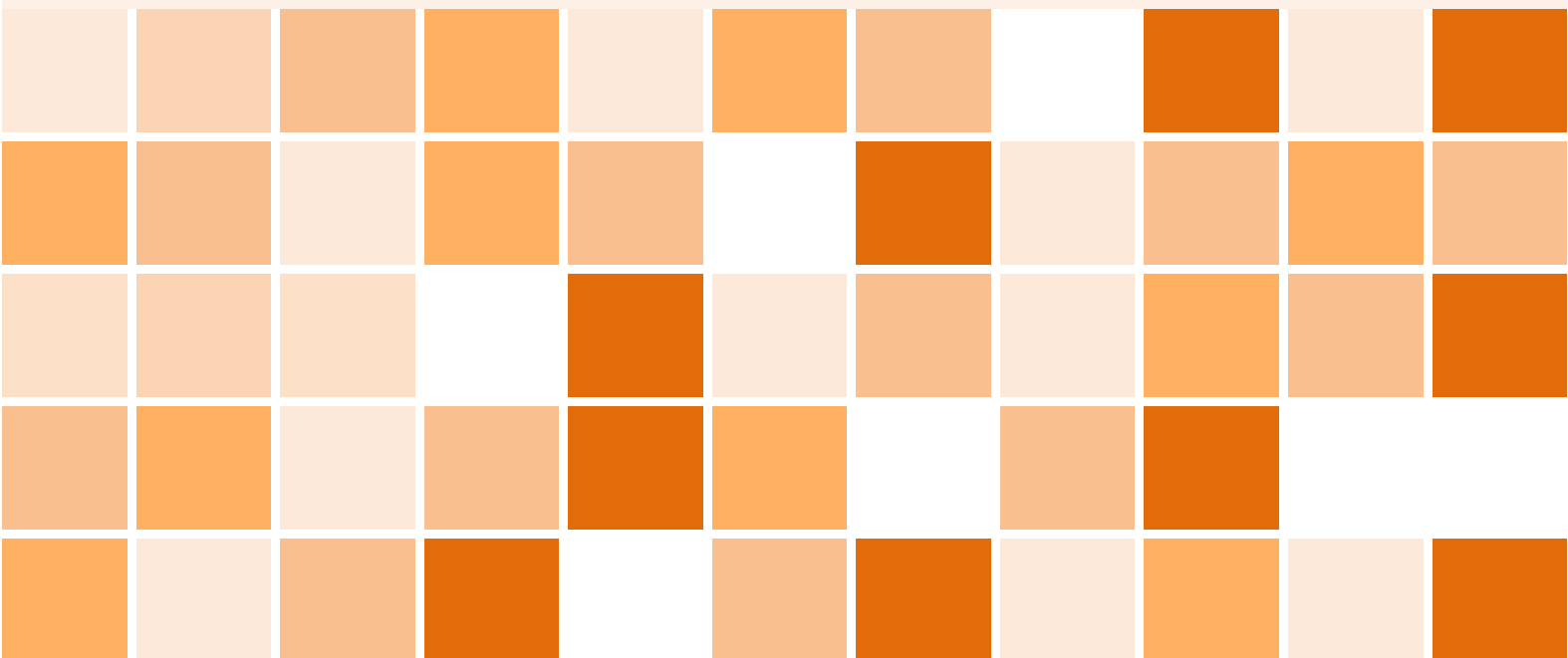


Introduction to discrete-event simulation

James Bekker

Simulation 442 – 2025

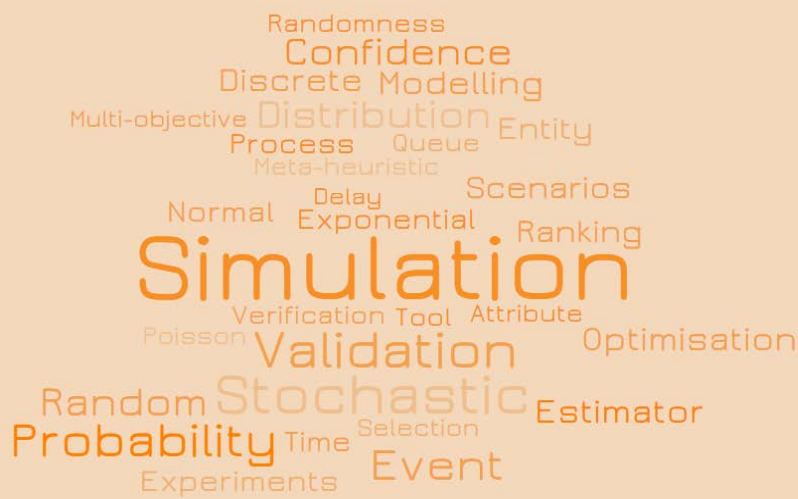


Copyright © 2025 Stellenbosch University

PUBLISHED BY JAMES BEKKER

DOCUMENT TEMPLATE: THE LEGRAND ORANGE BOOK BY MATHIAS LEGRAND, ADAPTED
BY JAMES BEKKER

First print, July 2019



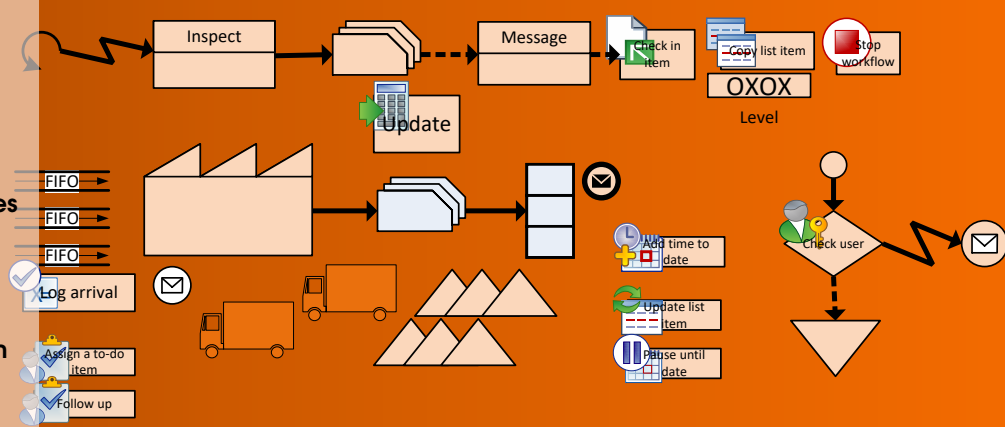
Contents

1	Introduction	7
1.1	Basic concepts needed	8
1.2	Systems thinking	8
1.3	Modelling	9
1.4	Why simulate?	10
1.5	What is simulation	11
1.6	Typical application areas of simulation	12
1.7	Discrete and continuous random variables	12
1.8	Simulation in perspective	13
1.9	Advantages and drawbacks	13
1.10	Pitfalls of simulation studies	14
1.11	Other modelling tools	15
1.12	The discrete-event simulation mechanism	15
1.13	Simulation software packages	20
1.14	Animation in simulation	20
2	Steps in a simulation study	21
2.1	Determine the problem and the objectives	21
2.2	Project planning	24
2.3	Define the boundaries for the study	24
2.4	Formulate a concept model	25
2.5	Preliminary experiment	28

2.6	Determine input data required, obtain the data and process it	29
2.7	Translate the model to a computer simulation language	29
2.8	Verify the model	29
2.9	Validate the model	30
2.10	Rework model	31
2.11	Execute the production runs	31
2.12	Perform the statistical analysis	31
2.13	Model modification and scenario analysis	32
2.14	Document the study concurrently	32
2.15	Implement the results of the study	32
2.16	Monitoring and maintenance	32
3	Random variates	35
3.1	Random number generation	35
3.2	Basics of random number generators	36
3.3	Generating random variates	37
3.3.1	The inverse transform: continuous case	37
3.3.2	The inverse transform: discrete case	40
3.3.3	Monte-Carlo simulation	42
4	Input data analysis	45
4.1	Sources of input data	45
4.2	No data available	46
4.3	Data available	47
4.4	Empirical distributions	47
4.5	Theoretical distributions	49
4.5.1	Selecting an input distribution	49
4.5.2	The Chi-squared test	50
4.5.3	The Kolmogorov-Smirnov test	52
4.5.4	Examples of distribution fits	54
5	Output data analysis	59
5.1	Output analysis Overview	59
5.2	Properties of output variables	60
5.2.1	Method of calculating of output	61
5.2.2	Types of simulation output	61
5.2.3	Point estimator	62
5.2.4	The confidence interval	63
5.3	Terminating systems	66
5.3.1	Determining the number of observations	66
5.3.2	Half-width of the confidence interval and the Two-phase method	67

5.4	Non-terminating systems	69
5.4.1	Determining the truncation point	69
5.4.2	Replication/Deletion approach	71
5.4.3	Batch-means approach	71
5.5	Evaluating and selecting scenarios	76
5.5.1	Analysis of variance	77
5.5.2	Applying ANOVA	78
5.5.3	The Kim-Nelson procedure	81
6	Optimisation with simulation	85
6.1	Introduction to simulation-optimisation	85
6.2	Some optimisation problems	88
6.2.1	Rosenbrock function	88
6.2.2	Rastrigin function	88
6.2.3	Shekel function	89
6.3	The genetic algorithm	90
6.4	Application of GA principles in Tecnomatix	95
6.5	Multi-objective optimisation in large decision spaces with simulation	96
6.6	Multi-objective optimisation problem formulation	97
6.7	Ranking of solutions	100
6.8	Solution selection	101
6.9	Some MOO algorithms	102
	Appendix A – Tables: statistical values	104
	Appendix B – The four modelling paradigms	109
	References	111
	Index	115

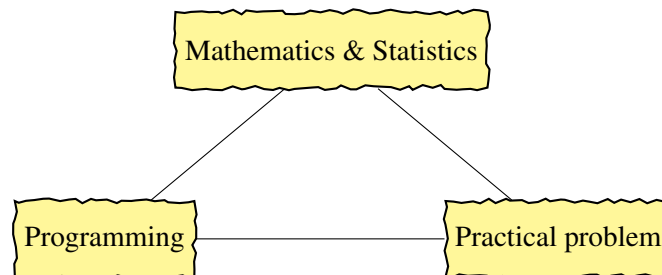
Basic concepts needed
 Systems thinking
 Modelling
 Why simulate?
 What is simulation
 Typical application areas of simulation
 Discrete and continuous random variables
 Simulation in perspective
 Advantages and drawbacks
 Pitfalls of simulation studies
 Other modelling tools
 The discrete-event simulation mechanism
 Simulation software packages
 Animation in simulation



1. Introduction

Simulation is beautiful

SIMULATION is beautiful because it combines the application of some mathematical and statistical principles with computer programming to solve real-world problems. The following schematic shows the idea.



This set of short notes provides the essential content of the Simulation 442 module. The module content mainly covers the simulation of *discrete-event stochastic processes*. At the same time, special attention is paid to the basic statistical principles required for application in the simulation of discrete, stochastic processes. Using a simulation software package is a secondary aspect that is addressed, and *Tecnomatix Plant Simulation* will be introduced in this module. It is important to understand the statistical and other principles needed for discrete stochastic simulation – many people study the simulation software only, but this is insufficient to conduct a simulation study sensibly.

1. Overview, simulation in perspective, systems thinking, steps in a simulation study, concept models, basic statistics, demonstration and discussion of some models.
2. Modelling world views and imitating reality.
3. Analysis of input data for simulation models.
4. Analysis of output data for simulation models – terminating systems.
5. Analysis of output data for simulation models – non-terminating systems.
6. Techniques to select the best from a finite number of simulated scenarios.
7. Principles of simulation-optimisation.
8. Building computer models in Tecnomatix Plant Simulation followed by analysis of output data.

R Module survival hints: Study the notes, then do the tutorials with vigour. Always try to relate the tutorial questions to the theory in this book, and attend class. Attempt the tutorials during the week, and come to the tutorial classes to put the finishing touches to the tutorial submissions. This is an *applied* module.

1.1 Basic concepts needed

The following basic topics of Statistics should be well understood for the purpose of this module:

1. Sample space.
2. Probability and proportions.
3. Randomness, random numbers and random variables.
4. Distributions – Discrete (mass function) and continuous (density function).
5. Cumulative distribution functions.
6. Line graphs and histograms.
7. The mean, mode, median and variance of distributions.
8. The Poisson, exponential, Student t, normal and other distributions.
9. Parameters of distributions.
10. Hypothesis tests.
11. The Central Limit Theorem.
12. Statistical inference: point estimators and confidence intervals.

The student also needs a good understanding of queueing theory, which may require a revision of Chapter 20 in Winston (2003), while simulation is covered in Chapter 21 in Winston (2003). Further (not complete!) sources of simulation practice and theory are listed at the end of this book.

On completion of this module, the student should be sufficiently equipped to execute a simulation study with some senior guidance in practice. When some experience has been gained, the engineer should be able to conduct simulation studies independently. The primary aim is to explain the art and science of simulation analysis to the student, with applications developed and analysed with a specific simulation software package as the secondary aim. A thorough knowledge of descriptive and inferential statistics and hypothesis testing is recommended.

The significant simulation activities, which are input analysis, modelling, and output analysis, are explained. These can be subdivided into several chronological and concurrent steps, which are treated in detail. The steps are practically applied and illustrated with examples. The Tecnomatix Plant Simulation package is used in this module. The focus is mainly on stochastic, dynamic, discrete event simulation models. Before we look at simulation, we should briefly consider two fundamental topics: *Systems thinking* and *modelling*. Each topic can be studied independently, but we shall explore it briefly to understand the topic better.

1.2 Systems thinking

One of the most important skills of an industrial engineer is to be able to see the whole, as well as the hole (the doughnut principle). This is an art required for successful simulation – first to isolate a section of a whole for simulation purposes, and then to be able to describe this selected section at a sufficient level of detail (*resolution*). We must first understand what a system is, at least from a simulation point of view:

Definition 1. A system is a **group** or **collection** of **interrelated objects** that **cooperate** to achieve one or more **objectives**.

We could for example see the university as a system, and the objects could include *administration, finance, residences, maintenance* and the *academic* unit. This view comprises the bigger

picture, but we can analyse for example the academic unit down to module codes, where we can have a lengthy discussion on the length of the module code, *i.e.* whether it must be five or six digits long.

Another example is a motorcar: The objects could be the *engine*, the *body*, the *support structure* and the *wheels*. They are interrelated/not interrelated in some ways, and these objects mentioned (usually) cooperate to transfer the system from one point to another. One or more human beings often accompany this system, so that they also get displaced. The motorcar could also be viewed as consisting of a *mechanical* object and an *electric/electronic* object. This viewpoint is also valid, which shows that there are many ways to view a system, and no view is correct or incorrect, but we must realise that some views are more applicable to simulation and/or the problem at hand than others.

More examples:

1. A manufacturing plant, comprising

- people
- machines
- material
- information
- products

It could also be viewed as consisting of

- reception
- processing
- dispatch

2. A hospital with objects

- theatres
- wards
- casualty
- food preparation division

3. A law practice having objects

- administration support
- pending cases
- completed cases
- junior partners
- senior partners

We can thus represent a system from different viewpoints. The industrial engineer must thus be able to identify the correct/applicable system for simulation purposes. Of course, what comprises a system for one person is only a subsystem or a super-system for someone else.

1.3 Modelling

To understand simulation, we also have to understand the idea of *modelling*. We may quote many dictionary definitions of modelling, but in simple terms we define modelling as follows:

Definition 2. *A model is the representation of an entity/object in a form other than the entity/object itself.*

We may thus model a house by making a simple drawing of it, as shown in Figure 1.1.

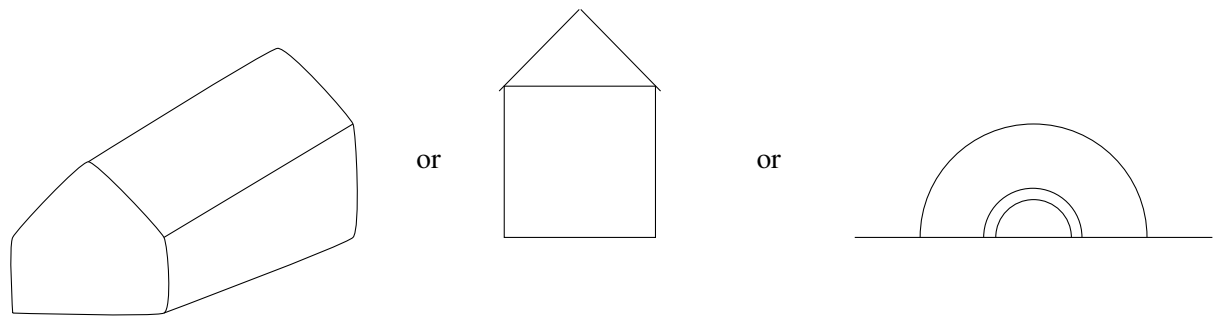


Figure 1.1: Schematics of house models

You would agree that these ‘models’ require some imagination to believe what they represent. A ‘better’ model could be one that we have built with a Lego set, which gives better views on layout and relative scales *etc.* Models like these are physical or iconic models, and are useful in many instances. This shows that a process of abstraction is present during modelling, and is also required by the person to whom the model is presented. We could also model other entities or systems with other approaches by following different viewpoints.

Typical viewpoints are

- Functional
- Logical
- Geographical
- Data flow.

Some types of models are

- Physical scale models: aeroplane models for wind tunnel tests, the human body for medical students
- Mathematical models: Linear programming, queueing theory, and economic models
- Social behaviour/psychological: Maslow’s five levels of human needs
- Ecological models: organism populations increase as a function of temperature.

Currently, four modelling paradigms are recognised. These are

1. Simulation, which is discussed in the next section.
2. System dynamics
3. Dynamic systems
4. Agent-based modelling, incorporating simulation.

These are summarized in Appendix B.

When we want to study a system, we can follow the map as defined by Law (2015), shown in Figure 1.2. We can experiment with the actual system, or develop a model of it, which we then experiment with. It is assumed that the behaviour of the model reflects the behaviour of the system it represents.

P “All models are false, but some models are useful”. (George E.P. Box, famous statistician)

1.4 Why simulate?

We must ask ourselves “Why do we want to study a system?” A system study usually originates because of one or more problems that have been identified by those operating/managing the system. The manager may for example wish to increase throughput at a certain point in a process, and suppose they have several possible ways to achieve this, they will first have to do some analysis, starting with Figure 1 as guide. When a system is of such a complex nature that it

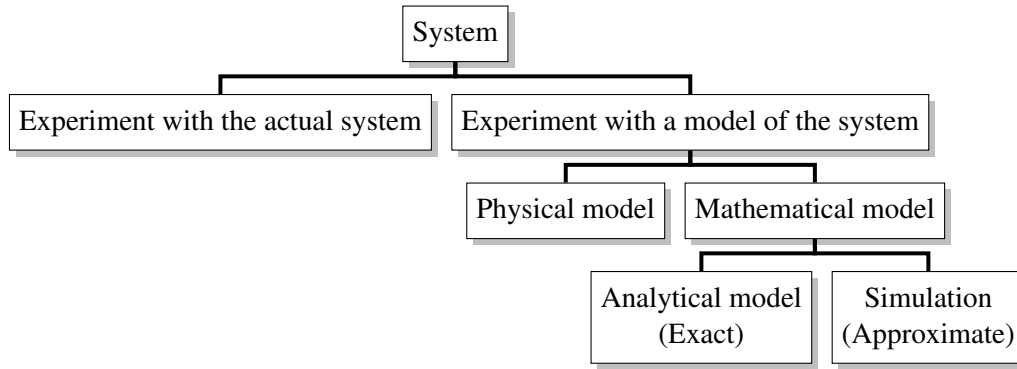


Figure 1.2: Ways to study a system (Law, 2015)

cannot be analysed analytically, simulation is strongly considered. In addition, if the system is of complex stochastic nature, then simulation is again indicated.

An example of a complex, stochastic process: Consider a simple queueing system with block arrivals from a finite population of size N , where the size of the block is a random variable X with a certain distribution. Assume that the system has four parallel service channels, and that there is only space for M entities in the system, while the queueing discipline is based on a general rule. It can be presented in Kendall-Lee notation (see Winston (2003)) as

$$GI^{[X]}|GI|4|GD|MIN.$$

Suppose we want to estimate the time an arrival spends in the common queue in front of the four service channels. There is no analytical solution to this queueing system, to the best of the author's knowledge. How practical is this system? Well, it represents the coal train arrival-service process at a South African coal terminal. Trains arrive with some variation of the time between arrivals, and each train consists of a different number of wagon sets (blocks). There are four service channels in the coal terminal that empty the wagons, and only 10 wagon sets can be hosted in the terminal, while the fleet of wagons is finite, *i.e.* 6 000 wagons. Each set consists of 50 wagons, so $N = 120$ in this case. The random variable X (the number of wagon sets in a train) could follow the distribution

$$f(x) = \begin{cases} 0.3 & x = 2 \\ 0.5 & x = 3 \\ 0.2 & x = 4. \end{cases}$$

A train thus consists of three wagon sets with a probability of 0.5. Since the sets are rotated (mine to coal terminal and back to mine) the population from which arrivals originate is finite. We see that a (practical) system can quickly become complex when we try to describe it analytically.

1.5 What is simulation

Simulation is a widely used term that is (over) utilised to cover virtually all perceptions on mock-ups of real/complex/non-existent entities. The industrial engineer views simulation as a computer model that yields the responses of a (usually) highly variable system, given a certain input data set. The economist simulates return on investment by applying different interest rates over different periods by way of example. A pilot exercising skills in a flight simulator is subjected to simulated situations.

The Handbook of Simulation (Banks, 1998) defines simulation as follows (p. 3):

Definition 3. *Simulation is the imitation of the operation of a real-world process or system over time.*

Simulation can thus be viewed as the experimentation with a model of a real-world system in order to study the behaviour of the model, given certain starting conditions. It is assumed that the behaviour of the model is a sufficient predictor of the real system's behaviour. This definition implies that we generally try to answer *what-if* questions with simulation, *i.e.* an existing or proposed system is modelled with a simulation software package and the behaviour of the system is studied when parameters are changed. Systems are thus analysed to find an acceptable solution for their operating parameters.

Simulation is very versatile, because it can be used in *manufacturing, agriculture, mining* and the *services* sector.


1.6 Typical application areas of simulation

The following is a list (incomplete) of some of the major fields where simulation is applied:

- Manufacturing (you guessed it!)
- Services (hospitals, business processes)
- Military
- Distributed systems
- Transport
- Computer and communication systems
- Physical sciences
- Aerospace systems
- Environmental, ecological studies
- Financial decision support systems
- Undersea systems
- Simulators
- Supply chains and logistics

1.7 Discrete and continuous random variables

A clear distinction should be made between *discrete* and *continuous* variables. A discrete variable represents countable things like number of products made, number of machine failures registered and number of hamburgers sold. The discrete variable can have finite or infinite limits on the integer numbers range. The continuous variables associates with time, distance, mass, and proportions like service level and utilisation. Discrete and continuous random variables consequently have different accompanying distributions. Some fundamental properties of a random variable, say X , are shown in Table 1.1.

 The reader should be aware of the various forms in which some probability density functions are presented – the Weibull p.d.f. has at least five forms, each presented within a certain context in the literature. The exponential distribution has the forms

$$f_X(x) = \lambda \exp^{-\lambda x}, \quad x > 0, \quad (1.1)$$

and

$$f_X(x) = \frac{1}{\beta} \exp^{-x/\beta}, \quad x > 0, \beta > 0 \quad (1.2)$$

and $\lambda = 1/\beta$. Thus, if $\lambda = 2$ arrivals/minute, then $\beta = 0.5$ minutes between arrivals.

Table 1.1: Discrete and continuous random variable properties

Discrete	Continuous
Probability <i>mass</i> function is $f_X(x)$.	Probability <i>density</i> function is $f_X(x)$.
$f_X(x) \geq 0, \quad x \in A$ (A is the definition range of X)	$f_X(x) \geq 0, \quad x \in \mathbb{R}$
$P(X = x) = f_X(x)$	$P(a < X < b) = \int_a^b f_X(x) dx$
$\sum_{x \in A} f_X(x) = 1$	$\int_{-\infty}^{\infty} f_X(x) dx = 1$
$F_X(x) = P(X \leq x) = \sum_{t \leq x} f_X(t)$	$F_X(x) = P(X \leq x) = \int_{-\infty}^x f_X(t) dt$

1.8 Simulation in perspective

The focus in this book is on *dynamic, stochastic, discrete event* simulation (DES) models. In dynamic, stochastic, discrete event simulation models, the state of the system being modelled changes at discrete and usually random-spaced points in time. A simulation model is characterised by specific features of the system that is being modelled. These features are *time dependency*, specific *characteristics of the variables* and the *simulation time increment*.

Time in this context means that the system or process being modelled *evolves over time*, for example the business processes of a retail shop: customers come and go, inventory is replenished, maintenance is being done, and more.

Deterministic variables assume exact values: there are 12 cashiers on duty in the retail shop, and the shop has 24 shelves.

A *stochastic variable* describes uncertainty, for example the number of customers that visit the shop per day. If the simulation time increment is *discrete*, it means that events are modelled that happen at discrete points in time, for example, when a customer arrives, it is considered an event. Other events are when the customer service starts and ends.

Figure 1.3 puts simulation in perspective within the mathematical modelling domain.

With this perspective in mind, one can consider the advantages and some drawbacks of simulation.

1.9 Advantages and drawbacks of simulation

Some advantages of simulation are:

- Analyse systems operations before they are implemented – avoiding and/or minimising cost
- Optimisation or sub-optimisation
- Tactical and strategic planning
- Changes to a system can be investigated without disrupting the operations of the system: New designs/acquisitions of resources can be investigated before the actual capital layout is made.
- Long ('never ending') processes are studied in a relatively short time.
- Critical parameters in a system can be identified and studied.
- Evaluation of alternatives.

It is true that simulation also has some drawbacks, some of which are:

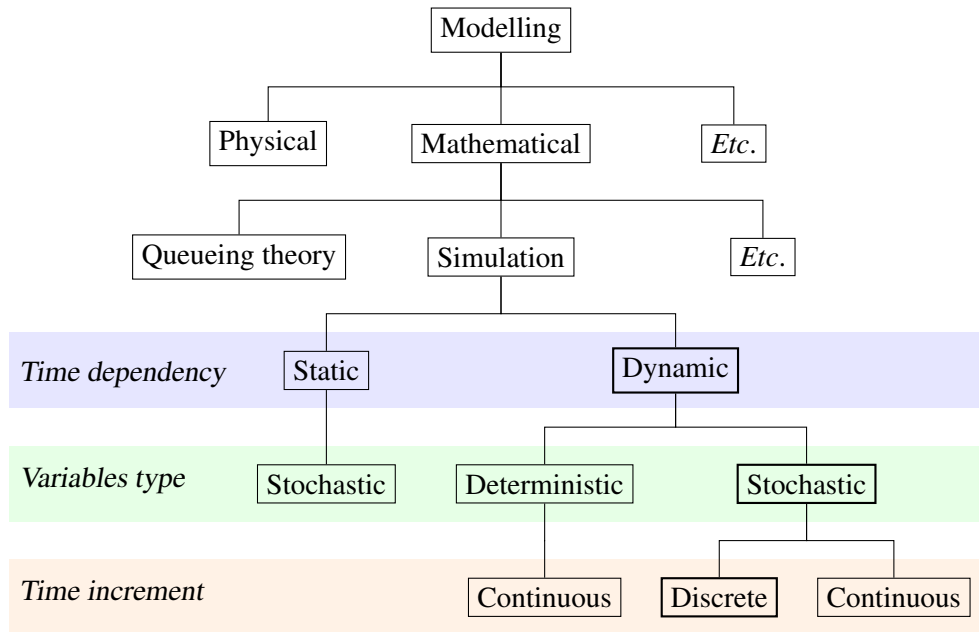


Figure 1.3: Simulation in perspective (Kruger and Du Preez (1988))

- Simulation studies can be *expensive* and *time-consuming* in many cases. (Software packages may become cheaper in some cases).
 - It is time-consuming – a simulation study takes time if properly executed.
 - A simulation study could be too costly for a given problem.
- It requires a certain level of expertise, although many software vendors claim that their products can be used by virtually anyone, not only bald professors with many years of experience. This is however the danger – anyone can use a simulation software package, but not everyone understands system dynamics and in many cases mathematical statistics. The software packages make analysis much easier today, but the author is still convinced that a person who understands the underlying concepts is required for simulation. A good simulation analyst needs good training and experience.
- Other tools may be more appropriate to analyse a particular problem, for example linear programming.
- The interpretation of simulation results requires a sound statistical background, regardless of what simulation software vendors tell you.

1.10 Pitfalls of simulation studies

Law (2015) discusses pitfalls to the successful completion of a simulation study. These are extremely important and some are quoted verbatim (Law (2015), pp. 71–72):

- “Failure to have a well-defined set of objectives at the beginning of the simulation study.
- Misunderstanding of simulation by management.
- Treating a simulation study as if it were primarily an exercise in computer programming.
- Failure to have people with knowledge of simulation methodology and statistics on the modeling team.
- Failure to collect good system data.
- Misuse of animation.

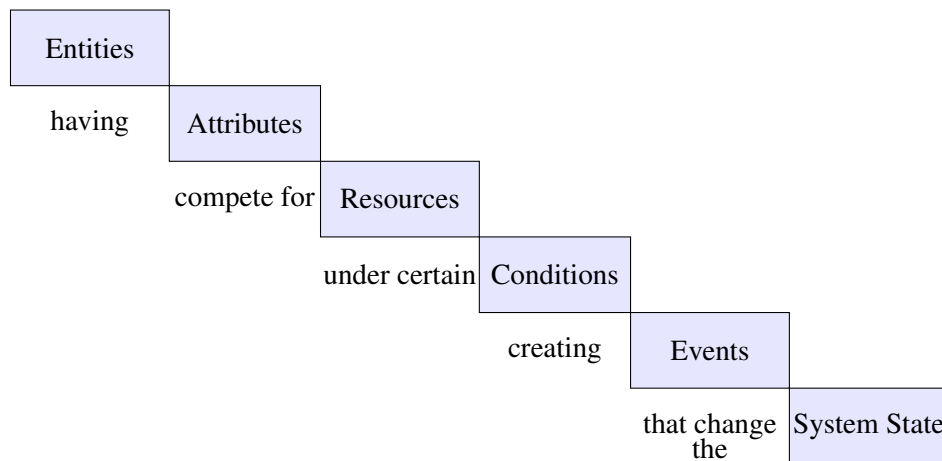


Figure 1.4: A typical simulation world view

- *Making a single replication of a particular system design and treating the output statistics as the “true answers”.*
 - *Comparing alternative system designs on the basis of one replication for each design.”*
- Also refer to Law (2003) and Sturrock (2018) for detailed discussions.

1.11 Other mathematical-based problem-solving alternatives

The industrial engineer is equipped with several problem-solving skills and tools. Although simulation is versatile and powerful (in the right hands), other mathematical-based problem solution tools can be used where appropriate, and some are:

- Queueing theory
- Linear programming
- Assignment algorithms
- Integer programming
- Graph theory
- Markov chains
- Stochastic inventory models
- Dynamic programming (deterministic & stochastic)

The problem must allow the tool to be used. For example, linear programming cannot be used if a problem is non-linear.

Next, the very important concept of the discrete-event mechanism is introduced.

1.12 The discrete-event simulation mechanism

We desire to model a complex real-world problem on a computer, which is fundamentally a discrete machine. How can that be done, especially in when a problem domain is continuous? To answer this, some important aspects regarding the *simulation mechanism* must be understood, namely the

World view: How is a real-world system conceptualized in a computer language? That is, what is the implicit view of the simulation software that we must follow to implement a real world system’s behaviour? A typical world view is shown in Figure 1.4.

This is a model according to R. Shannon (1975), and it is the basic one we will often use in this course. The simulation modelling world view is a framework for defining a system in

sufficient detail so that the behaviour of the system can be simulated (Pegden, R.E. Shannon, and R. Sadowski, 1995). A simulation model contains system state variables which must be changed over time, and the world view provides a set of rules for advancing simulation time and changing of state variables. Pegden, R.E. Shannon, and R. Sadowski (1995) describe three approaches to world views, these are: *event modelling*, *process modelling* and *object modelling*. The elements of Shannon's world view are discussed next. Also see Ingalls (2013) and White and Ingalls (2016).

Entities

Entities are objects that 'flow' through a process or system. They drive the business and are the reason for the existence of the process. Examples are:

1. Customers entering and leaving a bank or retail store.
2. Bottles in a bottling plant.
3. Containers in a ship yard.
4. Vehicles on an assembly line.

Consider a vehicle on an assembly line: it is modelled as an entity, but the subsystems making up the vehicle could also be modelled as entities. This means that several entities are merged into a final entity, so there is often more than one type of entity in a model.

Attributes

An attribute describes an entity, and is thus defines a property of the entity. It can be viewed as a local variable, as the attribute value flows with the entity through the process simulated. An entity can have attributes like colour, mass, volume, type and dimensions. Bottle entities in a bottling plant can be for example of *colour* green and *volume* 375 ml, while another type of bottle can have *colour* amber and *volume* 750 ml.

Resources

Entities compete for resources in the processes we typically simulate. Resources provide services to entities, and often transform entities. Resources can be manufacturing machines, cranes, people like receptionists, doctors, machine operators, managers, supervisors, and also moving units such as forklifts, trucks and carts.

Conditions

The conditions refer to the business rules built to be followed by a real-world process. Rules include

1. first-in-first-out when several entities compete for a resource
2. a task must be completed before an operator can go home at the end of a work day
3. priority tasks must first be processed by a resource.

When developing a simulation model, the analyst spends a lot of time understanding the business logic.

Events

In discrete-event simulation, events are 'holding points' in time. The simulation model stops at a scheduled point in time and make changes to entities, which consequently leads to model state changes. Events include arrival of entities in the model, departure of entities, an entity starting or completing its processing at a resource, a resource fails or is repaired, the simulation models starting and ending execution.

System state

The system state is described by a set of variables. The concept of system state variables can be explained as follows: suppose we observe the arrivals, processing and departure of customers at


a single automated teller machine (ATM). What variables can we use to describe the process at the ATM? First, we draw a boundary around it, and we identify what ‘drives’ this system. In this case, it would be customers doing transactions at the ATM. So they enter the system by crossing the boundary drawn around it, and they exit the system by crossing the boundary again. Each customer also needs to occupy the ATM for some time, but this time is not known beforehand. Now we can say that ‘the number of customers in the system’ is a state variable, because it gives us a quantification of one property of the system. Our intuition tells us that if this variable has a high value, the system is ‘busy’, and if the number is small, the system is ‘not busy’. Some other system state variables are

1. Number of customers waiting in the queue.
2. Number of customers served up to time T .
3. The utilisation of the ATM.
4. The average/minimum/maximum time a customer spends in the system.
5. The total amount of cash provided by the ATM at time $= T$.

A simple trick to identify system variables is to imagine that you sit back and observe the system, then take a photo of its operations. What do you count? What do you see the resources doing? Are they idle/busy/failed/on leave/maintained? The choice of system state variables depends on the analyst and the purpose of the simulation study. Usually, the performance measures, which are a subset of the system state variables, are the only state variables mentioned by the analyst.

Transaction-flow world view

The transaction-flow world view is explained by Schriber, Brunner, and Smith (2016). A very important principle to understand from this approach is that an entity proceeds through a model until it encounters a delay or is disposed of. Travelling from one point to another also implies a delay. Many simulation languages (e.g. Arena, Tecnomatix Plant Simulation) follow this approach. As explained by Schriber, Brunner, and Smith (2016), in simulation parallel real-world processes must be emulated on a serial processor, which means that two or more traffic units often have to be manipulated at one and the same time point. The traffic units are manipulated serially at those time points to achieve ‘simultaneous’ movement. The question now is in which order should the units be treated at a given time point, leading to logical complexities for both the simulation software developer and in many cases, the simulation analyst.

 **P** Point to ponder: Can you suggest a better approach than the transaction-flow? How would you program it?

Entity control

To realise the flow of entities and maintain control over them, many simulation software packages use entity states and associated control structures. These support by implication Shannon’s world view. The entity states and associated control structures are shown in Table 1.2, according to Schriber, Brunner, and Smith (2016).

Given the above, the focus area, namely *discrete-event simulation* (DES), can now be defined as

Definition 4. *Discrete-event simulation is one in which the state of a model changes only at a discrete, but possibly random, set of simulated time points, called event times (Schriber, Brunner, and Smith, 2016).*

Table 1.2: Entity states and control structures

Entity state	Control structure
Active: This is the entity the simulation currently deals with. It could be an entity that has just been created, or an entity that is placed in a queue, or at a resource. Once the entity has been advanced until it encounters a delay, or is disposed of, it is not the active entity anymore.	Not applicable, as no control structure is needed.
Ready: Entities in this state are all candidates to become the Active entity. The simulation software has built-in rules to decide in what order to serve the entities. When the simulation software ‘stops’ to serve entities in this state, it serves them all before it proceeds to the next event in the model.	Current events list (CEL).
Time-delayed: all entities that are served by resources are in this state, including entities being transported by vehicles or on conveyors.	Future events list (FEL).
Condition-delayed: Entities awaiting one or more conditions to become true are in this state. Typically, entities waiting to reach the front of a queue/buffer are kept in this state. Entities waiting for a merge to occur are also in this state.	Delayed list.
Dormant: The analyst must provide programming logic in the model to deal with special entities. Usually, entities follow flow logic through a model, and do so according to the preceding states, structures and Shannon’s world view, but sometimes the analyst needs to take full control of entities.	User-managed list.

P Points to ponder: Current computers’ processors are discrete machines. How would you simulate a continuous process on a discrete machine? Suppose you can execute the sample path on two processors, how would you manage the events then?

Sample path

The dynamic processes we study require us to consider a time-line on which events are super-imposed. These events include entity arrivals in the process or system being modelled, entity departures, entities joining buffers or queues, and entities being served or processed by resources like human beings, machines, transporters or combinations of these. A sample path is used to understand the interactions and delays of entities when simulating a real process that evolves over time.

A simple sample path for three arriving entities at different points in time at a single server is shown in Figure 1.5. The length of the inter-arrival and processing times are arbitrarily chosen and the queue space is very large. The sample path shows events, server busy and idle times, and queue times.

One can apply some of the states and structures in Table 1.2 to the sample path in Figure 1.5. Assume it is an MIM1FIFO|∞|∞ system that is simulated. So the inter-arrival times and service times follow exponential distributions. Assume the software to simulate this is called ‘Simulation Package’ or SP for short. The process for three arrivals is described in Table 1.3. The values of the time intervals are not explicitly shown, but we know they are exponentially distributed. Note that only one entity can be in the Active state at a time.

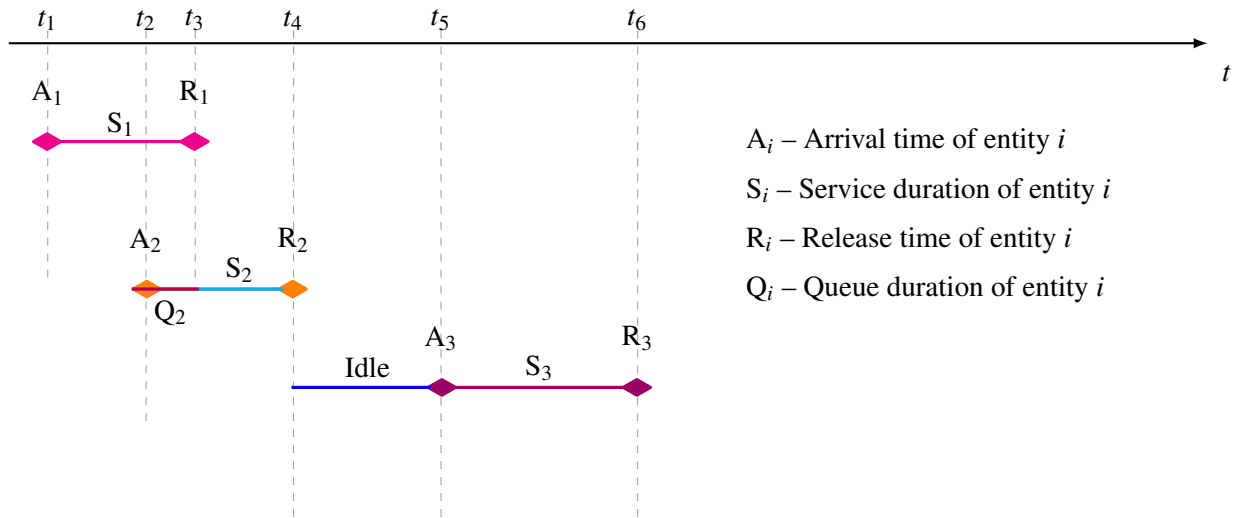


Figure 1.5: A simple sample path illustrated (single server, very large queue space)

Table 1.3: Example to illustrate the association of sample path and entity states and structures in Figure 1.5

At time	SP Action	State	Control structure
t_1	SP creates an arriving entity which is immediately in Active state, and since there are no entities in the system, the entity is assigned to the server. It is placed in the	Time-delayed state on the	Future Events list
t_2	SP creates a second arrival, which is immediately in Active state, and since there is an entity being served, this arrival is placed in a queue, which means it is placed in the	Condition-delayed state on the	Delayed list
t_3	The two entities both become SP takes the served entity, which is now in the The second entity on the Current events list is put in the The Active entity is assigned to the server and put in the	Ready and are placed on the Active state and disposes it Active state Time-delayed state on the	Current events list n/a n/a Future Events list
t_4	The service time of Entity 2 is over, and it is moved to the No other candidates currently exist in the model. The entity is moved to the	Ready state on the Active state and is disposed	Current events list n/a
t_5	A third arrival is created which is immediately in Active state, and since there are no entities in the system, the entity is assigned to the server. It is placed in the	Time-delayed state on the	Future Events list
t_6	The service time of Entity 3 is over, and it is moved to the No other candidates currently exist in the model. The entity is moved to the	Ready state on the Active state and is disposed	Current events list n/a
	There are no more entities to serve and the model terminates.		

1.13 Some simulation software packages

There are many simulation software packages available, including open source and proprietary. In this module, Tecnomatix Plant Simulation (TPS) of Siemens Digital Industries Software will be used. For a complete list, see <https://www.capterra.com/simulation-software/> and https://en.wikipedia.org/wiki/List_of_computer_simulation_software (viewed on 21 July 2023).


1.14 Animation in simulation

Animation is a feature that allows for a dynamic graphic representation of the simulation problem on the computer screen. This makes it easier to study the system's dynamics as one can see what happens while the model evolves over time. One danger is that one can get so involved with the animation that the simulation study is neglected. People who validate (see later) simulation models (usually managers) must therefore realise that a balance between studying the problem and representing it graphically must be maintained. Many hours can be spent on creating detailed animations, which usually contribute only marginally to the understanding of the problem under study.

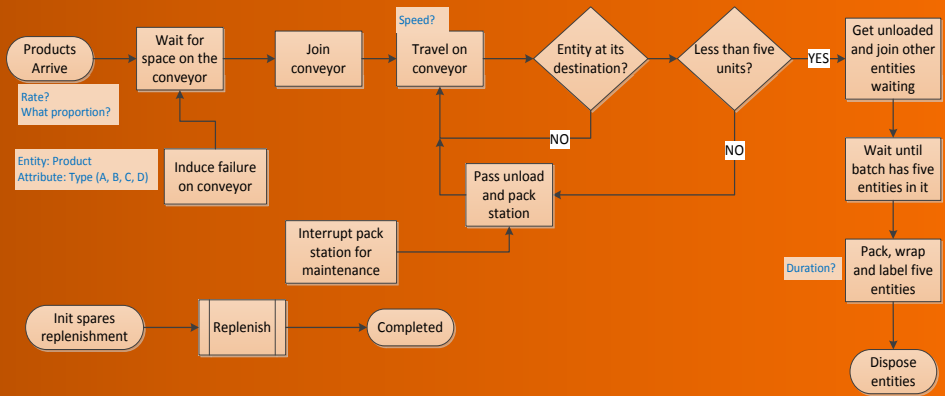
Survival kit:

Construct your simulation foundation using the following:

- 1 Know the basic concepts of Statistics presented in Section 1.1.
- 2 Know the difference between *discrete* and *continuous* random variables.
- 3 Know the generic definition of simulation, and the definition of discrete-event simulation.
- 4 Know the entity states and associated structures and apply the knowledge through an example of a real-world process.
- 5 Apply the world-view shown in Figure 1.4.
- 6 List some advantages and some drawbacks of simulation.
- 7 List some simulation application areas.
- 8 Draw a simple sample path, similar to Figure 1.5. More complicated sample paths (more entities that are simultaneously present) should be examined.
- 9 Describe entity states and control structures similar to what is presented in Table 1.3.

 All the statistics in the world can't measure the warmth of a smile – Chris Hart

Determine the problem and the objectives
 Project planning
 Define the boundaries for the study
 Formulate a concept model
 Preliminary experiment
 Determine input data required, obtain the data and process it
 Translate the model to a computer simulation language
 Verify the model
 Validate the model
 Rework model
 Execute the production runs
 Perform the statistical analysis
 Model modification and scenario analysis
 Document the study concurrently
 Implement the results of the study
 Monitoring and maintenance



2. Steps in a simulation study

A SIMULATION study is initiated by a problem in an existing system or while designing a new system. The analyst should however carefully consider the nature of the problem: is simulation really the tool to be used for investigation, or is any other modelling or problem solving tool more suitable? Simulation is one of several tools available to the engineer for problem solving purposes. The other alternatives usually provide quicker results and are not as expensive, but do not always provide suitable solutions to the problems at hand.

Once the analyst has established that simulation is indeed the preferred analysis tool, the following steps are suggested to model the problem. The steps should not be followed as a recipe, but may be tailored to suit the problem. Some are mandatory, while others may be reduced to the essential. Some steps may also be executed concurrently. The steps are discussed next.

2.1 Determine the problem and the objectives

This step is divided into three sub-steps: define the problem, *i.e.* compile a problem statement, do orientation and finally state the desired objective(s) with the study (Chung, 2004).

It often happens that a problem is analysed without really understanding what the real problem is (or what the root causes are), and often only the side effects or symptoms of a problem are investigated. Einstein stated that the formulation of a problem is even more essential than its solution, and in defining a problem one gets a better understanding of it. (The Japanese are well known for their belief that when a problem is properly defined, the solution is already 50% in sight). All members of a simulation project team, as well as the client (management, colleagues or an outside company) should therefore collaborate to define and agree on the problem definition/formulation.

To assist with the problem formulation, one could use techniques like the Fishbone and Pareto chart (Chung, 2004). It is important to state the expectations of all stakeholders during this phase to avoid scope creep of the study. It is further essential to clearly define the goals – what are the purposes of the study, why is it done, what is expected and what can be expected within the limitations of available resource allocation? The expectations must be stated for the short, medium and long term. It is also important that all parties agree on the goals, and that conflicting objectives be resolved. One or more of the following, depending on the real-world system, can drive the purpose of the study (Law and Kelton, 2000):

Evaluation	Comparing a system when evaluated against specific criteria.
Comparison	Comparing competing systems of similar functionality and comparing proposed alternatives.
Prediction	Estimate system performance under different operating conditions.
Sensitivity analysis	Determine which of one or several factors influence system performance.
Systems Improvement	Determine which of one or several factors influence system performance.
Optimisation	Determine which combination of factor levels results in best overall system performance.
Simulation optimisation	Integrating conventional simulation techniques with optimisation techniques to produce an optimal solution.
Functional relations	Determine the nature of relationships and the effects on the system's performance.
Other	Confirmation of proposals.

The industrial engineer should also remember their six Ms during problem formulation. The Ms tell us what we work with, and the efficiency of one or more Ms in a system could be improved using simulation. The Ms are summarised in Table 2.1.

Table 2.1: The six Ms guiding industrial engineers

M	Explanation
Man	<p>The human being is part of our systems: it could be clients, personnel (operators, nursing staff), stakeholders, colleagues, contractors and competitors.</p> <p><u>Questions:</u></p> <ol style="list-style-type: none"> 1. How many construction workers do we need for a given task? 2. What is the best schedule for a number of nursing staff? 3. How do we motivate employees?
Machine	<p>The non-human resources doing work in the system, for example manufacturing machines, trucks, fork lifts, conveyors, robots, computers.</p> <p><u>Questions:</u></p> <ol style="list-style-type: none"> 1. How many of a resource is needed in a process, for example, how many drills are needed in a job shop? 2. When must resources be taken out of production for maintenance? 3. When must a resource be salvaged? 4. Which is the best capacity of a potential new resource?
Material	<p>This is processed or transformed into something useful: Steel, wood, fresh produce/food.</p> <p><u>Questions:</u></p> <ol style="list-style-type: none"> 1. What is the best inventory level for a material to minimise inventory cost but maximise availability? 2. How can the cost of material handling be reduced (or contained)? 3. When must a material be acquired, and how much of it?
Money	<p>Finances are always paramount in a capitalistic world. Costs of the other Ms and profits are considered, usually over time.</p> <p><u>Questions:</u></p> <ol style="list-style-type: none"> 1. Which resources are the cheapest to acquire, but will give the best service? 2. What is the cost of a personnel schedule? 3. What is the expected profit of a given combination of personnel, resources and materials?

Continued on next page

M	Explanation
Method	<p>“Work smart, not hard!” Work methods, procedures and processes can be improved using simulation.</p> <p>Questions:</p> <ol style="list-style-type: none"> 1. How can a process, consisting of a series of activities be streamlined without losing its essential outcome? 2. How can the duration of an activity be shortened? 3. What is the best layout for a planned workstation?
Mother Nature	<p>Always find ways to cause the least harm to the planet. When solving problems, keep the environment in mind.</p> <p>Questions:</p> <ol style="list-style-type: none"> 1. How can the energy consumption of resources be minimised? 2. What is the best operating procedure in a process to minimise carbon emission? 3. What are good recycling policies for my company?

During orientation, the analysts familiarise themselves with the system that will be studied. This activity comprises an initial orientation visit to the problem area, during which high-level information regarding operations is collected. A second visit focuses on details, for example, resources, buffers, business rules, exceptions, and performance measures or *key performance indicators* (KPIs). There are two main types of performance measures. The first type is *qualitative*, like customer satisfaction, employee morale or operator fatigue. The second type is *quantitative* and examples of these are shown in Table 2.2 (adapted from Pooch and Wall (2000)). A third visit serves as a review opportunity during which the analyst will confirm previous observations and finalise outstanding issues (Chung, 2004). Knowing and understanding the

Table 2.2: Examples of simulation performance measures

Measure	Example(s)	Where, typically?
Throughput (rate)	Number of Car model X manufactured (per day)	Car assembly plant
	Number of freight containers loaded by a crane (per day)	Sea port
	Number of patients treated (per 24h)	Doctors' rooms
	Volume/mass of table grape packed (per shift)	Farm packhouse
Performance	Percentage of time a resource works/idles	Factory, construction site
	Percentage of time a resource is available/down	Factory
	Proportion of calls an operator accepts	Call centre
	Proportion of calls of a specific type	Call centre
Finance	Measure of cost over time	Supply chain
	Measure of income over time	Factory, warehouse
	Measure of net profit over time	Factory, transport fleet
Operations	Number of products delayed per time	Factory
	Buffer occupation by parts at a machine	Factory
	Number of patients waiting in emergency	Trauma unit
	Duration of patient waiting time in emergency	Trauma unit
	Duration of caller waiting time	Call centre
	Service level	Call centre, bank
	Number of customers balking (what is this term?)	Retail store
Time measures	Makespan of a simulated schedule	Job shop
	Average overtime worked per 24h	Factory, rail network
	Average lateness of orders	Online store
	Time to produce a specified number of products	Factory

process(es) and the (sub)system to be studied is also important for the development of a concept model as discussed in Section 2.4 on p. 25.

Finally, the following maxims from *The Handbook of Simulation* (Banks (1998), pp. 724 – 726) are worth considering:

- *A strong finish begins with an effective start.*
- *Fuzzy objectives lead to unclear successes.*
- *It is easier to correct an expectation now than to change a belief later. (Perception often becomes fact.)*
- *Focus on the problem more than on the model.*

Another good guideline for successful practice of simulation is given by Sturrock (2013).

2.2 Project planning

Project planning is a crucial element in ensuring the success of a simulation project, as it is the integrator of the efforts performed by the different resources of the simulation project. The major components of a project are

- Personnel
 - expertise
 - availability
- Hardware and software
- Funds
- Time

From a simulation project point of view, the personnel required to execute the project include the analysts, members of management and the system's operating staff. The analysts may be divided into subgroups executing parts of the project, and may include consultants from one or more external parties.

Simulation software often requires additions to standard available hardware to operate efficiently. If hardware and/or software are not available, they must be obtained, incurring costs and time delays that include training because the required skills and expertise are not always readily available.

Secondary costs may be incurred by travelling and the labour hours consumed by personnel who have to supply information for the study.

A complete, agreed project plan must be compiled, with detailed documentation of all the activities listed. Management must support the project and make it known to all concerned. *It is better to work with many intermediate milestones than with one absolute deadline* – maxim from *The Handbook of Simulation* (Banks (1998), p. 733).

2.3 Define the boundaries for the study

We have seen before that a system is a collection of interrelated objects that function towards one or more stated goals. The system can be viewed from different perspectives: functional, data flow, physical, *etc.*

The simulation analyst needs to view a part of the real world (the problem) as a system in order to define boundaries for the study as well as the level of detail. The boundaries define what will be included in and excluded from the model. Two boundaries are of concern (Law and Kelton, 2000):

- The boundary that separates the system (problem) from the rest of the universe it belongs to (the physical boundary).

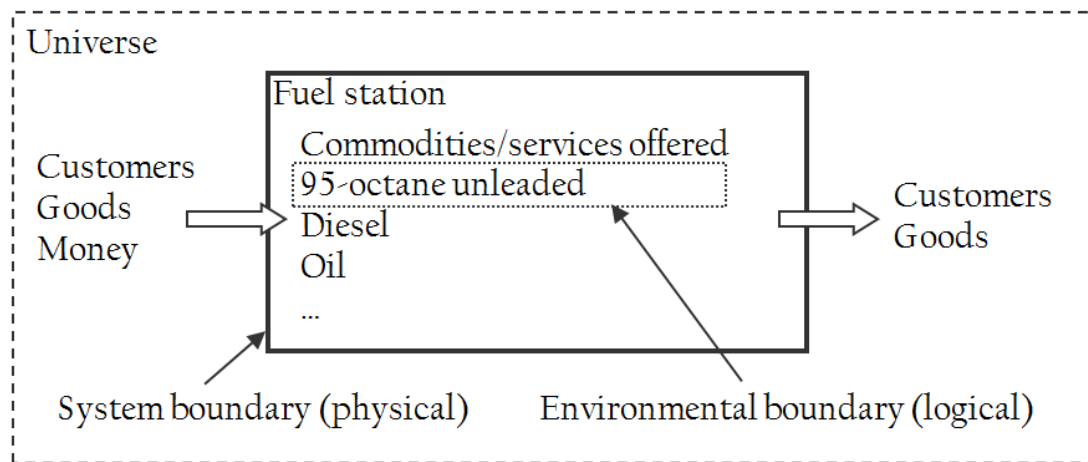


Figure 2.1: Example of a system boundary

- The boundary that separates the system defined from the environment in which it operates (the business boundary).

It is important not to exclude the interactions of the total system from the bounded system, unless they are insignificant. Consider a fuel station as a system. The fuel station sells several commodities and services, but the manager wants to determine the optimum reorder point for 95-octane unleaded petrol. The two boundaries are shown in Figure 2.1.

The purpose of boundary definition is therefore primarily to *simplify* the simulation study by reducing the amount of detail included. It is recommended that the smallest and least detailed model that provides the required information be used. The author has learnt through experience that large models should be developed from the top down and in two or more levels of detail. The first 'sweep' is usually performed at a rather high level of abstraction thus providing the least detailed model. This model is then embellished during a second and subsequent sweeps to a more detailed model. Verification and validation are also generally easier and faster for each sweep. The level of detail is generally reduced by aggregating related interdependent processes into 'black boxes' with a single input and output, and should vary in proportion to the impact it has on the results.

2.4 Formulate a concept model

During this step, the digital model is planned, and a proposed model is laid out 'on paper' in pseudo-code or graphical format, which we refer to as the *concept model*. A concept model describes the model logic in programming language-agnostic fashion so that a common reference for stakeholders from different backgrounds can be established. The real-world process or system is thus translated into the concept model from which the digital model is derived. The idea is shown in Figure 2.2, where detail is removed by making assumptions. The real-world is detailed and complex, but an approximate model is created, which must still be useful, but with less detail.

Boundary definition as discussed under the previous step, and model abstraction (developing a concept model) also include making assumptions. These should be *thoroughly and unambiguously* documented and referred to during the study. The list of assumptions is dynamic and should be updated during model development and must be included in the final report. The output results should also be evaluated against the assumptions made, and the project team and client(s) should all agree on the assumptions.

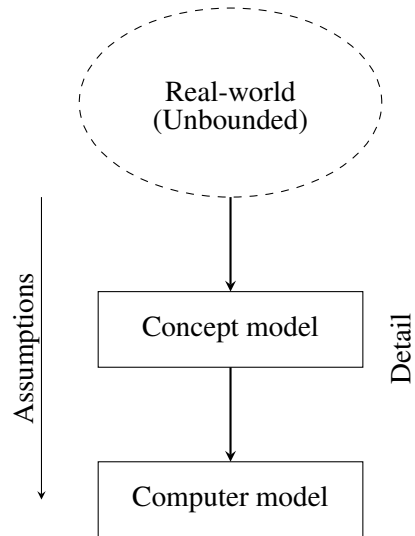


Figure 2.2: Real-world process via concept model to digital model

A typical relationship between the utility and complexity of a model is shown in Figure 2.3. The usefulness of a model can be the same for various levels of complexity, as shown by the red lines and blue circles. A less complex model (shown with a '1' on the vertical axis) representing the essentials of a system is usually a sufficient imitation of the real world. A more complex model (shown with a '2') requires more effort and cost but yields the same utility.

There are several advantages to the concept model, some are

1. The analyst(s) will quickly realise how well the system under study is understood.
2. Lack of information on system operations will be identified, which can be supplemented during a second round of interviews with those involved in the operations.
3. The model's logic is established (at least on a first order level).
4. Input data requirements are identified.
5. Assumptions are verified.
6. The need for more/less assumptions is identified.
7. First order validation can be done with those involved in the system's operations. This has the secondary advantage that these people do not need any computer knowledge.
8. A thoroughly designed concept is easier (and usually faster) to code.
9. The various levels of detail become apparent and can be adjusted.

This step can be viewed as a form of planning the computer model, and is usually neglected, because we tend to try to start coding the model as soon as possible, and often sooner than possible! We diagram the flow of a typical entity through the process logic. Along the way, we consider

- The entity type and possible transformations into other entity types.
- Entity attribute changes.
- Resources needed.
- Buffer/queue areas needed.
- Business rules to obey.
- Input data required.
- Performance measures required.

When constructing the concept model, consider the following maxims from *The Handbook of Simulation* (Banks (1998), pp. 733–737):

- Add detail, do not start with it.

- Let the model work for you.
- If it does not make sense, check it out.
- People decide, models do not.

A worked example of a concept model follows below. A narrative is usually obtained from one or more stakeholders and is used to develop the concept model. The model is usually presented in a graphical format. An informal set of symbols is used in the example, but the simulation analyst may use formal languages like activity diagrams from UML or Business Process Modelling Notation (BPMN, see www.bpmn.org). The business rules, decisions, storage, buffering, processing, and entity interaction are modelled.

Example of a concept model

Question: Develop a concept model for the following description:

Cars arrive at a parking area in Bitterfontein. An operator at the only access/exit gate regulates entries. If a parking space is available, the currently arriving car is given a ticket with the time stamped on it, and the car is allowed into the parking area. If the area is full, any cars arriving are denied entry. Drivers and their cars wishing to leave the parking area must first pay the operator at the gate a fee that depends on the duration of the parking time. A car waiting to pay before leaving the parking lot is considered to be taking up parking space. That space is thus only released when the payment transaction has been completed. All cars are the same size, so one car occupies one parking space. Ignore the driving of the cars in the parking area and assume the operator does not get tired. Also, assume that the technology does not fail.

Answer: How should we approach the concept model development? Look for (an) object(s) that flow(s) through a process, as the documented narrative mentions. If the absence of this object causes the process to cease, then it must be an entity. Identify objects (resources) for which the entities compete, for example, transporters, personnel, or machines. Look out for business rules which dictate the flow of entities. Also, identify holding areas or buffer spaces. Points along the flow of the entities will require input data, for example, the duration of a service by a resource, while some KPI values could be observed and registered along the way. It is possible that an entity can be split into other entities (an order becomes an order and a picking slip), or entities

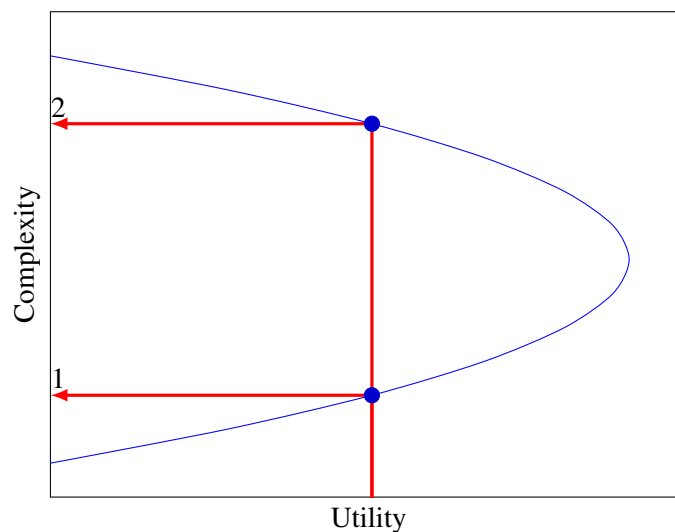


Figure 2.3: Relationship between complexity and utility (Banks, 1998)

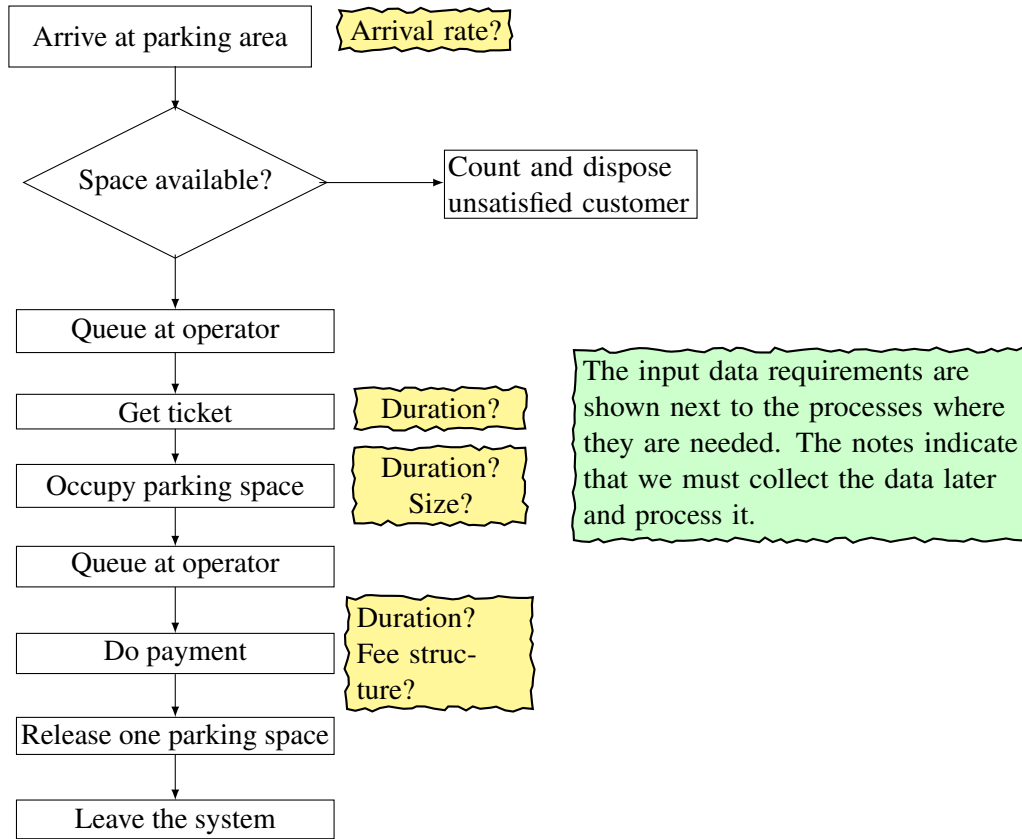


Figure 2.4: An example concept model

can be merged (a truck entity and load entity become a dispatched entity). The description of the example shows that the model entity represents arriving cars. The cashier is a resource, while the parking area is a holding area where each entity (car) will be held until it starts to compete for the cashier again (on exit). An example answer of the model is shown in Figure 2.4.

Once the concept model has been developed, the preliminary experiment must be defined, as is subsequently discussed.

2.5 Preliminary experiment

This step comprises the establishment of

- Level of confidence for the confidence intervals (usually 95%).
- Model time span, for example 8 hours per day. This determines if we deal with a terminating or non-terminating system.
- Input variables to be varied during the study: These are the *decision variables* over which the simulation analyst takes control.
- Limits of the decision variables: The ranges of the decision variables have an impact on the simulation effort. Many decision variables with large ranges may cause long simulation times because many scenarios must be simulated. Suppose we have two discrete variables X and Y , and we vary each from 1 to 10, then the decision space has $10 \times 10 = 100$ scenarios.
- Identify the model output parameters (also known as *key performance indicators* (KPIs)) that will be studied to supply the desired information from which conclusions of the system's performance will be made. These will comprise the *output parameters* on which output

statistical analysis will be done, and they will drive the required number of replications and/or the length of the simulation run. These parameters are identified and translated from the simulation study objectives to variables, *e.g.* the variable Throughput will describe the output parameter *Products completed per day*. An objective of the study could be to determine the scenario with the highest number of products produced per day.

- Definition of the entity or the entities, and possible conversions of entities into other entity types.
- Attributes per entity where applicable.
- Scale of units of measurement: Time and other units to be used, *e.g.* m/s vs. km/h *etc.* Tecnomatix uses SI units of measurement.
- Resources in the model.

2.6 Determine input data required, obtain the data and process it

Collect the data required for the model as identified by the concept model definition and the preliminary experimental design. This is usually a time-consuming process. The details of this step are discussed in Chapter 4.

2.7 Translate the model to a computer simulation language

The computer model is developed from the concept model as defined in Section 2.4.

2.8 Verify the model (debugging)

This is a detailed step in the simulation study and is simulation software orientated, and we ask the question “Did we build the model right?”. The main action here is debugging to verify that the computerised model works correctly. Other actions include:

- Syntax errors are corrected.
- Logic is checked.
- Compiler (if applicable) errors are corrected.
- A secondary source of errors is *Run-time errors* that only manifest during program execution, which must be corrected.

If no errors occur, it does not mean that the model is error-free, the errors have simply not manifested with a given data set. Because the absence of errors does not guarantee that the model is correct, our attitude must be *destructive*, *i.e.* we must focus on *finding* and correcting errors rather than demonstrating model correctness. Several approaches can be followed to find errors:

- Establish a doubtful frame of mind and try to ‘crash’ the model by supplying various data sets. A successful model run with a given data set must be viewed as a failure to locate errors.
- Outside ‘doubters’ may be incorporated. They should understand the model’s intention but prior involvement in the model development is not a prerequisite for such collaborators.
- Conducting ‘walk through’ sessions by manually emulating the execution of the model.
- Execute the model with a single entity created, and follow it through the model with a tracing facility. The latter is a feature included in most simulation packages. Note that a single entity may never follow certain routes through the model due to logic, so that these excluded paths are never followed. The logic may be modified for the moment to force the model execution through those routes. A complete model execution may also be traced, but it becomes difficult to keep track of all entities in a complex model because of the parallel execution of the model.

- The model can be executed with certain parameters set to certain values, so that the effect on the model can be anticipated.
- Animation can be used to verify a model's execution.

2.9 Validate the model

Verification allows us to confirm that we have built the model right. In contrast, validation allows us to confirm that we have built the *right model* for the particular objective(s) of the study (Law, 2015). A verified model has code that executes correctly, given the tests it has been subjected to. There may be latent errors in the model code not identified by verification tests. A valid model is an approximate but adequate representation of the real-world process being modelled.

Verification and validation are interrelated: when verifying, the model is partially validated in terms of correct output. Verification is first done, and some validation is often required to demonstrate that a model is verified. For example, a model gives the correct output when all input is made deterministic, and the output can be calculated. The relationship can be visualised with a Venn diagram as shown in Figure 2.5.

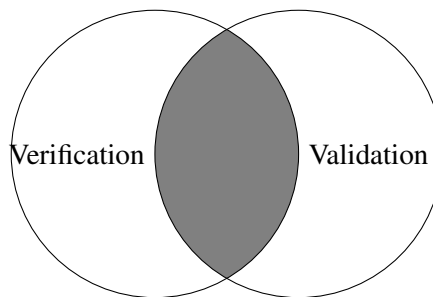


Figure 2.5: Verification and validation

Three questions must be considered during validation (Law and Kelton, 2000):

Conceptual validity:	Does the model represent the real world system adequately?
Operational validity:	Can the model's generated data be associated with the real world system's behavioural data?
Credibility:	Does the model's end-user have confidence in the model's results?

From this it follows that validation takes place from three different perspectives (Law and Kelton, 2000):

The analyst	Design and conduct confidence-building tests.
The technical evaluator	Reviews the technical data and information provided by the modeller.
The user and/or decision-maker	Usually an outsider who does not understand the validation tests executed; they only require that the model maps onto the real world.

The Handbook of Simulation (Banks, 1998) presents 75 verification, validation and testing techniques. Of these, the informal techniques like auditing, desk checking, face validation, walk-through sessions, inspections and reviews are arguably the most intuitive, easiest and efficient in terms of detecting most errors/discrepancies in models.

Formal statistical techniques like the paired t-test require that some assumptions be taken into account, which are not always practical. An example is the assumption of normality of the test sets. One can thus not compare simulation output of the model with that of the real world

unless there is reason to believe that the two data-sets are normally distributed. In cases where this is not possible, the non-parametric tests like the Wilcoxon rank-sum test can be used. In this case, the equality of the means of two non-normal continuous distributions of which samples are independent, is tested (Walpole and Myers (1993), pp. 634–635).

A model must exhibit reasonableness, measured against the following factors:

Continuity:	If small changes are made to input parameters, consequent small and appropriate changes should be reflected by the model's output and variables.
Consistency:	Similar runs of the model should reflect similar results – the output should not be influenced to a large extent because a random number generator seed has changed. (Tecnomatix has several random number streams; the modeller may select the stream number and seed.)
Degeneracy:	The model should reflect the removal of one or more features of the model. If, for example, there are two instances of a certain resource and one is removed, we expect less output from the set of resources, and a possible overload of the remaining resource.
Absurd conditions:	If absurd conditions are introduced, the model should not necessarily produce equally absurd outputs. Absurd conditions should also not arise during model execution, for example negative mass, negative times. Variables should always be in their range where applicable – if we know that the start-up time for a machine is at least x minutes, then the variable should never be less than this value.

A model should also be tested at its extremes, that is at the minimum and maximum limits that certain variables may assume.

Validation is done at various stages of the simulation project. It starts when the concept model is developed, it is done while and after the computer model is developed, and leads to credibility once the model's results are accepted.

2.10 Rework the model where necessary

It will almost always be necessary to rework the model after verification and validation. After the changes have been made, the model must again be verified and validated. This is an iterative process, which is repeated until all parties concerned deem the model to be acceptable.

A word of advice may be added: the author has experienced that it is well worth the effort and time to ensure that the model is widely accepted and 'correct' before commencing with subsequent steps in the project. These steps are time-consuming and labour intensive. If any one or more of them are executed and it is found that the model is not working properly, a great deal of time and money are wasted in reworking the model and regenerating output data.

2.11 Execute the production runs

Refer to Section 5.1 (p. 59), Section 5.3 (p. 66) and Section 5.4 (p. 69). Note that to do validation, preliminary production runs are needed.

2.12 Perform the statistical analysis

Refer to Section 5.1 (p. 59), Section 5.3 (p. 66) and Section 5.4 (p. 69).

2.13 Model modification and scenario analysis

Modify the model for alternatives where required and repeat Steps 2.8 to 2.12. The output analysis must be repeated for a modified model.

2.14 Document the study concurrently

Although this step is listed as one of the last steps in the simulation process, the study must be documented right from the start. Ideally, only final touches should be required at this point, apart from adding the results and their interpretation, conclusions and recommendations. Experience has taught that it is good practice to update the simulation report after each step is completed while the actions involved in the particular step are fresh in memory. It is very difficult to recall decisions and their rationale after several months of other simulation activities.

2.15 Implement the results of the study

This step can be the responsibility of the client only, or it can be in co-operation with the modeller(s). Third party consultants may also be used for certain aspects of the implementation, *i.e.* human resources and/or issues with unions.

2.16 Maintenance, feedback, monitoring and refining

The modeller(s) should monitor that the implementation is maintained and must obtain feedback from the client to evaluate the success of the study. This normally leads to refining the model where possible and to subsequent improvement in their level of experience.

What proportion of budgeted project time should be spent on each of the steps above? Various sources recommend different time allocations to the steps above. The '40-20-40' rule is suggested by Pegden, R.E. Shannon, and R. Sadowski (1995), which states that 40% of the overall project time should be devoted to steps 1 to 7, 20% to step 8 and 40% to steps 9 to 19. R. Shannon (1975) suggests a different allocation, but the central message is that model translation (computer model development) is not the major effort of a simulation study, there are more important issues that deserve more time.

The steps above set the most formal requirement for a simulation study; the analyst should however judge to what extent each should be executed by tailoring where necessary. It will hardly be possible to follow the steps in strict sequential order – some steps may be conducted concurrently, especially when working in a project team. Some members may, for example, collect input data while others may start developing a concept model, which may impose new and/or different requirements for the input data. A study may often have a false start, in which case much of the work in some steps must be abandoned, or some rework must be done.

These are some realities that the analyst may face, and one should try to plan a simulation study in terms of these potential problems: the study would usually take longer than planned, cost more and require more effort.

P Point to ponder: Suppose you lead a team doing a simulation study for a large enterprise, spanning over a few months. How would you coordinate each team member?

Survival kit:

Ensure you have the following in your assessment pack:

- 1** Know the steps in a simulation study in broad terms.
- 2** Know the objectives drivers of a simulation study (Section 2.1).
- 3** Know the six Ms guiding industrial engineers (Table 2.1).

-
- 4 Know the simulation performance measure types and give examples (Table 2.2).
 - 5 Know the types of system boundaries (Section 2.3).
 - 6 Create and draw a concept model (Section 2.4).
 - 7 Verification and validation are very important (Sections 2.8 and 2.9).

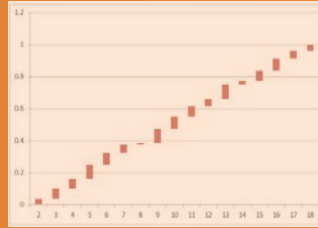
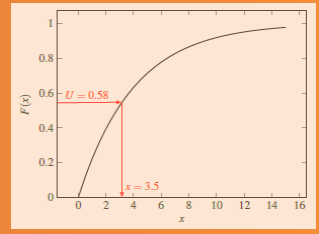
Random number generation

Basics of random number generators

Generating random variates

The inverse transform: continuous case
The inverse transform: discrete case
Monte-Carlo simulation

$$\begin{aligned}
F(x) &= \int_0^x f(z) dz \\
&= \int_0^x \frac{1}{\beta} \exp\left(\frac{-z}{\beta}\right) dz \\
&= -\exp\left(\frac{-z}{\beta}\right) \Big|_0^x \\
&= 1 - \exp\left(\frac{-x}{\beta}\right) \\
&= U. \\
X &= F^{-1}(U) \\
&= -\beta \ln(1 - U).
\end{aligned}$$



0.71464	0.41472	0.45395	0.20527	0.43824
0.72700	0.95528	0.93084	0.92111	0.25805
0.36477	0.94243	0.17866	0.36138	0.00827
0.46218	0.20972	0.22791	0.65858	0.87798
0.36458	0.49537	0.43067	0.65950	0.37429
0.26494	0.17691	0.51296	0.10539	0.71491
0.24901	0.72310	0.83471	0.64589	0.18068
0.27080	0.02045	0.39797	0.51331	0.95084
0.14335	0.24891	0.58877	0.91678	0.20341
0.39069	0.71483	0.95367	0.23682	0.21499
0.28527	0.30922	0.23930	0.67698	0.88679
0.05985	0.62431	0.65183	0.00875	0.52420
0.57042	0.16075	0.93422	0.89802	0.44419
0.55230	0.03058	0.35627	0.19582	0.14914
0.58464	0.31230	0.62784	0.77463	0.05814

3. Random numbers and random variates

THIS chapter covers the topics of random number generation and random variate generation. We need random numbers to induce (pseudo)randomness in our stochastic models, and we use these numbers when generating observations from distributions. These observations are used to replicate stochastic events, for example the inter-arrivals of entities in a process. The content of the chapter covers the main principles only; for in-depth discussions see for example Banks (1998) and Law (2015).

3.1 Random number generation

Stochastic processes originate due to randomness. *Randomness* is a thought-provoking concept, also from a philosophical point of view. It leads to uncertainty, which in turn can be classified as *objective uncertainty* and *subjective uncertainty*. Objective uncertainty is due to inherent randomness (in a process or system), while subjective uncertainty follows from lack of information (Denardo, 2002).

Since random numbers form the core of a stochastic simulation, we must investigate how a simulation software program (or any computer program) generates random numbers. This exercise seems to be unnecessary, because the general response is ‘Well, the computer simply does it’. A computer only follows instructions, and a ‘program’, ‘method’ or ‘recipe’ is therefore required to generate random numbers. Because of this ‘recipe’ followed, the random numbers generated are not truly random, and we use the term ‘pseudo-random’ when referring to these

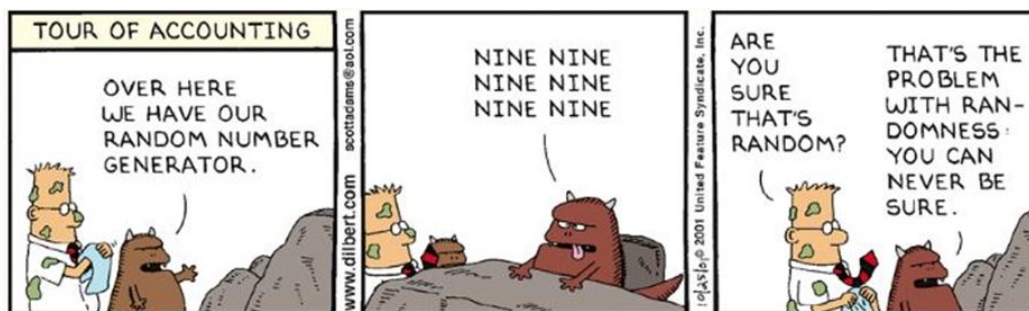


Figure 3.1: Randomness is not always well-understood

computer-generated numbers.

3.2 Basics of random number generators

There exist many types of random number generators (RNGs), for example

- the midsquare method,
- lagged Fibonacci generator (Knuth, 1997),
- linear congruential generators (LCGs) (based on the work of Derrick Lehmer (see Law (2015)), and the
- permuted congruential generator (O'Neill, 2014).

The LCGs produce a sequence of integers Z_i using

$$Z_i = (aZ_{i-1} + c) \bmod m \quad (3.1)$$

where m is the modulus, a is a multiplier, c is an increment and Z_0 is the seed or starting value. It is required that $m > 0$, $a < m$, $c < m$ and $Z_0 < m$. These Z_i are then converted to $U(0,1)$ numbers using $U_i = Z_i/m$. The LCG in (3.1) is *recursive*, Z_1 is thus determined by Z_0 , Z_2 is determined by Z_1 , and so on, which means that once we have selected the constants, the random number stream is completely defined. These random numbers are therefore called *pseudorandom* numbers. It is also necessary to specify Z_0 , the *seed number* of the RNG.

- **Example 3.1** Generate five random numbers using (3.1) with $a = 3$, $c = 0$, $m = 23$ and $Z_0 = 7$. Table the resulting calculations.

Answer:

i	Z_i	$U_i = Z_i/m$
0	7	0.304
1	21	0.913
2	17	0.739
3	5	0.217
4	15	0.652

Z_1 for example, was calculated by $Z_1 = (aZ_0 + c) \bmod m = (3 \times 7) \bmod 23 = 21$.

This illustrates the ‘recipe’ mentioned earlier in this section.

Random number generators have important characteristics, some are

1. They should be $U(0,1)$ distributed.
2. They can take only discrete values due to integer division, which yields the numbers (*when sorted*) $\frac{1}{m}, \frac{2}{m}, \dots, \frac{m-1}{m}$. They are thus not perfectly continuous.
3. A random number stream is periodic, *i.e.* a set of numbers is generated before the stream starts to repeat itself. The length of this period is at most m , since $0 \leq Z_i \leq m-1$.
4. It can degenerate. This means that the same random number is generated over and over. The mid-square method is a good example of a random number generation method that can cause quick degeneration.
5. A seed number is required to initiate the RNG.
6. The sequence of numbers generated should be reproducible for debugging purposes and also to evaluate different alternative systems on common grounds.

The choice for the various parameters has been researched extensively, as well as the development of new RNG methods, see for example the ‘combined multiple recursive generator’ (CMRG) in Kelton, R. Sadowski, and D. Sadowski (2002), pp. 475–476.

Here is a cool random number generator at <https://www.youtube.com/watch?v=1cUUfMe0ijg>.

The site <https://www.random.org/> claims to generate true random numbers.

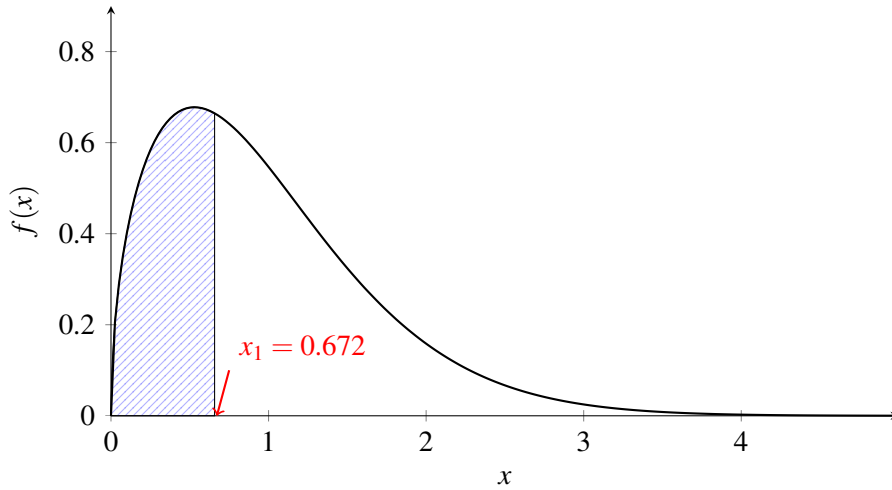


Figure 3.2: The area under the Weibull function $f(x, \alpha, \beta) = f(x, 1.5, 0.9)$ is associated with the value of $U = 0.37$. The associated x -value is $x_1 = 0.672$.

P Point to ponder: Our world seems to be driven by randomness. You cannot tell when the next client will arrive at your fruit stall, or what the client will buy. Is it really randomness or do we only perceive it?

3.3 Generating random variates

Referring to the discussion above, one can ask why we need random numbers? This question can be answered by asking: How does a simulation software program generate random observations from the various statistical distributions? There are several methods, but only the inverse transform will be discussed.

3.3.1 The inverse transform: continuous case

The *inverse transform* method is based on the use of the cumulative distribution function F of a variate, and the assumption is that 1) the inverse of F , namely F^{-1} , exists and 2) can be determined.

The inverse transform method for continuous distributions works as follows: We know how to generate a random number U , and $0 < U < 1$. But $0 \leq F(x) \leq 1$. So why not generate a U value and set it equal to $F(x)$? But what then? Let us first find $F(x)$ with

$$F(x) = \int_{-\infty}^x f(t) dt \quad (3.2)$$

$$= U. \quad (3.3)$$

The value of $U = 0.37$ is shown as an area in Figure 3.2. The corresponding value is $X_1 = 0.672$.

In general, we generate a random number U from $U(0,1)$, and return $X = F^{-1}(U)$. This can be proved as follows:

Let F_X denote the distribution function of $X = F^{-1}(U)$. Then

$$\begin{aligned}
 F_X(x) &= P\{X \leq x\} \\
 &= P\{F^{-1}(U) \leq x\} \\
 &= P\{F(F^{-1}(U)) \leq F(x)\} \text{ (because } F(x) \text{ is a monotone} \\
 &= P\{U \leq F(x)\} \quad \text{increasing function of } x \\
 &= F(x).
 \end{aligned} \tag{3.4}$$

We thus determine the cumulative distribution function ($F(x)$) of a distribution from which we want to sample, then we ‘randomly’ select a value between 0 and 1 and insert it into the obtained inverse cumulative distribution function ($F(x) = U$) which returns an observed value $X = F^{-1}(U)$. By repeating this ‘many times’, the desired distribution is generated.

The mechanism is graphically illustrated in Figure 3.3 for an exponential distribution with $\beta = 4$. A random number on the vertical axis is selected, and at that height, a horizontal line is projected to the curve of $F(x)$, then a vertical line is followed downwards to the x -axis. The value at the intersection is the observation generated from the distribution. Note that this example uses $X = -\beta \ln(1 - U)$.

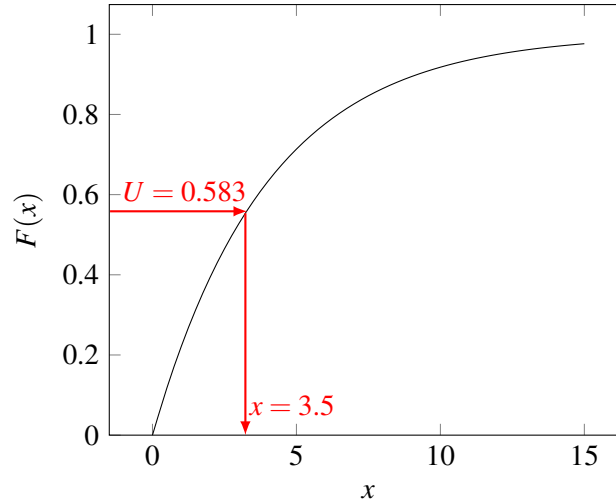


Figure 3.3: Random variate generation through an inverse transform (continuous)

■ **Example 3.2** Suppose a random variate X is exponentially distributed with probability density function

$$f_X(x) = \frac{1}{\beta} \exp^{-x/\beta}, \quad x > 0, \quad \beta > 0. \tag{3.5}$$

The function can be inverted to obtain an expression for X . We use the result from (3.4): first determine the integral of $f(x)$, then set it equal to U , and solve for x . We denote x as X because

it is a variate.

$$\begin{aligned}
 F(x) &= \int_0^x f(z) dz \\
 &= \int_0^x \frac{1}{\beta} \exp\left(\frac{-z}{\beta}\right) dz \\
 &= -\exp\left(\frac{-z}{\beta}\right) \Big|_0^x \\
 &= 1 - \exp\left(\frac{-x}{\beta}\right) \\
 &= U. \\
 X &= F^{-1}(U) \\
 &= -\beta \ln(1 - U).
 \end{aligned} \tag{3.6}$$

P Points to ponder:

- In (3.5) it is shown that $x > 0$, but the result in (3.6) has a negative in the equation. Can you explain?
- In (3.6), analysts often use $-\beta \ln(U)$. Is that valid?

■ **Example 3.3** Determine the inverse transform of the Weibull distribution $f(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-(x/\beta)^\alpha}$.

$$\begin{aligned}
 F(x) &= \int_0^x f(t) dt \\
 &= \int_0^x \alpha\beta^{-\alpha}t^{\alpha-1}e^{-(t/\beta)^\alpha} dt \\
 \text{Let } u &= \left(\frac{t}{\beta}\right)^\alpha \\
 \frac{du}{dt} &= \alpha\beta^{-\alpha}t^{\alpha-1} \\
 \text{If } t &= 0 \text{ then } u = 0, \\
 \text{if } t &= x \text{ then } u = \left(\frac{x}{\beta}\right)^\alpha. \\
 F(x) &= \int_0^{(x/\beta)^\alpha} e^{-u} du \\
 &= 1 - e^{-(x/\beta)^\alpha} \\
 &= U. \\
 X &= \beta(-\ln(1 - U))^{1/\alpha}.
 \end{aligned}$$

■ **Example 3.4** Determine the inverse transform of the continuous uniform distribution on $[a, b]$.

$$\begin{aligned}
 f(x) &= \frac{1}{b-a} \\
 F(x) &= \int_a^x \frac{1}{b-a} dt \\
 &= \frac{x-a}{b-a} \\
 &= U. \\
 X &= a + (b-a)U.
 \end{aligned}$$

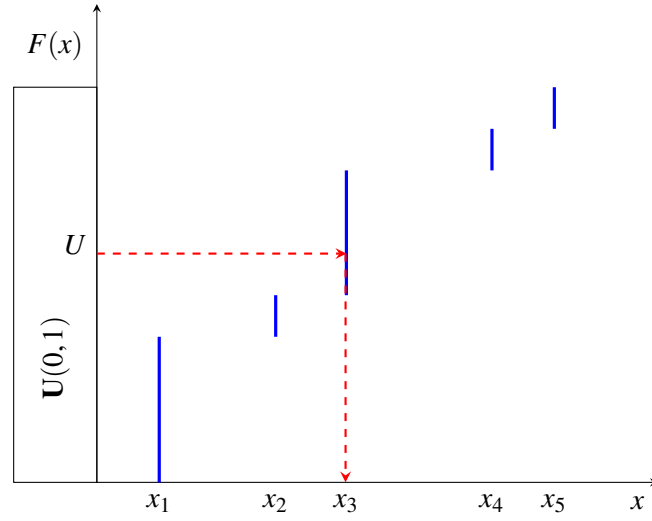


Figure 3.4: Random variate generation through an inverse transform (discrete)

P Point to ponder: Why should the result of the continuous uniform distribution inverse transform be intuitive?

3.3.2 The inverse transform: discrete case

Assume we have a probability mass function

$$f(x) = \begin{cases} p_0 & \text{if } x = x_0 \\ p_1 & \text{if } x = x_1 \\ \vdots & \vdots \\ p_j & \text{if } x = x_j, \end{cases}$$

and the discrete inverse transform must be determined. The cumulative distribution function for the probability mass function $f(x) = P(X = x)$ is $F(x) = \sum_{t \leq x} f(t)$. Take the $\text{Unif}(1,6)$ distribution for example: $F(3) = f(1) + f(2) + f(3) = 1/6 + 1/6 + 1/6 = 0.5$. Now generate $U \sim U(0,1)$, and determine the smallest positive integer i so that $U \leq F(x_i)$ and return $X = x_i$. Formally, we use

$$X = \begin{cases} x_0 & \text{if } U < p_0 \\ x_1 & \text{if } p_0 \leq U < p_0 + p_1 \\ \vdots & \vdots \\ x_j & \text{if } \sum_{i=0}^{j-1} p_i \leq U < \sum_{i=0}^j p_i \end{cases}$$

as an algorithm. The inverse transform approach for discrete random variates is graphically illustrated in Figure 3.4.

The variates are thus generated in discrete steps, while the probability of creating a variate is proportional to the probability mass function.

■ **Example 3.5** Using $U_1 = 0.33$, $U_2 = 0.65$ and $U_3 = 0.45$, generate three observations from

$$f(x) = \begin{cases} 0.35, & x = 1 \\ 0.20, & x = 2 \\ 0.45, & x = 3. \end{cases}$$

Figure 3.4 has been adapted to the given $f(x)$, as shown in Figure 3.5.

1. $U_1 = 0.33 < 0.35$, return $X = 1$.
2. $0.35 + 0.20 = 0.55 < U_2 = 0.65 < 0.35 + 0.20 + 0.45 = 1.00$, return $X = 3$.
3. $0.35 < U_3 = 0.45 < 0.35 + 0.20 = 0.55$, return $X = 2$.

The observations 1, 2, and 3 are returned in any order, as dictated by the random numbers U_i . Once ‘many’ observations are generated, the proportions of 1, 2 and 3 should approach 0.35, 0.20 and 0.45 respectively.

P Points to ponder: How would you make the process applied in the example efficient? Here ‘efficient’ means the computer makes the least number of comparisons when executing the process.

■ **Example 3.6** The discrete, uniform distribution $U(a, b)$, $a < b$ and both integers, is defined by

$$f(x) = \begin{cases} \frac{1}{b-a+1} & \text{if } x \in \{a, a+1, \dots, b\} \\ 0 & \text{otherwise.} \end{cases}$$

To sample from this distribution, we use $X = a + \text{INT}((b-a+1)U)$, where ‘INT’ returns the integral part of a number. It is also denoted by $\lfloor x \rfloor$. Examples: $\text{INT}(0.9) = 0$, $\text{INT}(6.2) = 6$.

P Points to ponder: How would you generate observations from a Bernoulli distribution with p.m.f. $f(x) = p^k(1-p)^{1-k}$ for $k \in \{0, 1\}$, with p the probability of success? And the Poisson distribution with p.m.f. $f(x) = e^{-\lambda} \lambda^x / x!$, $x = 0, 1, 2, \dots$?

Bringing it together

Here is an example to show how the generation of random numbers and random variates are done.

A random number generator and a distribution are given below. Let $z_0 = 54$ and $k = 2$, and create three random variables from the distribution, starting with z_1 .

$$\begin{aligned} z_{i+1} &= (39z_i + 71) \bmod 513 \\ f(x) &= kx^{-k-1}, \quad x \geq 1. \end{aligned}$$

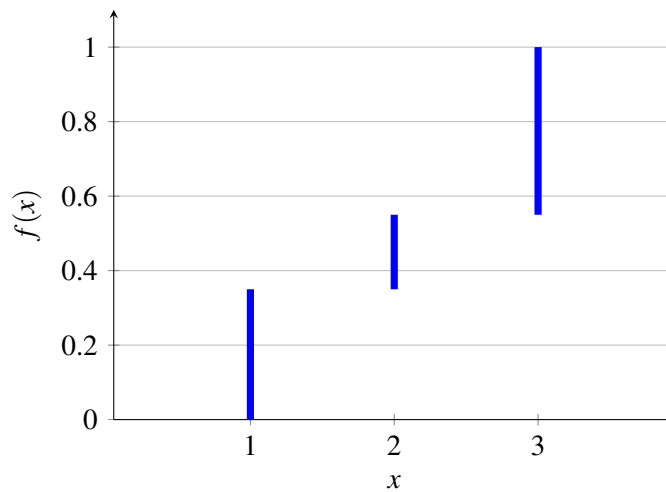


Figure 3.5: Random variate generation through an inverse transform (discrete)

Answer:

Use the inverse transform:

$$\begin{aligned} F(x) &= \int_1^x f(y)dy \\ &= \int_1^x 2y^{-3}dy \\ &= 1 - x^{-2}, \end{aligned}$$

and the inverse then is $X = (1 - U)^{-1/2}$. Starting with $z_0 = 54$ we construct the following table, with the required three observations shown in the last column.

$z_1 =$	125	$U_1 = Z_1/513 =$	0.243	$X_1 = 1.150$
$z_2 =$	329	$U_2 = Z_2/513 =$	0.641	$X_2 = 1.670$
$z_3 =$	77	$U_3 = Z_3/513 =$	0.150	$X_3 = 1.085$

3.3.3 Monte-Carlo simulation

Now that we know how random numbers are generated, we can study a specific application in the static, stochastic simulation domain. This application is called *Monte-Carlo (M-C) simulation*. It originated during World War II and was developed by nuclear scientists to solve difficult mathematical problems, including double and triple integrals. M-C simulation can be illustrated as follows: Suppose we want to determine

$$I = \int_a^b f(x)dx, \quad -\infty < a < b < \infty, \quad (3.7)$$

and I does not have an exact solution. We could possibly determine I with numerical methods, but for illustration, M-C simulation is used. Let $X \sim U(a, b)$, then $g(x) = 1/(b - a)$. Define

$$Y = (b - a)f(X),$$

then

$$\begin{aligned} E[Y] &= E[(b - a)f(X)] \\ &= \int_a^b (b - a)g(x)f(x)dx \\ &= \int_a^b f(x)dx \\ &= I. \end{aligned}$$

If we can estimate the $E[Y]$, we have an approximation for I . To do so, we first transform the limits $[a, b]$ of the integral to the region $(0, 1)$ and obtain a function $f(y)$, then draw pseudorandom numbers U_i in the place of y and determine $f(U_i)$. The variable x has been transformed into the variable y , which is defined on $(0, 1)$. We do this n times (simulating), and estimate an approximation for I , as follows:

$$\bar{Y} = \frac{(b - a) \sum_{i=1}^n f(U_i)}{n}. \quad (3.8)$$

Suppose in (3.7), $a = 0$ and the upper limit is infinity, then we transform the range to $[0, 1]$ by letting $y = 1/(x + 1)$, then $dy = -dx/(1 + x)^2$ and $I = \int_0^1 h(y)dy$. Different transformations are needed for the integral boundaries. The cases are shown in Table 3.1, with $f(x) = e^{x^2}$ as example.

Table 3.1: Monte-Carlo simulation transformations

	Bound- aries	Transformation	Transformed integral
Case 1: $I = \int_a^b f(x)dx = \int_a^b e^{x^2} dx$	$[a,b]$	$y = \frac{x-a}{b-a}$ $x = a + (b-a)y$ $x = a: y = 0$ $x = b: y = 1$ $\frac{dy}{dx} = \frac{1}{b-a}$ $dx = (b-a)dy$	$I = \int_0^1 (b-a)f(y)dy$ $I = \int_0^1 (b-a)e^{(a+(b-a)y)^2} dy$
Case 2: $I = \int_0^\infty f(x)dx = \int_0^\infty e^{-x^2} dx$	$[0,\infty]$	$y = \frac{1}{x+1}$ $x = \frac{1}{y} - 1$ $x = 0 : y = 1$ $x \rightarrow \infty: y = 0$ $\frac{dy}{dx} = \frac{-1}{(x+1)^2}$ $= -y^2$ $dx = -\frac{dy}{y^2}$	$I = -\int_1^0 \frac{f(y)}{y^2} dy$ $= \int_0^1 \frac{f(y)}{y^2} dy$ $I = \int_0^1 \frac{e^{-((1/y)-1)^2}}{y^2} dy$
Case 3: $I = \int_{-\infty}^\infty f(x)dx$ $= \int_{-\infty}^\infty e^{x^2} dx$ $= \int_{-\infty}^0 e^{x^2} dx + \int_0^\infty e^{x^2} dx$ $= I_1 + I_2$ (We address I_1 here).	$[-\infty,0]$	$y = \frac{1}{1-x}$ $x = 1 - \frac{1}{y}$ $x \rightarrow -\infty: y = 0$ $x = 0 : y = 1$ $\frac{dy}{dx} = \frac{1}{(1-x)^2}$ $= y^2$ $dx = \frac{dy}{y^2}$	$I_1 = \int_0^1 \frac{f(y)}{y^2} dy$ $I_1 = \int_0^1 \frac{e^{(1-1/y)^2}}{y^2} dy$ I_2 is according to Case 2 .

To do the actual simulation, one would generate ‘many’ values for $y \sim U(0,1)$ and calculate *many values* of I in column 4, then average them. This will give an *estimation* of the integral.

Example of implementation: Take the transformed integral of Case 2 and estimate its value

by generating 10 000 observations, then take the average. The exact answer is $\sqrt{\pi}/2 = 0.886$.

Obs. no.	$y = U$	$\frac{e^{((1/y)-1)^2}}{y^2}$
1	0.981	1.039692
2	0.753	1.583729
3	0.915	1.184991
4	0.674	1.742165
...
10 000	0.412	0.772859
Ave. =		0.88945

Survival kit:

Conquer Mount Randomness easily by carrying the following in your assessment pack:

- 1 Understand the need for and the principle of generating random numbers.
- 2 Know the properties of random number generators.
- 3 Be able to generate random numbers using a given formulation.
- 4 Do the inverse transform for both the discrete and continuous cases.
- 5 Do Monte-Carlo simulation, given any of the three cases.

Sources of input data

No data available

Data available

Empirical distributions

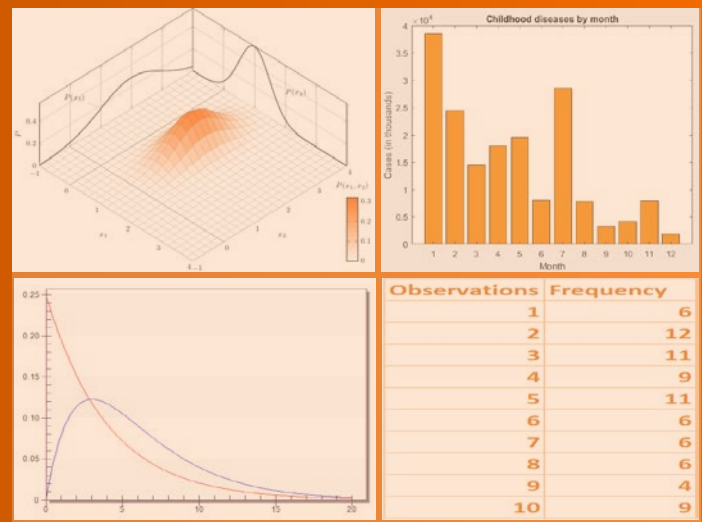
Theoretical distributions

Selecting an input distribution

The Chi-squared test

The Kolmogorov-Smirnov test

Examples of distribution fits



4. Input data analysis

SOME or all of the events in a real-world system are driven by a mechanism of randomness. The nature of these mechanisms must be established and quantified to conduct a simulation study of the system under investigation. The major concepts in this process are discussed in this section. These concepts comprise the acquisition, preparation and definition of input data. Input data is very important and must be carefully specified, as input data sets are the *drivers* of a simulation model.

4.1 Sources of input data

Input data must be acquired either for an existing system or a proposed system. The sources of input data could be any one or more of the following:

1. Historical records – from information systems of production, quality control, time studies, reports:
 - Data may or may not be up to date.
 - Data is usually voluminous.
 - The format of the data may require special computer programs for extraction and translation.
 - The history and context of the data should also be investigated – when was it collected, how, why and by whom. See Leemis (2001) and Banks (1998) (pp. 59-60) for particular annoyances with input data collection.
2. Observational data:
 - Observe a system in operation and gather the data personally.
 - May induce the Hawthorne effect.
 - Applicable in existing systems.
 - It is often a time-consuming task to collect data.
 - Observations may reflect only a snapshot of seasonal trend.
 - A secondary advantage is that while observing, the analyst may find suggestions for process improvement.
3. Similar systems:
 - Many designs are variations of others (for example Durban harbour versus Cape Town harbour).
 - Validation may be difficult.

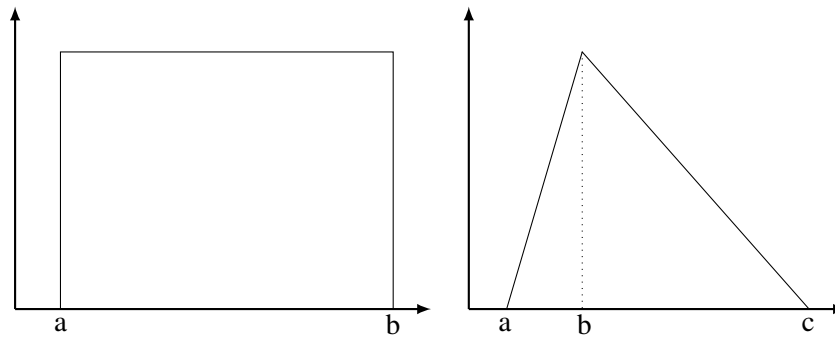


Figure 4.1: The uniform and triangular distribution

4. Operator estimates:

- Can be utilised if insufficient time is available to do a complete study.
- Research has shown that people are very poor estimators of parameters/events when they are highly familiar with them – extreme cases are usually forgotten and most recent ones overemphasised.

5. Vendor's claims:

- Provide estimates of the operating characteristics of new equipment.
- Estimates are invariably highly optimistic.
- Similar equipment currently in use at other clients may be compared to temper the estimates.

6. Designer estimates:

- May be the only source of data (new system).
- Estimations can also be far off, as the designer expects that the system will operate as intended.

7. Theoretical considerations – the analyst may use accepted theoretical principles:

- Mean Time Between Failures (MTBFs) of electronic equipment are usually Weibull distributed.
- Time between arrivals is usually exponentially distributed.
- Extreme care must be exercised when selecting a theoretical distribution, especially in terms of the range. For example, time cannot be negative, so the normal distribution cannot be used to represent times, unless it is shifted to the positive range. This will be discussed later in this section.

4.2 No data available

Often, no data is available for a simulation study, typically when a new system or process is designed. Several procedures to deal with this situation are suggested in the literature, including the use of vendor- and designer claims as discussed previously. These claims may be extended to be a mean with a deviation or tolerance, for example $x \pm 20\%x$, or a range $[x_1, x_2]$ where x_1 is the minimum and x_2 is the maximum. When a range is specified, the uniform distribution is used to obtain random observations from the range, but simulation parameters are rarely uniformly distributed.

The most useful and convenient way to deal with this case is to use the triangular distribution. A minimum, mode (most likely value) and maximum estimators are used instead of a single estimation or a mean with tolerance. There is, of course, possible subjectivity included in such estimations. The shapes of the distributions are shown in Figure 4.1.

4.3 Data available

There are two schools of thought regarding the use of available data. One approach followed is to sample directly from the available empirical distribution, while the second is to sample from the theoretical distribution if the data fits a certain distribution.

The implications are as follows: The empirical distribution allows for replicating the past, but no other values outside the range of the observed data are used. The theoretical distribution may return values from the tails, which are bigger or smaller than the observations of the sample, which may be inaccurate if the fit is relatively poor.

Law (2015) suggests the use of one of the following alternatives, in increasing order of desirability:

1. Use the observed data values themselves in the simulation. Whenever a value is needed, for example a machine downtime, a value is extracted (at random) from the set of observations.
2. An empirical distribution is deduced from the available data, and sampling is done from this distribution.
3. A theoretical distribution is fitted to the data via any standard technique, and sampling is done from this distribution during the simulation run.

Alternative 1 is useful for validation purposes. Arguments in favour of alternative 3 (which is supported by the second school of thought as discussed earlier) are:

- An empirical distribution may have irregularities, whereas the theoretical distribution is smooth and provides information on the overall underlying distribution.
- A theoretical distribution is a compact way of representing data values compared to the storage required by empirical data.
- The theoretical distribution allows for values that are not revealed by the empirical distribution.

Using the empirical and theoretical distributions for data specification is subsequently discussed.

4.4 Empirical distributions

We construct a continuous, empirical distribution from a data sample using the histogram; then we specify, per range/bin/category, the relative proportion of occurrences. Consider the following data set of repair times (continuous data):

2.999	6.486	7.198	0.306
4.327	4.336	5.928	8.506
1.508	1.882	4.847	6.336

The proportions for the n observations can be calculated using the empirical distribution, which is given by

$$F_n(t) = \frac{\text{Number of observations} \leq t}{n}. \quad (4.1)$$

Applying (4.1) yields the values in Table 4.1. Note that the observations were first sorted in *ascending order*.

Suppose we want to sample from this distribution, then we first draw a random number U , say $U = 0.6$, then we find the t (the bin upper limit) in the table for which the maximum of $F_n(t)$ would give $F_n(t) \leq U$. For $U = 0.6$, it would return $t = 5.928$. Strictly speaking, in the continuous case we should interpolate, as shown in Figure 4.2. The value to return is $t = 5.0632$.

The discrete case is treated similarly, but no interpolation is required. An example data set and calculations are shown in Table 4.2.

Table 4.1: Sample data and the continuous empirical distribution

Observations	$F(t)$
0.306	0.083
1.508	0.167
1.882	0.250
2.999	0.333
4.327	0.417
4.336	0.500
4.847	0.583
5.928	0.667
6.336	0.750
6.486	0.833
7.198	0.917
8.506	1.000

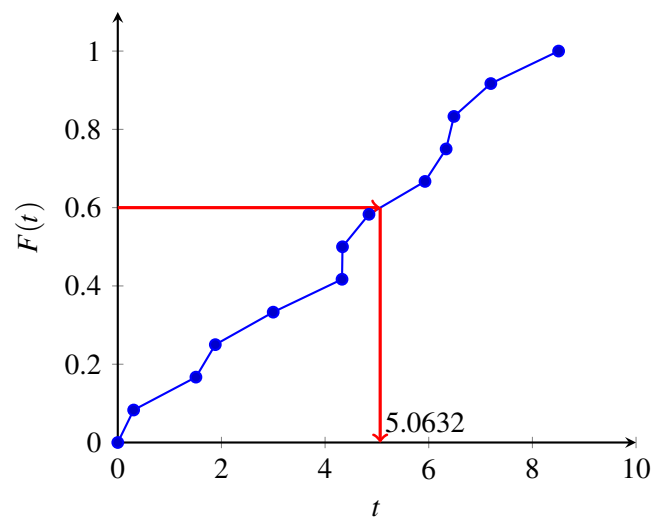


Figure 4.2: An empirical distribution and an observation sampled from it

Table 4.2: Example data set and the discrete empirical distribution

Data (original)	Data (sorted)	$F_n(x)$
3	1	1/12=0.083
5	2	3/12=0.250
2	2	3/12=0.250
8	3	4/12=0.333
6	5	7/12=0.583
5	5	7/12=0.583
7	5	7/12=0.583
5	6	8/12=0.667
2	7	10/12=0.833
1	7	10/12=0.833
9	8	11/12=0.917
7	9	12/12=1.000

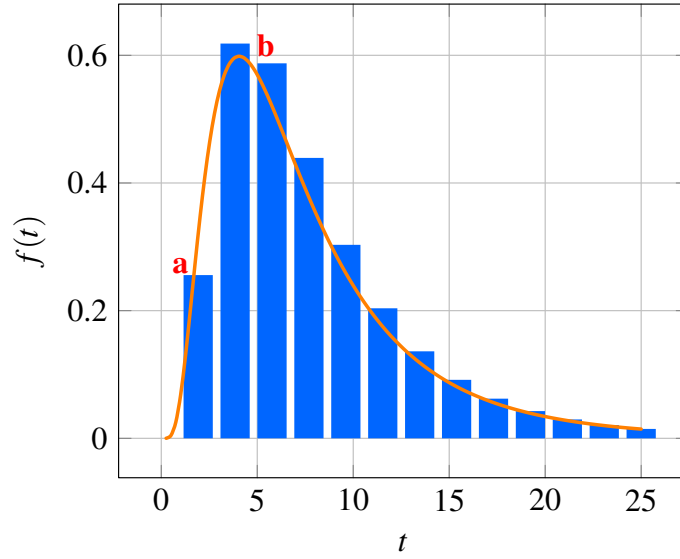


Figure 4.3: An empirical distribution and the true underlying distribution

For a $U = 0.55$, one would find the maximum $F_n(x)$ such that $F_n(x) \leq U$, which, from Table 4.2, would return $x = 5$. Suppose the data in Table 4.2, in column “Data sorted” is indexed by j , then any entry in the column can be denoted by x_j and the first entry is x_0 . Now, in general,

$$P(X = x_j) = F(x_{j-1}) \leq U < F(x_j). \quad (4.2)$$

If we apply (4.2) using Table 4.2 and $U = 0.55$, we see that $F(x_{j-1}) = F(3)$ and $F(x_j) = F(5)$, so $F(3) = 0.333 \leq 0.55 < F(5) = 0.583$. So $x = 5$.

Note the following drawbacks when using empirical distributions: Suppose we have a data set of repair times represented by the histogram in Figure 4.3 (blue bars), while the true, but unknown distribution is represented by the function in orange. If we use the data as is, and we sample from the empirical distribution, we shall find too few observations from the range defined by the first bar (marked with a red ‘a’), because the proportion represented by the first bar is smaller than the area under the curve in that range. Similarly, we shall find more observations from the range defined by the third bar (marked with a red ‘b’) than what we should. Again, we shall not be able to sample values less than the left edge of the first bar, or greater than $t = 25$. (Is it possible to get a bar height that is higher than the true distribution, as shown at ‘b’, or is the argument superfluous?)

4.5 Theoretical distributions

We could ‘fit’ a theoretical distribution to the observed data. It is proposed, through a hypothesis, that the observed data comes from a certain theoretical distribution. This hypothesis is then tested and a conclusion is made. There are many tests available, but we shall focus on the χ^2 (or chi-squared) and the Kolmogorov-Smirnov (K-S) goodness-of-fit tests. The two tests are compared in Table 4.3.

4.5.1 Selecting an input distribution

To specify a distribution, its parameters are needed – the β is the parameter of the exponential function, while μ and σ are the parameters of the normal distribution. If we observed a sample of IID random variables $X_1 \dots X_n$, we can use these to estimate the values of the parameter(s) of the proposed distribution.

Table 4.3: The χ^2 and K-S goodness-of-fit tests

χ^2 test	Kolmogorov-Smirnov test
a. Discrete and continuous data.	a. Only continuous data.
b. All distributions.	b. Only for these cases: <ol style="list-style-type: none"> 1. All parameters known. 2. Normal. 3. Exponential. 4. Weibull.
c. More than 100 data points, but over-sensitive to large number of data points.	c. ‘Any’ (?) number of data points.
d. Fundamental: Compare <i>frequencies</i> .	d. Fundamental: Compare <i>cumulative distributions</i> .
e. Test a hypothesis with critical value	e. Test a hypothesis with critical value following from a table in Law (2015). See Table 4.4.
f. The critical value is given by	The critical values are shown in Table 4.4.

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

These estimators are numerical functions of the observations. Several methods of estimation exist, among which are least squares-, unbiased- and maximum-likelihood estimators (MLEs). It is beyond the scope of this module to go into the detail of MLEs, but the interested reader is referred to Law (2015) for a thorough discussion.

Finding the MLEs for some distributions is difficult, and numerical solutions are sometimes used. Some important MLEs are listed in Table 4.5 (Law, 2015) at the end of this chapter.

Once a distribution is chosen to be representative of the data set, the quality of the fit must be evaluated. Although the fit will virtually never be exact, a measure of the deviation from the observed data set must be determined, especially if it appears that more than one distribution may be an acceptable fit.

One of the goodness-of-fit tests presented earlier (χ^2 , K-S) is used to assess if the observations X_1, X_2, \dots, X_n are an independent sample from a distribution with distribution function \hat{F} . The null hypothesis is

H_0 : The X_i ’s are independent random variables with distribution function \hat{F} .

The goodness-of-fit test is one technique of assessing the quality of fits, and the two tests mentioned are now presented.

4.5.2 The Chi-squared test

This test is applicable to *discrete* as well as *continuous* data, and is a formal comparison of a line graph or histogram with the fitted mass or density function. The following must be pointed out:

- Suppose the fitted distribution is divided into k intervals, and the test is done at level α , then the critical region is defined by $\chi_{k-m-1, 1-\alpha}^2$ if *all parameters of the fitted distribution are known*. This is shown in Figure 4.4.
- If m parameters were used ($m \geq 1$) to specify the fitted distribution, the degrees of freedom

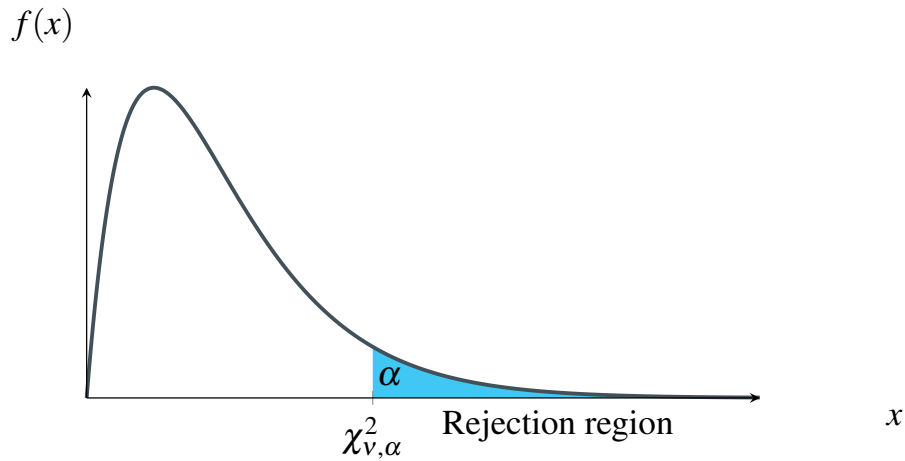


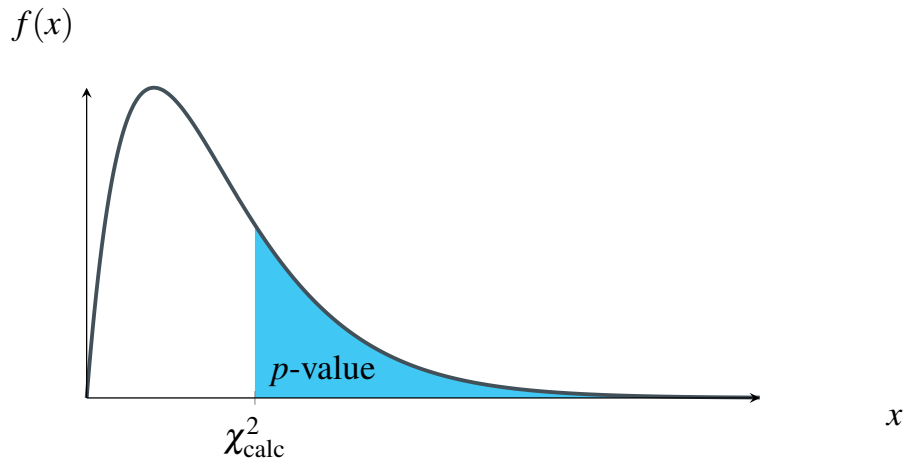
Figure 4.4: Critical regions for the chi-square test

- $v = k - m - 1$ are reduced by m , that is, H_0 will be rejected if $\chi^2_{\text{calc}} > \chi^2_{v, 1-\alpha}$. Typical parameters that could be estimated from the observed data are the mean and the variance.
- No simple and definite guidelines yet exist for the choice of the number and size of the intervals. Research has shown (Law, 2015) that it is good practice to use $k \geq 3$ and at least five observations (theoretical) per interval. If necessary, adjacent intervals are grouped until the number of observations per class exceeds five.
 - The chi-square test is not applicable to small samples because at least five observations per interval are needed as we desire reasonably large degrees of freedom.
 - The chi-square test is insensitive to small n , *i.e.* a small sample size.
 - It is however oversensitive to large n ; only small deviations will result in the rejection of H_0 .
 - A corresponding *p-value* can be calculated for a given/calculated χ^2 -value. This value is the area to the right of the calculated χ^2 -value, and has several interpretations:
 - It is the probability of obtaining a test statistic equal to or more extreme than the result observed, given H_0 is true.
 - It gives an indication of the quality of the fit, the larger p , *i.e.* the closer it is to 1, the *weaker* the evidence that one must *reject* H_0 , given H_0 is true. The converse is true, with a cut-off at α . Beyond α , one rejects the H_0 , and the conclusion is that the evidence against H_0 is strong. The concept of the *p-value* is shown in Figure 4.5. Also see Greenland et al. (2016) for a discussion on *p-values*, especially the misuse and wrong interpretation.
 - The test will be investigated in detail with the aid of examples during the module. The process is subsequently described.

The Chi-square goodness-of-fit test procedure

The test is executed by following this procedure:

1. Group the raw data in intervals if necessary; the frequency in each interval is required. An indication of the required number of intervals (continuous data) is given by Sturges's rule: $k = \lfloor 1 + 3.322 \log_{10} n \rfloor$, where n = number of data points or observations.
2. Determine the theoretical distribution you want to fit. Now estimate any parameters if necessary (use MLEs), typically the mean and variance. This will give the variable m a value of *e.g.* 1 or 2. The Poisson distribution has parameter λ , so when a Poisson distribution is fitted to a data set and λ has to be estimated from the data set, then $m=1$.

Figure 4.5: The p -value for the chi-square test

The normal distribution has parameters μ and σ , and if they are estimated from a data set, then $m=2$.

3. State the null hypothesis:

H_0 : The observed data is from the theoretical distribution $f(x)$ with mean μ and variance σ^2 . (This is the general statement).

4. Decide on the level of significance. It is usually 5%. (Note: The level of confidence is the complement, *i.e.* 100% - 5% = 95% confidence.)
5. Arrange the observed frequencies (O_i) of each class interval in a table. Add further class intervals to cover the definition range of the theoretical distribution, *e.g.* $[0, \infty)$. (The observed data will rarely cover the complete range).
6. Determine the expected frequencies (E_i) of each corresponding class interval and list them in the table. $E_i = np_i$, where n = number of observations in the raw data set, and p_i is the proportion for the i -th interval as returned by $f(x)$. The latter may be discrete or continuous. How would you treat the two possibilities? The values of continuous distributions are sometimes approximated by the mid-point of the class interval.
7. Important: Group the **expected** frequency classes if $E_i < 5$, so that $E_i \geq 5$. Group the corresponding O_i and determine the sum of the groups. The number of classes after the grouping gives the k -value.
8. Determine $\chi^2_{calc} = \sum_{i=1}^k \frac{(E_i - O_i)^2}{E_i}$.
9. Refer to the table of chi-square critical values (Table 6.2 in Appendix 6.9) and find $\chi^2_{k-m-1, 1-\alpha} = \chi^2_{crit}$.
10. If $\chi^2_{calc} > \chi^2_{k-m-1, 1-\alpha}$ we reject H_0 at the $\alpha * 100\%$ level of significance. If it is not the case, we have no sufficient evidence to reject H_0 . We **never** accept a hypothesis!

4.5.3 The Kolmogorov-Smirnov test

This test compares an empirical distribution function with the distribution function \hat{F} of the hypothetical distribution. The following points are important to note before we describe the procedure:

- The K-S test does not require the grouping of data, so no information is lost and interval selection is eliminated.
- K-S tests tend to be more powerful than chi-square tests.
- The range of applicability is more limited than that for chi-square tests. The original form

Table 4.4: Critical values for $c_{1-\alpha}$, $c'_{1-\alpha}$, $c''_{1-\alpha}$, $c^*_{1-\alpha}$

				1 - α					
Case	Test Condition (We reject H ₀ if:)			0.850	0.900	0.950	0.975	0.990	
All parameters known	$\left(\sqrt{n} + 0.12 + \frac{0.11}{\sqrt{n}}\right) D_n > c_{1-\alpha}$			1.138	1.224	1.358	1.480	1.628	
Normal N (X̄(n), S ² (n))	$\left(\sqrt{n} - 0.01 + \frac{0.85}{\sqrt{n}}\right) D_n > c'_{1-\alpha}$			0.775	0.819	0.895	0.955	1.035	
Exponential (X̄(n))	$\left(D_n - \frac{0.2}{n}\right) \left(\sqrt{n} + 0.26 + \frac{0.5}{\sqrt{n}}\right) > c''_{1-\alpha}$			0.926	0.990	1.094	1.190	1.308	
Weibull	$\sqrt{n}D_n > c^*_{1-\alpha}$		n	10	-	0.760	0.819	0.880	0.944
				20	-	0.779	0.843	0.907	0.973
				50	-	0.790	0.856	0.922	0.988
				∞	-	0.803	0.874	0.939	1.007

of the K-S test is valid only if all parameters of the hypothesised distribution are known and the distribution is continuous. The parameters may thus in the strict sense not be estimated from the data.

- The K-S test has been applied in its original form on both discrete and continuous distributions with estimated parameters, but this resulted in conservative tests. The probability of a Type I error will be smaller than specified, leaving a false impression.
- The K-S test should be used only on continuous distributions.
- The K-S test is valid for any sample size.

The Kolmogorov-Smirnov goodness-of-fit test procedure

The task is to compare the cumulative probability distributions of the observed data and the stated theoretical distribution.

1. State the null hypothesis H_0 : The observed data is from the theoretical distribution $F(x)$ with mean μ and variance σ^2 . (This is the general statement).
2. Define an empirical distribution function $F_n(x)$ from the observations $X_1, X_2, X_3, \dots, X_n$ as

$$F_n(x) = \frac{\text{number of } X_i \leq x}{n}.$$

$F_n(x)$ is a *right-continuous* step-function. List the values in a table.

Let $\hat{F}_n(x)$ be the fitted cumulative distribution function, *i.e.* the hypothesised distribution, then D_n is the K-S test statistic, which determines the largest vertical distance between the two functions at each x . We calculate D_n using the *left-difference* values D_n^- and the right-difference values D_n^+ (assuming unique observation values):

$$D_n^+ = \max_{1 \leq i \leq n} \left\{ \frac{i}{n} - \hat{F}(X_i) \right\}$$

$$D_n^- = \max_{1 \leq i \leq n} \left\{ \hat{F}(X_i) - \frac{i-1}{n} \right\}$$

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

3. If D_n exceeds some critical value, *i.e.* there is a ‘large’ difference at a certain point between the two distribution functions; we reject the null hypothesis (H_0). The critical value depends on the type of distribution tested for, and the various cases are summarised in Table 4.4 (Law, 2015).

We reject the null hypothesis if the critical c value is exceeded. Note that for the Weibull distribution we need to consider the number of observations as well (n).

If the data is presented in grouped format, we proceed as follows: Define the upper limits of the groups as new observations X'_i , to comply with the empirical function

$$F_n(x) = \frac{\text{number of } X'_i \leq x}{n},$$

with n = Total number of original observations.

We define $F_n(X'_0) = 0$

$$D_n^+ = \max_{1 \leq i \leq m} \{F_n(X'_i) - \hat{F}(X'_i)\}$$

$$D_n^- = \max_{1 \leq i \leq m} \{\hat{F}(X'_i) - F_n(X'_{i-1})\}, \quad m = \text{number of intervals}$$

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

The author has formulated this procedure since no guidelines could be found in the literature.

4.5.4 Examples of distribution fits

Three examples of hypothesis tests for data fits are presented here. There are three cases that one must identify, namely

1. A fairly large discrete data set, which requires the chi-squared test.
2. A fairly large continuous data set, which requires the chi-squared test.
3. A continuous but small data set, which requires the K-S test.

Discrete data: chi-squared test

■ **Example 4.1** The percentage of sulphur in a certain tyre should not exceed 4%. A chemical engineer has noted the number of days (X) on which violations of the sulphur limit occurred. Determine if the random variable X is Poisson distributed.

Violations per day	0	1	2	3	4	5	6
Number of days	33	44	10	5	5	2	1

Solution:

H_0 : The data is Poisson distributed with parameter λ .

Determine λ : From Table 4.5, we can estimate λ with

$$\begin{aligned} \bar{X} &= \sum_{i=1}^n X_i/n \\ &= \frac{(0 \times 33) + (1 \times 44) + (2 \times 10) + \dots + (6 \times 1)}{100} \\ &= \hat{\lambda} \\ &= 1.15 \text{ violations per day.} \end{aligned}$$

Now set up a table with each row representing a class of the distribution:

# Days	Observed freq. O_i	Expected freq. E_i	E'_i	O'_i	$(E_i - O_i)^2/E_i$
0	33	31.66	31.66	33.00	0.06
1	44	36.41	36.41	44.00	1.58
2	10	20.94	20.94	10.00	5.71
3	5	8.03	10.99	13.00	0.37
4	5	2.31			
5	2	0.53			
6	1	0.10			
7	0	0.02			
>7 ^[1]	0	0.00			
				$\chi^2_{\text{calc}} =$	7.72

The degrees of freedom is $\nu = k - m - 1 = 4 - 1 - 1 = 2$, so the value of $\chi^2_{\text{crit}} = 5.99$. We reject H_o because $\chi^2_{\text{calc}} > \chi^2_{\text{crit}}$. The corresponding p -value is 0.0211.

Please note:

1. The range of the Poisson distribution goes to infinity, and the last class is added to cover that range, even if no observations were made there (green cell, last row of first column).
2. The coloured cells are merged so that $E_i \geq 5$ (blue cells).
3. The value of k follows *after merging*, and in this case, $k = 4$.

Continuous data: chi-squared test

■ **Example 4.2** The times between arrivals of calls at a call centre were observed. The boss suspects the data is Weibull distributed and proposes $\alpha=2$ and $\beta=4$, as *estimated from the data* by her. Now do a goodness-of-fit test to see if the data can be associated with the proposed distribution. The observations were grouped and the frequencies determined, as follows:

Class upper limits	O_i
1.6289	772
3.2578	1610
4.8867	1478
6.5157	789
8.1446	270
9.7735	70
11.4024	10
13.0313	1

Solution:

H_o : The data is Weibull distributed with parameters α and β .

We note that $m = 2$ because α and β were estimated from the data by the boss of the call centre (if the parameters were not estimated, $m = 0$). The test is done using the above data and an extended table:

Class upper limits	O_i	Nett Areas	E_i	E'_i	O'_i	χ^2 terms
1.6289	772	0.1528	764.0711	764.0711	772	0.0823
3.2578	1610	0.3321	1660.2920	1660.2920	1610	1.5234
4.8867	1478	0.2903	1451.6068	1451.6068	1478	0.4799
6.5157	789	0.1544	771.9605	771.9605	789	0.3761
8.1446	270	0.0546	272.9223	272.9223	270	0.0313
9.7735	70	0.0133	66.3771	66.3771	70	0.1977
11.4024	10	0.0023	11.2915	12.7703	11	0.2454
13.0313	1	0.0003	1.3559			
10000	0	0.0000	0.1229			
$\chi^2_{\text{calc}} =$						2.9361

The nett areas were determined by integrating the Weibull distribution from 0 to the class upper limit, then subtract the area from 0 to the previous upper limit. If this area is A_i , then the $E_i = n \times A_i$. In Excel, one can use the formula function =WEIBULL.DIST(x,alpha,beta,cumul). Note the purple coloured cell – it indicates the approximation of the distribution to infinity.

The degrees of freedom is $\nu = k - m - 1 = 7 - 2 - 1 = 4$, so the value of $\chi^2_{\text{crit}} = 9.488$. We do not reject H_0 because $\chi^2_{\text{calc}} < \chi^2_{\text{crit}}$. The corresponding p -value is 0.569. (Please check on the table in Appendix A that you agree with the values stated here.)

Continuous data: K-S test

■ **Example 4.3** A state vehicle inspection station has been designed so that inspection time follows a uniform distribution with limits of 10 and 15 minutes. A sample of 10 duration times during low peak and peak traffic conditions was taken. Use the K-S test with $\alpha=0.05$ to determine if the sample is from this distribution. The times are:

11.3 10.4 10.2 12.6 14.8 13 14.3 13.3 11.5 13.6

Solution:

H_0 : The data is uniformly distributed $U(10, 15)$.

The parameters (a and b) were given. We need to compare cumulative distribution functions, so the cumulative of the theoretical uniform distribution is

$$\begin{aligned}
 F(x) &= \int_a^x f(t) dt \\
 &= \int_a^x \frac{1}{b-a} dt \\
 &= \frac{x-a}{b-a} \\
 &= \frac{x-10}{5}.
 \end{aligned}$$

The empirical distribution is applied as before with (4.1). The solution is developed using a table, as follows:

Observations	Freq	Number of $X_i \leq x$	$F_n(x)$	$F(x) = \frac{x-10}{5}$	D^-	D^+	$\text{Max}(D^-, D^+)$
10.20	1.00	1.0	0.100	0.0400	0.0400	0.0600	0.0600
10.40	1.00	2.0	0.200	0.0800	0.0200	0.1200	0.1200
11.30	1.00	3.0	0.300	0.2600	0.0600	0.0400	0.0600
11.50	1.00	4.0	0.400	0.3000	0.0000	0.1000	0.1000
12.60	1.00	5.0	0.500	0.5200	0.1200	0.0200	0.1200
13.00	1.00	6.0	0.600	0.6000	0.1000	0.0000	0.1000
13.30	1.00	7.0	0.700	0.6600	0.0600	0.0400	0.0600
13.60	1.00	8.0	0.800	0.7200	0.0200	0.0800	0.0800
14.30	1.00	9.0	0.900	0.8600	0.0600	0.0400	0.0600
14.80	1.00	10.0	1.000	0.9600	0.0600	0.0400	0.0600

$D_{\max} = D_n = 0.12$. We select the row ‘All parameters known’ from Table 4.4, and determine $c_{1-\alpha}$: the critical value is 1.358 in the ‘0.95’ column. The test condition is calculated using

$$\begin{aligned} \left(\sqrt{n} + 0.12 + \frac{0.11}{\sqrt{n}} \right) D_n &> c_{1-\alpha} \\ \left(\sqrt{10} + 0.12 + \frac{0.11}{\sqrt{10}} \right) \times 0.12 &= 3.317 \\ &> 1.358. \end{aligned}$$

Conclusion: We reject H_0 . Plots of the true and empirical distributions are shown in Figure 4.6.

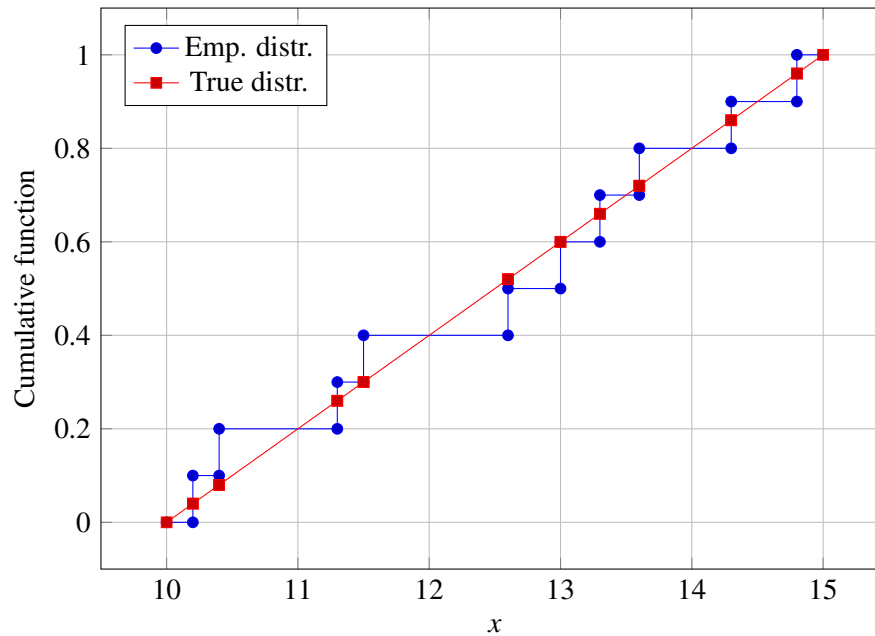


Figure 4.6: The true and empirical distributions for Example 4.3

Table 4.5: Typical maximum likelihood estimators

Distribution	MLE(s)
Uniform $f(x) = \frac{1}{b-a}$ or $f(x) = \frac{1}{k}$ (discrete)	$\hat{a} = \min_{1 \leq i \leq n} X_i, \hat{b} = \max_{1 \leq i \leq n} X_i$
Exponential $f(x) = \frac{1}{\beta} \exp\left(-\frac{x}{\beta}\right), x > 0.$	$\hat{\beta} = \bar{X}(n)$
Normal $f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$ $-\infty < x < \infty.$	$\hat{\mu} = \bar{X}(n)$ $\hat{\sigma} = \left[\frac{n-1}{n} S^2(n)\right]^{1/2}$
Lognormal $f(x) = \frac{1}{x} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\ln x - \mu)^2}{2\sigma^2}\right),$ $x > 0.$	$\hat{\mu} = \frac{\sum_{i=1}^n \ln X_i}{n} \quad X_i > 0 \forall i$ $\hat{\sigma} = \left[\frac{\sum_{i=1}^n (\ln X_i - \hat{\mu})^2}{n}\right]^{1/2}$
Poisson $f(x) = e^{-\lambda} \frac{\lambda^x}{x!}, \quad x = 0, 1, 2, \dots$	$\hat{\lambda} = \bar{X}(n)$
Weibull $f(x) = \alpha \beta^{-\alpha} x^{\alpha-1} e^{-(x/\beta)^\alpha} \quad x > 0.$	$\frac{\sum_{i=1}^n X_i^{\hat{\alpha}} \ln X_i}{\sum_{i=1}^n X_i^{\hat{\alpha}}} - \frac{1}{\hat{\alpha}} = \frac{\sum_{i=1}^n \ln X_i}{n}$ $\hat{\beta} = \left(\frac{\sum_{i=1}^n X_i^{\hat{\alpha}}}{n}\right)^{1/\hat{\alpha}}$

Survival kit:

Ensure you have the following in your assessment pack:

- 1 Know the sources of input data.
- 2 Know how to estimate distributions for a model if no data is available: uniform and triangular distribution.
- 3 Know what an empirical distribution is, how to develop it and how to sample from it.
- 4 Know when and how to apply the chi-squared and K-S tests.

Output analysis Overview

Properties of output variables

- Method of calculating of output
- Types of simulation output
- Point estimator
- The confidence interval

Terminating systems

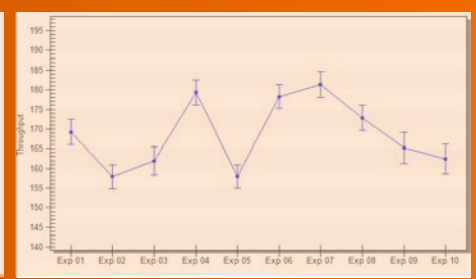
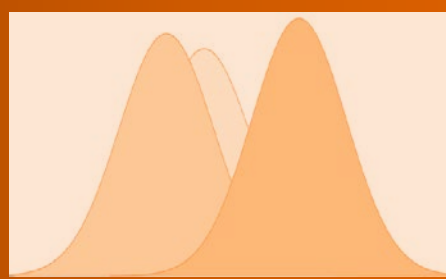
- Determining the number of observations
- Half-width of the confidence interval and the Two-phase method

Non-terminating systems

- Determining the truncation point
- Replication/Deletion approach
- Batch-means approach

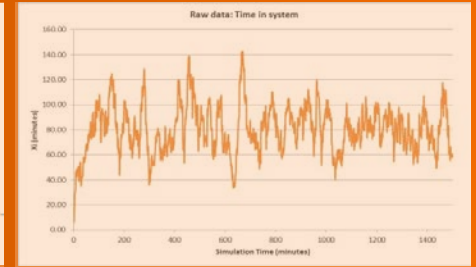
Evaluating and selecting scenarios

- Analysis of variance
- Applying ANOVA
- The Kim-Nelson procedure



Procedure KN

1. *Setup.* Select the overall desired $P(CS) = 1 - \alpha$, the value for δ , and common first stage sample size $n_0 \geq 10$. Set $\eta = \frac{1}{2} \left[\left(\frac{2\alpha}{k-1} \right)^{-2/(n_0-1)} - 1 \right]$.
2. *Initialization.* Let $I = \{1, 2, \dots, k\}$ be the set of scenarios still in contention, and let $k^2 = 2\eta(n_0 - 1)$. Obtain n_0 outputs X_{ij} ($j = 1, 2, \dots, n_0$) from each scenario i ($i = 1, 2, \dots, k$) and let $\bar{X}_i(n_0) = \sum_{j=1}^{n_0} X_{ij}/n_0$ denote the sample mean of the first n_0 outputs from scenario i .
For all $i \neq l$ calculate $S_{il}^2 = \frac{1}{n_0 - 1} \sum_{j=1}^{n_0} (X_{ij} - \bar{X}_i(n_0) - [\bar{X}_l(n_0) - \bar{X}_i(n_0)])^2$, the sample variance of the difference between scenarios i and l . Set $r = n_0$.
3. *Screening.* Set $I = I^{std}$. Let $I = \{i : i \in I^{std} \text{ and } \bar{X}_i(r) \geq \bar{X}_l(r) W_{\delta}(r), \forall l \in I^{std}, i \neq l\}$ where $W_{\delta}(r) = \max \left\{ 0, \frac{\delta}{2r} \left(\frac{k^2 S_{il}^2}{\delta^2} - r \right) \right\}$.
4. *Stopping rule.* If $|I| = 1$, then stop and select the scenario whose index is in I as the best. Otherwise, take one additional output $X_{i,r+1}$ from each system $i \in I$, set $r \leftarrow r + 1$, and go to Step Screening.



5. Output data analysis

R “Climate is what you expect; weather is what you observe.” (Morrison, 2019)

OUTPUT of stochastic models provide estimations of output parameters identified by the modeller. These are also referred to as *output variables*, *key performance indicators* (KPIs) or *performance measures*, and in the case of simulation optimisation, objective functions (see Table 2.2 and Chapter 6). In this chapter we study the statistical techniques used to configure simulation models to provide point and interval estimators for KPIs. We also study techniques to evaluate finite scenarios.

5.1 Simulation output analysis – overview

Systems containing stochastic elements/processes respond stochastically (see Kelton (1997)). A system is partially or completely described in terms of input and output (response) variables of a model. The following two concepts are important:

- *Response*: the consequence(s) of model logic and input data.
- *Statistical inference*: The methods by which one makes generalisations about populations.

When at least one input variable is stochastic, the model is not deterministic anymore, but has a random response that must be analysed with applicable statistical techniques. The analysis techniques applied in this chapter rely on aspects of statistics theory, of which a cornerstone is the Central Limit Theorem. It forms the basis of inference on expected values, and is stated next.

Theorem 5 (Central Limit Theorem). *If \bar{X} is the mean of a sample of size n taken from a distribution with mean μ and variance σ^2 , then $N(0, 1)$ is the limiting form of the distribution of*

$$Z = \sqrt{n} \frac{\bar{X} - \mu}{\sigma} \quad \text{as } n \rightarrow \infty. \quad (5.1)$$

The application of the theorem will become apparent later in the chapter.

A single response of a simulation model may not be used to draw conclusions from. It is similar to concluding that if person X has blue eyes, then every person in the world must have blue eyes. *A stochastic simulation executed on a computer is simply a computer-based statistical sampling experiment, and we rely on principles of Statistical Inference to analyse and draw conclusions from the results.*

Consider the simple deterministic system in Figure 5.1, where an entity arrives every minute, and is serviced for 30 seconds. It is obvious that the server will be occupied for 50% of the time, and that the time spent in the system is constant, that is 30 seconds. The time between exits will be 1 minute from the second entity and on.



Figure 5.1: Simple deterministic system

If the mechanisms in this system are changed to stochastic (arrivals and service) as shown in Figure 5.2, what will the effect on the exit times be?

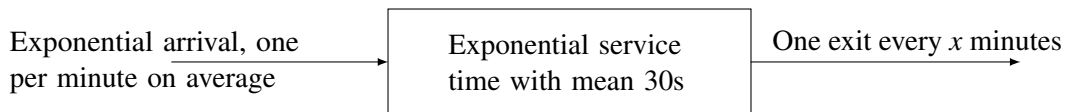


Figure 5.2: Simple stochastic system

It is not possible to predict the exit time with certainty anymore. This is what simulation is all about – to create a sufficient number of arrivals to draw a conclusion on the unknown parameter at the exit. The “sufficient number of arrivals” is determined with statistical analysis, as described later. A simulation model of a stochastic process yields different output values when compared to a deterministic system. The rule is often described as RIRO: Randomness in, randomness out. The principle of GIGO also applies in simulation, and more importantly, the principle of *RIRO* must be respected and correctly treated.

Figure 5.3 shows that we supply various forms of input to a stochastic simulation at different points in a model, and the resulting output parameters each has a response distribution.

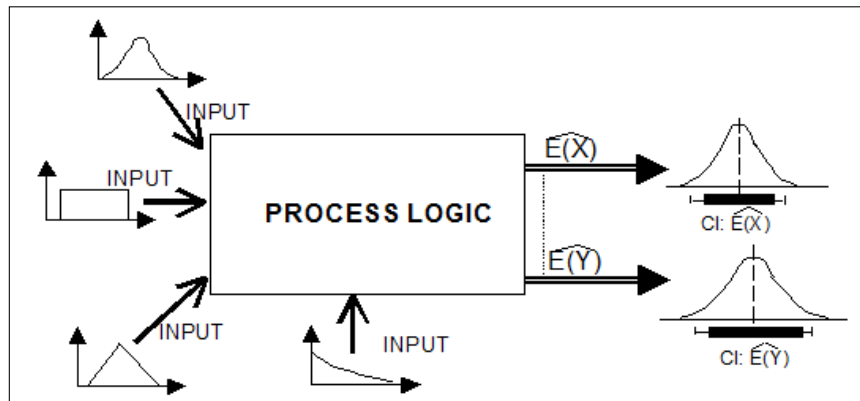


Figure 5.3: Input/output concept for a stochastic simulation model

Analysis of the output requires a different approach, which firstly depends on the type of system analysed (terminating *vs.* non-terminating), and other features like underlying statistical assumptions and the nature of the output variables, discussed next.

5.2 Properties of output variables

Output variables can be 1) classified according to the type of output they estimate, and 2) how they are observed.

5.2.1 Method of calculating of output

Given a set of observations $\{X_i\}$ generated with simulation, the methods of calculating estimated output are

- expected values ($\bar{X} = \sum_{i=1}^n X_i/n$),
- minimums ($\min\{X_i\}$),
- maximums ($\max\{X_i\}$),
- percentile (“How long do 95% of my customers wait at a specific point in the system?”), and
- proportions (“What proportion of set-up times take longer than 5 minutes?”).

5.2.2 Types of simulation output

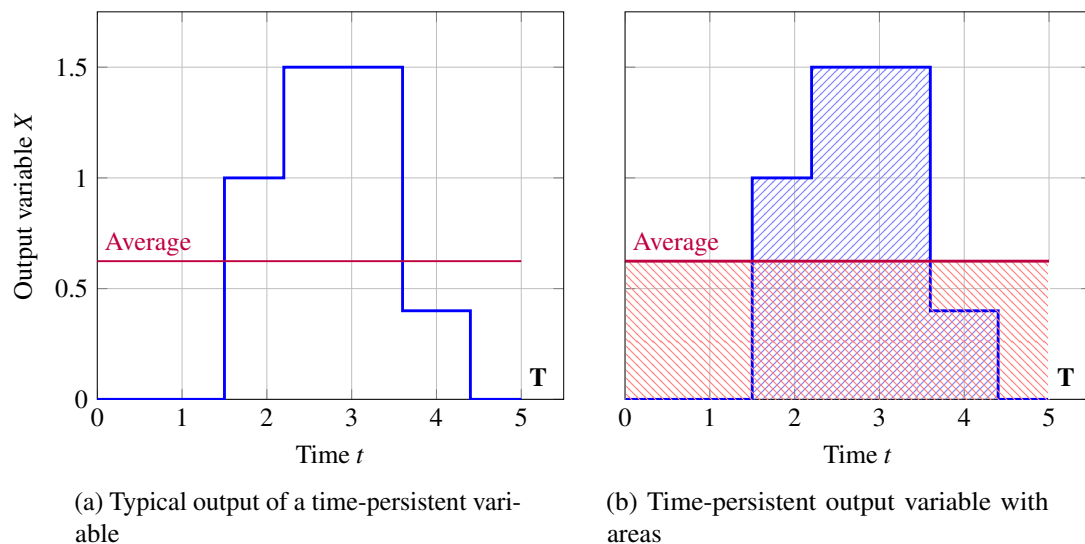
There are two main types of simulation output; these are:

1. **Observational** output: These simply contain single numbers which could be integer or real, for example
 - (a) the time it takes an operator to switch a jig,
 - (b) number of tasks completed,
 - (c) time to complete a task,
 - (d) time between arrivals.

In conjunction with the previous subsection, we determine the arithmetic mean of the observed values.

2. **Time-persistent** output: We often measure an output that persists at a level over time, for example the
 - (a) average queue length at a paypoint,
 - (b) utilisation of a resource,
 - (c) fluid level in a container,
 - (d) inventory level of a commodity.

An example of a time-persistent output variable X is shown in Figure 5.4a.



The level is observed over time from $t = 0$ until $T = 5$, then the average over that time length is calculated as the purple line. The average is the height of the horizontal line that defines a rectangle over the time of observation so that the red area and the blue areas in Figure 5.4b are equal. Note that the duration of the calculation starts at 0, even if the level is zero from $t = 0$ to $t = 1.5$.

An example of how to calculate the average for a time-persistent observation follows, in this case, the utilisation of two resources. In Figure 5.5 it is shown that two resources are operated over time, from $t = 0$ to $T = 15$. When no resource operates, the level is indicated as zero, while when any one resource is active, the active level is 1, and if both resources are operating, the level is 2. The levels on the graph change at discrete points in time, and the *combined utilisation* of the two resources is calculated as

$$\begin{aligned}
 \eta &= \sum \text{Subareas} / (\text{Number of resources} \times T) \\
 &= [((4-3) \times 1) + ((7-4) \times 2) + ((9-7) \times 1) + ((12-11) \times 1) + \\
 &\quad ((13-12) \times 2) + ((15-13) \times 1)] / (2 \times 15) \\
 &= 14/30 \\
 &= 0.467.
 \end{aligned}$$

Note that, as mentioned before, we divide in this example by 2×15 time units, because that is the duration of the time measurement for two resources – irrespective of the zero levels from $t = 0$ to $t = 3$. One can also think of the utilisation is the ratio of the area under the blue line divided by the rectangle enclosed by the vertical line $(0,0)$; $(0,2)$ and the horizontal line $(0,0)$; $(15,0)$.

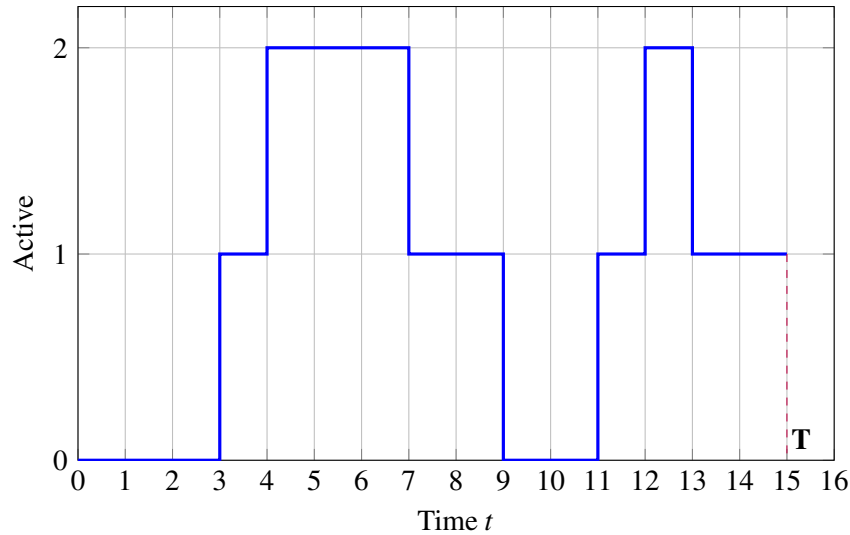


Figure 5.5: Time-persistent example for two resources that are operating or not

We focus on estimating *Expected values* of KPIs or output variables in this module, and typical variables are presented in Table 2.2 in Chapter 2. The objective of the output analysis is to estimate the value(s) of KPI(s) or output variable(s) of the system that is studied, and to describe them in terms of

1. *point* and
2. *interval estimators*.

These will be subsequently discussed.

5.2.3 Point estimator

The estimated mean values in Figure 5.9 are determined with simulation of a terminating system. The useful result from the Central Limit Theorem is that these means are approximately normally distributed (assuming that the number of observations per day is large), *i.e.* the random variable

\bar{X} is approximately normally distributed with mean $\bar{\bar{X}}$ and variance S^2 , where

$$\bar{\bar{X}} = \frac{\sum_{i=1}^n \bar{X}_i}{n} \quad (5.2)$$

and the sample variance $S_{\bar{X}}^2$ is an unbiased estimator of the population variance σ^2 , where

$$S_{\bar{X}}^2 = \frac{\sum_{i=1}^n (\bar{X}_i - \bar{\bar{X}})^2}{n-1} \quad (5.3)$$

with n = sample size ($n = 100$ in Figure 5.9). The estimated mean value calculated from (5.2) is a *point estimator* of the true population mean. The estimated variance of the mean follows from

$$S_{\bar{X}}^2 = \frac{S_{\bar{X}}^2}{n}. \quad (5.4)$$

R (5.4) can be proved as follows: We have a sample of \bar{X}_i , with sample size n . The observations are identically distributed (i.i.d.) and were sampled from the same distribution. Their variance is thus $\sigma_{\bar{X}}^2$. This is the *population variance*. Now, the sample variance is, under assumptions of i.i.d:

$$\begin{aligned} \text{Var}(\bar{\bar{X}}) &= \sigma_{\bar{\bar{X}}}^2 \\ &= \text{Var}\left(\sum_{i=1}^n \bar{X}_i / n\right) \\ &= \frac{1}{n^2} \text{Var}\left(\sum_{i=1}^n \bar{X}_i\right) \\ &= \frac{1}{n^2} \sum_{i=1}^n \text{Var}(\bar{X}_i) \quad \text{due to independence} \\ &= \frac{1}{n^2} \times n \times \sigma_{\bar{X}}^2 \quad \text{because we summed } n \text{ times the same variance of the same distribution} \\ &= \frac{\sigma_{\bar{X}}^2}{n}. \end{aligned}$$

The sample variance decreases as n increases.

P Points to ponder: Suppose the observations are not normally distributed, but we do not know it. What will be the effect on the simulation result? If the observations are not independent, what will be the effect on the simulation result?

Next, an interval estimator for $\bar{\bar{X}}$ is explained.

5.2.4 The confidence interval

The result of (5.2) is called a *point estimator* of the population parameter μ . Another (more descriptive) estimator of a parameter is the *confidence interval*. This is an interval estimator of the population parameter, and *specifies a range in which the unknown population parameter is to be expected*. The confidence interval is also used to determine the number of replications required for the estimation of an output parameter, and we shall use the *t*-distribution to determine the confidence interval. The rationale behind the confidence interval and the use of the *t*-distribution is as follows:

- Recall from the CLT that $Z_n = \frac{\bar{X} - \mu}{\sqrt{\sigma^2/n}}$, and $Z_n \xrightarrow{D} N(0, 1)$ as $n \rightarrow \infty$.
- It follows from the Central Limit Theorem that the random variable Z_n is approximately normal distributed if n is ‘large’.
- The mean $\bar{X}(n)$ is approximately normally distributed with unknown mean μ and variance σ^2/n , where σ^2 is the variance of the distribution of the population of X_i .
- In practice, σ^2 is usually unknown, but can be estimated with S^2 if n is large.
- The random variable $T_n = \frac{\bar{X}(n) - \mu}{\sqrt{S^2/n}}$ is approximately standard normal distributed ($N(0, 1)$).
- Therefore, $P(-z_{1-\alpha/2} \leq \frac{\bar{X}(n) - \mu}{\sqrt{S^2/n}} \leq z_{1-\alpha/2}) \approx 1 - \alpha$, where $z_{1-\alpha/2}$ is the upper $1 - \alpha/2$ critical point from the standard normal distribution.
- If n is ‘sufficiently large’, an approximate confidence interval for μ is

$$\bar{X} \pm z_{1-\alpha/2} \sqrt{S^2/n} \quad (5.5)$$

and the confidence interval half-width

$$h = z_{1-\alpha/2} \sqrt{S^2/n}.$$

- From the above, the question of what is ‘large’ arises.
- Firstly, if the X_i ’s are normally distributed, the random variable

$$T_n = \frac{\bar{X}(n) - \mu}{\sqrt{S^2/n}}$$

has a t -distribution with $n - 1$ degrees of freedom, and an *exact* $100 \times (1 - \alpha)$ confidence interval for μ is given by

$$\begin{aligned} CI &= \bar{X} \pm h \\ &= \bar{X} \pm t_{n-1, 1-\alpha/2} \sqrt{S^2/n}, \quad n \geq 2 \end{aligned} \quad (5.6)$$

where

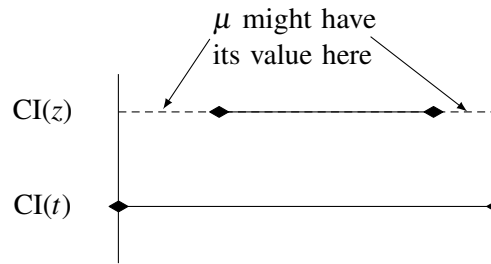
$t_{n-1, 1-\alpha/2}$ is the upper $1 - \alpha/2$ critical point from the Student t -distribution (upper critical values are shown in Table 6.4, Appendix A)

\bar{X} = Sample Mean

S^2 = Estimation of the population variance

n = Sample size, *i.e.* the number of observations.

- The X_i ’s are usually not exactly normally distributed (except where it can be assumed that they are approximately normally distributed due to the CLT, *e.g.* averages as output in a simulation study), so that the confidence interval from (5.6) is also approximate in terms of coverage.
- But $t_{n-1, 1-\alpha/2} > z_{1-\alpha/2}$, so that a wider confidence interval will result from (5.6) compared to that from (5.5), and the proportion of coverage will thus be closer to $1 - \alpha$ if the t -distribution is used. Refer to Figure 5.6.
- As $n \rightarrow \infty$, then $t_{n-1, 1-\alpha/2} \rightarrow Z_{1-\alpha/2}$, which will eventually differ very little. If n is small, the difference is significant, but at $n = 40$ the difference is only 3%.
- The interpretation of the confidence interval is as follows: If k confidence intervals are constructed for a given parameter, then it is expected that $(1 - \alpha) * k$ of these intervals

Figure 5.6: Coverage of the confidence interval: normal- and t -distribution

includes the true population parameter. This is known as the *coverage* of the confidence interval. Refer to Figure 5.7.

- We also conclude that the t -distribution must be used when the population variance must be estimated from the sample, *i.e.* when it is unknown.

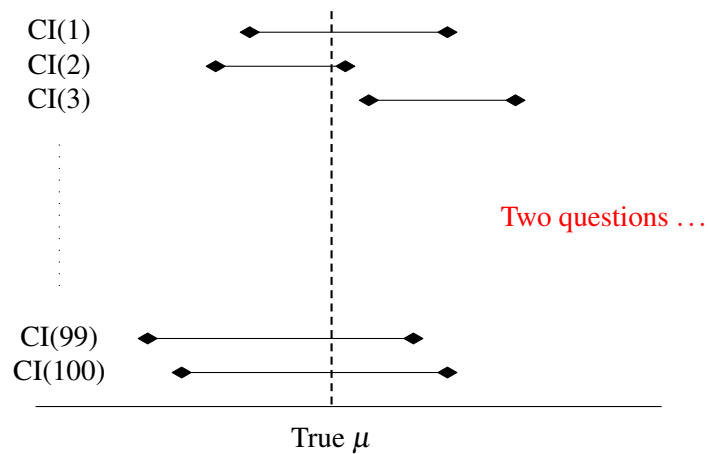


Figure 5.7: Interpretation of the coverage of a confidence interval

The coverage of the confidence interval and the error in estimating the true population mean is further explained in Figure 5.8. Note that ε indicates the error.

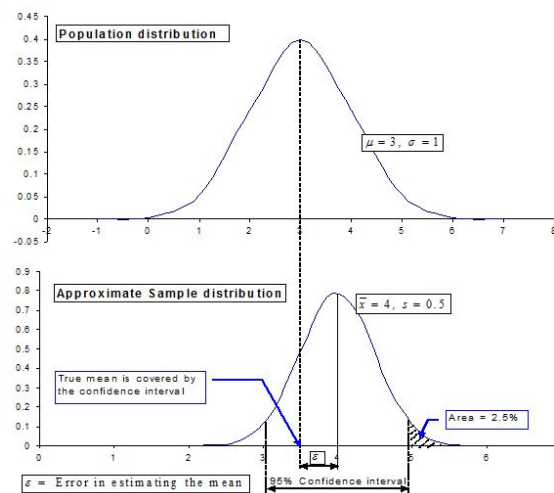


Figure 5.8: Error in estimating the true population mean

We shall now discuss the specifics of output analysis of terminating- and non-terminating systems when we simulate discrete-event, stochastic processes.

5.3 Terminating systems

A *terminating system* is defined as a system that starts in the empty state with operations idle, and after a logical event, ends in the empty state with operations idle again. The operations of a retail store stop after a specific time period and when there are no more customers to be served. The same principle applies to a restaurant. Every termination event usually supplies the analyst with an observation per output parameter, which may be assumed to be statistically independent of the observations from other replications during the same simulation run.

Suppose we are interested in the average time a party spends in a restaurant, and the true, but unknown value is denoted by μ . We can observe the time for every party per day since the restaurant is open until closing time. The arithmetic mean of these values can be calculated, which will result in one observation for that particular day. In simulation terminology, a model run that results in such an observation is called a *replication*. One replication in this case thus corresponds to a day in the restaurant. In terminating systems, each replication results in one observation, so that the ultimate objective is to determine how many replications are required to achieve a certain confidence level for a given output parameter.

The observation process can be executed for a number of days, say 100 days, and 100 averages will be available. The question is, did we make sufficient observations to draw a conclusion on this parameter, which is *Average time in restaurant*? These 100 values can be used to determine how many observations are actually needed to determine the parameter under study to a *specified level of confidence*. We therefore **estimate** the expected value of the parameter with a certain level of confidence, and this estimation process is a subset of *Statistical inference*. (See Gogg and Mott (1992), p. 9-3.) The process is described in Figure 5.9; the entry ‘Party x ’ denotes the time spent in the restaurant for the specific party. The value of μ is estimated in two ways: using a *point estimator* and an *interval estimator*.

Day 1	Day 2	Day 3	Day 4	...	Day 100
Party 1	Party 1	Party 1	Party 1	...	Party 1
Party 2	Party 2	Party 2	Party 2	...	Party 2
Party 3	Party 3	Party 3	Party 3	...	Party 3
\vdots	\vdots	\vdots	\vdots	...	\vdots
Party i	Party j	Party k	Party l	...	Party z

\bar{X}_1	\bar{X}_2	\bar{X}_3	\bar{X}_4	...	\bar{X}_{100}
-------------	-------------	-------------	-------------	-----	-----------------

Figure 5.9: Estimation of the mean of a parameter – terminating system

As stated earlier, simulation of stochastic systems is statistical sampling with the aid of a computer. A sufficient number of observations must therefore be made to draw a conclusion with a certain level of confidence on a parameter under study. This process is outlined next.

5.3.1 Determining the number of observations – terminating system

The main driver for the analysis is the distribution of the observations for the parameter under study. We will aim to limit the variance of the distribution to obtain a good estimation of the output parameter(s) under study. For this, we need the confidence interval half-width h and the *Two-phase method*, explained next.

5.3.2 Half-width of the confidence interval and the Two-phase method

We usually work towards a desired confidence interval width (a zero-width means that we have estimated the population parameter exactly), for which we require a number of simulation observations. One of several methods to determine the required number of observations (replications) is the *Two-phase method*, which is as follows (Pegden, R.E. Shannon, and R. Sadowski, 1995):

1. Phase 1: Do a trial run of 10 replications, *i.e.* $n = 10$. We therefore make 10 observations of the mean of each output parameter.
2. Estimate $\bar{\bar{X}} = \frac{\sum_{i=1}^n \bar{X}_i}{n}$ (using (5.2)) with the observations $\bar{X}_1, \bar{X}_2, \dots, \bar{X}_n$. Now estimate $S_{\bar{X}}^2 = \frac{\sum_{i=1}^n (\bar{X}_i - \bar{\bar{X}})^2}{n-1}$ from (5.3).
3. Determine the confidence interval half-width with

$$h = t_{n-1; 1-\alpha/2} \frac{S_{\bar{X}}}{\sqrt{n}}. \quad (5.7)$$

The value of t is the upper $1 - \alpha/2$ critical point on the cumulative Student t -distribution — make sure you understand what this means. It has for example a value of 2.262 at $n = 10$ and $\alpha = 0.05$. It can be calculated in MS-Excel 2010 with the formula “=T.INV(α , degrees of freedom)”, *i.e.* “=T.INV(0.975, 9)” will give 2.262158887, which is the upper critical t -value which has 97.5% of the area of the distribution with nine degrees of freedom to its left. This is for $\alpha=5\%$, so you must halve its value to get the two-tailed t -value. If one changes (5.7) to


$$h = t_{n-1; 1-\alpha} \frac{S_{\bar{X}}}{\sqrt{n}} \quad (5.8)$$

then one can use ‘=T.INV.2T(α , degrees of freedom)’, *i.e.* ‘=T.INV.2T(5%, 9)’ will also give $t=2.262158887$. Remember we use the ‘INV’ function because we need a t -value associated with a given probability.

4. If the calculated h is judged ‘too wide’ by the simulation analyst, they then select a smaller h called h^* , and the actual number of replications required to (hopefully) realise this narrower half-width is given by (Law and Kelton, 2000)

$$n^* = \left\lceil n \left(\frac{h}{h^*} \right)^2 \right\rceil \quad (5.9)$$

where h^* is the desired confidence interval half-width and n^* is the required number of replications. To use (5.9), the simulation analyst has to choose a target value for h^* . There are no recipes or rules for choosing a value for h^* , but it should at least be equal to or smaller than h (Why?). Students cringe when they have to choose a value without any guidance. If you understand the concept, it is easy. A smaller chosen value for h^* will result in a larger value for n^* which results in more replications and improved quality of estimation, but requires more simulation time.

 (5.9) can be proved as follows: let the variance estimator based on n observations be $S^2(n)$. Also, $n^* \geq n$ by definition, and we can assume that $S^2(n) \leq S^2(n^*)$, because

more observations make the variance smaller. Now,

$$\begin{aligned}
 h &= t_{n-1;1-\alpha/2} S(n) / \sqrt{n} \\
 h^* &= t_{n^*-1;1-\alpha/2} S(n^*) / \sqrt{n^*} \\
 \frac{h}{h^*} &= \frac{t_{n-1;1-\alpha/2} S(n) / \sqrt{n}}{t_{n^*-1;1-\alpha/2} S(n^*) / \sqrt{n^*}} \\
 &= \frac{t_{n-1;1-\alpha/2} S(n)}{t_{n^*-1;1-\alpha/2} S(n^*)} \cdot \frac{\sqrt{n^*}}{\sqrt{n}} \\
 h \times t_{n^*-1;1-\alpha/2} S(n^*) \times \sqrt{n} &= h^* \times t_{n-1;1-\alpha/2} S(n) \times \sqrt{n^*} \\
 h\sqrt{n} &\leq \sqrt{n^*} h^* \\
 \therefore n^* &\geq \left\lceil n \left(\frac{h}{h^*} \right)^2 \right\rceil.
 \end{aligned}$$

We introduce the “ \leq ” sign in the second last step because $t_{n-1;1-\alpha/2} \geq t_{n^*-1;1-\alpha/2}$ and $S(n) \geq S(n^*)$, so $t_{n-1;1-\alpha/2} S(n) \geq t_{n^*-1;1-\alpha/2} S(n^*)$. Because n^* is an integer, we round up as in (5.9), using the symbols \lceil and \rceil .

5. Phase 2: The simulation is run for n^* replications, and the calculations above are repeated (using (5.2), (5.3), (5.7) and (5.9)) to obtain a ‘final’ point and interval estimator, which are presented to the stakeholders of the study. The analysis is done *per output parameter*, and we take the maximum of n^* that is determined for each parameter, then run the simulation model for this number of replications.

General comments – number of replications

Why do we make 10 replications initially? Why not 15, 50, or 100? The number of 10 is a rule of thumb, but we also know that the t -distribution is useful for observations less than 30. We could make more than 30 observations, but the initial simulation runs may take too long. The use of 10 runs is thus an effort to compromise between simulation time (computer time) and the requirements of statistical procedures. If the observations are not normally distributed, then the results for the confidence interval hold for approximately 10 replications or more.

P Point to ponder: Does Statistics make uncertainty certain, or does it make us certain of the level of uncertainty?

Example: Calculating the desired number of replications

Assume the 10 observations given are means from 10 independent simulation runs:

\bar{X}_1	\bar{X}_2	\bar{X}_3	\bar{X}_4	\bar{X}_5	\bar{X}_6	\bar{X}_7	\bar{X}_8	\bar{X}_9	\bar{X}_{10}
16.818	8.124	18.895	1.879	4.394	11.654	1.086	2.353	16.500	5.435

From (5.2), the estimated mean is calculated as 8.714, and from (5.3) the estimated variance follows to be 45.968. Of course, $n = 10$, and the confidence interval half-width at $\alpha = 0.05$ is given by

$$\begin{aligned}
 h &= t_{n-1;1-\alpha/2} S_{\bar{X}} / \sqrt{n} \\
 &= 2.262 \times \sqrt{45.968/10} \\
 &= 4.850.
 \end{aligned}$$

Now select an $h^* < h$. The effect of the different values of h^* on n^* is shown in the following list.

h^*	2	1.5	1	0.75	0.25
n^*	59	105	236	941	3 764

A small h^* is desirable, but it can be seen that smaller values require many more replications.

P Point to ponder: Why $h^* < h$?

5.4 Non-terminating systems

Analysis of the simulation output of a non-terminating stochastic model requires more effort and knowledge than that of a terminating system. The main reasons are the presence of the transient phase, correlation, and the dependence of observations per parameter. The run length in terms of model time is also unknown. On the positive side, once certain data processing techniques have been applied, the remainder of the analysis is similar to terminating systems, as outlined in the previous section.

When a non-terminating model is simulated, observations are recorded as a sequential series of data for the duration of the simulation run. These observations are made as they occur, and they are not statistically independent due to the random number generation process and in-process variation. The transient phase also induces bias in the statistics. The techniques required to overcome these problems will now be discussed, and

- the moving average,
- correlation and the correlogram, and
- the replication/deletion and batching of observations

will be introduced.

5.4.1 Determining the truncation point

In the analysis of non-terminating systems it is usually desired to study the long-term or steady state behaviour of a system. (There are cases, however, where the transient phase is of particular interest: consider a proposed/new system – typical questions could be the length of the transient phase, the distribution of the parameters during that phase that is worst case, *etc.*). The truncation point is the point in simulation time where the steady state starts, and all data collected up to this point is discarded in analysis to eliminate bias.

The truncation point is determined by iteration and visual inspection using the moving average. Since we do not know where the truncation point is located, the simulation is run for an arbitrary length of simulated time. This might be too short, or sufficient. The output per parameter is used to draw a moving average. Suppose we study parameter X , and a number of observations were made over time, as follows:

Observation number	1	2	3	4	5
Time	0.2	3.3	5.1	5.8	7.1
Value of X_i	2.88	3.03	4.16	2.60	3.75
Average, $w=3$		3.357	3.263	3.503	

A moving average with window size $w = 3$ is used in this example. It means that the average of the first three values is taken (equal to 3.357, shown in column labelled '2'), then the average of values X_2, X_3 and X_4 is taken, and finally, the average of X_3, X_4 and X_5 is taken. A larger window size results in less averages. The effect the moving average has is to 'smooth' the original time-series signal (the values of X_i), so that a truncation point can be selected after visual

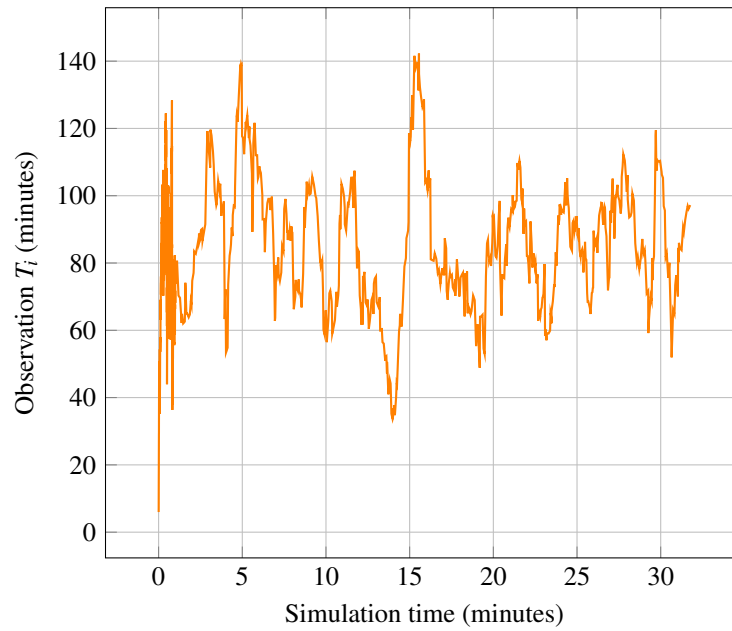


Figure 5.10: Typical output of a non-terminating system

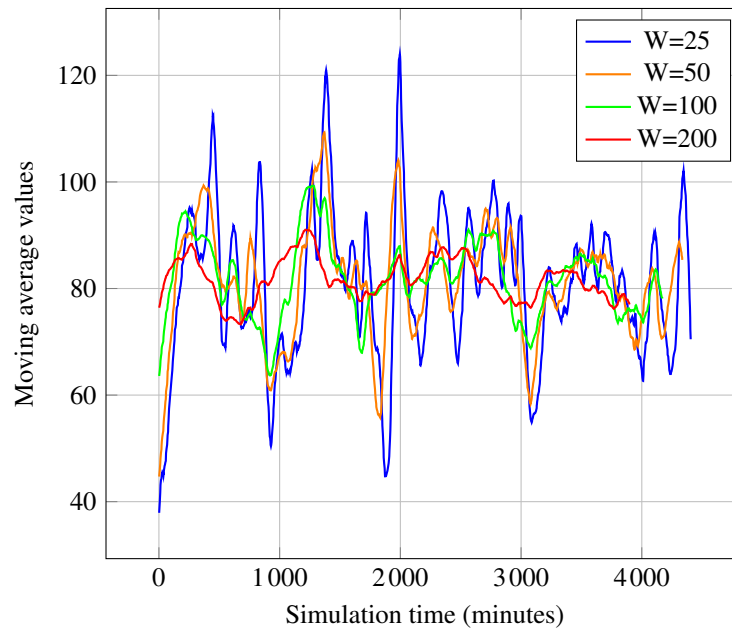


Figure 5.11: Moving averages with different window sizes

inspection. The observations shown in Figure 5.10 were smoothed using different window sizes, and the results are shown in Figure 5.11. The truncation point can be selected anywhere from 100 time units onwards, as it seems that the observations start to vary around a mean, while it seems as if the values increased from $t = 0$ up to 100. Choosing the truncation point is subjective and there are many correct answers for a given output series.

Once the truncation point is determined, the production runs can be executed with a warm-up period of length equal to l . There are several methods for the analysis of the steady state observations (see for example Chen and Kelton (2000)), but only two, namely the *Replication/Deletion*

and *Batch means*, will be discussed.

5.4.2 Replication/Deletion approach

The analysis in this method is similar to that for terminating systems, but the observations of the warm-up period are discarded. The method seeks to obtain statistically independent observations to make the application of the methods of the terminating system analysis valid.

1. Make N replications of length M , where M is at least $2l$.
2. The averages across replications are given by

$$\bar{Y}_j = \frac{\sum_{i=l+1}^M Y_{ji}}{(M-l)} \quad \text{for } j = 1, 2, \dots, N.$$

The Y_j uses only the observations from the j th replication of the steady state, that is $Y_{j,l+1}, Y_{j,l+2}, \dots, Y_{j,M}$.

3. The \bar{Y}_j 's are IID variables with $\mu \approx E(Y_j)$. $\bar{Y}(N)$ is an approximate unbiased point estimator for μ , and an approximate $100(1 - \alpha)$ CI is for μ is given by

$$\bar{Y}(N) \pm t_{N-1, 1-\alpha/2} \sqrt{\frac{S^2(N)}{N}}$$

where $\bar{Y}(N)$ and $S^2(N)$ are calculated from equations 5.2 and 5.3, respectively.

5.4.3 Batch-means approach

This method also seeks to obtain statistically independent observations to make the application of terminating system analysis possible. Only one replication is required with this method, as opposed to the Replication/Deletion approach; the simulation run therefore passes only once through the warm-up period.

Assume that the l observations of the warm-up period have been discarded, so that we deal only with observations X_{l+1}, X_{l+2}, \dots . Suppose the run produced m observations, then the observations are divided into n batches of length k . Batch 1 therefore consists of observations X_1, X_2, \dots, X_k , batch 2 consists of observations $X_{k+1}, X_{k+2}, \dots, X_{2k}$ and so forth. These observations are now averaged per batch, so that the n batches produce n averages $\bar{X}_1, \bar{X}_2, \dots, \bar{X}_n$, which form the new set of observations (also see Fishman (1978)). The overall sample mean is

$$\bar{\bar{X}}(n, k) = \frac{\sum_{j=1}^n \bar{X}_j(k)}{n} \tag{5.10}$$

which will be used as the point estimator for μ . The terms *warm-up*, *batches* and *steady state* are explained in Figure 5.12.

The main problem with this method is the choice of the batch size k . Many textbooks state that this method is acceptable if the batch size is *large enough*. But what is *large enough*? The correlation structure of the observations and the correlogram may provide a solution to this restriction, but before this concept is explained, additional background will be provided on *covariance* and *correlation*.

Covariance

The covariance between two random variables X and Y is a measure of their linear dependence, and is defined as

$$\sigma_{XY} = \text{Cov}(X, Y) = E[(X - \mu_X)(Y - \mu_Y)]. \tag{5.11}$$

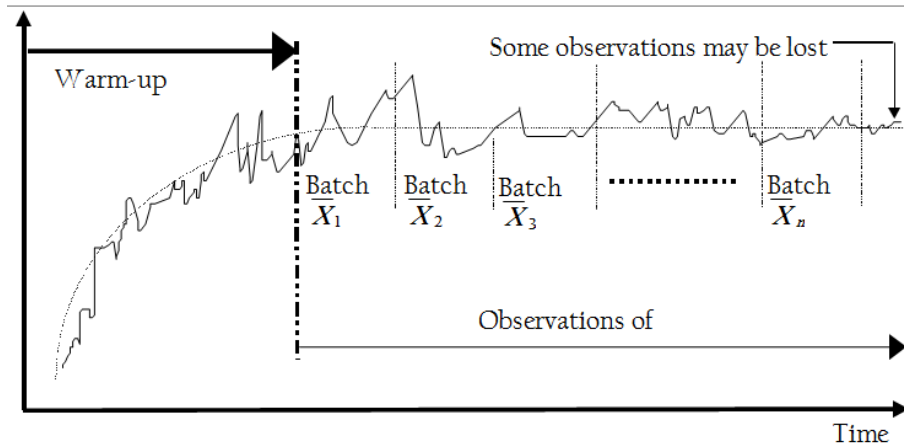


Figure 5.12: Discarding the transient phase and batching observations

- X and Y are uncorrelated if $C_{XY} = 0$, and if X and Y are independent, then $C_{XY} = 0$. The converse however, is *not generally true*.
- If $C_{XY} > 0$, then X and Y are positively correlated, so that $X > \mu_X$ and $Y > \mu_Y$ tend to occur together, and $X < \mu_X$ and $Y < \mu_Y$ occur together. Thus, if the one random variable is large, the other tends to be large (think of waiting times in a queue – if the customer in front of you waits a long time to be served, it is likely that the time you spend in the queue will also be long).
- If $C_{XY} < 0$, then X and Y are *negatively correlated*, so that $X > \mu_X$ and $Y < \mu_Y$ tend to occur together, and $X < \mu_X$ and $Y > \mu_Y$ occur together. Thus, if the one random variable tends to be large, the other tends to be small.

Correlation

The covariance σ_{XY} as a measure of dependence between two random variables is not dimensionless, which makes its interpretation difficult. If the random variables are measured in dimension x for example, then the covariance is measured in dimension x^2 . A dimensionless measure of linear independence is the *correlation* or *coefficient of correlation*, defined as

$$\rho_{XY} = \frac{\sigma_{XY}}{\sqrt{\sigma_X^2 \sigma_Y^2}} \quad (5.12)$$

and $-1 \leq \rho \leq 1$. Things to note:

- ρ_{XY} will have the same sign as σ_{XY} .
- If ρ_{XY} is close to +1, then the random variables X and Y are highly positively correlated.
- If ρ_{XY} is close to -1, then the random variables X and Y are highly negatively correlated.

These definitions imply that we have two sets of data from which we want to determine a covariance or correlation, for example X = mass of a person and Y = height of a person. If we sample a group of people by determining the mass of each, and measure the height of each, we could determine whether there is a correlation between a person's mass and height or not.

P **Point to ponder:** What is the difference between 'causation' and 'correlation'?

Covariance and correlation in a single data set

With simulation, we usually have a single set of observations per parameter under study, for example the flow time of patients through an emergency room. (We can of course have more

than one set of observations, which we want to study). It is often desired to determine if there is correlation among the observations in order to apply certain statistical operations, which assume independent observations. How would we determine this? The answer is to use the correlation coefficient as defined above with the single data set implicitly divided into subsets. This coefficient gives a measure of correlation/dependence between two observations and is used instead of the covariance, because it is dimensionless and therefore easier to interpret.

A different way to determine the covariance and correlation of a *single data set* must now be followed. The covariance in a single data set is defined as

$$C_j = \sum_{i=1}^{n-j} \frac{(X_i - \bar{X})(X_{i+j} - \bar{X})}{n-j} \quad (5.13)$$

where n is the number of observations. The sample correlation ρ_j is defined as

$$\rho_j = \frac{C_j}{\sigma^2} \quad (5.14)$$

where j is the so-called *lag*. The application of (5.13) and (5.14) is illustrated in the following paragraph.

Determining the correlation in a single data set

As mentioned earlier, we usually have a single set of observations per parameter under study which we want to analyse statistically, but we first have to know whether the observations are independent or not, or alternatively the degree of dependence on a scale of -1 to $+1$.

We now do not have two random variables X and Y that are each represented by a set of data, and we cannot use the strict definitions as defined above. We will have to work with data sets within our single data set (which reduces this explanation to one of explaining the use of indices of variables in a vector).

Suppose we have $n = 20$ observations, say random numbers between 1 and 10 inclusive. To determine the degree of dependence among the observations, we calculate the correlation. The process is as follows:

Calculate the sample mean with

$$\bar{X} = \sum_{i=1}^n \frac{X_i}{n}.$$

Estimate the *population* variance

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

and calculate the covariance among the observations within the vector, applying the indices as indicated in

$$C_j = \sum_{i=1}^{n-j} \frac{(X_i - \bar{X})(X_{i+j} - \bar{X})}{n-j}. \quad (5.15)$$

This process in (5.15) is repeated for various lag sizes, and a great number of calculations are to be done to generate a correlogram when the number of lags is fairly large. It is often sufficient to draw the correlogram for 50 lags, but the final number depends on the problem. An example with a small sample in a spreadsheet with $j = 5$ correlations calculated is shown in Table 5.1; note the many numbers that are needed to be stored to obtain five results. It can be seen that, if there are say n observations in the data set under study, then for $j = 1$ one needs to make $n - 1$ subtractions of \bar{X} and $n - 1$ multiplications, then sum the terms. In the example given, this is executed 85 times.

Table 5.1: Example: Correlation calculations on a single data set

Obs. No.	Obs. i	$(X_i - \bar{X})(X_{i+j} - \bar{X})$					
		$j = 1$	$j = 2$	$j = 3$	$j = 4$	$j = 5$	
	1	7.984	0.132	-7.281	2.946	3.810	1.867
	2	6.830	-0.607	0.246	0.318	0.156	0.278
	3	0.942	-13.531	-17.496	-8.573	-15.301	17.512
	4	9.065	7.080	3.469	6.192	-7.087	-5.533
	5	9.750	4.486	8.007	-9.164	-7.154	9.443
	6	8.207	3.923	-4.490	-3.506	4.627	-2.120
	7	9.371	-8.014	-6.256	8.258	-3.783	6.250
	8	3.696	7.161	-9.452	4.330	-7.153	-5.003
	9	4.360	-7.379	3.380	-5.584	-3.906	13.357
	10	9.845	-4.462	7.371	5.156	-17.631	1.038
	11	5.295	-3.377	-2.362	8.077	-0.475	-3.673
	12	9.087	3.902	-13.344	0.785	6.067	-9.061
	13	8.377	-9.333	0.549	4.244	-6.338	4.911
	14	1.075	-1.878	-14.512	21.673	-16.793	-1.938
	15	7.057	0.854	-1.276	0.988	0.114	-0.705
	16	9.293	-9.855	7.636	0.881	-5.446	
	17	2.888	-11.404	-1.316	8.133		
	18	9.697	1.020	-6.302			
	19	7.068	-0.727				
	20	4.605					
Average $\bar{X} =$	6.725	Sum/ $(n - j) =$	-2.211	-2.968	2.656	-4.768	1.775
Pop. Variance =	8.025	$\rho_j =$	-0.275	-0.370	0.331	-0.594	0.221
$n =$	20						

The Correlogram

The correlogram is a graphical representation of the correlations against the lag numbers and is presented in a bar chart format. It is used to determine where the correlation in the data set diminishes to a satisfactory level *i.e.* close to 0. A typical correlogram is shown in Figure 5.13.

Note how the correlation values decrease with increasing lag number.

The correlogram is used to visually determine the lag number where the correlation becomes insignificant, *i.e.* close to 0. The need for this lag number will become apparent in the following paragraph. Note that the lags may or may not cross the zero line. The single lag where a zero crossing occurs may not be taken as the number where no correlation exists; a range of lags must be close to zero and a number in this range should be considered for the desired lag number. The first number in such a range is usually taken, provided the correlations further on are lower than the selected value.

It is also important to note that the window size, the value of j in (5.13), influences the correlogram. A small value for the window will result in a partial correlogram that will not reveal the approximate zero correlation lag. The maximum value for the window is also limited – if n observations were made, and the window size is j , it follows from equation 5.13 that only $n - j$ lags can be computed.

Determining the production run length for the Batch-means approach

The first step in analyzing a stochastic non-terminating model, a pilot run, must be executed to determine the truncation point and the approximate zero correlation lag number. The replication

length must be determined by iteration, or a very long run may be used but with a computer time penalty. The iteration process may be further extended if the correlogram does not show acceptable zero correlation levels for all parameters under study.

Once the lag number where correlation is acceptably small is identified, it can be used to determine the batch length (in simulation time), and consequently the actual length of the single replication for the production run. Suppose we select $j = 150$ in Figure 5.13, which means that the batch size is 150. As a rule, the batch size is selected at least one order of magnitude larger, *i.e.* 10 times larger than required. In this example, the batch size is thus $150 \times 10 = 1\,500$ observations. Suppose the duration of the pilot run was 1 200 simulation hours during which 800 observations of the parameter under study were made. Assume it is given that 800 observations required 1 200 hours of simulated time, so that 1 500 observations will require

$$\begin{aligned} \text{Replication length} &= \frac{1\,500 \text{ obs.} \times 1\,200\text{h}}{800 \text{ obs}} \\ &= 2\,250 \text{ simulation hours.} \end{aligned}$$

The analysis is now similar to that of a terminating system – a single run of $10 \times 2\,250 \text{ h} = 22\,500$ simulated hours will be executed to produce 10 batch-means. The warm-up period must also be added to this time. Suppose the warm-up period was determined to be 1 000 simulated hours, then the actual run length will be $22\,500 + 1\,000 = 23\,500$ simulated hours. The 10 observations are determined by calculating the average of each batch, that is 10 averages of the 15 000 observations produced by the 22 500 simulated hours run. The first average (observation) follows from the average of the first 1 500 observations, and the second observation from the average of the second set of 1 500 observations *etc.* The observations made during the warm-up period must of course, first be discarded.

A confidence interval can now be constructed per parameter, and the required number of replications can be determined. The production run will then be done for n^* times the time length per batch length, which is 2 250 simulated hours in this case. The warm-up period must once again be added to the overall time. It is recommended that the truncation point and the zero

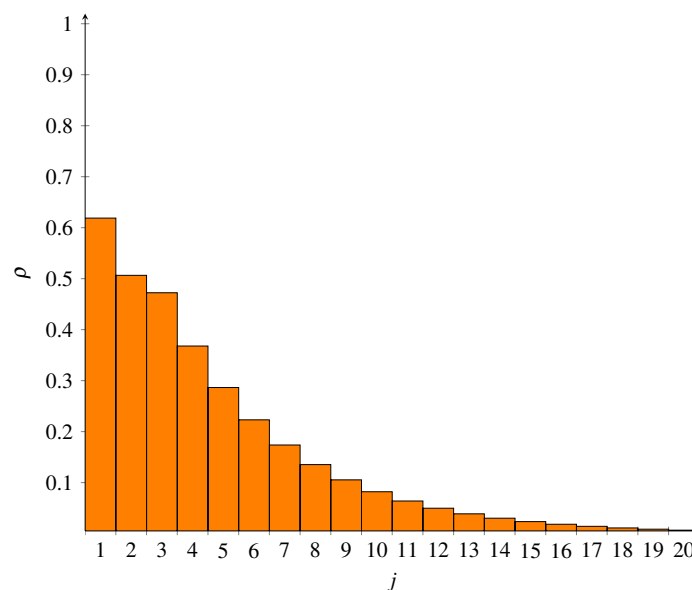


Figure 5.13: A typical correlogram

correlation lag be confirmed prior to the production run, by again analysing the output of the 10 batch-pilot run.

It is possible to test whether the batch size is sufficient or not. The von Neumann test as outlined in Banks (1998) can be used to determine if the batch means are independent.

Summary: Output analysis of non-terminating systems with Batch means

The analysis of a non-terminating system with the batch means approach is summarised in the following steps:

1. Do an initial run of arbitrary length.
2. Determine the length of the warm-up period per parameter with the method of moving averages. Iterate with the window sizes if necessary.
3. Do a long enough run so that sufficient data can be collected in the steady state. Use the data of the steady state to draw the correlogram.
4. If the length of the warm-up period as well as the approximate zero correlation lags can be determined, continue with the following steps, otherwise repeat the steps above with increasing run length.
5. Determine the number of observations required per batch by multiplying the approximate zero correlation lag number with 10. If more than one parameter is studied, the approximate zero correlation lag is the maximum of the lags of all parameters. If parameter 1 has an approximate zero correlation lag of 120, and parameter 2 has an approximate zero correlation lag of 95, then $120 \times 10 = 1\,200$ observations will be required per batch.
6. Determine the run time required to produce the required number of observations, and include the warm-up period.
7. Do a single replication pilot run of simulation time length as determined above.
8. Verify the truncation point and the approximate zero correlation lags.
9. Calculate the averages per batch.
10. Construct preliminary confidence intervals per parameter, and determine the required number of batches for each parameter's desired half-width.
11. Determine the production run length by multiplying the maximum number of replications required by the time required per batch, and add the warm-up period length.
12. Execute the single replication production run.
13. Calculate the averages of the observations per batch.
14. Draw confidence intervals per parameter.

P Points to ponder: What will the effect be on your mean estimation of you choose the warm-up length 'too short'? And 'too long'? Can it be 'too long'? What effect does the choice of the lag number j have on the estimation of the mean?

5.5 Evaluating and selecting scenarios

Simulation studies usually focus on a predefined set of scenarios to answer 'what-if' questions and one or more output parameters are used to determine which scenario is best. The number of scenarios could be small or large. Often, the simulation analyst formulates the scenarios, with or without stakeholders, then determines the best of these. A simple example is: do I need three, four, or five cashiers in a particular retail shop on Saturdays? This question implies three scenarios.

Defining scenarios thus means that we know what we want to investigate; in this case, it is usually a 'small' number. A 'large' number of scenarios usually constitutes simulation problems with such a large number of scenarios that it is impractical, or even impossible, to define them manually. For these problems, we need computerised methods to ease our task, which is the topic

of Chapter 6, but for now, the focus is on finding the best among a small number of scenarios. Several methods are available, collectively called *ranking and selection* (R&S) methods. For an overview of these methods, see Moonyoung Yoon (2017).

In this section, we shall study two of these methods, the first being *analysis of variance* (ANOVA), as it is the method Tecnomatix Plant Simulation uses. A second approach, namely the *Kim-Nelson method* (K-N method), will also be introduced and applied in this module using MS Excel.

5.5.1 Analysis of variance

Analysis of variance (ANOVA) is implemented in Tecnomatix Plant Simulation to assist the simulation analyst with finding the best scenario in a small set of scenarios. When analysing stochastic systems, the estimated parameter values may be numerically different, but they may not be *statistically significantly different*. A throughput value of $\bar{X}_1 = 22.4$ is not necessarily different from $\bar{X}_2 = 23.5$. ANOVA allows for finding scenarios that differ statistically significantly.

We use one-way ANOVA to determine the ‘better’ of a **small** number of scenarios. ‘One-way’ means a single output parameter is studied and compared. ‘Better’ is based on the estimated values of the output parameter, *e.g.* Throughput.

The general hypothesis test using ANOVA for k scenarios is

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \dots = \mu_k$$

$$H_1 : \mu_1 \neq \mu_2 \neq \mu_3 \neq \dots \neq \mu_k.$$

In our simulation studies, we test scenarios pair-wise: assume we have three scenarios to compare, then we compare Scenario 1 with 2, 1 with 3, and 2 with 3. For this purpose, the *paired t-test* is used while assuming unknown, unequal variances. The paired t-test is considered to be a special case of the general ANOVA.

R If three or more scenarios are to be compared, we should use the *F-test*, followed by a *post-hoc* test like Scheffe *post hoc* F test or the Tukey *post hoc* test to determine which pair of means is unequal. The paired t-test for more than two scenarios may result in compromising the Type I error rate. It could be used but the α -value should be corrected using the Bonferroni correction α/k , where k is the number of scenarios.

The critical test value is obtained by specifying a value for α , typically 5%. The test yields a p -value, and if $p < \alpha$, H_0 is rejected. This indicates that at least one μ_i is not equal to the others. The assumptions for the test are

- The samples are from normally distributed populations.
- Samples are independent.

The test is fairly robust and deviations from these assumptions do not affect results too much. The test values are determined as follows (Walpole and Myers, 1993):

H_0	Test statistic	H_1	Critical region
$\mu_1 - \mu_2 = d_0$	$t = \frac{(\bar{x}_1 - \bar{x}_2) - d_0}{\sqrt{(s_1^2/n_1) + (s_2^2/n_2)}}$	$\mu_1 - \mu_2 < d_0$	$t < -t_\alpha$
		$\mu_1 - \mu_2 > d_0$	$t > t_\alpha$
		$\mu_1 - \mu_2 \neq d_0$	$t < -t_{\alpha/2}$
			and $t > t_{\alpha/2}$
	$v = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}}$		
	$\sigma_1 \neq \sigma_2$ and unknown		

Tecnomatix does pair-wise comparisons to test pairs of means. If a test for any pair (μ_i, μ_j) , shows that $p < \alpha$, then $\mu_i \neq \mu_j$. If $\mu_i < \mu_j$ and we maximise, then Scenario j is better than

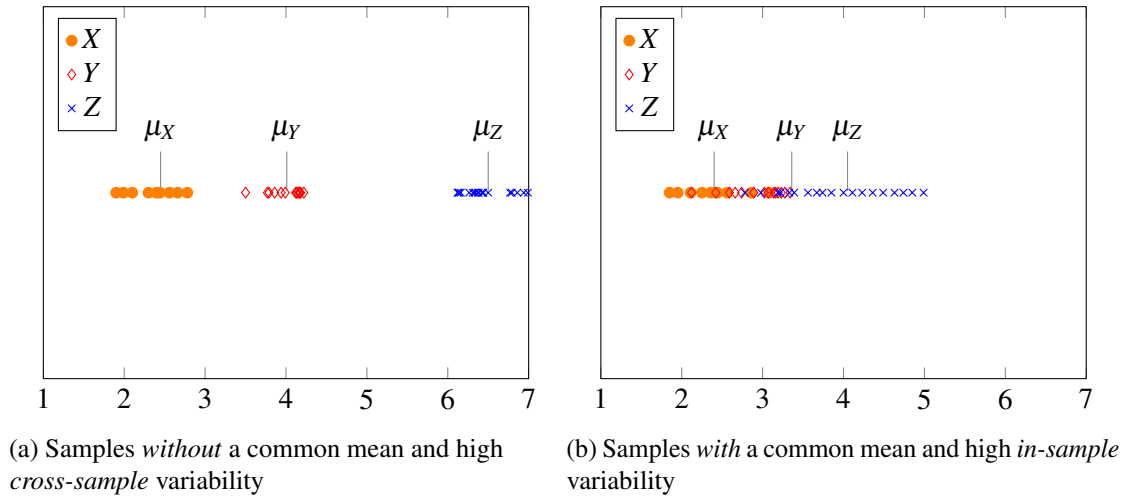


Figure 5.14: Cross-sample vs. in-sample variability

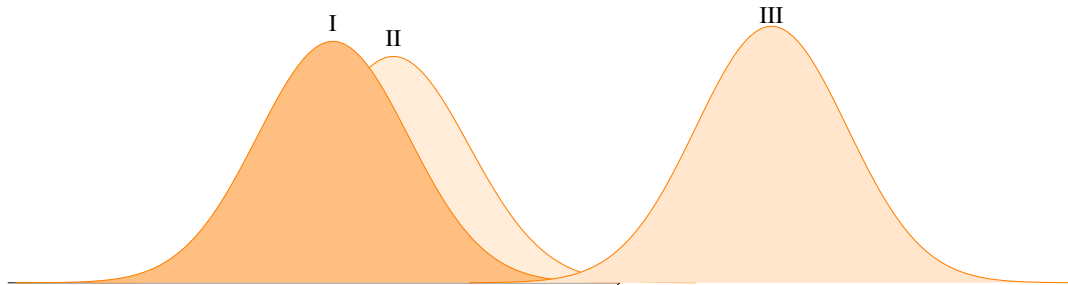


Figure 5.15: Samples with different distribution characteristics

Scenario i , and they differ statistically significantly.

The difference can be explained using Figure 5.14a. It shows that

- The populations do not have a *common mean*.
- The variability *between* samples is high.
- These imply that the populations *are different*, *i.e.* in simulation terms: the scenarios are different.

In Figure 5.14b, the populations **do** have a *common mean* while the variability *within* samples is high. These imply that the populations are not different, *i.e.* in simulation terms: the scenarios are *not different*.

Figures 5.14a and 5.14b show the difference between in-sample and cross-sample variation, and we conclude that if two scenarios are similar, *in-sample variation* is observed, while if two scenarios are different, their output exhibit *cross-sample variation*. The two figures can also be interpreted using Figure 5.15.

The two distributions labelled ‘I’ and ‘II’ are from the same population and exhibit in-sample variation, while the third distribution (labelled ‘III’) is from a different population, hence cross-sample variation between I, II *versus* III.

5.5.2 Applying ANOVA

Tecnomatix gives us a plot of the CI per experiment (scenario). It also gives us a pair-wise comparison of the estimated means of each experiment, and we need to determine which experiment is best. Consider an example with five experiments, where the real-life scenarios defined by them are increasingly more expensive, *i.e.* Exp1 is the cheapest to implement,

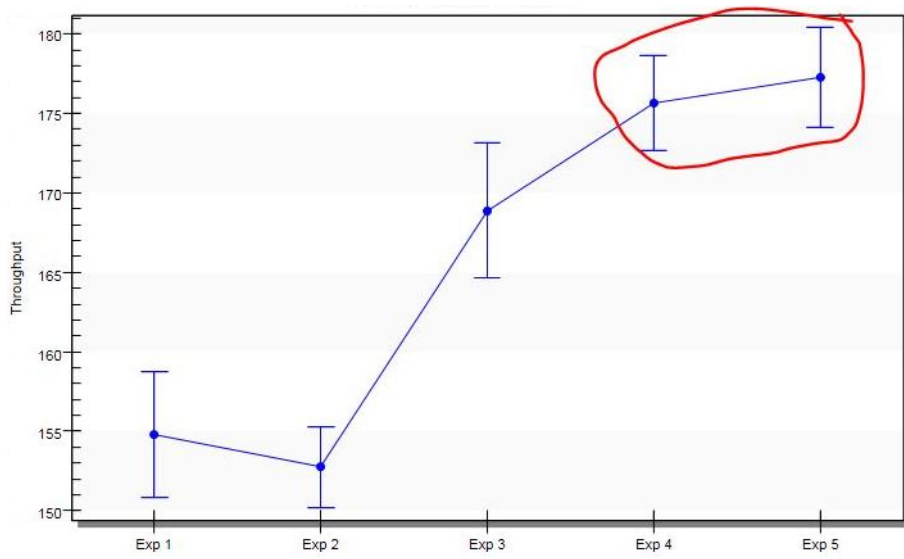


Figure 5.16: CI plot for ANOVA with highest scenario values

Table 5.2: Structure of p -value comparisons

	Exp 2	Exp 3	Exp 4	Exp 5
Exp 1	0.359	0	0	0
Exp 2		0	0	0
Exp 3			0.01	0.002
Exp 4				0.438

followed by Exp2, while Exp5 is the most expensive to implement. Suppose that, as an example, we simulate processes in a day-care hospital, and we want to maximise the *throughput* of the system, which is defined as *e.g.* Number of patients served per time period. A confidence interval plot for the five experiments is shown in Figure 5.16, with $\alpha = 5\%$. Each vertical bar shows a CI for an experiment, with each expected mean shown as a dot in the middle. The length of the line indicates the 95% confidence interval of the observations. Since we want to maximise throughput in this example, the values for Exp1 and Exp2 are too low for consideration and we reject these scenarios (experiments). The highest values are for Exp4 and Exp5, as shown in Figure 5.16.

It is clear that Exp5 has the highest numerical value for throughput, but is it really better than Exp4? We use the table with p -values from the ANOVA to determine that. The p -values associated with Figure 5.16 are shown in Table 5.2, for $\alpha = 5\%$. The direction of the ANOVA is symmetrical, *i.e.* comparing Exp4 and Exp5 is the same as comparing Exp5 and Exp4.

One can see that Exp1 and Exp2 have a common mean, as the p -value=0.359. Exp1 and Exp3 differ significantly, because the p -value<5%. This is also true for Exp(1, 4), Exp(1, 5), Exp(2, 3), Exp(2, 4), Exp(2, 5), Exp(3, 4) and Exp(3, 5). But Exp4 and Exp5 do not differ significantly because $p = 0.438 > \alpha$. We choose Exp4 as the best, since it has the highest throughput; although Exp5 has a numerically higher throughput, it is similar to Exp4 but more *expensive*. Remember the experiments were arranged to become progressively more expensive. We would thus not select Exp5, as it *yields the same output but is more expensive*. We can interpret this statement using an example: suppose the five scenarios represented appointment of one to five doctors in the day-care hospital, then Experiment 5 yields similar patient service, but with one more (expensive) doctor. The example shows the effect of in-sample variation (Figure

5.14b (Exp4 and Exp5), and cross-sample variation, *e.g.* Exp1 and Exp5 (Figure 5.14a).

Example of t-test

This example illustrates how the p -values in Table 5.2 were obtained. Suppose we have three scenarios, S1, S2 and S3, and the output parameter is *Cost*. Since we want to minimise cost, we seek the scenario which has the lowest cost and which differs statistically significantly from the others. The observations are shown in Table 5.3.

We need to do t-tests for (S1;S2), (S1;S3) and (S2;S3), and we test for no difference of observations, *i.e.* $\mu_1 - \mu_2 = d_0 = 0$. If the test fails, we can say that there is a significant difference between the two tested data sets. The results are shown in Table 5.4.

To calculate for example the t -statistic for (S1; S2), we use

$$\begin{aligned} t &= \frac{(\bar{x}_1 - \bar{x}_2) - d_0}{\sqrt{(s_1^2/n_1) + (s_2^2/n_2)}} \\ &= \frac{(26.3 - 27.0) - 0}{\sqrt{(0.357/10) + (1.875/10)}} \\ &= -1.6212 \end{aligned}$$

Table 5.3: Example observations for paired t-test

	S1	S2	S3
	27.0	28.3	24.2
	25.8	27.8	23.7
	26.6	27.8	23.6
	26.4	27.5	23.4
	24.9	24.2	21.4
	26.3	27.2	23.3
	26.8	27.5	23.5
	26.7	28.5	24.0
	26.1	25.4	22.7
	26.1	26.2	23.2
n_i	10	10	10
Mean	26.3	27.0	23.3
Variance	0.357	1.875	0.605

Table 5.4: Results for t-test example

	S1;S2	S1;S3	S2;S3
v	12	17	14
t-stat	-1.6212	9.5134	7.4622
p -value (two tails)	0.1309	0	0
t critical	2.1788	2.1098	2.1448

and the degrees of freedom with

$$\begin{aligned}
 v &= \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}} \\
 &= \frac{(0.357/10 + 1.875/10)^2}{\frac{(0.357/10)^2}{10 - 1} + \frac{(1.875/10)^2}{10 - 1}} \\
 &= 12 \text{ (rounded)}.
 \end{aligned}$$

Note that $n_1 = n_2 = n_3 = 10$, but it is not required, so the n_i may differ. The p -value can be found on statistical tables while taking the t -statistic value as absolute. If the table specifies a one-tailed value, then the obtained probability must be multiplied by 2. The t -test can also be done in MS-Excel: on the menu ribbon, click Data, Data Analysis, and choose t -Test: Two Sample Assuming Unequal Variances and follow the intuitive input requirements. In the next subsection, a procedure with a different approach to R&S is presented.

5.5.3 The Kim-Nelson procedure

The second ranking and selection procedure we study is the Kim-Nelson procedure (K-N procedure), due to Kim and Nelson (2001). There are two important concepts associated with the K-N procedure (and similar R&S procedures) which we must understand; they are:

- The *probability of correct selection*, $P(\text{CS})$. This is the probability of *correctly* selecting the scenario with the best output parameter value (*e.g.* lowest cost, highest throughput, shortest delay time). Since we are studying stochastic-driven scenarios, there is a probability of selecting the wrong one after analysis. This probability can be minimised but then many simulation replications must be done. We choose the probability of correct selection beforehand, and indicate it by $P(\text{CS}) = 1 - \alpha$, where typically $\alpha = 5\%$.
- The *indifference zone*, and indifference zone values, usually indicated by δ , δ_i or δ_{ij} , depending on the problem description. This value is specified by the simulation analyst beforehand, and is a value which indicates that the analyst is *indifferent* to two scenarios that have estimated means within the indifference zone value. The theoretical condition is $|\mu_1 - \mu_2| \leq \delta$, while the simulated condition would be $|\bar{X}_1 - \bar{X}_2| \leq \delta$. Of course, the value of δ must be carefully selected, keeping practical considerations in mind. Suppose we study the time a part spends waiting in a process, then it does not make sense to set $\delta = 0.01$ seconds, if the typical time is 3 hours and 20 minutes. So the simulation analyst must use judgement when choosing values for δ .

The K-N procedure is a *sequential* procedure, which means one estimates the output values using an initial number of replications n_0 , then sequentially does more replications per scenario, until sufficient observations are made to ensure $P(\text{CS})$. Estimates are thus calculated for n_0 replications, then for $n_0 + 1$ replications, and so on. The procedure is as follows (Kim and Nelson, 2001), assuming k scenarios:

Procedure KN

1. *Setup*. Select the overall desired $P(\text{CS}) = 1 - \alpha$, the value for δ , and common first stage sample size $n_0 \geq 10$. Set $\eta = \frac{1}{2} \left[\left(\frac{2\alpha}{k-1} \right)^{-2/(n_0-1)} - 1 \right]$.
2. *Initialisation*. Let $I = \{1, 2, \dots, k\}$ be the set of scenarios still in contention, and let $h^2 = 2\eta(n_0 - 1)$. Obtain n_0 outputs X_{ij} ($j = 1, 2, \dots, n_0$) from each scenario i ($i = 1, 2, \dots, k$) and let $\bar{X}_i(n_0) = \sum_{j=1}^{n_0} X_{ij} / n_0$ denote the sample mean of the first n_0 outputs from scenario i .

Table 5.5: Snapshot of observations X_{ij} for K-N procedure example

	i	$j \rightarrow$					
Scenario	1	7.20	5.61	8.44	6.30	10.48	...
Scenario	2	9.25	6.58	8.07	6.36	5.90	...
Scenario	3	8.32	7.34	16.81	9.89	10.96	...

For all $i \neq l$ calculate

$S_{il}^2 = \frac{1}{n_0 - 1} \sum_{j=1}^{n_0} (X_{ij} - X_{lj} - [\bar{X}_i(n_0) - \bar{X}_l(n_0)])^2$, the sample variance of the difference between scenarios i and l . Set $r = n_0$.

3. *Screening*. Set $I^{\text{old}} = I$. Let

$$I = \{i : i \in I^{\text{old}} \text{ and } \bar{X}_i(r) \geq \bar{X}_l(r) - W_{il}(r), \forall l \in I^{\text{old}}, l \neq i\}$$

$$\text{where } W_{il}(r) = \max \left\{ 0, \frac{\delta}{2r} \left(\frac{h^2 S_{il}^2}{\delta^2} - r \right) \right\}.$$

4. *Stopping rule*. If $|I| = 1$, then stop and select the scenario whose index is in I as the best. Otherwise, take one additional output $X_{i,r+1}$ from each system $i \in I$, set $r \leftarrow r + 1$, and go to Step *Screening*.

In Step 2 of the procedure, the subscript ‘ il ’ means ‘between Scenario i and Scenario l ’; the same applies to Step 3. The requirement in Step 2 “For all $i \neq l$ calculate” means we take all combinations of scenarios, for example, if we have three scenarios, then the combinations would be ‘1-2’, ‘1-3’ and ‘2-3’. The combinations are symmetric, so we do not consider ‘2-1’ and so on. We do not consider ‘1-1’, ‘2-2’ and ‘3-3’.

In Step 3, $W_{il}(r)$ determines how far the sample mean from Scenario i can deviate from the sample means of the other systems without being eliminated. If $W_{il}(r)$ becomes large, then Scenario i is eliminated. In the case of maximising the best scenario, the scenario that is finally selected has the largest true mean of all scenarios, and the true mean of the best scenario is also at least δ better than the second best.

Example: Kim-Nelson ranking and selection procedure

Three scenarios are present in this example. One thousand observations were created in advance from three normal distributions with means and standard deviations of $\mu_1 = 5, \sigma_1 = 1.8, \mu_2 = 10, \sigma_2 = 2$ and $\mu_3 = 12, \sigma_3 = 4$, respectively.

The 1 000 observations were created for the *purpose of this example* and may be too many – in practice, the simulation model will start with n_0 observations and sequentially make additional observations when necessary, which is the whole idea of the K-N procedure. A snapshot of the observations X_{ij} is shown in Table 5.5. We maximise, so we expect Scenario 3 to come out best because $\mu_3 = 12$.

The settings for Procedure K-N for the example are shown in Table 5.6. The value of η is calculated as in Step 1 of the procedure. Note that the indifference value was chosen as $\delta = 0.5$; this will/may dictate the outcome of the procedure: if it is chosen very small, many observations may be needed to distinguish scenarios, and it may become impossible for the procedure to do so. If it was chosen too large, the wrong scenario may be chosen as best because we ‘tell’ the procedure that we are very indifferent about the outcome. It is recommended to start with a relatively large value for δ , then decrease it as long as the optimisation does not take ‘too long’.

Table 5.6: Settings for the example of the K-N procedure

$\alpha = 5\%$	$k = 3$	$\delta = 0.5$	$n_0 = 10$	$\eta = 0.473$	$h^2 = 8.513$
----------------	---------	----------------	------------	----------------	---------------

The initial values for the K-N procedure are shown in Table 5.7.

Table 5.7: Initial values for K-N procedure example

	$\bar{X}_i(n_0)$	S_{il}^2
$il: 1,2:$	6.77	11.29
$il: 2,3:$	8.085	23.56
$il: 1,3:$	11.93	19.63

The progress of the K-N procedure is shown in Table 5.8. The scenario pairs were considered in the order (1,2), (2,3) and (1,3). The combinations are symmetric due to the squared term in Step *Initialisation*. First, the \bar{X}_s , $s = 1, \dots, k$ and the S_{il}^2 are determined, based on n_0 observations. These are shown in Table 5.7. Initially, 10 observations were made, then 11, then 12, until after 40 observations, only Scenario 3 was left in the set I . The note in the far right column indicates whether or not the scenario remains in set I , if not, it is not further considered.

The term $\bar{X}_i(r) \geq \bar{X}_l(r) - W_{il}(r)$ in Step *Screening* can be rearranged and written as $W_{il}(r) \geq \bar{X}_l(r) - \bar{X}_i(r)$. Scenario i stays as long as this is true, otherwise it is removed from the set I . In Table 5.8, at $r = 27$, $\bar{X}_1 = 5.92$, $\bar{X}_2 = 8.61$, and $W_{12}(27) = 2.57$. Because $W_{12}(27) = 2.57 \not\geq 8.61 - 5.92 = 2.69$, Scenario i is removed from I .

In the row at $r = 40$, the value of $W_{12} = 3.56$ is striked through, because it is not considered in the screening; only $W_{23} \geq \bar{X}_3 - \bar{X}_2$ is considered. In this row, $W_{23} = 2.93 \not\geq \bar{X}_3 - \bar{X}_2 = 3.71$, and Scenario 2 is eliminated, leaving Scenario 3 as the winner.



 Why bother with Procedure K-N if we have ANOVA?

Table 5.8: The sequential progress of the K-N procedure example











r	\bar{X}_i	$W_{il}(r)$	Stay/Out?
10	6.77	9.36	Stay
	8.08	19.80	Stay
	11.93	16.46	Stay
11	6.76	8.17	Stay
	8.21	17.32	Stay
	11.91	14.39	Stay
...
27	5.92	2.57	Out
	8.61	5.63	Stay
	12.43	4.65	Stay
...
40	5.92	2.57	Out
	8.91	3.56	Out
	12.62	2.93	Stay

There are many more procedures to rank scenarios and select the best, and the R&S field is still being actively researched.

 “Statistics means never having to say you’re certain” (Senn, 2018).

Survival kit:

If you want to successfully wade through the output analysis assessment jungle, make sure the following tools are in your bag:

-  1 Know the difference between a TS and NTS.
-  2 Know what is a point and interval estimator.
-  3 Interpret the confidence interval.
-  4 Know how to calculate n^* for a TS.
-  5 Know the two NTS analysis methods.
-  6 Find the truncation point for each output parameter of an NTS.
-  7 Know and use correlation when simulating an NTS and apply the correlogram.
-  8 Determine the batch size per output parameter of an NTS.
-  9 Determine n^* for an NTS.
-  10 Determine the best in a small set of scenarios using the t-test (manually) and the *properties* of the K-N procedure (MS-Excel).

Introduction to simulation-optimisation

Some optimisation problems

- Rosenbrock function
- Rastrigin function
- Shekel function

The genetic algorithm

Application of GA principles in Tecnomatix

Multi-objective optimisation in large decision spaces with simulation

Multi-objective optimisation problem formulation

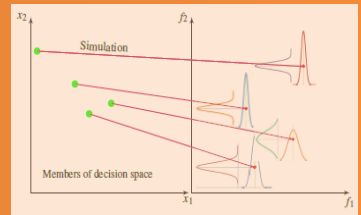
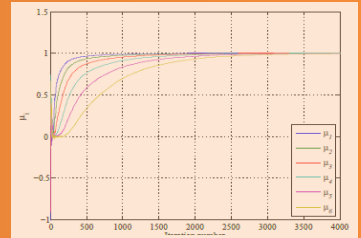
Ranking of solutions

Solution selection

Some MOO algorithms

Algorithm 4 MOO CEM Algorithm

```
1: Set Elite =  $\emptyset$ ,  $t = 1$ ,  $k = 1$ .
2: Initialise variable vectors  $X_i$ ,  $1 \leq i \leq D$ , and compute initial objective values.
3: For each decision variable  $x_i$ ,  $1 \leq i \leq D$ , initialise a histogram class vector  $C_i = [c_{i1}, c_{i2}, \dots, c_{i(p-1)}, c_{ip}]$  and histogram frequency vector  $R_i = [r_{i1}, r_{i2}, \dots, r_{i(p-1)}, r_{ip}]$ .
4: Set  $i = 1$ .
5: Set  $\kappa = 0$ .
6: Increment  $\kappa$ .
7: For each frequency element  $r_{ik}$  in  $R_i$  do
8:   Generate a class-based  $\hat{y}$  in the range  $[c_{ik}, c_{i(k+1)}]$ ,  $1 \leq \kappa \leq p+2$ .
9:   Generate a subsample  $Y$  according to the pdf  $\phi(x_i, Y)$ .
10:  with  $x_i \in [c_{ik}, c_{i(k+1)})$  and  $|Y| = r_{ik}$ ,  $1 \leq \kappa \leq p+2$ .
11:  Append  $Y$  to  $X_i$ .
12: end for
13: If  $\kappa < p+2$  return to Step 6.
14: Invert the histogram counts with probability  $p_k$ .
15: Increment  $i$ .
16: If  $i < D$ , return to Step 5.
17: Compute the  $NK$  objective function values using  $X_i$ ,  $1 \leq i \leq D$ .
18: Rank the objective function values using the Pareto ranking of Algorithm 3 with a relaxed  $p_R = 2$  to obtain an updated elite vector Elite.
19: Form new histogram class vectors  $C_i$  and histogram frequency vectors  $R_i$  based on Elite,  $1 \leq i \leq D$ .
20: Use the values in Elite and compute  $v_{ik}$  for all  $i$ ,  $1 \leq i \leq D$ .
21: Smooth the vectors  $v_{ik}$  for all  $i$ ,  $1 \leq i \leq D$ , using (3.15).
22: If all  $\sigma_k > \epsilon_k$  or less than the allowable number of evaluations has been done, increment  $t$  and reiterate from Step 4.
23: Rank the elite vector Elite using the Pareto ranking of Algorithm 3 with  $p_R = 1$ .
24: Increment  $k$ .
25: If  $k$  is smaller than the allowable number of loops, return to Step 2.
26: Rank the elite vector Elite using the Pareto ranking of Algorithm 3 with  $p_R = 0$  to obtain the final elite vector.
```



6. Optimisation with simulation

IN the previous chapter we learnt that one can use ANOVA or the K-N method, among others, to find the best scenario from a small number of scenarios in a simulation study. In this chapter we address the problem when we have a large decision space to investigate with simulation – spaces that are so large it would be impossible, due to time, to define and evaluate all solutions in search of the optimum.

6.1 Introduction to simulation-optimisation

Many simulation problems are of *combinatorial nature*, i.e. the decision variables can take on many values, resulting in a large decision space. The Buffer-allocation problem (BAP) is a good example of a combinatorial optimisation problem. Suppose we have four buffers, and we allow the size to vary between one and 10, all inclusive, then the decision space consists of 10^4 possibilities. If we want to evaluate each of these, we need to run $10^4 \times n^*$ replications. This could result in very long, impractical simulation times. Suppose the BAP takes one minute to run its n^* replications for a given buffer allocation, then it would take almost seven days to cover the full decision space of this example. Of course, if we have strong computational capability, the time can be reduced.

In optimisation we may use *exact algorithms* and *heuristics*. The exact algorithm finds the guaranteed optimal solution in finite time, provided the problem is not too large in terms of the number of decision variables. Linear programming is an example of an exact method, provided that certain assumptions are adhered to. If we cannot formulate a problem to solve it using exact methods, we often revert to simulation, and we call this *simulation-optimisation* (SO). In simple terms we can say that the simulation model becomes the ' $f(x)$ ' of an optimisation problem, i.e. it is the objective function, and it represents the complex but unknown relationships among decision variables.

To address this problem, meta-heuristics are used in conjunction with simulation to intelligently search the decision space and find good answers while evaluating combinations of decision variables. Because the computational time required to solve many real-life problems with large decision spaces may grow extremely quickly as the problem size increases, it is often better to try to find a solution for these problems that is approximate and almost as good as the best answer than to try to find the absolute best answer. Techniques that yield good, though not necessarily optimal solutions, are known as *heuristic techniques*. The word 'heuristic' is from

the Greek word ‘heuriskein’, which means to search. Heuristics are problem-specific, while an extension of them, namely *meta-heuristics*, is more generic and can solve a wide variety of problems, often without the need to understand the problem in detail. The term *meta-heuristic* is defined by Sörensen (2015) as

Definition 6. *A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms. The term is also used to refer to a problem-specific implementation of a heuristic optimization algorithm according to the guidelines expressed in such a framework.*

For a discussion and formal definition, see Fred Glover (2018). A meta-heuristic thus uses a heuristic (‘search guidance’ if you want to) in a specific way, hence the term *meta* which means ‘high-level’ or ‘beyond’. Meta-heuristics reduce the number of decision variable combinations so that a smaller subset of the decision space needs to be explored to find good objective function values. The meta-heuristic provides values for the decision variables in the simulation model, the model runs a number of replications with these settings, then returns point estimators for the output parameters (the ‘ $f(x)$ ’s). This process is repeated a number of times, until the meta-heuristic (usually) converges. If at least one of the decision variables is stochastic, we can only estimate “ $f(x)$ ”, and in particular, $\mathbb{E}[f(x)]$.

Rosen, Harmonosky, and Traband (2007) define the traditional simulation optimisation problem as

$$\text{Minimise} \quad f(\theta) \quad (6.1)$$

$$\text{subject to} \quad \theta \in \Theta, \quad (6.2)$$

where $f(\theta) = E[\psi(\theta, \xi)]$ is the expected system performance value, and is estimated by $\hat{f}(\theta)$ from samples of a simulation model using instances of discrete or continuous feasible and possibly constrained input $\theta \in \Theta \subset \mathbb{R}^D$. The stochastic elements of the model are represented by ξ .

The search algorithm mentioned previously uses the current response value(s) to adjust the control variables (parameters) towards a (hopefully) better solution, and it terminates when the response value(s) do not show significant improvement. The search algorithms contain procedures to prevent the algorithms getting stuck into local optima. Once the search algorithm terminates, the values of the control variables are considered the near-optimum combination of values.

Many simulation packages now include optimisation procedures: Simio is integrated with OptQuest, which uses scatter- and tabu searches as well as neural networks, while the WITNESS Optimizer includes simulated annealing and tabu searches.

The simulation-optimisation process using a metaheuristic is shown in Figure 6.1. A trajectory-based optimisation (*e.g.* simulated annealing) works similarly, but no population is used.

■ **Example 6.1** Consider the 7-11 example to illustrate the size of the decision space (Michalewicz and Fogel, 2004). Here is a story: I went to the 7-11 shop last night and bought four items. I noticed the cashier multiplied the prices of the four items, then she said I owe her R7-11. I objected and insisted that she added the four prices; she did so and again found that I owe her R7-11. What were the prices of the four items?

To answer this, first note that the prices can be expressed in cents, so the problem is discrete and the prices can range from 1¢ to 708¢. Let the prices be $X_i, i = 1, 2, 3, 4$. The following

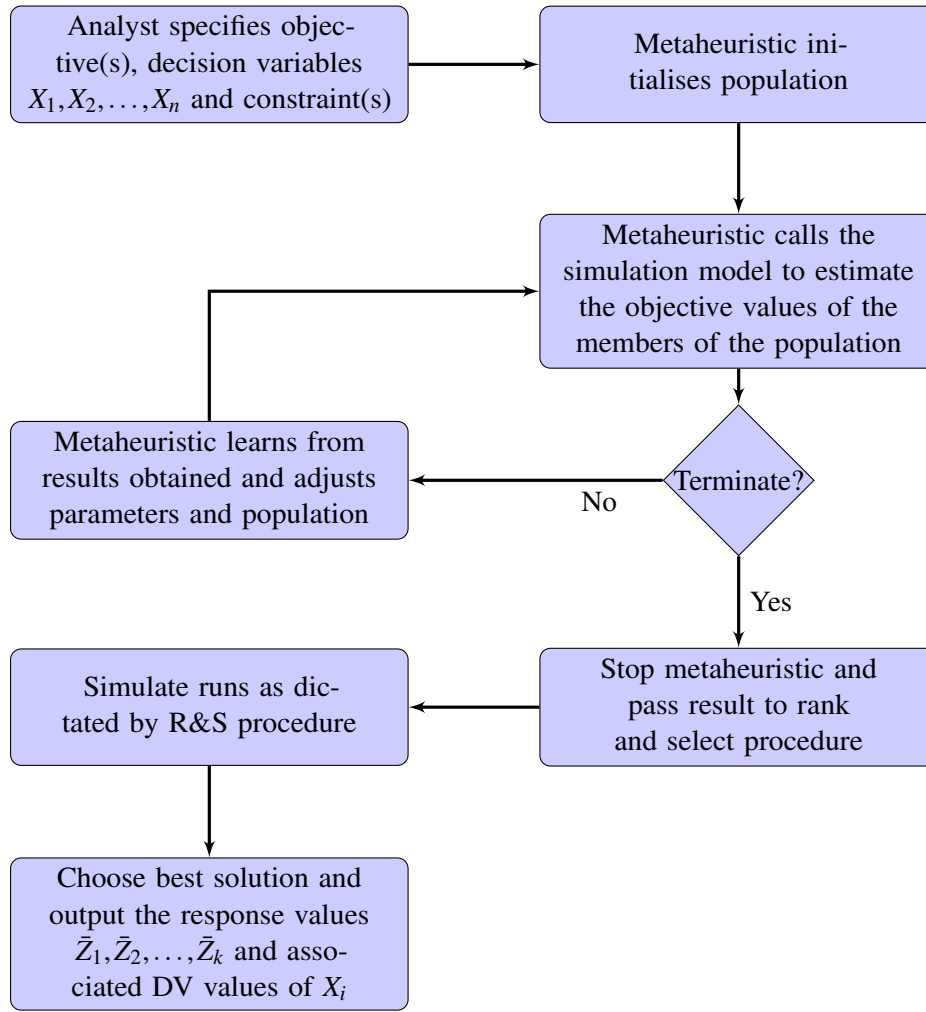


Figure 6.1: Simulation-optimisation process with a population-based metaheuristic

equations must be satisfied:

$$\sum_{i=1}^4 X_i = 711 \text{ and}$$

$$\prod_{i=1}^4 X_i = 711\,000\,000.$$

Since a product must have a price, the minimum value for X_i is 1¢, and therefore the maximum price is 708¢. The size of the decision space of this problem is thus $708 \times 708 \times 708 \times 708 = 708^4 = 251\,265\,597\,696 > 2.5 \times 10^{11}$. This is a large space to search, and if each solution is calculated, it could take a long time to find the answer. We call problems of this nature *combinatorial problems*, because we have to use different combinations of decision variable values to find the optimum. (The answer is $X_1=\text{R}1.20$, $X_2=\text{R}1.25$, $X_3=\text{R}1.50$, $X_4=\text{R}3.16$, you can determine it at home.)

There are many meta-heuristics available, and many of them are nature-inspired. The best-known meta-heuristic is the genetic algorithm (GA, Goldberg (1989)), which will be used in this module. It is a built-in feature of Tecnomatix. Other examples of meta-heuristics are

- Simulated annealing,

- Ant colony optimisation,
- Tabu search, and
- Particle swarm optimisation.

A list of these algorithms is available at https://en.wikipedia.org/wiki/List_of_metaphor-based_metaheuristics.

6.2 Some optimisation problems

A few (nasty) optimisation test problems, for GAs and other meta-heuristics are now presented. These problems are included here to show the reader some of the challenging shapes of optimisation problems that meta-heuristics have to deal with.

6.2.1 Rosenbrock function

The Rosenbrock function is an accepted benchmark for optimisation algorithms. It must be minimised and one variant is defined as

$$f(\mathbf{x}) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2] \quad (6.3)$$

with D variables x_1, \dots, x_D , where $-2 \leq x_i \leq 2$ for all $i = 1, \dots, D$. A plot for $D = 2$ is shown in Figure 6.2. The exact minimum is a vector of ones, that is $\mathbf{x}^* = (1, 1, 1, \dots, 1)$, with $f(\mathbf{x}^*) = 0$, while a local minimum exists at $(-1, 1, 1, \dots, 1)$ for $4 \leq D \leq 7$.

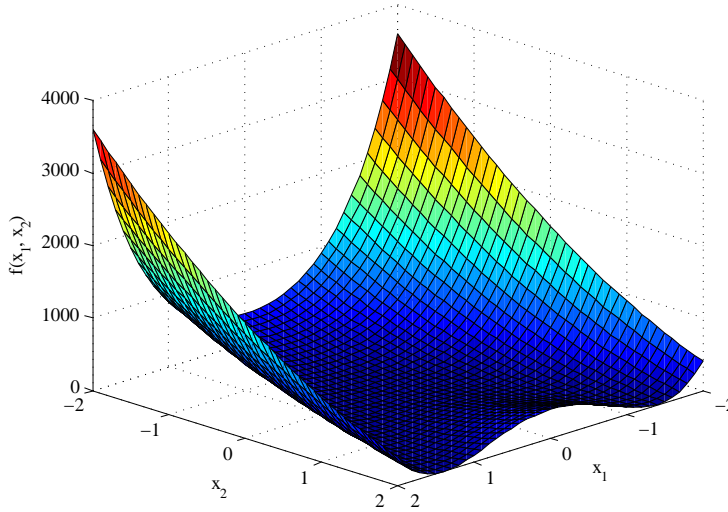


Figure 6.2: The Rosenbrock function with $D = 2$ and $-2 \leq x_i \leq 2$

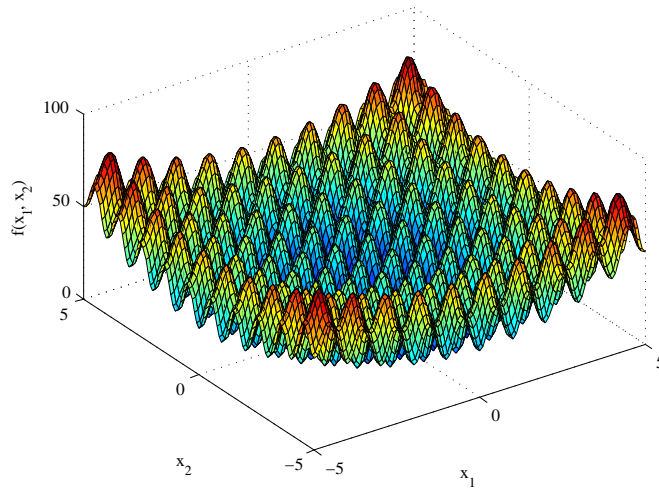
The function has a deceptive optimum at $(0,0)$, while many other coordinates seem to be optimal. They are, however, all local optima.

6.2.2 Rastrigin function

The Rastrigin function has many local optima. It is the function

$$f(\mathbf{x}) = 10D + \sum_{i=1}^D [x_i^2 - 10(\cos(2\pi x_i))], \quad (6.4)$$

which has to be minimised. A plot for the case of $D = 2$ decision variables, where $-5.12 \leq x_i \leq 5.12$ for $i = 1, 2$, is shown in Figure 6.3. The absolute minimum is at $(0,0)$ with $f(0,0) = 0$.

Figure 6.3: Rastrigin function with $D = 2$

6.2.3 Shekel function

The Shekel function

$$f(\mathbf{x}) = - \sum_{i=1}^{m_s} \frac{1}{c_i + \sum_{j=1}^D (x_j - a_{ij})^2} \quad (6.5)$$

is another widely accepted benchmark for evaluating optimisation algorithms. It has to be minimised and depends on D variables x_1, \dots, x_D with $0 \leq x_i \leq 10$, m_s being the number of local minima and can take, by definition, values of 5, 7 or 10.

The values of a_{ij} and c_i are determined via an algorithm for each problem instance. A plot of $-f(\mathbf{x})$ for the case $m_s = 10$ and $D = 2$ is shown in Figure 6.4 (the negative of the function is plotted for clarity). The absolute minimum of $f(\mathbf{x})$ is at $(4, 4)$ with $f(4, 4) = -11.0298$.

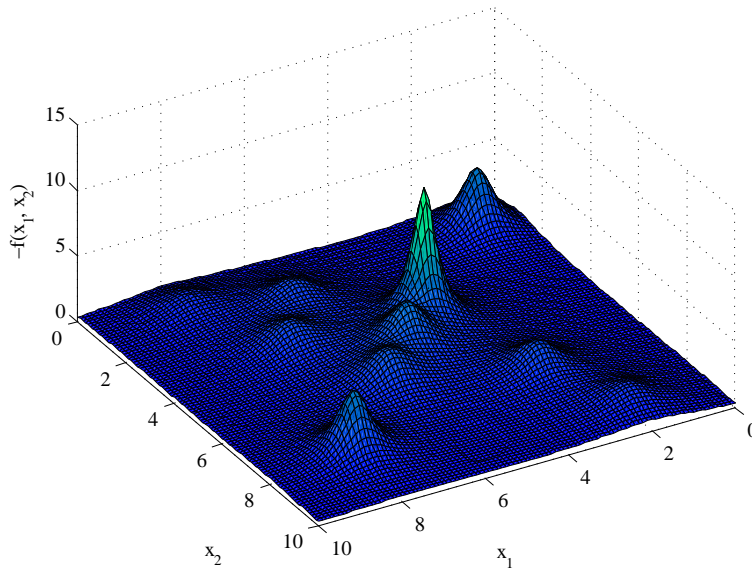


Figure 6.4: Negative Shekel function with 10 peaks

Tecnomatix includes a GA for optimisation, and we must understand the GA principles in

order to do simulation-optimisation. The interested student may consult books by Goldberg (1989) and Mitchell (1997) for more information on GAs.

6.3 The genetic algorithm

Genetic algorithms are based on the biological principle of evolution with the survival of the fittest being a fundamental property. It is an iterative procedure that operates on a constant-size population. The GA was developed by John Holland and others (Mitchell (1997), Goldberg (1989)).

When applied to an optimisation problem the analogy is as follows: A *population* of possible solutions is created from the problem's *solution space* by varying the values of the *decision variables* (search space). Each individual (referred to as a *chromosome*) in the population is an *encoding* of a solution. Some of the individuals combine with each other to form new individuals or *offspring*, similar to parents producing children. Good individuals are more likely to be used during this combination phase (*crossover* phase). The process is also subject to random variation (known as *mutation*) where some part of the chromosome is randomly changed. The new (on average 'better' or 'fitter') individuals replace some or all of the individuals in the old population thus causing the population of solutions to improve over time. A given population represents a *generation*, and once new (improved) individuals are added to the population, a new generation is created. This process of combination and replacement continues until a sufficiently good solution has evolved.

The GA thus effectively searches in the decision space of the problem, and learns from the outcomes in the solution space to adjust the search. The improvement of the population is associated with forming chromosomes that drive the fitness function to a near-optimum (approximate) value. [Note: 'approximate' means we think it is a good guess of the true value of the solution, while 'approximation' means we know (we can prove it) how close to the true value of the solution we are.]

The GA is included under the term *evolutionary algorithms*, together with evolution strategies, evolutionary programming and genetic programming (GP (2018), MathWorks (2018)). There are many variations of GAs, while other types of optimisation algorithms also exist, including population-based incremental learning (PBIL), simulated annealing and neural networks. Each algorithm is more suitable or less suitable for a specific problem. Problems addressed in industrial engineering include complex scheduling problems, layout problems, vehicle routing problems and multi-objective optimisation problems. When exact techniques like linear programming and goal programming can address a problem (they can find the exact solution in finite time), we should use these rather than meta-heuristics.

The pseudo-code for a GA is outlined in Algorithm 1. There are many variants of the code given, but the principles are similar. A description of the components follows.

Problem representation

We recognise that our objective function, *i.e.* the function we want to maximise/minimise is a function of several decision variables that we may change until the objective reaches an accepted level of near-optimality. We may *e.g.* want to schedule a number of tasks on a finite number of machines so that the time to produce a number of products is the minimum.

To utilise a GA, we have to *encode* our decision variables to a form that facilitates the GA processes of *crossover* and *mutation*. The obvious question is how to encode a set of decision variables (which translates back to a solution of the objective function). One way to encode a solution is to represent it as a binary string (*i.e.* a string of 0's and 1's). If, for example, we were searching for an integer value that maximises the function

$$f(x) = x^3 - 60x^2 + 900x + 100 \quad (6.6)$$

Algorithm 1 Basic Genetic Algorithm

```

1: Generate an initial random population and evaluate members. Set  $t = 1$ .
2: Repeat
3:   If the cross-over probability is satisfied
4:     Randomly select two individuals to reproduce
5:     Do cross-over between the members
6:   Else
7:     Choose any of the parents to be the offspring
8:     Replace worst individual in population by new offspring
9:   End If
10:  For Each gene in the offspring
11:    If mutation probability is satisfied
12:      Flip value of gene
13:    End If
14:  End For Each
15:  Evaluate offspring with the objective function
16:  If offspring is better than least fit member in population
17:    Replace the least fit member with the offspring
18:  End If
19:   $t \leftarrow t + 1$ 
20: Until Termination condition

```

in the range between 0 and 31, with x the only decision variable, we could code the value of x as a binary string of length 5. For example, the string “01010” corresponds to the value $x = 10$. If we had a problem involving many variables then we would represent the decision variables as one long binary string. If we wanted to work with fractions then we could allocate extra bits to get the required precision. Note that the search space in this case is linear and discrete, *i.e.* all integers from 0 to 31.

As mentioned before, a specific encoded string of our set of decision variables is called a *chromosome*. In the simple example above, the chromosome consists of an encoding for x only, so that $x = 5$ is represented by the chromosome “00101”, while $x = 19$ is represented by chromosome “10011”. The components of a chromosome are referred to as *genes*. In a binary representation, each 0 or 1 is equivalent to a gene. If we have, say, three decision variables represented by x, y and z , then we encode them as a string:

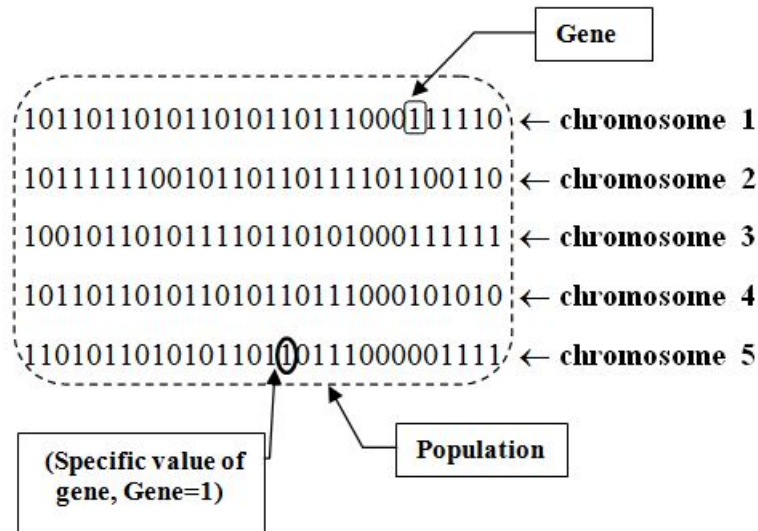
$$\underbrace{1001011010}_{x(=602)} \quad \underbrace{0111011011}_{y(=475)} \quad \underbrace{1000111110}_{z(=574)} \quad \leftarrow \text{chromosome}$$

The vertical lines in between are added to distinguish between each variable’s representation; the representation is thus one single string consisting of many ‘0’s and ‘1’s. Note that it is implied that each variable is represented by 10 bits, *i.e.* it consists of 10 genes, the latter assuming values of either ‘0’ or ‘1’.

Initial population and population size

The GA is initialised by forming a *population* of initial chromosomes, either via a random or a heuristic process. The population is thus made up of a collection of chromosomes. The *size of the population* is another parameter of the GA. In general, a larger population size will increase the likelihood of finding very good solutions. However, it will also increase the computation time. Typically, population sizes between 30 and 100 chromosomes are used.

Initially, the chromosomes need to be filled with data before the GA can operate on them. This can be done using random values for each bit in a gene by assigning it a value of '0' or '1' with a probability of 0.5 in each case. A population of five chromosomes consisting of three decision variables, each 10 bits (genes) long, may thus look as follows:



A specific instance of the population is referred to as a *generation*.

Evaluating the population

Every GA needs a so-called *fitness function* (objective function) to evaluate the fitness of the chromosomes in the population. In the example in (6.6), $f(x)$ is the fitness value, because it is this function that we want to maximise; the value of this function for each chromosome also drives the improvement of the GA. We thus need to decode the chromosomes into values for the decision variables again so that the fitness function can be evaluated with these values. Decoding thus comprises the translation of the binary substrings into representations understandable by man, for example real numbers.

Now suppose that we are using a genetic algorithm to try to determine the best schedule of tasks for a group of machines in a factory. In this case we might want to minimise the time it takes to complete the tasks and the fitness function can be the completion time of the last task.

Crossover

The crossover phase combines the genes of two chromosomes. There are several methods of performing crossover. The common method of *one-point crossover* works as follows (a bit different from what is shown in Algorithm 1). Two chromosomes (parents) are randomly chosen from the population, then, of these two, the one with the best objective function value is selected. Another chromosome (parent) is again randomly selected and it replaces the second chromosome. A random position, the crossover point, is then chosen between two genes. Crossover forms two offspring, the first of which is formed by combining the first parent's genes that are before the crossover point, with the second chromosome's genes that follow after the crossover point. The second chromosome is formed in the reverse manner. The next schematic depicts one-point crossover and uses grey colours to see how the combinations are formed.

1	0	1	1	0	1	0	1	1	0
1	0	1	1	0	1	0	1	1	0

Crossover is applied to pairs of chromosomes as determined by the selection mechanism of the GA (explained later). A *crossover probability* is applied to these pairs of chromosomes to determine if they are to be subjected to crossover. This crossover probability is a parameter that is set by the decision-maker. Typical crossover probabilities are in the region $0.7 - 1$. Crossover is required if we choose a random number U so that $U \leq P(\text{Crossover})$.

■ **Example 6.2** Suppose $U = 0.451$, and $P(\text{Crossover}) = 0.8$, then we would do crossover, since $U \leq P(\text{Crossover})$.

Mutation

Mutation is an important part of a genetic algorithm because it introduces new genetic material into the population and prevents the algorithm from getting trapped in a local optimum. The method of flip mutation as applied to a binary coded chromosome changes the value of the contents of a gene to a '0' if it contained a '1', or it changes the gene content to a '1' if it contained a '0'.

Mutation is applied to a chromosome by considering each gene in turn and mutating the gene according to a certain probability (in other words each gene has this probability of being mutated). This mutation probability is a parameter of the algorithm. Typical mutation rates lie in the region $0.001 - 0.03$. Mutation is applied to all offspring that are generated, but since the probability is low, only a few bits are flipped at a time. Mutation is done if we choose a random number U so that $U \leq P(\text{Mutation})$.

■ **Example 6.3** Suppose $U = 0.311$, and $P(\text{Mutation}) = 0.01$, then we would not do mutation on the specific gene (bit), since $U \geq P(\text{Mutation})$. An example of a mutated bit is shown below:

1	0	1	0	0	1	0	1	1	0	
					↓					
1	0	1	0	0	0	0	1	1	0	

Selection and replacement

The selection method used governs which chromosomes will be selected for crossover. There are several selection methods, for example roulette wheel, Boltzman, tournament and rank selection. The method of tournament selection works in the following way: Two (or more) different chromosomes are selected at random from the population. Of these selected chromosomes, the chromosome with the best fitness function is selected to be the first parent for crossover. The second parent is selected randomly from the population.

The replacement method determines how the offspring are to be inserted into the population. Some types of genetic algorithms replace the entire population with new offspring while others keep some chromosomes from the old population. One method of replacement is to replace only one of the members of the population. GAs that employ this method of replacement are known as *incremental GAs*. Only one offspring is generated in each generation and it replaces a chosen chromosome in the population. This chromosome is chosen in many ways, including

- Replace the chromosome with the worst fitness value in the population.
- Randomly replace a chromosome, which implies that a “good” chromosome can be replaced. This improves diversity of the population but results in possible longer times to convergence.

It does not matter which of the two offspring generated from crossover are used for this purpose because crossover chooses the parents in a random manner in any case.

Stopping conditions

The stopping conditions determine when the algorithm terminates. The algorithm could terminate after a certain number of iterations have been completed, which is called a hard stopping

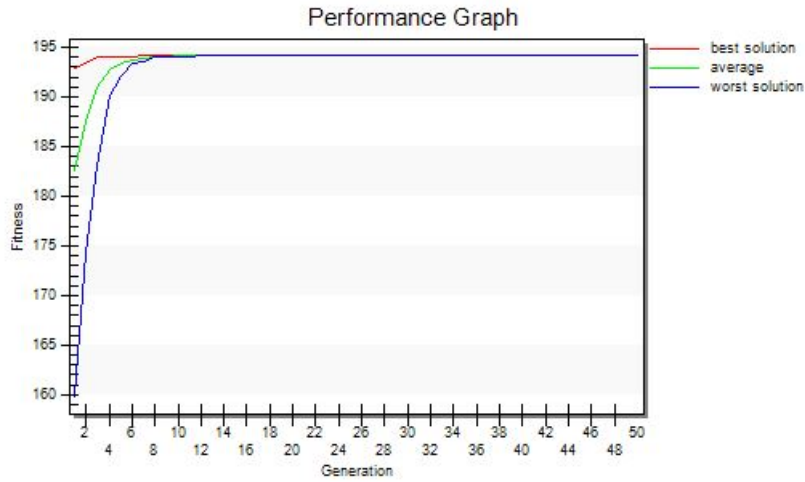


Figure 6.5: Generation progress of a GA as developed by Tecnomatix

							Fitness
1	0	1	0	1	0	1	11.2
1	0	1	0	1	0	1	12.4
0	1	0	1	0	1	1	14.9
1	0	1	0	1	0	1	10.9
0	0	0	0	1	1	0	19.7
1	1	1	1	0	0	1	12.3
0	1	1	0	0	1	1	21.4
1	1	0	1	0	0	1	18.4
0	0	0	0	0	1	0	14.3
1	0	1	1	1	1	1	13.2
Average for this generation:							14.87

condition: the algorithm thus stops after T iterations have been executed. The algorithm may also terminate due to some other condition such as when the rate of improvement slows down. One method is to determine the average value of the fitness function for each generation. If the current iteration yields an average of the fitness function that is within $\delta\%$ of the average of the last n averages, then we terminate the algorithm. We should also add a second stopping criterion T , as above. That prevents the algorithm from running indefinitely.

The graph in Figure 6.5 shows a plot of the average of the fitness function per generation (this example was taken from Tecnomatix). After some generations we see that the average stabilises.

Assuming that a local optimum is not reached, we can determine

$$\left| \frac{\overline{M}_c - \overline{\overline{M}}_n}{\overline{M}_c} \right| \leq \delta$$

where

\overline{M}_c = Average of the objective function values in current generation.

$\overline{\overline{M}}_n$ = Average of the previous n averages of the objective function.

Suppose we have the following population:

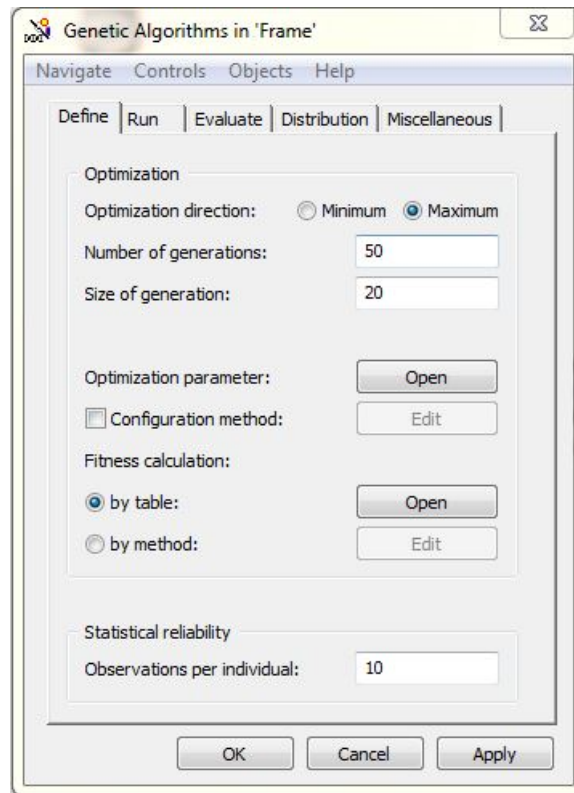


Figure 6.6: Tecnomatix GA Wizard settings

The value of 14.87 is the average for this generation, *i.e.* it is \bar{M}_c . We should thus record each average before replacing the current generation with the next. Also, if we decide to consider the averages of the previous n generations, we can only do so after we have formed at least $n + 1$ generations.

We can now use the GA in conjunction with simulation to solve large problems and find us near-optimal solutions.

6.4 Application of GA principles in Tecnomatix

Several simulation packages use meta-heuristics for optimisation, and Tecnomatix uses the genetic algorithm. We shall now see how the concepts mentioned are implemented in Tecnomatix. We now learn how the wizard is used. The main fields for the GA settings are shown in Figure 6.6.

The decision-maker should know whether the optimisation is minimisation or maximisation – we typically minimise machine down-times or maximise throughput. The fields of the GA Wizard are subsequently related to some of the GA principles.

1. The **Number of generations** is the maximum number of times you will allow the GA to create a new, improved generation. A small number specified here will often prevent the GA enough opportunity to ‘learn’ from previous generations. A large number will be advantageous, but it could take a very long time before the simulation-optimisation terminates.
2. The **Size of the generation** specifies how many individuals you allow in your *population* (Tecnomatix means ‘population’ here). A population with only a few individuals will not exhibit sufficient diversity to allow the GA to find good answers. A large population is

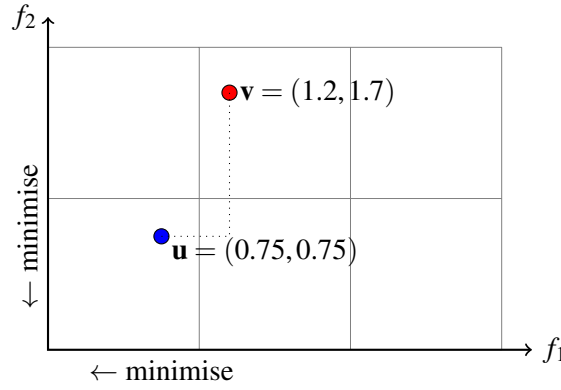


Figure 6.7: Vector dominance illustrated

good, but remember that each member must be evaluated using n^* replications. This can thus take long.

3. The **observations per individual** specifies your choice for n^* . When doing SO, one can usually not afford to enforce very small confidence intervals because of the resulting computational burden. This is a trade-off that the simulation analyst must make – if you have good computing power, you may increase the statistical reliability and the values for the GA optimisation parameters in 1. and 2. My experience has shown that it is better to assign more computing power to the number of generations and the size of the generation than to require ultimate statistical accuracy. (Why am I right?)
4. The **optimis[z]ation parameter** in Figure 6.6 refers to the set of decision variables (the X_i in Figure 6.1).
5. The **fitness calculation** refers to the set of response parameters (output values) (the \bar{Z}_i in Figure 6.1). (Note that we must strictly-speaking refer to \bar{Z}_i , because these are the parameters that we usually estimate.)

This concludes the brief introduction to simulation-optimisation. You should know how it works in principle, and also how to use the settings in the Tecnomatix GAWizard. Next, multi-objective optimisation with simulation is presented.

6.5 Multi-objective optimisation in large decision spaces with simulation

In multi-objective optimisation (MOO), two or more *conflicting* objectives are *simultaneously* optimised. The objectives are often *non-commensurate*, meaning they are measured in different units. Think, for example, of an investment: we want to *maximise* profit, but we also want to *minimise* the risk of losing money. Another example is: you want to buy as much airtime as possible but want to minimise the cost. MOO problems that can be solved do not yield a single best solution, but a set of optimal solutions called the *Pareto set*. We shall only deal with two objectives in this module, called *bi-objective optimisation*. For simplicity, we shall use ‘multi-objective optimisation’ throughout the text.

When doing MOO, we need to understand *vector dominance*. This is necessary to separate good and no-good solutions. Consider Figure 6.7 and the following definitions.

Definition 7. Assuming minimisation, given two vectors $\mathbf{u} = (u_1, \dots, u_K)$ and $\mathbf{v} = (v_1, \dots, v_K) \in \mathbb{R}^K$, then $\mathbf{u} \leq \mathbf{v}$ if $u_i \leq v_i$ for $i = 1, 2, \dots, K$, and $\mathbf{u} < \mathbf{v}$ if $\mathbf{u} \leq \mathbf{v}$ and $\mathbf{u} \neq \mathbf{v}$.

Definition 8. Given two vectors \mathbf{u} and \mathbf{v} in \mathbb{R}^K , then \mathbf{u} dominates \mathbf{v} (denoted by $\mathbf{u} \prec \mathbf{v}$) if $\mathbf{u} < \mathbf{v}$.

Definition 9. A vector of decision variables $\mathbf{x}^* \in \Omega$ (Ω is the feasible region) is Pareto optimal if there does not exist another $\mathbf{x} \in \Omega$ such that $\mathbf{f}(\mathbf{x}) \prec \mathbf{f}(\mathbf{x}^*)$.

The concept of domination is illustrated with the simple Pareto front plot in Figure 6.8 (both objectives are to be minimised). Note that we now focus on the *solution space*.

Assuming the blue and red dots represent all possible solutions to the problem shown in Figure 6.8, then the blue dots represent members of the Pareto set, and these are not dominated by the red dots. At 'A', $f_1(x) = 2, f_2(x) = 2$, but at 'B', $f_1(x) = 1, f_2(x) = 1$, so the point at 'B' is far 'better' than the point at 'A'. Although $f_1(x)$ at 'A' is less than $f_1(x)$ at 'C', many other points have $f_1(x)$ less than that of 'A', so 'A' is dominated.

► In MOO, the aim is to find the blue dots • and their associated green dots • in Figure 6.9; the problem in this case has two decision variables x_1 and x_2 , and two objectives, f_1 and f_2 , both to be minimised. The blue dots represent non-dominated solutions and form the *Pareto set*. For a given solution (blue dot) there is an associated set of decision variable values (the green dots). These are the values at which a system must be operated to ensure near-optimality.

We can now state the MOO problem formulation.

6.6 Multi-objective optimisation problem formulation

The MOO problem with K objectives and $M + Q$ constraints are formulated as follows (Tsou, 2008):

$$\text{Minimise } \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_K(\mathbf{x})]^T \quad (6.7)$$

$$\text{subject to } \mathbf{x} \in \Omega \quad (6.8)$$

$$\Omega = \{\mathbf{x} \mid g_i(\mathbf{x}) \leq 0, i = 1, 2, \dots, M; \quad (6.9)$$

$$h_j(\mathbf{x}) = 0, j = 1, \dots, Q\}. \quad (6.10)$$

The symbol \mathbf{f} implies a vector, *i.e.* two or more objective functions, while the symbol \mathbf{x} implies a vector of decision variables. In (6.8), the feasible region is defined. The constraints are defined in (6.9), which are the inequality constraints, and the equality constraints are defined in (6.10).

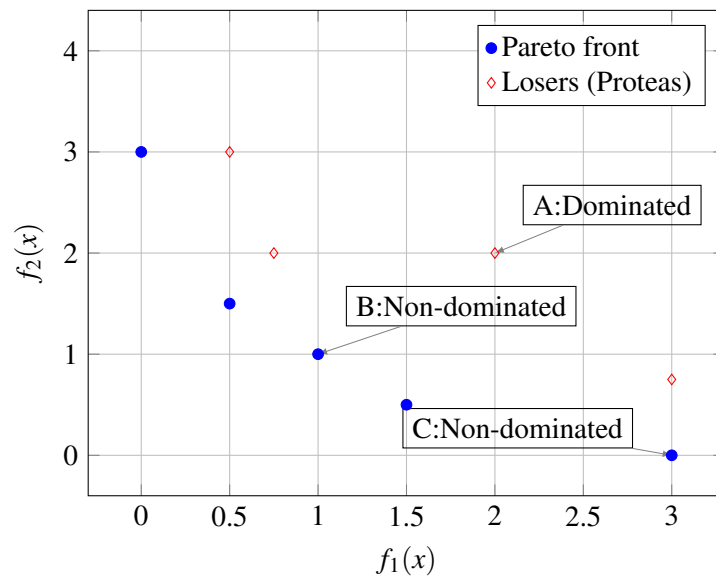


Figure 6.8: Concept of domination in MOO (both objectives to be minimised)

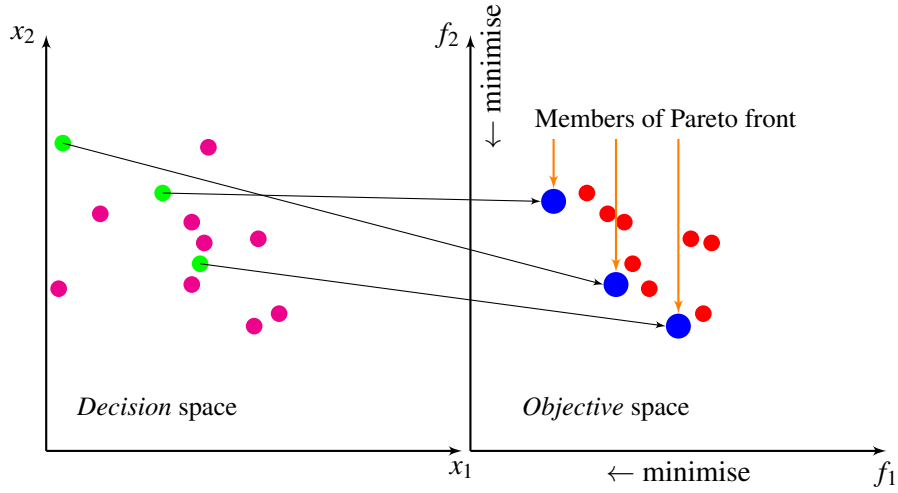


Figure 6.9: The decision space and objective space

MOO deterministic example

■ **Example 6.4** Consider the following simple deterministic, continuous example with one decision variable and two objective functions:

Minimise

$$f_1(x) = x^2 \quad (6.11)$$

$$f_2(x) = (x-2)^2 \quad (6.12)$$

with $-10^5 \leq x \leq 10^5$. The decision space is one-dimensional but large. We can find by inspection that $0 \leq x \leq 2$, while $0 \leq f_1(x) \leq 4$ and $0 \leq f_2(x) \leq 4$. We see that the objectives have many values. Indeed, we can plot a subset of them on a Cartesian coordinate system, as shown in Figure 6.10. The red curve shows the Pareto set, and it can be seen that the solutions on the blue part of the curve are dominated by those on the red curve.

Also note that, if we decrease $f_1(x)$, then $f_2(x)$ increases, illustrating the conflicting nature of the two functions.

MOO stochastic example

■ **Example 6.5** In simulation-optimisation, $\mathbf{f}(\mathbf{x})$ is estimated with a simulation model. We can reformulate the multi-objective simulation optimisation (MOSO) problem as (Moonyoung Yoon, 2017):

$$\begin{aligned} \min \quad & \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_p(\mathbf{x})]^T \\ & = [E(Y_1(\mathbf{x}, \xi)), \dots, E(Y_p(\mathbf{x}, \xi))]^T \\ \text{s.t.} \quad & \mathbf{x} \in S_F, \end{aligned} \quad (6.13)$$

where $Y_i(\mathbf{x}, \xi)$ represents the random variable of the simulation output for the i^{th} objective ($i = 1, \dots, p$). The symbol ξ indicates the stochastic elements of the optimisation problem and S_F the feasible region.

Multi-objective simulation-optimisation (MOSO) is very complex and challenging due to the stochastic output. The principle is illustrated in Figure 6.11. For each combination of (x_1, x_2) , the simulation model must execute a number of replications to determine a ‘good’ value for the point estimator, and this often takes time. Each combination also has its unique distribution, although

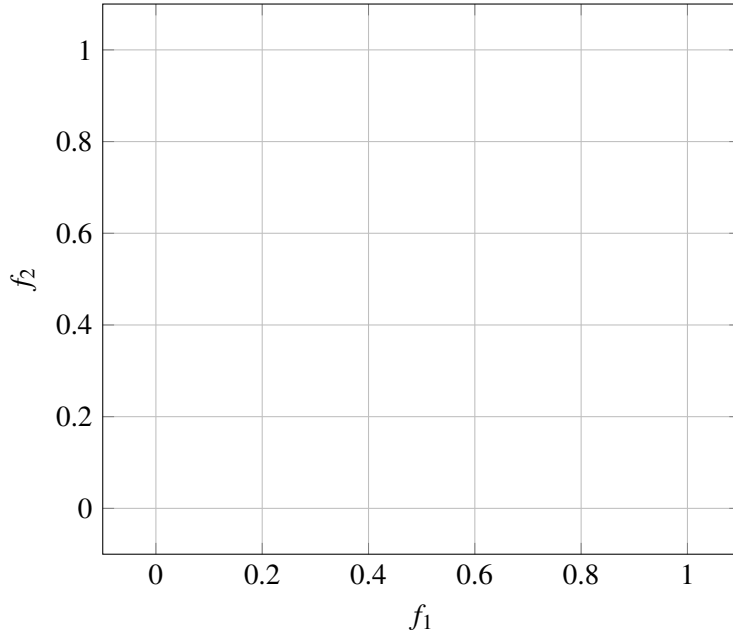


Figure 6.10: Solutions to (6.11) and (6.12) including the Pareto front

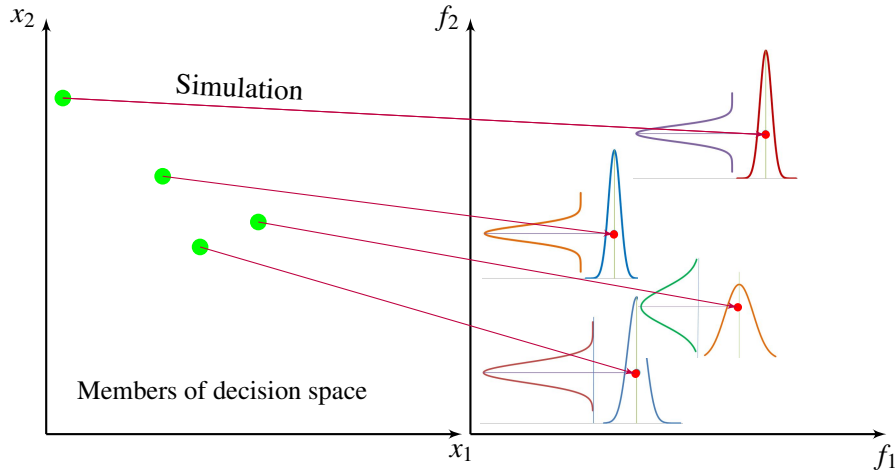


Figure 6.11: The computational burden of simulation-optimisation illustrated

one can assume that estimations of expected values are approximately normally distributed. Finding the approximate Pareto set while doing the smallest number of simulation replications is currently being actively researched (Yoon and Bekker, 2020; Yoon and Bekker, 2022).

In the classic buffer-allocation problem (BAP), one wants to *maximise throughput* and *minimise work-in-progress*, so a typical (approximate) Pareto front is shown in Figure 6.12, for 10 machines and 25 buffers. In this problem then, $f_1(x)$ represents the throughput and $f_2(x)$ the work-in-progress, while $S = \{1, 2, 3, \dots, 25\}$. The stochastic elements ξ in this example include the processing times on the machines, the number of cycles until failure per machine, and the times to repair the machines.

The blue ‘cloud’ of circles shows all solutions developed in search of the Pareto set (via simulation), while the red dots show the good solutions. Note the shape of the Pareto front – we *maximise* throughput. (How would a Pareto front look like if we maximised both objectives?)

For further practical applications of MOSO see Scholtz, Bekker, and Toit (2012), Stadler and Bekker (2014), Snyman (2019) and Yoon and Bekker (2022).

6.7 Ranking of solutions

When doing MOSO, many solutions may emerge, especially if meta-heuristics are used to search the decision space. To find the solutions of the approximate Pareto set, some form of *ranking* is needed. This ‘ranking’ should separate non-dominated solutions from those that are dominated. A version of the Pareto ranking method proposed by Fonseca and Fleming (1998) will be discussed, although it only applies to *deterministic* results. Suppose one wants to rank the data values of columns $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ in Table 6.1(a), which were created from (6.11) and (6.12).

The method is applied by *sorting* the data set, with the result shown in Table 6.1(b), where a rank column has been added. The data was sorted in *descending order* with the column of $f_1(\mathbf{x})$ as sort key, so that the *largest* values are at the top. The sort order of $f_1(\mathbf{x})$ is

- **descending** if $f_1(\mathbf{x})$ is to be **minimised**, or
- **ascending** if $f_1(\mathbf{x})$ is to be **maximised**.

The values of column $f_2(x)$ are now considered: if the element in the first row is larger than the element in the second row, the rank of row 1 is incremented by one. This is repeated until the last row of column $f_2(x)$ is reached. The process is repeated, but one now starts at row 2, and does the comparison from row 3 downwards. The rank value of the last row is defaulted to 0. Once all rows (except the last row) have been used as reference, all the rows having a rank of zero are taken out (Table 6.1(c)) and the $f_i(x)$ form the Pareto set. A rank value of 0 indicates that no other solution set $(f_1(x), f_2(x))$ dominates any solution set in the group (the Pareto set). The minimum and maximum values for x can also be retrieved from this set. In this example, x was not indicated as a vector although it strictly is one. Note that the process to extract the x -values is a bit more complicated when the Pareto set is discontinuous.

For a Pareto set obtained with stochastic methods, one must ensure that the solutions in the Pareto set are statistically significantly different, similar to the SOSO case (Subsection

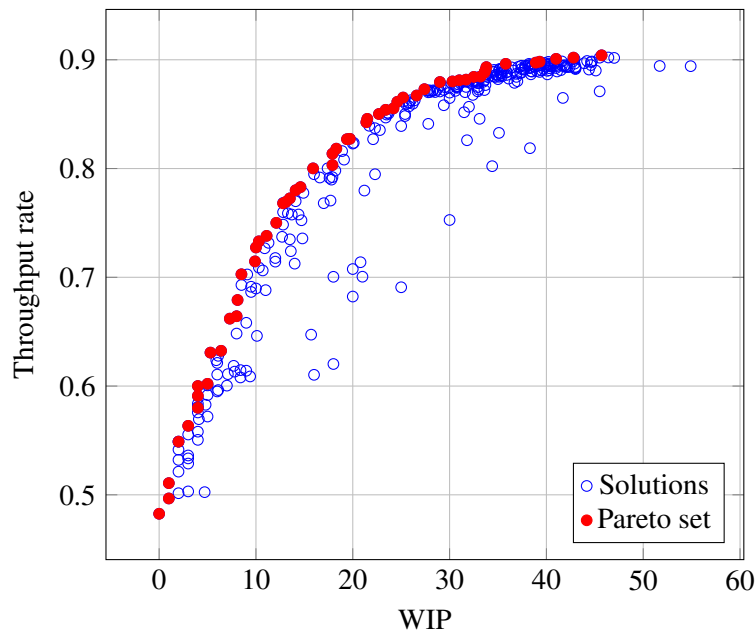


Figure 6.12: Approximate Pareto set with dominated vectors for the BAP

Table 6.1: Example values for Pareto ranking

Original data set			Ranked data set				Pareto set			
x	$f_1(x)$	$f_2(x)$	x	$f_1(x)$	$f_2(x)$	Rank	x	$f_1(x)$	$f_2(x)$	Rank
-2.0	4.00	16.00	3.0	9.00	1.00	10	2.0	4.00	0.00	0
-1.8	3.24	14.44	2.8	7.84	0.64	7	1.8	3.24	0.04	0
-1.6	2.56	12.96	2.6	6.76	0.36	6	1.6	2.56	0.16	0
-1.4	1.96	11.56	2.4	5.76	0.16	5	1.4	1.96	0.36	0
-1.2	1.44	10.24	2.2	4.84	0.04	2	1.2	1.44	0.64	0
-1.0	1.00	9.00	-2.0	4.00	16.00	20	1.0	1.00	1.00	0
-0.8	0.64	7.84	2.0	4.00	0.00	0	0.8	0.64	1.44	0
-0.6	0.36	6.76	-1.8	3.24	14.44	18	0.6	0.36	1.96	0
-0.4	0.16	5.76	1.8	3.24	0.04	0	0.4	0.16	2.56	0
-0.2	0.04	4.84	-1.6	2.56	12.96	16	0.2	0.04	3.24	0
0.0	0.00	4.00	1.6	2.56	0.16	0	0.0	0.00	4.00	0
0.2	0.04	3.24	-1.4	1.96	11.56	14				
0.4	0.16	2.56	1.4	1.96	0.36	0				
0.6	0.36	1.96	-1.2	1.44	10.24	12				
0.8	0.64	1.44	1.2	1.44	0.64	0				
1.0	1.00	1.00	-1.0	1.00	9.00	10				
1.2	1.44	0.64	1.0	1.00	1.00	0				
1.4	1.96	0.36	-0.8	0.64	7.84	8				
1.6	2.56	0.16	0.8	0.64	1.44	0				
1.8	3.24	0.04	-0.6	0.36	6.76	6				
2.0	4.00	0.00	0.6	0.36	1.96	0				
2.2	4.84	0.04	-0.4	0.16	5.76	4				
2.4	5.76	0.16	0.4	0.16	2.56	0				
2.6	6.76	0.36	-0.2	0.04	4.84	2				
2.8	7.84	0.64	0.2	0.04	3.24	0				
3.0	9	1.00	0.0	0.00	4.00	0				

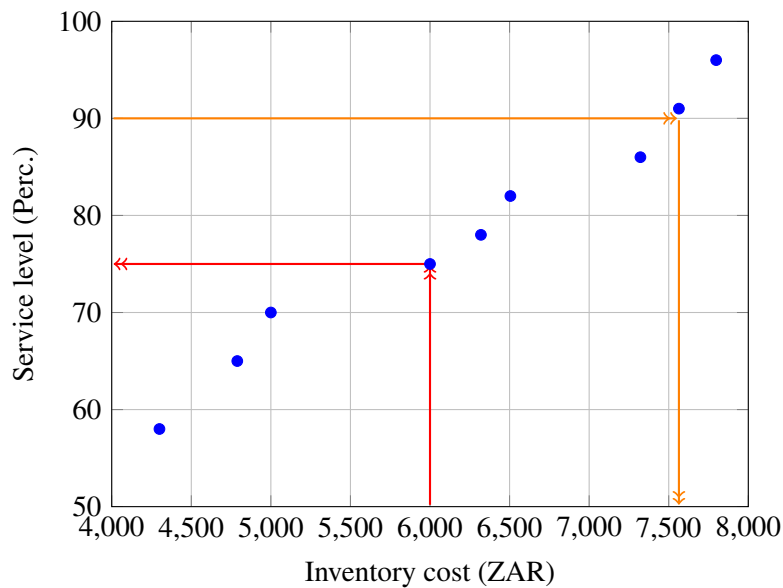
(a)
(b)
(c)

5.5.3). The work by Pierro, Khu, and Savić (2007) and Lee et al. (2010) explains and applies multi-objective ranking and selection (MORS) and their methods to ensure statistically different solutions in the Pareto set. The MORS field is actively researched, but falls outside of the scope of this module.

6.8 Selecting a solution for implementation

Assume that the Pareto set has been formed with an R&S procedure, and one of these solutions must now be selected for implementation. You must now advise Management. The solutions in the Pareto set are all ‘good’, and the decision-maker may select any of these. However, to determine which one to select for implementation, some preferences need to be defined, *e.g.* from a business perspective. There are methods like simple additive-weighting (SAW) and TOPSIS available, but these are outside of the scope of this module. The interested reader is referred to Hwang, Lai, and Liu (1993). One can also use business requirements to select a single solution from the approximate Pareto set. Consider the (r, Q) inventory problem, in which we want to minimise inventory cost, but maximise service level. A typical Pareto front is shown in Figure

6.13.

Figure 6.13: Example Pareto front for the (r, Q) inventory problem

Suppose Management decided that the maximum inventory cost they can afford per period is R6 000. The analyst reads the graph by starting on the horizontal axis at R6 000, then goes up to the dots along the red line. When a dot or nearby dot is found, the analyst goes horizontally to the left, still along the red line, and reads off the service level (75% in this case). The analyst can then tell Management that the maximum expected service level will be 75% if R6 000 is invested per period. Management will have to accept that 25% of the customers will not be served or increase the allowed cost.

On the other hand, suppose Management insists on a 90% service level, then the analyst can follow the orange line, starting at 90 on the vertical axis, then projecting a horizontal line to the right towards the (nearest) 'dot' on the Pareto front. From that point, by going down, the analyst determines that the cost is approximately R7 560. Management must now decide if this cost justifies the desired service level. The lesson for the industrial engineer here is that in MOO and MOSO, the best solutions are often found via methods other than mathematical methods.

6.9 Some MOO algorithms

Research is actively done to develop algorithms that can find the Pareto set with as few as possible simulation evaluations (Bekker and Aldrich, 2011). Examples include (Coello Coello, Lamont, and Veldhuizen, 2007):


1. Multi-objective genetic algorithm (MOGA, Fonseca & Fleming).
2. Multi-objective optimisation with the cross-entropy method (MOO CEM) ((Bekker and Aldrich, 2011))
3. Niche-Pareto Genetic Algorithm (NPGA, Erickson *et al.*).
4. Strength Pareto Evolutionary Algorithm (SPEA, Zitzler & Thiele).
5. Pareto Archived Evolution Strategy (PAES, Knowles & Corne).
6. Non-dominated sorting genetic algorithm (NSGA-II, Deb *et al.*).

Many other algorithms are available in literature, and commercial packages like OptQuest are used with several simulation software packages (*e.g.* Simio) for multi-objective simulation-optimisation.

Survival kit:

Optimise your SO skills and ensure you have the following in your assessment pack:

- 1** Understand the combinatorial nature of optimisation problems.
- 2** Understand the principles of simulation-optimisation (SO) and define it.
- 3** Understand meta-heuristics, and that they provide good, but not necessarily optimal solution(s), in relatively short time. It is important to understand Figure 6.1.
- 4** You must know the basic working of the GA, its parameters (population size, number of generations, termination conditions, mutation and selection and cross-over).
- 5** You must define the MOO problem, and understand the concept of dominance and the Pareto set.
- 6** Understand and define MOSO.
- 7** Always keep in mind that in SO, we can only estimate an approximate Pareto set, and that a *computational burden* is associated with SO. The concepts illustrated in Figure 6.11 are very important to understand.
- 8** You must be able to do Pareto ranking of solutions (although it is not strictly correct in the case of MOSO).

 Notice on the door to the School of the Gifted: *Pull*. I pushed and I pushed ...



v	1- α									
	0.050	0.100	0.150	0.200	0.250	0.300	0.350	0.400	0.450	0.500
1	0.004	0.016	0.036	0.064	0.102	0.148	0.206	0.275	0.357	0.455
2	0.103	0.211	0.325	0.446	0.575	0.713	0.862	1.022	1.196	1.386
3	0.352	0.584	0.798	1.005	1.213	1.424	1.642	1.869	2.109	2.366
4	0.711	1.064	1.366	1.649	1.923	2.195	2.470	2.753	3.047	3.357
5	1.145	1.610	1.994	2.343	2.675	3.000	3.325	3.655	3.996	4.351
6	1.635	2.204	2.661	3.070	3.455	3.828	4.197	4.570	4.952	5.348
7	2.167	2.833	3.338	3.822	4.255	4.671	5.082	5.493	5.913	6.346
8	2.733	3.490	4.078	4.594	5.071	5.527	5.975	6.423	6.877	7.344
9	3.325	4.168	4.817	5.380	5.899	6.393	6.876	7.357	7.843	8.343
10	3.940	4.865	5.570	6.179	6.737	7.267	7.783	8.295	8.812	9.342
11	4.575	5.578	6.336	6.989	7.584	8.148	8.695	9.237	9.783	10.341
12	5.226	6.304	7.114	7.807	8.438	9.034	9.612	10.182	10.755	11.340
13	5.892	7.042	7.901	8.634	9.299	9.926	10.532	11.129	11.729	12.340
14	6.571	7.790	8.696	9.467	10.165	10.821	11.455	12.078	12.703	13.339
15	7.261	8.547	9.499	10.307	11.037	11.721	12.381	13.030	13.679	14.339
16	7.962	9.312	10.309	11.152	11.912	12.624	13.310	13.983	14.655	15.338

Appendix A – Tables: statistical values

Table 6.2: χ^2 values

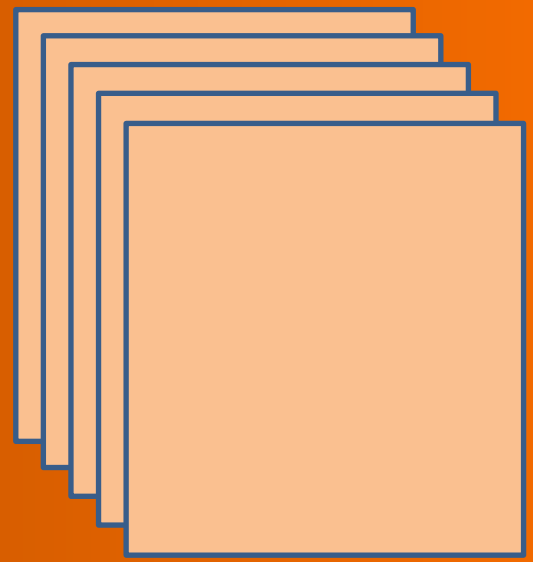
v	1- α									
	0.050	0.100	0.150	0.200	0.250	0.300	0.350	0.400	0.450	0.500
1	0.004	0.016	0.036	0.064	0.102	0.148	0.206	0.275	0.357	0.455
2	0.103	0.211	0.325	0.446	0.575	0.713	0.862	1.022	1.196	1.386
3	0.352	0.584	0.798	1.005	1.213	1.424	1.642	1.869	2.109	2.366
4	0.711	1.064	1.366	1.649	1.923	2.195	2.470	2.753	3.047	3.357
5	1.145	1.610	1.994	2.343	2.675	3.000	3.325	3.655	3.996	4.351
6	1.635	2.204	2.661	3.070	3.455	3.828	4.197	4.570	4.952	5.348
7	2.167	2.833	3.358	3.822	4.255	4.671	5.082	5.493	5.913	6.346
8	2.733	3.490	4.078	4.594	5.071	5.527	5.975	6.423	6.877	7.344
9	3.325	4.168	4.817	5.380	5.899	6.393	6.876	7.357	7.843	8.343
10	3.940	4.865	5.570	6.179	6.737	7.267	7.783	8.295	8.812	9.342
11	4.575	5.578	6.336	6.989	7.584	8.148	8.695	9.237	9.783	10.341
12	5.226	6.304	7.114	7.807	8.438	9.034	9.612	10.182	10.755	11.340
13	5.892	7.042	7.901	8.634	9.299	9.926	10.532	11.129	11.729	12.340
14	6.571	7.790	8.696	9.467	10.165	10.821	11.455	12.078	12.703	13.339
15	7.261	8.547	9.499	10.307	11.037	11.721	12.381	13.030	13.679	14.339
16	7.962	9.312	10.309	11.152	11.912	12.624	13.310	13.983	14.655	15.338
17	8.672	10.085	11.125	12.002	12.792	13.531	14.241	14.937	15.633	16.338
18	9.390	10.865	11.946	12.857	13.675	14.440	15.174	15.893	16.611	17.338
19	10.117	11.651	12.773	13.716	14.562	15.352	16.109	16.850	17.589	18.338
20	10.851	12.443	13.604	14.578	15.452	16.266	17.046	17.809	18.569	19.337
21	11.591	13.240	14.439	15.445	16.344	17.182	17.984	18.768	19.548	20.337
22	12.338	14.041	15.279	16.314	17.240	18.101	18.924	19.729	20.529	21.337
23	13.091	14.848	16.122	17.187	18.137	19.021	19.866	20.690	21.510	22.337
24	13.848	15.659	16.969	18.062	19.037	19.943	20.808	21.652	22.491	23.337
25	14.611	16.473	17.818	18.940	19.939	20.867	21.752	22.616	23.472	24.337
26	15.379	17.292	18.671	19.820	20.843	21.792	22.697	23.579	24.454	25.336
27	16.151	18.114	19.527	20.703	21.749	22.719	23.644	24.544	25.437	26.336
28	16.928	18.939	20.386	21.588	22.657	23.647	24.591	25.509	26.419	27.336
29	17.708	19.768	21.247	22.475	23.567	24.577	25.539	26.475	27.402	28.336
30	18.493	20.599	22.110	23.364	24.478	25.508	26.488	27.442	28.386	29.336
31	19.281	21.434	22.976	24.255	25.390	26.440	27.438	28.409	29.369	30.336
32	20.072	22.271	23.844	25.148	26.304	27.373	28.389	29.376	30.353	31.336
33	20.867	23.110	24.714	26.042	27.219	28.307	29.340	30.344	31.337	32.336
34	21.664	23.952	25.586	26.938	28.136	29.242	30.293	31.313	32.322	33.336
35	22.465	24.797	26.460	27.836	29.054	30.178	31.246	32.282	33.306	34.336
36	23.269	25.643	27.336	28.735	29.973	31.115	32.200	33.252	34.291	35.336
37	24.075	26.492	28.214	29.635	30.893	32.053	33.154	34.222	35.276	36.336
38	24.884	27.343	29.093	30.537	31.815	32.992	34.109	35.192	36.262	37.335
39	25.695	28.196	29.974	31.441	32.737	33.932	35.064	36.163	37.247	38.335
40	26.509	29.051	30.856	32.345	33.660	34.872	36.021	37.134	38.233	39.335
41	27.326	29.907	31.740	33.251	34.585	35.813	36.977	38.105	39.219	40.335
42	28.144	30.765	32.626	34.157	35.510	36.755	37.935	39.077	40.205	41.335
43	28.965	31.625	33.512	35.065	36.436	37.698	38.892	40.050	41.191	42.335
44	29.787	32.487	34.400	35.974	37.363	38.641	39.851	41.022	42.177	43.335
45	30.612	33.350	35.290	36.884	38.291	39.585	40.809	41.995	43.164	44.335
46	31.439	34.215	36.180	37.795	39.220	40.529	41.769	42.968	44.150	45.335
47	32.268	35.081	37.072	38.708	40.149	41.474	42.728	43.942	45.137	46.335
48	33.098	35.949	37.965	39.621	41.079	42.420	43.689	44.915	46.124	47.335
49	33.930	36.818	38.859	40.534	42.010	43.366	44.649	45.889	47.111	48.335
50	34.764	37.689	39.754	41.449	42.942	44.313	45.610	46.864	48.099	49.335

Table 6.3: χ^2 values (continued)

	1- α									
v	0.500	0.550	0.600	0.650	0.700	0.750	0.800	0.850	0.900	0.950
1	0.455	0.571	0.708	0.873	1.074	1.323	1.642	2.072	2.706	3.841
2	1.386	1.597	1.833	2.100	2.408	2.773	3.219	3.794	4.605	5.991
3	2.366	2.643	2.946	3.283	3.665	4.108	4.642	5.317	6.251	7.815
4	3.357	3.687	4.045	4.438	4.878	5.385	5.989	6.745	7.779	9.488
5	4.351	4.728	5.132	5.573	6.064	6.626	7.289	8.115	9.236	11.070
6	5.348	5.765	6.211	6.695	7.231	7.841	8.558	9.446	10.645	12.592
7	6.346	6.800	7.283	7.806	8.383	9.037	9.803	10.748	12.017	14.067
8	7.344	7.833	8.351	8.909	9.524	10.219	11.030	12.027	13.362	15.507
9	8.343	8.863	9.414	10.006	10.656	11.389	12.242	13.288	14.684	16.919
10	9.342	9.892	10.473	11.097	11.781	12.549	13.442	14.534	15.987	18.307
11	10.341	10.920	11.530	12.184	12.899	13.701	14.631	15.767	17.275	19.675
12	11.340	11.946	12.584	13.266	14.011	14.845	15.812	16.989	18.549	21.026
13	12.340	12.972	13.636	14.345	15.119	15.984	16.985	18.202	19.812	22.362
14	13.339	13.996	14.685	15.421	16.222	17.117	18.151	19.406	21.064	23.685
15	14.339	15.020	15.733	16.494	17.322	18.245	19.311	20.603	22.307	24.996
16	15.338	16.042	16.780	17.565	18.418	19.369	20.465	21.793	23.542	26.296
17	16.338	17.065	17.824	18.633	19.511	20.489	21.615	22.977	24.769	27.587
18	17.338	18.086	18.868	19.699	20.601	21.605	22.760	24.155	25.989	28.869
19	18.338	19.107	19.910	20.764	21.689	22.718	23.900	25.329	27.204	30.144
20	19.337	20.127	20.951	21.826	22.775	23.828	25.038	26.498	28.412	31.410
21	20.337	21.147	21.991	22.888	23.858	24.935	26.171	27.662	29.615	32.671
22	21.337	22.166	23.031	23.947	24.939	26.039	27.301	28.822	30.813	33.924
23	22.337	23.185	24.069	25.006	26.018	27.141	28.429	29.979	32.007	35.172
24	23.337	24.204	25.106	26.063	27.096	28.241	29.553	31.132	33.196	36.415
25	24.337	25.222	26.143	27.118	28.172	29.339	30.675	32.282	34.382	37.652
26	25.336	26.240	27.179	28.173	29.246	30.435	31.795	33.429	35.563	38.885
27	26.336	27.257	28.214	29.227	30.319	31.528	32.912	34.574	36.741	40.113
28	27.336	28.274	29.249	30.279	31.391	32.620	34.027	35.715	37.916	41.337
29	28.336	29.291	30.283	31.331	32.461	33.711	35.139	36.854	39.087	42.557
30	29.336	30.307	31.316	32.382	33.530	34.800	36.250	37.990	40.256	43.773
31	30.336	31.323	32.349	33.431	34.598	35.887	37.359	39.124	41.422	44.985
32	31.336	32.339	33.381	34.480	35.665	36.973	38.466	40.256	42.585	46.194
33	32.336	33.355	34.413	35.529	36.731	38.058	39.572	41.386	43.745	47.400
34	33.336	34.371	35.444	36.576	37.795	39.141	40.676	42.514	44.903	48.602
35	34.336	35.386	36.475	37.623	38.859	40.223	41.778	43.640	46.059	49.802
36	35.336	36.401	37.505	38.669	39.922	41.304	42.879	44.764	47.212	50.998
37	36.336	37.416	38.535	39.715	40.984	42.383	43.978	45.886	48.363	52.192
38	37.335	38.430	39.564	40.760	42.045	43.462	45.076	47.007	49.513	53.384
39	38.335	39.445	40.593	41.804	43.105	44.539	46.173	48.126	50.660	54.572
40	39.335	40.459	41.622	42.848	44.165	45.616	47.269	49.244	51.805	55.758
41	40.335	41.473	42.651	43.891	45.224	46.692	48.363	50.360	52.949	56.942
42	41.335	42.487	43.679	44.934	46.282	47.766	49.456	51.475	54.090	58.124
43	42.335	43.501	44.706	45.976	47.339	48.840	50.548	52.588	55.230	59.304
44	43.335	44.514	45.734	47.017	48.396	49.913	51.639	53.700	56.369	60.481
45	44.335	45.527	46.761	48.058	49.452	50.985	52.729	54.810	57.505	61.656
46	45.335	46.541	47.787	49.099	50.507	52.056	53.818	55.920	58.641	62.830
47	46.335	47.554	48.814	50.139	51.562	53.127	54.906	57.028	59.774	64.001
48	47.335	48.567	49.840	51.179	52.616	54.196	55.993	58.135	60.907	65.171
49	48.335	49.580	50.866	52.219	53.670	55.265	57.079	59.241	62.038	66.339
50	49.335	50.592	51.892	53.258	54.723	56.334	58.164	60.346	63.167	67.505

Table 6.4: Student t-distribution (upper critical one-tailed values)

$n - 1$	$1 - \alpha/2$											
	0.700	0.725	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925	0.950	0.975
1	0.727	0.854	1.000	1.171	1.376	1.632	1.963	2.414	3.078	4.165	6.314	12.706
2	0.617	0.713	0.816	0.931	1.061	1.210	1.386	1.604	1.886	2.282	2.920	4.303
3	0.584	0.671	0.765	0.866	0.978	1.105	1.250	1.423	1.638	1.924	2.353	3.182
4	0.569	0.652	0.741	0.836	0.941	1.057	1.190	1.344	1.533	1.778	2.132	2.776
5	0.559	0.641	0.727	0.819	0.920	1.031	1.156	1.301	1.476	1.699	2.015	2.571
6	0.553	0.633	0.718	0.808	0.906	1.013	1.134	1.273	1.440	1.650	1.943	2.447
7	0.549	0.628	0.711	0.800	0.896	1.001	1.119	1.254	1.415	1.617	1.895	2.365
8	0.546	0.624	0.706	0.794	0.889	0.993	1.108	1.240	1.397	1.592	1.860	2.306
9	0.543	0.621	0.703	0.790	0.883	0.986	1.100	1.230	1.383	1.574	1.833	2.262
10	0.542	0.619	0.700	0.786	0.879	0.980	1.093	1.221	1.372	1.559	1.812	2.228
11	0.540	0.617	0.697	0.783	0.876	0.976	1.088	1.214	1.363	1.548	1.796	2.201
12	0.539	0.615	0.695	0.781	0.873	0.972	1.083	1.209	1.356	1.538	1.782	2.179
13	0.538	0.614	0.694	0.779	0.870	0.969	1.079	1.204	1.350	1.530	1.771	2.160
14	0.537	0.613	0.692	0.777	0.868	0.967	1.076	1.200	1.345	1.523	1.761	2.145
15	0.536	0.612	0.691	0.776	0.866	0.965	1.074	1.197	1.341	1.517	1.753	2.131
16	0.535	0.611	0.690	0.774	0.865	0.963	1.071	1.194	1.337	1.512	1.746	2.120
17	0.534	0.610	0.689	0.773	0.863	0.961	1.069	1.191	1.333	1.508	1.740	2.110
18	0.534	0.609	0.688	0.772	0.862	0.960	1.067	1.189	1.330	1.504	1.734	2.101
19	0.533	0.609	0.688	0.771	0.861	0.958	1.066	1.187	1.328	1.500	1.729	2.093
20	0.533	0.608	0.687	0.771	0.860	0.957	1.064	1.185	1.325	1.497	1.725	2.086
21	0.532	0.608	0.686	0.770	0.859	0.956	1.063	1.183	1.323	1.494	1.721	2.080
22	0.532	0.607	0.686	0.769	0.858	0.955	1.061	1.182	1.321	1.492	1.717	2.074
23	0.532	0.607	0.685	0.769	0.858	0.954	1.060	1.180	1.319	1.489	1.714	2.069
24	0.531	0.606	0.685	0.768	0.857	0.953	1.059	1.179	1.318	1.487	1.711	2.064
25	0.531	0.606	0.684	0.767	0.856	0.952	1.058	1.178	1.316	1.485	1.708	2.060
26	0.531	0.606	0.684	0.767	0.856	0.952	1.058	1.177	1.315	1.483	1.706	2.056
27	0.531	0.605	0.684	0.767	0.855	0.951	1.057	1.176	1.314	1.482	1.703	2.052
28	0.530	0.605	0.683	0.766	0.855	0.950	1.056	1.175	1.313	1.480	1.701	2.048
29	0.530	0.605	0.683	0.766	0.854	0.950	1.055	1.174	1.311	1.479	1.699	2.045
30	0.530	0.605	0.683	0.765	0.854	0.949	1.055	1.173	1.310	1.477	1.697	2.042
31	0.530	0.604	0.682	0.765	0.853	0.949	1.054	1.172	1.309	1.476	1.696	2.040
32	0.530	0.604	0.682	0.765	0.853	0.948	1.054	1.172	1.309	1.475	1.694	2.037
33	0.530	0.604	0.682	0.765	0.853	0.948	1.053	1.171	1.308	1.474	1.692	2.035
34	0.529	0.604	0.682	0.764	0.852	0.948	1.052	1.170	1.307	1.473	1.691	2.032
35	0.529	0.604	0.682	0.764	0.852	0.947	1.052	1.170	1.306	1.472	1.690	2.030
36	0.529	0.603	0.681	0.764	0.852	0.947	1.052	1.169	1.306	1.471	1.688	2.028
37	0.529	0.603	0.681	0.764	0.851	0.947	1.051	1.169	1.305	1.470	1.687	2.026
38	0.529	0.603	0.681	0.763	0.851	0.946	1.051	1.168	1.304	1.469	1.686	2.024
39	0.529	0.603	0.681	0.763	0.851	0.946	1.050	1.168	1.304	1.468	1.685	2.023
40	0.529	0.603	0.681	0.763	0.851	0.946	1.050	1.167	1.303	1.468	1.684	2.021
41	0.529	0.603	0.681	0.763	0.850	0.945	1.050	1.167	1.303	1.467	1.683	2.020
42	0.528	0.603	0.680	0.763	0.850	0.945	1.049	1.166	1.302	1.466	1.682	2.018
43	0.528	0.603	0.680	0.762	0.850	0.945	1.049	1.166	1.302	1.466	1.681	2.017
44	0.528	0.602	0.680	0.762	0.850	0.945	1.049	1.166	1.301	1.465	1.680	2.015
45	0.528	0.602	0.680	0.762	0.850	0.944	1.049	1.165	1.301	1.465	1.679	2.014
46	0.528	0.602	0.680	0.762	0.850	0.944	1.048	1.165	1.300	1.464	1.679	2.013
47	0.528	0.602	0.680	0.762	0.849	0.944	1.048	1.165	1.300	1.463	1.678	2.012
48	0.528	0.602	0.680	0.762	0.849	0.944	1.048	1.164	1.299	1.463	1.677	2.011
49	0.528	0.602	0.680	0.762	0.849	0.944	1.048	1.164	1.299	1.462	1.677	2.010
50	0.528	0.602	0.679	0.761	0.849	0.943	1.047	1.164	1.299	1.462	1.676	2.009
51	0.528	0.602	0.679	0.761	0.849	0.943	1.047	1.164	1.298	1.462	1.675	2.008
52	0.528	0.602	0.679	0.761	0.849	0.943	1.047	1.163	1.298	1.461	1.675	2.007
53	0.528	0.602	0.679	0.761	0.848	0.943	1.047	1.163	1.298	1.461	1.674	2.006
54	0.528	0.602	0.679	0.761	0.848	0.943	1.046	1.163	1.297	1.460	1.674	2.005
55	0.527	0.601	0.679	0.761	0.848	0.943	1.046	1.163	1.297	1.460	1.673	2.004
Inf	0.525	0.598	0.675	0.756	0.842	0.935	1.037	1.151	1.282	1.441	1.646	1.962



Appendix B – Modelling paradigms

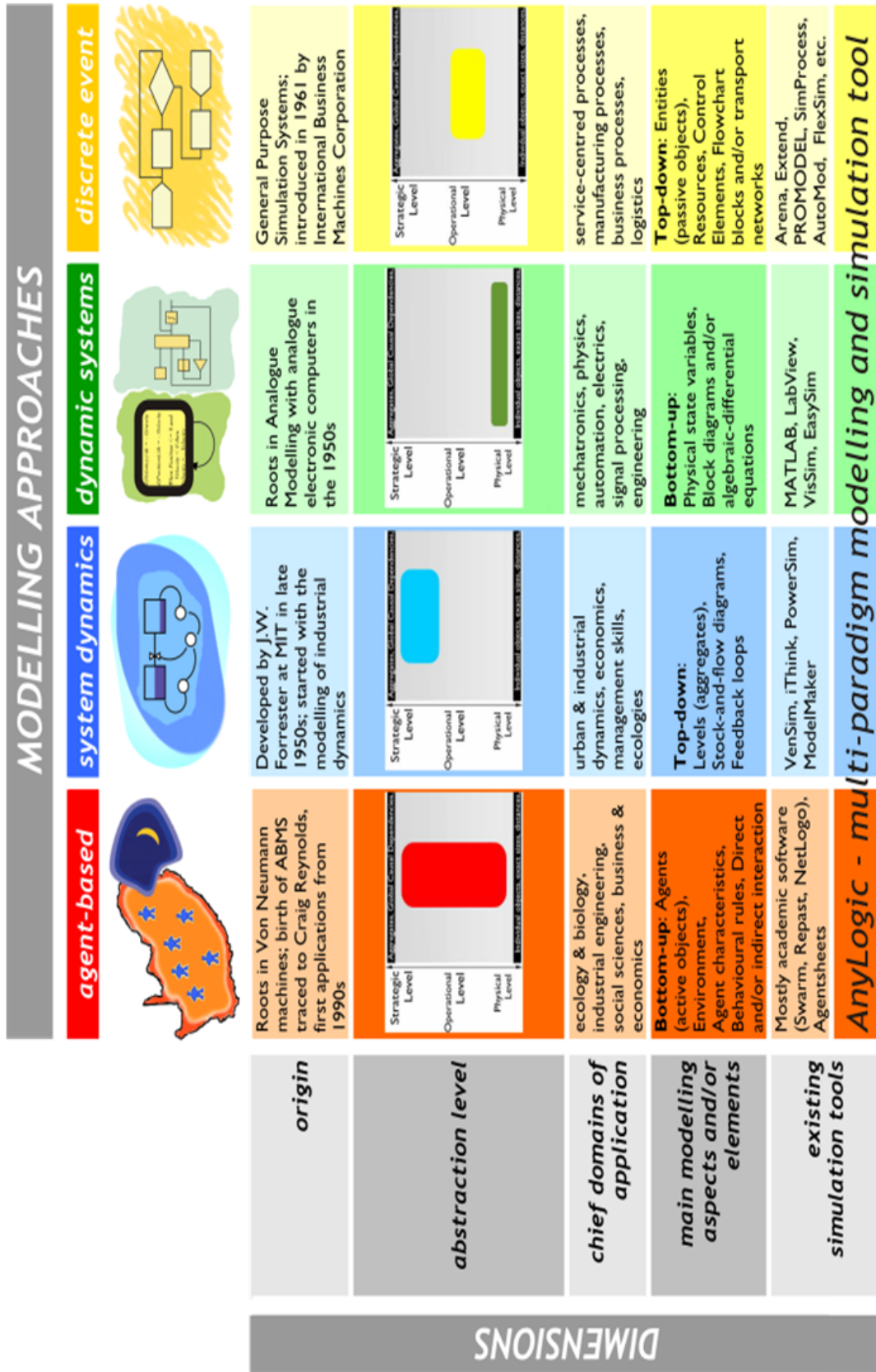


Figure 6.14: The four modelling paradigms



Bibliography

Books

- Banks, J. (1998). *Handbook of Simulation: Principles, Methodology, Advances, Applications and Practice*. New York: John Wiley & Sons Inc. (cited on pages 11, 24, 26, 27, 30, 35, 45, 76).
- Chung, C.A. (2004). *Simulation modeling handbook – A practical approach*. New York: CRC Press LL (cited on pages 21, 23).
- Coello Coello, C.A., G.B. Lamont, and D.A. van Veldhuizen (2007). *Evolutionary Algorithm for Solving Multi-Objective Problems*. Edited by D.E. Goldberg and J.R. Koza. Springer (cited on page 102).
- Denardo, E.V. (2002). *The Science of Decision Making: A Problem-Based Approach Using Excel*. John Wiley & Sons Inc. (cited on page 35).
- Gogg, T.J. and J.R.A. Mott (1992). *Improve Quality and Productivity with Simulation*. JMI Consulting Group (cited on page 66).
- Goldberg, D.E. (1989). *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley Publishing Company (cited on pages 87, 90).
- Kelton, W.D., R.P. Sadowski, and D.A. Sadowski (2002). *Simulation with Arena*. McGraw-Hill (cited on page 36).
- Knuth, Donald E. (1997). *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Third. Boston: Addison-Wesley. ISBN: 0201896842 9780201896848 (cited on page 36).
- Law, A.M. (2015). *Simulation Modeling & Analysis*. 5th. New York: McGraw-Hill (cited on pages 10, 11, 14, 30, 35, 36, 47, 50, 51, 53).
- Law, A.M. and W.D. Kelton (2000). *Simulation Modeling & Analysis*. 3rd. New York: McGraw-Hill (cited on pages 21, 24, 30, 67).
- Michalewicz, Z. and DB Fogel (2004). *How to Solve It: Modern Heuristics*. Springer, Berlin, Heidelberg (cited on page 86).
- Mitchell, Melanie (1997). *An introduction to genetic algorithms*. MIT Press (cited on page 90).
- O'Neill, Melissa E. (2014). *PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation*. Technical report. Computer Science Department, Harvey Mudd College (cited on page 36).
- Pegden, D., R.E. Shannon, and R.P. Sadowski (1995). *Introduction to Simulation using SIMAN*. 2nd. New York: McGraw-Hill (cited on pages 16, 32, 67).

- Pooch, Udo W. and James A. Wall (2000). *Discrete event simulation: A practical approach* (cited on page 23).
- Shannon, R. (1975). *System Simulation – the art & science*. Prentice-Hall Inc. (cited on pages 15, 32).
- Walpole, R.E. and R.H. Myers (1993). *Probability and Statistics for Engineers and Scientists*. 5th. MacMillan Publishing Company (cited on pages 31, 77).
- Winston, W.L. (2003). *Operations Research: Applications and Algorithms*. 4th edition. Cengage Learning (cited on pages 8, 11).

Articles

- Bekker, James and Chris Aldrich (May 2011). “The cross-entropy method in multi-objective optimisation: An assessment”. In: *European Journal of Operational Research* 211.1, pages 112–121. DOI: 10.1016/j.ejor.2010.10.028. URL: <http://dx.doi.org/10.1016/j.ejor.2010.10.028> (cited on page 102).
- Fishman, G.S. (Jan. 1978). “Grouping Observations in Digital Simulation”. In: *Management Science* 24.5, pages 510–521 (cited on page 71).
- Fonseca, C.M. and P.J. Fleming (1998). “Multiobjective optimization and multiple constraint handling with evolutionary algorithms. I. A unified formulation”. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 28.1, pages 26–37. DOI: 10.1109/3468.650319 (cited on page 100).
- Greenland, Sander et al. (2016). “Statistical tests, P-values, confidence intervals, and power: a guide to misinterpretations.” In: *European Journal of Epidemiology* 31.4, pages 337–350. DOI: <https://doi.org/10.1007/s10654-016-0149-3> (cited on page 51).
- Hwang, Ching-Lai, Young-Jou Lai, and Ting-Yun Liu (1993). “A new approach for multiple objective decision making”. In: *Computers & Operations Research* 20.8, pages 889–899. ISSN: 0305-0548. DOI: [https://doi.org/10.1016/0305-0548\(93\)90109-V](https://doi.org/10.1016/0305-0548(93)90109-V). URL: <http://www.sciencedirect.com/science/article/pii/030505489390109V> (cited on page 101).
- Kim, S-H. and B. Nelson (July 2001). “A fully sequential procedure for indifference-zone selection in simulation”. In: *ACM Transactions on Modeling and Computer Simulation* 11.3, pages 251–273. DOI: 10.1145/502109.502111. URL: <http://dx.doi.org/10.1145/502109.502111> (cited on page 81).
- Lee, Loo Hay et al. (2010). “Finding the non-dominated Pareto set for multi-objective simulation models”. In: *IIE Transactions* 42.9, pages 656–674. DOI: 10.1080/07408171003705367. eprint: <http://dx.doi.org/10.1080/07408171003705367>. URL: <http://dx.doi.org/10.1080/07408171003705367> (cited on page 101).
- Moonyoung Yoon, James Bekker (2017). “Single- and multi-objective ranking and selection procedures in simulation: a historical review”. In: *The South African Journal Of Industrial Engineering* 28.2, pages 37–45 (cited on pages 77, 98).
- Pierro, F. di, Soon-Thiam Khu, and D.A. Savić (Feb. 2007). “An Investigation on Preference Order Ranking Scheme for Multiobjective Evolutionary Optimization”. In: *Evolutionary Computation, IEEE Transactions on* 11.1, pages 17–45. ISSN: 1089-778X. DOI: 10.1109/TEVC.2006.876362 (cited on page 101).
- Rosen, Scott L., Catherine M. Harmonosky, and Mark T. Traband (2007). “A simulation optimization method that considers uncertainty and multiple performance measures”. In: *European Journal of Operational Research* 181.1, pages 315–330. ISSN: 0377-2217. DOI: DOI:10.1016/j.ejor.2006.05.040 (cited on page 86).

- Scholtz, Esmarie, James Bekker, and Delyno du Toit (2012). “Multi-objective optimisation with stochastic discrete-event simulation in retail banking: a case study”. In: *ORiON* 28.2 (cited on page 100).
- Sörensen, K. (2015). “Metaheuristics – the metaphor exposed”. In: *International Transactions in Operational Research* 22.1, pages 3–18 (cited on page 86).
- Stadler, Johan and James Bekker (2014). “Multi-objective optimisation in CO gas management at Tronox KZN Sands”. In: *South African Journal of Industrial Engineering* 25.2 (cited on page 100).
- Tsou, Ching-Shih (2008). “Multi-objective inventory planning using MOPSO and TOPSIS”. In: *Expert Systems with Applications* 35.1-2, pages 136–142. ISSN: 0957-4174. DOI: DOI : 10.1016/j.eswa.2007.06.009 (cited on page 97).
- Yoon, Moonyoung and James Bekker (2020). “Multi-objective simulation optimisation on discrete sets: a literature review”. In: *International Journal of Operational Research* 39.3, pages 364–405 (cited on page 99).
- (2022). “Bayesian-based indifference-zone multi-objective ranking and selection procedures”. In: *Computers & Industrial Engineering*. DOI: <https://doi.org/10.1016/j.cie.2022.108007> (cited on pages 99, 100).

Proceedings

- Ingalls, Ricki G. (2013). “Introduction to simulation”. In: *Proceedings of the 2013 Winter Simulation Conference*. Edited by R. Pasupathy et al., pages 291–305 (cited on page 16).
- Kelton, W.D. (1997). “Statistical Analysis of Simulation Output”. In: *Proceedings of the 1997 Winter Simulation Conference*. Edited by S. Andradóttir et al. SCSI, pages 23–30 (cited on page 59).
- Leemis, Lawrence (2001). “Input Modeling Techniques For Discrete-Event Simulations”. In: *Proceedings of the 2001 Winter Simulation Conference*. Piscataway, pages 62–73 (cited on page 45).
- Schriber, T.J., D.T. Brunner, and J.S. Smith (2016). “Inside discrete-event simulation software: how it works and why it matters”. In: *Proceedings of the 2016 Winter Simulation Conference*. Edited by T. M. K. Roeder et al., pages 221–235 (cited on page 17).
- Sturrock, David T. (2013). “Tutorial: Tips for successful practice of simulation”. In: *Proceedings of the 2013 Winter Simulation Conference*. Edited by R. Pasupathy et al., pages 354–361 (cited on page 24).
- (2018). “Avoid failures! Tested success tips for simulation project excellence”. In: *Proceedings of the 2018 Winter Simulation Conference*. Edited by M. Rabe et al., pages 252–260 (cited on page 15).
- White, K. Preston and Ricki G. Ingalls (2016). “The basics of simulation”. In: *Proceedings of the 2016 Winter Simulation Conference*. Edited by T. M. K. Roeder et al., pages 38–52 (cited on page 16).

Online

- Fred Glover, Kenneth Sörensen (2018). *Metaheuristics*. Online. Accessed on 10 December 2018 (cited on page 86).
- GP (2018). *About genetic programming*. <http://www.geneticprogramming.com>. Online, accessed on 10 December 2018 (cited on page 90).

- Kruger, P.S. and N.D. Du Preez (Dec. 1988). *Introduction to Simulation and SIMAN*. Course presented by The Institute for Industrial Engineering (University of Stellenbosch) and The Centre for Computer Integrated Production Systems (University of Pretoria) (cited on page 14).
- MathWorks (2018). *About genetic programming*. <https://uk.mathworks.com/help/gads/what-is-the-genetic-algorithm.html>. Online, accessed on 10 December 2018 (cited on page 90).
- Morrison, Aaron (2019). *Climate is what you expect; Weather is what you get*. <https://www.abc17news.com/weather/climate-is-what-you-expect-weather-is-what-you-get/565192216>. Online, accessed June 2019 (cited on page 59).
- Senn, Stephen (2018). *Being a statistician means never having to say you are certain*. <https://errorstatistics.com/2018/01/13/s-senn-being-a-statistician-means-never-having-to-say-you-are-certain-guest-post/>. Online, accessed June 2019 (cited on page 84).



<https://www.adazing.com/wp-content/uploads/2019/02/open-book-clipart-07-300x300.png>

Index

A

Advantages of simulation	13
Analysis of variance	77
Application areas	12
Attributes	16

B

Basic concepts needed	8
Batch-means approach	71
Boundaries	24

C

Central Limit Theorem	59
Chi-squared test	50
Chi-squared test procedure	51
Concept model	25
Confidence interval	63
Continuous random variable	12
Correlogram	74
Crossover	92

D

Data sources	45
Definition of simulation	11
DES definition	17
Discrete random variable	12
Drawbacks of simulation	13

E

Empirical distributions	47
Entities	16
Entity control structures	17
Entity states	17
Events	16
Examples of distribution fits	54

G

Genetic algorithm	90
Genetic algorithm in Tecnomatix	95

I

Inverse transform	37
-------------------------	----

K

Kim-Nelson procedure	81
Kolmogorov-Smirnov test	52
Kolmogorov-Smirnov test procedure	53

M

Maximum likelihood estimators	58
Modelling	9
Monte-Carlo simulation	42
MOO deterministic example	98
MOO problem formulation	97
MOO stochastic example	98

Multi-objective optimisation	96
Mutation	93

N

Non-terminating systems	69
-----------------------------------	----

P

Performance measures	23
Pitfalls of simulation	14
Point estimator	62
Population	91

R

Random number generation	35
Ranking of solutions	100
Replication-deletion approach	71
Resources	16

S

Sample path	18
Scenarios	76
Selection and replacement	93
Simulation and optimisation	85
Simulation mechanism	15
Simulation perspective	13
Six Ms	22
Solution selection	101
Some MOO algorithms	102
Steps in a study	21
Stochastic elements	98
Stopping conditions	93
System conditions	16
System state	16
Systems thinking	8

T

t-test	80
Terminating system	66
Theoretical distributions	49
Transaction flow view	17
Truncation point	69
TS number of observations	66

U

Uniform distribution	36, 39
--------------------------------	--------

V

Validation	30
Verification	29