\alpha_E = \sum_{i=1}^{C} \alpha_i$. In practice, we use the following equivalent form of the Bonferroni inequality:

$$P(\text{all } C \text{ confidence intervals contain the parameter being estimated}) \geq 1 - \alpha_E$$

Thus, to conduct an experiment that involves making C comparisons, first select the overall error probability, say, $\alpha_E = 0.05$. The individual α_i may be chosen to be equal to α_E/C.

1. Case 1: comparison of each performance measure θ_i to a benchmark θ_1 (e.g., θ_1 could represent the mean performance of existing system). In this case, $C = K - 1$.
2. Case 2: All pair of comparisons, $\theta_i - \theta_j$ for $i \neq j$. In this case, $C = K(K-1)/2$.
3. Case 3: Selection of the best θ_i (i.e., the largest or the smallest). In this case, we use a special procedure introduced below.

Example 9.2.

Bonferroni Approach for Comparing System Designs

Assume that we have $K = 4$ system designs with unknown mean performances $\theta_1, \theta_2, \theta_3, \theta_4$. From simulation, we obtain the following sample means and standard errors (SEs) of the means:

Design i	1	2	3	4
$Y_{.i}$	10.0	10.7	9.6	10.1
SE($\bar{Y}_{.i}$)	0.224	0.268	0.179	0.246

We want to compare all pairs $\theta_i - \theta_j$ for $i \neq j$ with overall (family-wise) error probability $\alpha_E = 0.05$. The number of pairwise comparisons is

$$C = \frac{K(K-1)}{2} = \frac{4 \cdot 3}{2} = 6.$$

By the Bonferroni approach, we take

$$\alpha_i = \frac{\alpha_E}{C} = \frac{0.05}{6} \approx 0.0083.$$

Each individual two-sided confidence interval thus has confidence level

$$1 - \alpha_i \approx 1 - 0.0083 = 0.9917.$$

Assuming a large number of degrees of freedom, we use the normal critical value

$$z_{1-\alpha_i/2} \approx z_{0.9958} \approx 2.65.$$

For each pair (i, j), the estimated difference is $\hat{d}_{ij} = \bar{Y}_{\cdot i} - \bar{Y}_{\cdot j}$ with

$$\mathrm{SE}(\hat{d}_{ij}) = \sqrt{\mathrm{SE}(\bar{Y}_{\cdot i})^2 + \mathrm{SE}(\bar{Y}_{\cdot j})^2},$$

and the Bonferroni confidence interval is

$$\hat{d}_{ij} \pm z_{1-\alpha_i/2}\, \mathrm{SE}(\hat{d}_{ij}).$$

(1) Design 1 vs Design 2.

$$\hat{d}_{12} = 10.0 - 10.7 = -0.7,$$
$$\mathrm{SE}(\hat{d}_{12}) = \sqrt{0.224^2 + 0.268^2} \approx \sqrt{0.050 + 0.072} \approx \sqrt{0.122} \approx 0.349.$$

Halfwidth:

$$H_{12} = 2.65 \times 0.349 \approx 0.92.$$

CI:

$$-0.7 \pm 0.92 \;\Rightarrow\; [-1.62,\ 0.22].$$

This interval contains 0, so we cannot conclude a significant difference between Designs 1 and 2 at the family-wise level 0.05.

(2) Design 1 vs Design 3.

$$\hat{d}_{13} = 10.0 - 9.6 = 0.4,$$
$$\mathrm{SE}(\hat{d}_{13}) = \sqrt{0.224^2 + 0.179^2} \approx \sqrt{0.050 + 0.032} = \sqrt{0.082} \approx 0.286.$$
$$H_{13} = 2.65 \times 0.286 \approx 0.76,$$
$$0.4 \pm 0.76 \;\Rightarrow\; [-0.36,\ 1.16].$$

Again, 0 is inside the interval.

(3) Design 1 vs Design 4.

$$\hat{d}_{14} = 10.0 - 10.1 = -0.1, \quad \mathrm{SE}(\hat{d}_{14}) = \sqrt{0.224^2 + 0.246^2} \approx \sqrt{0.050 + 0.061} = \sqrt{0.111} \approx 0.333,$$
$$H_{14} = 2.65 \times 0.333 \approx 0.88,$$
$$-0.1 \pm 0.88 \;\Rightarrow\; [-0.98,\ 0.78].$$

(4) Design 2 vs Design 3.

$$\hat{d}_{23} = 10.7 - 9.6 = 1.1,$$
$$\mathrm{SE}(\hat{d}_{23}) = \sqrt{0.268^2 + 0.179^2} \approx \sqrt{0.072 + 0.032} = \sqrt{0.104} \approx 0.322,$$
$$H_{23} = 2.65 \times 0.322 \approx 0.85,$$
$$1.1 \pm 0.85 \;\Rightarrow\; [0.25,\ 1.95].$$

This interval is strictly positive, so Design 2 has a significantly larger mean performance than Design 3 at overall family-wise level $\alpha_E = 0.05$.

(5) Design 2 vs Design 4.

$$\hat{d}_{24} = 10.7 - 10.1 = 0.6, \quad \mathrm{SE}(\hat{d}_{24}) = \sqrt{0.268^2 + 0.246^2} \approx \sqrt{0.072 + 0.061} = \sqrt{0.133} \approx 0.365,$$
$$H_{24} = 2.65 \times 0.365 \approx 0.97,$$
$$0.6 \pm 0.97 \;\Rightarrow\; [-0.37,\ 1.57].$$

(6) Design 3 vs Design 4.

$$\hat{d}_{34} = 9.6 - 10.1 = -0.5,$$
$$\mathrm{SE}(\hat{d}_{34}) = \sqrt{0.179^2 + 0.246^2} \approx \sqrt{0.032 + 0.061} = \sqrt{0.093} \approx 0.305,$$
$$H_{34} = 2.65 \times 0.305 \approx 0.81,$$
$$-0.5 \pm 0.81 \;\Rightarrow\; [-1.31,\ 0.31].$$

Conclusion Drawn from Simulation Study.

Among the six Bonferroni-adjusted confidence intervals, only the interval for $(\theta_2 - \theta_3)$ is strictly positive. Thus, at overall family-wise error level $\alpha_E = 0.05$, we can conclude that Design 2 has a significantly larger mean performance than Design 3. For the other pairs, we cannot claim significant differences between those designs under the Bonferroni criterion.

9.2.2 Selection of the Best

Selection of the best usually involves two stages: In stage 1, those systems that are significantly inferior to the others are filtered out. If there is more than one system after stage 1, we conduct more replications (determined by a function of Rinott's constant) and make the final decision based on the final sample mean.

In the two-stage procedure, we need to use Rinott's constant to determine the extra number of replications required in the second stage. As a piece of background knowledge, Rinott's constant is a parameter used in the context of sequential sampling in the field of statistics, specifically in the Two-Stage Group Sequential Procedure. The Two-Stage Group Sequential Procedure is a technique used in experimental designs, where the goal is to minimize the total number of observations required to achieve a predetermined level of statistical significance. The Rinott's constant is used to determine the minimum sample size required during the second stage of the procedure.

The value of Rinott's constant, h, can be obtained from a table (listed at the end of this chapter) or calculated using a mathematical formula or numerical methods. Python statistical packages such as scipy.stats also include a function to calculate h. The constant itself does not have any direct interpretation; rather, it is a scaling factor that ensures the two-stage procedure maintains the desired significance level.

Note that the value of ϵ in the algorithm depends on the specific application under study. Also, note that in the second stage, we no longer estimate the confidence intervals but directly use the overall sample mean.

Below is the two-stage procedure for selecting the best out of K system designs.

Stage 1: Filter out clear losers:

1. Select an α such that $\frac{1}{K} < 1 - \alpha < 1$, the practically significant difference $\epsilon > 0$, an initial number of replications R_0.
2. Conduct R_0 replications of each system (CRN preferred). Calculate the first-stage sample means and sample variances,

$$\bar{Y}_{\cdot i} = \frac{1}{R_0} \sum_{r=1}^{R_0} Y_{ri}$$

$$S_i^2 = \frac{1}{R_0 - 1} \sum_{r=1}^{R_0} (Y_{ri} - \bar{Y}_{\cdot i})^2$$

   for $i = 1, 2, \ldots, K$.
3. Calculate the filtering thresholds:

$$W_{ij} = t \sqrt{\frac{S_i^2 + S_j^2}{R_0}}$$

   for all $i \neq j$, where $t = t_{1-(1-\alpha/2)^{\frac{1}{K-1}}, R_0 - 1}$. In other words, we apply Bonferroni's approach when setting the confidence level (i.e., $(1 - \alpha/2)^{\frac{1}{K-1}}$) here.
4. Filtering out obvious losers:
   - If bigger is better, then form the survivor subset S containing every system design i such that

$$\bar{Y}_{\cdot i} \geq \bar{Y}_{\cdot j} - \max(0, W_{ij} - \epsilon) \text{ for all } i \neq j.$$

     That is, designs that are significantly smaller than others are losers and are dropped out.
   - If smaller is better, then form the survivor subset S containing every system design i such that

$$\bar{Y}_{\cdot i} \leq \bar{Y}_{\cdot j} + \max(0, W_{ij} - \epsilon) \text{ for all } i \neq j.$$

     That is, designs that are significantly larger than others are losers and are dropped out.

Stage 2: Select the final winner:

1. If S only includes one system, then stop and select this system as the final winner. Otherwise, for all design i in S, compute the replications that we need to conduct:

$$R_1 = \max(R_0, \lceil (hS_i/\epsilon)^2 \rceil)$$

where h is Rinott's constant, which could be found either in a table or with Python code.
2. Conduct $R_1 - R_0$ extra replications for all designs i in S (CRN preferred).
3. Compute the overall sample means for all designs i in S:

$$\bar{Y}_{\cdot i} = \frac{1}{R_1} \sum_{r=1}^{R_1} Y_{ri}$$

4. Final winner: if bigger is better, then select the system with the largest $\bar{Y}_{\cdot i}$; if smaller is better, then select the system with the smallest $\bar{Y}_{\cdot i}$.

Remark 9.3. The above procedure is from a book [2]. In Step 3 of Stage 1, however, we can use $\frac{S_D}{\sqrt{R_0}}$ (as defined in Section 9.1.2 to replace $\sqrt{\frac{S_i^2+S_j^2}{R_0}}$ when calculating W_{ij} if the CRN method is used.

Example 9.4.

Selecting the Best of $K = 4$ Designs

Assume that bigger is better. Assume the following parameters:

- **Confidence Level $(1-\alpha)$:** 0.95 (i.e., $\alpha = 0.05$)
- **Initial Replications (R_0):** 10
- **Practically Significant Difference (ϵ):** 0.20
- **Critical t-value (t):** $t = t_{1-(1-\alpha/2)^{\frac{1}{K-1}}, R_0-1} = t_{1-0.975^{\frac{1}{3}},9} = t_{0.0084,9} = \mathbf{2.928}$ (Used in W_{ij} for $R_0 - 1 = 9$ degrees of freedom)

Stage 1: Filter Out Clear Losers

First-Stage Sample Means and Variances (Hypothetical)

Let's assume the following output results:

Table 9.1. First-Stage Results ($R_0 = 10$)

System (i)	$\overline{Y}_{\cdot i}$	S_i^2
1	10.0	0.80
2	10.5	0.90
3	11.6	0.70
4	12.0	0.80

Calculate Filtering Thresholds (W_{ij})

$W_{ij} = t\sqrt{\frac{S_i^2+S_j^2}{R_0}}$. Using $t = 2.928$:

- $W_{14} = 2.928\sqrt{\frac{0.80+0.80}{10}} = 2.928 \times 0.40 = \mathbf{1.1712}$
- $W_{24} = 2.928\sqrt{\frac{0.90+0.80}{10}} \approx \mathbf{1.208}$
- $W_{34} = 2.928\sqrt{\frac{0.70+0.80}{10}} \approx \mathbf{1.135}$

Filtering Out Obvious Losers

Survival condition: $\overline{Y}_{.i} \geq \overline{Y}_{.4} - \max(0, W_{i4} - \epsilon)$, where $\overline{Y}_{.4} = 12.0$ and $\epsilon = 0.20$.

Table 9.2. Filtering Check (Survival Barrier: $12.0 - \max(0, W_{i4} - 0.20)$)

System (i)	$\overline{Y}_{.i}$	$W_{i4} - 0.20$	Survival Barrier	$\overline{Y}_{.i} \geq$ Barrier?
1	10.0	0.9712	11.0288	**NO** (Failed)
2	10.5	1.008	10.992	**NO** (Failed)
3	11.6	0.935	11.065	**YES** (Survived)
4	12.0	-	-	**YES** (Survived)

The survivor subset remains $\mathbf{S} = \{\mathbf{3}, \mathbf{4}\}$.

Stage 2: Select the Final Winner

Compute Total Required Replications (R_1)

The required replications depend only on R_0, ϵ, and Rinott's constant $h = 3.476$.

Table 9.3. Required Replications Calculation ($h = 3.476$)

System (i)	S_i	$\lceil (hS_i/\epsilon)^2 \rceil$	$R_1 = \max(10, \dots)$
3	$\sqrt{0.70} \approx 0.837$	211	**211**
4	$\sqrt{0.80} \approx 0.894$	242	**242**

Final Winner Selection

For designs 3 and 4, perform their corresponding extra number of runs, $R_1 - R_0$. Assuming the final overall means are computed:

Table 9.4. Final Overall Sample Means

System (i)	Overall $\overline{Y}_{.i}$
3	11.75
4	12.05

Since Bigger is Better, we select the system with the largest overall sample mean:

$$\max(11.75, 12.05) = 12.05$$

Conclusion: The system with the largest overall sample mean is **System 4**.

Values of Rinott's Constant h

$1-\alpha$	R_0	K								
		2	3	4	5	6	7	8	9	10
0.90	5	2.291	3.058	3.511	3.837	4.093	4.305	4.486	4.644	4.786
	6	2.177	2.871	3.270	3.552	3.771	3.951	4.103	4.235	4.352
	7	2.107	2.758	3.126	3.384	3.582	3.744	3.881	3.999	4.103
	8	2.059	2.682	3.031	3.273	3.459	3.609	3.736	3.845	3.941
	9	2.025	2.628	2.963	3.195	3.372	3.515	3.635	3.738	3.829
	10	1.999	2.587	2.913	3.137	3.307	3.445	3.560	3.659	3.746
	11	1.978	2.556	2.874	3.092	3.258	3.391	3.503	3.598	3.682
	12	1.962	2.531	2.843	3.056	3.218	3.349	3.457	3.551	3.632
	13	1.948	2.510	2.817	3.027	3.186	3.314	3.420	3.512	3.592
	14	1.937	2.493	2.796	3.003	3.160	3.285	3.390	3.480	3.558
	15	1.928	2.479	2.779	2.983	3.138	3.261	3.364	3.453	3.530
	16	1.919	2.467	2.764	2.966	3.119	3.241	3.343	3.430	3.506
	17	1.912	2.456	2.751	2.951	3.102	3.223	3.324	3.410	3.485
	18	1.906	2.447	2.739	2.938	3.088	3.208	3.308	3.393	3.467
	19	1.901	2.438	2.729	2.926	3.075	3.194	3.293	3.378	3.451
	20	1.896	2.431	2.720	2.916	3.064	3.182	3.280	3.364	3.437
	30	1.866	2.387	2.666	2.855	2.997	3.110	3.204	3.284	3.354
	40	1.852	2.366	2.641	2.827	2.966	3.077	3.169	3.247	3.315
	50	1.844	2.354	2.627	2.810	2.948	3.057	3.148	3.225	3.292
0.95	5	3.107	3.905	4.390	4.744	5.025	5.259	5.461	5.638	5.797
	6	2.910	3.602	4.010	4.303	4.533	4.722	4.884	5.025	5.150
	7	2.791	3.424	3.791	4.051	4.253	4.419	4.559	4.681	4.789
	8	2.712	3.308	3.649	3.889	4.074	4.225	4.353	4.463	4.561
	9	2.656	3.226	3.550	3.776	3.950	4.091	4.210	4.313	4.404
	10	2.614	3.166	3.476	3.693	3.859	3.993	4.106	4.204	4.290
	11	2.582	3.119	3.420	3.629	3.789	3.918	4.027	4.121	4.203
	12	2.556	3.082	3.376	3.579	3.734	3.860	3.965	4.055	4.135
	13	2.534	3.052	3.340	3.539	3.690	3.812	3.915	4.003	4.080
	14	2.517	3.027	3.310	3.505	3.654	3.773	3.874	3.960	4.035
	15	2.502	3.006	3.285	3.477	3.623	3.741	3.839	3.924	3.998
	16	2.489	2.988	3.264	3.453	3.597	3.713	3.810	3.893	3.966
	17	2.478	2.973	3.246	3.433	3.575	3.689	3.785	3.867	3.938
	18	2.468	2.959	3.230	3.415	3.556	3.669	3.763	3.844	3.914
	19	2.460	2.948	3.216	3.399	3.539	3.650	3.744	3.824	3.894
	20	2.452	2.937	3.203	3.385	3.523	3.634	3.727	3.806	3.875
	30	2.407	2.874	3.129	3.303	3.434	3.539	3.626	3.701	3.766
	40	2.386	2.845	3.094	3.264	3.392	3.495	3.580	3.652	3.716
	50	2.373	2.828	3.074	3.242	3.368	3.469	3.553	3.624	3.687

10

Extra Content 1: Simulation Software

Please read Chapter 3, Simulation Software, in the textbook [1]. Because this course is cross-listed for **fourth-year** undergraduates and **graduate** students, we will not provide detailed lectures on each individual simulation tool. Instead, students are expected to be capable of self-learning and exploring these tools independently.

As a recommendation, AnyLogic is a widely used general-purpose simulation platform. Self-learning materials for AnyLogic are available on Brightspace. You may choose to use simulation software for your course project, which can significantly reduce the amount of coding required but requires extra time to be familiar with the tool. Pay attention to the following two notes:

- Using such software is completely optional.
- When you learn AnyLogic, keep in mind that this general-purpose simulator is capable of performing different types of simulations, including discrete-event simulation, agent simulation, system dynamics, and Monte Carlo simulation. **We only care about discrete-event simulation in this course.**

In addition to general-purpose tools like AnyLogic, there are also special-purpose simulators, such as Mininet, OMNeT++, and NS-3, which are popular simulators for computer networks.

11

Extra Content 2: Simulation in Machine Learning

Simulation has been extensively used in machine learning. This chapter uses Reinforcement learning (RL) as the application background to illustrate the usage of simulation. As a nutshell, RL involves an agent interacting with an environment in order to learn an optimal decision-making policy. In many real-world problems (inventory control, queuing networks, traffic systems, robot manipulation, and dynamic scheduling), a closed-form model of the environment is unavailable. Simulation provides a practical tool for generating trajectories, evaluating policies, and driving RL algorithms when analytical models cannot be constructed.

Below, we first provide an overview of how simulation is used in RL. We then provide two simple examples: two-stage queueing control and Gridworld.

11.1 Why Simulation Is Needed in Reinforcement Learning

In RL, an agent repeatedly observes a state s_t, takes an action a_t, and receives a reward r_t, while the environment transitions to a new state s_{t+1}. To compute expectations such as

$$E[r_t \mid s_t, a_t], \qquad E[V(s_{t+1}) \mid s_t, a_t],$$

we usually need the transition probabilities $P(s_{t+1} \mid s_t, a_t)$. However, for many complex systems, the dynamics are *unknown but simulatable*.

Simulation allows the RL agent to "sample" transitions:

$$(s_t, a_t) \longrightarrow (r_t, s_{t+1}),$$

thus enabling model-free algorithms such as Q-learning and deep RL.

The simulation–RL Interaction Loop can be found in Figure 11.1

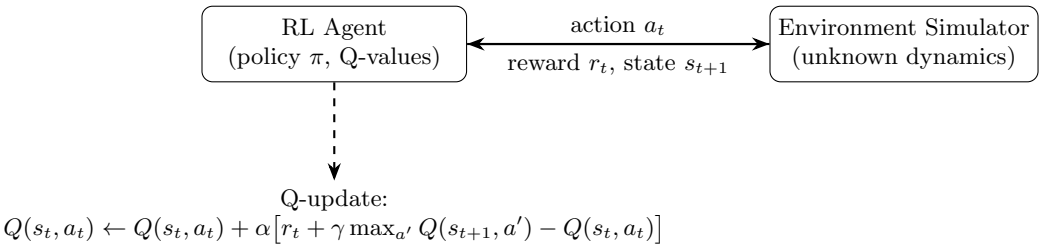


Fig. 11.1. Simulation–RL interaction loop.

11.2 Q-Learning Using Simulation

Q-learning estimates the action-value function

$$Q(s,a) = E\Big[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,\Big|\, s_t = s,\, a_t = a\Big],$$

using the update rule

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\Big[r_t + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)\Big].$$

A simulator supplies (r_t, s_{t+1}) for each executed action.

Below, we use two concrete examples to show how simulation is used in Q-learning.

Queueing System Control

Consider a simple queueing system with two possible states:

$$S = \{0,1\},$$

where state 0 means the system is empty and state 1 means a customer is waiting. The agent can choose an action:

$$A = \{\text{serve quickly},\ \text{serve slowly}\}.$$

Suppose the environment simulator behaves as follows:

- If the agent serves quickly, service costs more energy but clears the queue with high probability, e.g., 0.9
- If the agent serves slowly, service is cheaper but clears the queue with high probability, e.g, 0.6
- A new customer arrives with probability 0.4 each time step.

We define the one-step cost as

$$c(s_t,a_t) = h \cdot \mathbf{1}\{\text{queue nonempty at time } t\} + g(a_t),$$

where $h > 0$ is a holding (waiting) cost and $g(a)$ is an action cost (e.g., energy cost), where $g(\text{quick}) > g(\text{slow})$.

The immediate reward is then

$$r_t = -c(s_t,a_t),$$

The objective of the reinforcement-learning agent is to learn a policy π that minimizes the long-run expected total cost, or equivalently maximizes the discounted reward:

$$\pi^\star = \arg\max_\pi E_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] = \arg\min_\pi E_\pi\left[\sum_{t=0}^{\infty} \gamma^t c(s_t,a_t)\right].$$

Simulation Step

Let the discount factor be $\gamma = 0.9$ and the learning rate $\alpha = 0.2$. At each time step, given the current state s_t and chosen action a_t:

1. The simulator draws a Bernoulli outcome for whether the service succeeds.
2. The queue becomes empty or remains full based on the drawn service outcome.
3. A Bernoulli(0.4) arrival is simulated to determine whether a new job arrives.
4. The simulator computes the resulting next state s_{t+1} and reward r_t.

This generates the sample transition (s_t, a_t, r_t, s_{t+1}) for the Q-learning update.

Numerical Example for One Update

Suppose at time t:

$$s_t = 1, \qquad a_t = \text{serve slowly}.$$

Assume the simulator produces the following outcomes:

$$\text{service succeeds with probability } 0.6, \qquad \text{arrival occurs with probability } 0.4.$$

Simulated outcome:

$$\text{service succeeds (empty queue)}, \quad \text{new arrival occurs},$$

so the next state is

$$s_{t+1} = 1.$$

Assume that reward calculated based on $r_t = -c(s_t, a_t)$ is

$$r_t = -1.$$

Assume the current Q-table contains:

$$Q(1, \text{slow}) = -2.0, \quad Q(1, \text{quick}) = -1.0, \quad Q(0, \text{slow}) = -0.2, \quad Q(0, \text{quick}) = -0.1.$$

The Q-update is

$$Q(1, \text{slow}) \leftarrow -2.0 + 0.2\Big(-1 + 0.9 \max\{-2.0, -1.0\} + 2.0\Big).$$

Since $\max\{-2.0, -1.0\} = -1.0$,

$$Q(1, \text{slow}) \leftarrow -2.0 + 0.2\big(-1 + 0.9(-1) + 2\big) = -2.0 + 0.2(-1 - 0.9 + 2) = -2.0 + 0.2(0.1) = -1.98.$$

Thus Q-learning (driven entirely by simulation) updates the action value slightly upward:

$$\boxed{Q(1, \text{slow}) = -1.98.}$$

After running the simulation for many steps, the values in the Q-table will converge to the stable values, based on which we can obtain the optimal control policy.

5. Gridworld Q-Learning

Gridworld Environment

We consider a 3×3 Gridworld:

$$S = \{(i, j) : 1 \le i, j \le 3\},$$

start at $(1, 1)$ and goal at $(3, 3)$. Actions:

$$A = \{\text{Up, Down, Left, Right}\}.$$

Rewards:

$$r_t = \begin{cases} +10, & s_{t+1} = (3, 3), \\ -1, & \text{otherwise}. \end{cases}$$

Dynamics:

- move succeeds with probability 0.8,
- agent stays in place with probability 0.2,
- hitting a wall keeps the agent in place.

S $(1,1)$	$(1,2)$	$(1,3)$
$(2,1)$	$(2,2)$	$(2,3)$
$(3,1)$	$(3,2)$	$(3,3)$ G

Fig. 11.2. A 3×3 Gridworld. S$(1,1)$ is the start; G$(3,3)$ is the goal.

Example Simulated Q-Update

Suppose the agent is at

$$s_t = (2,2), \quad a_t = \text{Right}.$$

The simulator generates:

$$r_t = -1, \qquad s_{t+1} = (2,3) \quad \text{(movement succeeds)}.$$

Assume current Q-values satisfy

$$Q((2,2), \text{Right}) = 2.5, \qquad \max_{a'} Q((2,3), a') = 3.0.$$

With $\alpha = 0.1$, $\gamma = 0.95$:

$$\begin{aligned} Q((2,2), \text{Right}) &\leftarrow 2.5 + 0.1\,[-1 + 0.95(3.0) - 2.5] \\ &= 2.5 + 0.1(-0.65) = 2.435. \end{aligned}$$

Thus the simulation-driven update yields:

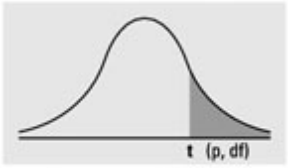
$$\boxed{Q((2,2), \text{Right}) = 2.435.}$$

After running the simulation for many steps, the values in the Q-table will converge to the optimal values, which can be used to find the optimal path from the start state to the goal state.

References

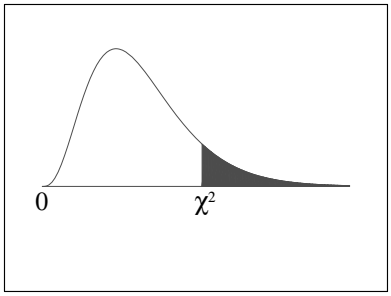
1. A. M. Law, *Simulation modeling and analysis (6e)*. Mcgraw-hill New York, 2024.
2. J. Banks, J. S. CARSON II, L. Barry, and D. Nicol, *Discrete-event system simulation (5e)*. Pearson, 2009.

Numbers in each row of the table are values on a *t*-distribution with (*df*) degrees of freedom for selected right-tail (greater-than) probabilities (*p*).



df/p	0.40	0.25	0.10	0.05	0.025	0.01	0.005	0.0005
1	0.324920	1.000000	3.077684	6.313752	12.70620	31.82052	63.65674	636.6192
2	0.288675	0.816497	1.885618	2.919986	4.30265	6.96456	9.92484	31.5991
3	0.276671	0.764892	1.637744	2.353363	3.18245	4.54070	5.84091	12.9240
4	0.270722	0.740697	1.533206	2.131847	2.77645	3.74695	4.60409	8.6103
5	0.267181	0.726687	1.475884	2.015048	2.57058	3.36493	4.03214	6.8688
6	0.264835	0.717558	1.439756	1.943180	2.44691	3.14267	3.70743	5.9588
7	0.263167	0.711142	1.414924	1.894579	2.36462	2.99795	3.49948	5.4079
8	0.261921	0.706387	1.396815	1.859548	2.30600	2.89646	3.35539	5.0413
9	0.260955	0.702722	1.383029	1.833113	2.26216	2.82144	3.24984	4.7809
10	0.260185	0.699812	1.372184	1.812461	2.22814	2.76377	3.16927	4.5869
11	0.259556	0.697445	1.363430	1.795885	2.20099	2.71808	3.10581	4.4370
12	0.259033	0.695483	1.356217	1.782288	2.17881	2.68100	3.05454	43178
13	0.258591	0.693829	1.350171	1.770933	2.16037	2.65031	3.01228	4.2208
14	0.258213	0.692417	1.345030	1.761310	2.14479	2.62449	2.97684	4.1405
15	0.257885	0.691197	1.340606	1.753050	2.13145	2.60248	2.94671	4.0728
16	0.257599	0.690132	1.336757	1.745884	2.11991	2.58349	2.92078	4.0150
17	0.257347	0.689195	1.333379	1.739607	2.10982	2.56693	2.89823	3.9651
18	0.257123	0.688364	1.330391	1.734064	2.10092	2.55238	2.87844	3.9216
19	0.256923	0.687621	1.327728	1.729133	2.09302	2.53948	2.86093	3.8834
20	0.256743	0.686954	1.325341	1.724718	2.08596	2.52798	2.84534	3.8495
21	0.256580	0.686352	1.323188	1.720743	2.07961	2.51765	2.83136	3.8193
22	0.256432	0.685805	1.321237	1.717144	2.07387	2.50832	2.81876	3.7921
23	0.256297	0.685306	1.319460	1.713872	2.06866	2.49987	2.80734	3.7676
24	0.256173	0.684850	1.317836	1.710882	2.06390	2.49216	2.79694	3.7454
25	0.256060	0.684430	1.316345	1.708141	2.05954	2.48511	2.78744	3.7251
26	0.255955	0.684043	1.314972	1.705618	2.05553	2.47863	2.77871	3.7066
27	0.255858	0.683685	1.313703	1.703288	2.05183	2.47266	2.77068	3.6896
28	0.255768	0.683353	1.312527	1.701131	2.04841	2.46714	2.76326	3.6739
29	0.255684	0.683044	1.311434	1.699127	2.04523	2.46202	2.75639	3.6594
30	0.255605	0.682756	1.310415	1.697261	2.04227	2.45726	2.75000	3.6460
z	0.253347	0.674490	1.281552	1.644854	1.95996	2.32635	2.57583	3.2905
CI	———	———	80%	90%	95%	98%	99%	99.9%

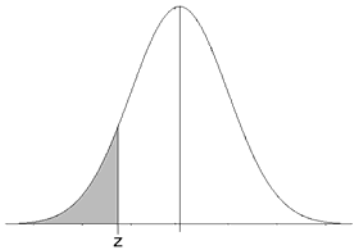
Chi-Square Distribution Table



The shaded area is equal to α for $\chi^2 = \chi^2_\alpha$.

df	$\chi^2_{.995}$	$\chi^2_{.990}$	$\chi^2_{.975}$	$\chi^2_{.950}$	$\chi^2_{.900}$	$\chi^2_{.100}$	$\chi^2_{.050}$	$\chi^2_{.025}$	$\chi^2_{.010}$	$\chi^2_{.005}$
1	0.000	0.000	0.001	0.004	0.016	2.706	3.841	5.024	6.635	7.879
2	0.010	0.020	0.051	0.103	0.211	4.605	5.991	7.378	9.210	10.597
3	0.072	0.115	0.216	0.352	0.584	6.251	7.815	9.348	11.345	12.838
4	0.207	0.297	0.484	0.711	1.064	7.779	9.488	11.143	13.277	14.860
5	0.412	0.554	0.831	1.145	1.610	9.236	11.070	12.833	15.086	16.750
6	0.676	0.872	1.237	1.635	2.204	10.645	12.592	14.449	16.812	18.548
7	0.989	1.239	1.690	2.167	2.833	12.017	14.067	16.013	18.475	20.278
8	1.344	1.646	2.180	2.733	3.490	13.362	15.507	17.535	20.090	21.955
9	1.735	2.088	2.700	3.325	4.168	14.684	16.919	19.023	21.666	23.589
10	2.156	2.558	3.247	3.940	4.865	15.987	18.307	20.483	23.209	25.188
11	2.603	3.053	3.816	4.575	5.578	17.275	19.675	21.920	24.725	26.757
12	3.074	3.571	4.404	5.226	6.304	18.549	21.026	23.337	26.217	28.300
13	3.565	4.107	5.009	5.892	7.042	19.812	22.362	24.736	27.688	29.819
14	4.075	4.660	5.629	6.571	7.790	21.064	23.685	26.119	29.141	31.319
15	4.601	5.229	6.262	7.261	8.547	22.307	24.996	27.488	30.578	32.801
16	5.142	5.812	6.908	7.962	9.312	23.542	26.296	28.845	32.000	34.267
17	5.697	6.408	7.564	8.672	10.085	24.769	27.587	30.191	33.409	35.718
18	6.265	7.015	8.231	9.390	10.865	25.989	28.869	31.526	34.805	37.156
19	6.844	7.633	8.907	10.117	11.651	27.204	30.144	32.852	36.191	38.582
20	7.434	8.260	9.591	10.851	12.443	28.412	31.410	34.170	37.566	39.997
21	8.034	8.897	10.283	11.591	13.240	29.615	32.671	35.479	38.932	41.401
22	8.643	9.542	10.982	12.338	14.041	30.813	33.924	36.781	40.289	42.796
23	9.260	10.196	11.689	13.091	14.848	32.007	35.172	38.076	41.638	44.181
24	9.886	10.856	12.401	13.848	15.659	33.196	36.415	39.364	42.980	45.559
25	10.520	11.524	13.120	14.611	16.473	34.382	37.652	40.646	44.314	46.928
26	11.160	12.198	13.844	15.379	17.292	35.563	38.885	41.923	45.642	48.290
27	11.808	12.879	14.573	16.151	18.114	36.741	40.113	43.195	46.963	49.645
28	12.461	13.565	15.308	16.928	18.939	37.916	41.337	44.461	48.278	50.993
29	13.121	14.256	16.047	17.708	19.768	39.087	42.557	45.722	49.588	52.336
30	13.787	14.953	16.791	18.493	20.599	40.256	43.773	46.979	50.892	53.672
40	20.707	22.164	24.433	26.509	29.051	51.805	55.758	59.342	63.691	66.766
50	27.991	29.707	32.357	34.764	37.689	63.167	67.505	71.420	76.154	79.490
60	35.534	37.485	40.482	43.188	46.459	74.397	79.082	83.298	88.379	91.952
70	43.275	45.442	48.758	51.739	55.329	85.527	90.531	95.023	100.425	104.215
80	51.172	53.540	57.153	60.391	64.278	96.578	101.879	106.629	112.329	116.321
90	59.196	61.754	65.647	69.126	73.291	107.565	113.145	118.136	124.116	128.299
100	67.328	70.065	74.222	77.929	82.358	118.498	124.342	129.561	135.807	140.169

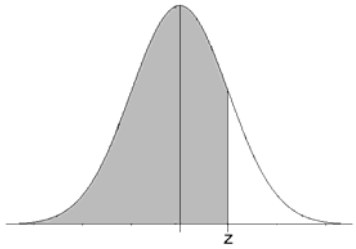
Standard Normal Cumulative Probability Table



Cumulative probabilities for NEGATIVE z-values are shown in the following table:

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
-3.4	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0002
-3.3	0.0005	0.0005	0.0005	0.0004	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
-3.2	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006	0.0006	0.0005	0.0005	0.0005
-3.1	0.0010	0.0009	0.0009	0.0009	0.0008	0.0008	0.0008	0.0008	0.0007	0.0007
-3.0	0.0013	0.0013	0.0013	0.0012	0.0012	0.0011	0.0011	0.0011	0.0010	0.0010
-2.9	0.0019	0.0018	0.0018	0.0017	0.0016	0.0016	0.0015	0.0015	0.0014	0.0014
-2.8	0.0026	0.0025	0.0024	0.0023	0.0023	0.0022	0.0021	0.0021	0.0020	0.0019
-2.7	0.0035	0.0034	0.0033	0.0032	0.0031	0.0030	0.0029	0.0028	0.0027	0.0026
-2.6	0.0047	0.0045	0.0044	0.0043	0.0041	0.0040	0.0039	0.0038	0.0037	0.0036
-2.5	0.0062	0.0060	0.0059	0.0057	0.0055	0.0054	0.0052	0.0051	0.0049	0.0048
-2.4	0.0082	0.0080	0.0078	0.0075	0.0073	0.0071	0.0069	0.0068	0.0066	0.0064
-2.3	0.0107	0.0104	0.0102	0.0099	0.0096	0.0094	0.0091	0.0089	0.0087	0.0084
-2.2	0.0139	0.0136	0.0132	0.0129	0.0125	0.0122	0.0119	0.0116	0.0113	0.0110
-2.1	0.0179	0.0174	0.0170	0.0166	0.0162	0.0158	0.0154	0.0150	0.0146	0.0143
-2.0	0.0228	0.0222	0.0217	0.0212	0.0207	0.0202	0.0197	0.0192	0.0188	0.0183
-1.9	0.0287	0.0281	0.0274	0.0268	0.0262	0.0256	0.0250	0.0244	0.0239	0.0233
-1.8	0.0359	0.0351	0.0344	0.0336	0.0329	0.0322	0.0314	0.0307	0.0301	0.0294
-1.7	0.0446	0.0436	0.0427	0.0418	0.0409	0.0401	0.0392	0.0384	0.0375	0.0367
-1.6	0.0548	0.0537	0.0526	0.0516	0.0505	0.0495	0.0485	0.0475	0.0465	0.0455
-1.5	0.0668	0.0655	0.0643	0.0630	0.0618	0.0606	0.0594	0.0582	0.0571	0.0559
-1.4	0.0808	0.0793	0.0778	0.0764	0.0749	0.0735	0.0721	0.0708	0.0694	0.0681
-1.3	0.0968	0.0951	0.0934	0.0918	0.0901	0.0885	0.0869	0.0853	0.0838	0.0823
-1.2	0.1151	0.1131	0.1112	0.1093	0.1075	0.1056	0.1038	0.1020	0.1003	0.0985
-1.1	0.1357	0.1335	0.1314	0.1292	0.1271	0.1251	0.1230	0.1210	0.1190	0.1170
-1.0	0.1587	0.1562	0.1539	0.1515	0.1492	0.1469	0.1446	0.1423	0.1401	0.1379
-0.9	0.1841	0.1814	0.1788	0.1762	0.1736	0.1711	0.1685	0.1660	0.1635	0.1611
-0.8	0.2119	0.2090	0.2061	0.2033	0.2005	0.1977	0.1949	0.1922	0.1894	0.1867
-0.7	0.2420	0.2389	0.2358	0.2327	0.2296	0.2266	0.2236	0.2206	0.2177	0.2148
-0.6	0.2743	0.2709	0.2676	0.2643	0.2611	0.2578	0.2546	0.2514	0.2483	0.2451
-0.5	0.3085	0.3050	0.3015	0.2981	0.2946	0.2912	0.2877	0.2843	0.2810	0.2776
-0.4	0.3446	0.3409	0.3372	0.3336	0.3300	0.3264	0.3228	0.3192	0.3156	0.3121
-0.3	0.3821	0.3783	0.3745	0.3707	0.3669	0.3632	0.3594	0.3557	0.3520	0.3483
-0.2	0.4207	0.4168	0.4129	0.4090	0.4052	0.4013	0.3974	0.3936	0.3897	0.3859
-0.1	0.4602	0.4562	0.4522	0.4483	0.4443	0.4404	0.4364	0.4325	0.4286	0.4247
0.0	0.5000	0.4960	0.4920	0.4880	0.4840	0.4801	0.4761	0.4721	0.4681	0.4641

Standard Normal Cumulative Probability Table



Cumulative probabilities for POSITIVE z-values are shown in the following table:

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998